Equivariant Generative Modeling of Buckling in Mechanical Metamaterials

Fleur Hendriks

Equivariant Generative Modeling of Buckling in Mechanical Metamaterials

Fleur Hendriks

Equivariant Generative Modeling of Buckling in Mechanical Metamaterials

Fleur Hendriks
Technische Universiteit Eindhoven, 2026

A catalogue record is available from the Eindhoven University of Technology Library
ISBN digital version: 978-90-386-6818-5

The printed version is published by Global Academic Press.
ISBN printed version: 978-94-6534-681-6

Cover explanation: the top half of the cover shows a metamaterial in the reference configuration. The bottom half shows the same metamaterial deformed after buckling.

Cover design: Fleur Hendriks
Printed by: ProefschriftMaken

This research was funded by EAISI.

© Copyright, 2026, Fleur Hendriks. All rights reserved.



Equivariant Generative Modeling of Buckling in Mechanical Metamaterials

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus prof.dr. S.K. Lenaerts, voor een commissie aangewezen door het College voor Promoties, in het openbaar te verdedigen op dinsdag 13 oktober 2026 om 13:30 uur

door

Fleur Hendriks

geboren te Weert

De samenstelling van de promotiecommissie is als volgt:

Voorzitter:	prof.dr.ir. J. den Toonder
Promotoren:	dr.ing. O. Rokoš
	prof.dr.ir. M.G.D. Geers
	dr. V. Menkovski
Leden:	dr. J. Brandstetter (Johannes Kepler University Linz)
	prof.dr. D. Kochmann (ETH Zürich)
	dr. S. Kumar (Technische Universiteit Delft)
	prof.dr. K.P. Veroy-Grepl

Het onderzoek of ontwerp dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

Contents

Summary **vii**

1 Introduction **1**
1.1 Mechanical metamaterials and the design problem 1
1.2 State of the art: from simulations to learned surrogate models 2
1.3 Symmetry, buckling, and multimodality 3
1.4 Datasets for metamaterial design 5
1.5 Research questions 5
1.6 Contributions and thesis outline 6

2 Similarity Equivariant Graph Neural Networks for Homogenization of Metamaterials **13**
2.1 Introduction 13
2.2 Dataset Generation 18
2.3 In- and Equivariance 26
2.4 GNN Approach 28
2.5 Results 40
2.6 Conclusion 48
2.7 Data Statement 49
Appendices 50
2.A Dataset 50
2.B Implementation 51
2.C Full results 52

3 Wallpaper Group-Based Mechanical Metamaterials: Dataset Including Mechanical Responses **71**
3.1 Background & Summary 71
3.2 Methods 74
3.3 Data Records 88
3.4 Technical Validation 93
3.5 Usage Notes 94
3.6 Code availability 96
Appendices 103
3.A Periodic graph connectivity check 103

4 Equivariant Flow Matching for Symmetry-Breaking Bifurcation Problems **105**
4.1 Introduction 105
4.2 Equivariance and Symmetry Breaking 107
4.3 Experiments and Results 108
4.4 Conclusion 117

5 Probabilistic Modeling of Buckling in Mechanical Metamaterials 121
5.1 Introduction . . . 121
5.2 Problem Formulation . . . 125
5.3 Architectural Building Blocks . . . 128
5.4 Implementation . . . 146
5.5 Results . . . 159
5.6 Conclusion . . . 178

6 Discussion and Outlook 189
6.1 Revisiting the research questions . . . 189
6.2 Main achievements and novelties . . . 190
6.3 Implications for metamaterial design . . . 191
6.4 Limitations and outlook . . . 191

Curriculum Vitae 194

List of publications 195

Awards 195

Acknowledgements 196

Summary

Mechanical metamaterials are engineered materials whose mechanical behavior is governed not only by the material they are made from, but also by the geometry of their internal microstructure. By carefully designing this microstructure, these materials can exhibit unusual and useful properties, such as tunable stiffness, negative Poisson's ratio, programmable shape changes, and controllable wave propagation. A particularly powerful mechanism behind these functionalities is buckling. Under loading, a structure may suddenly switch from one deformation pattern to another, giving rise to large changes in the overall mechanical response. This makes buckling useful for design, but also difficult to simulate and predict.

The main difficulty addressed in this thesis is that the mechanical response of buckling metamaterials is nonlinear, strongly geometry-dependent, and governed by symmetry. In addition, the post-buckling solution space can be multimodal: a single microstructure under a fixed loading condition may admit several physically valid deformation patterns. This is challenging not only for machine-learning models, but also for conventional finite-element simulations, which can accurately compute individual deformation paths, but do not automatically reveal the full set of possible post-buckling solutions. The solution that is obtained may depend on imperfections, perturbations, or numerical choices. Exploring multiple admissible responses therefore requires repeated simulations or post-processing. This can become computationally prohibitive when many simulations are needed for design, optimization, or uncertainty analysis. Machine-learning surrogate models offer a promising route to faster prediction, but standard deterministic models often ignore important physical structure. They may fail to respect geometric symmetries, generalize poorly to unseen microstructures, or collapse several valid post-buckling solutions into one averaged, nonphysical prediction.

This thesis investigates how machine learning can be adapted to this setting by embedding physical and geometric structure directly into the learning problem. The work is organized around four research questions.

1. How can neural-network surrogate models be designed so that they respect the symmetries and physical constraints of mechanical metamaterials? The first part of the thesis addresses the need for fast and physically consistent surrogate models. A reliable model should not depend on arbitrary choices such as the position or orientation of the coordinate system, the selected representative volume element, or the scale at which an equivalent periodic cell is described. These constraints result from fundamental symmetries of the underlying mechanical problem.

To incorporate these symmetries, this thesis develops a similarity-equivariant graph neural network, SimEGNN. The model represents the microstructure as a graph and predicts both effective macroscopic quantities, such as energy, stress, and stiffness,

and local deformation fields. The key idea is that relevant symmetries are not only learned from data, but built directly into the architecture. Translation, rotation, reflection, scaling, and periodicity are treated as inductive biases that guide the model towards physically consistent predictions.

Numerical comparisons with less constrained graph neural networks show that these architectural choices improve the consistency and accuracy of the model for the considered metamaterial system. In this way, the approach demonstrates how equivariant graph neural networks can be adapted to nonlinear mechanical metamaterial simulations while preserving essential geometric and physical structure.

2. How can a dataset be constructed to study and learn mechanical responses across a broad range of loading conditions and metamaterial designs, including all periodic planar symmetry groups? The development of robust machine-learning models depends on training data that captures the diversity of the design space. For mechanical metamaterials, this diversity is not limited to the size or shape of holes in a unit cell. It also includes the symmetry group, and the loading paths under which instabilities may occur. Existing datasets often cover only a limited range of geometries, symmetries, or deformation regimes, which makes it difficult to train models that generalize beyond a narrow set of examples.

The second part of this thesis addresses this gap by introducing a dataset of two-dimensional mechanical metamaterials based on the 17 wallpaper groups. These groups describe all possible planar periodic symmetries, including translations, rotations, reflections, and glide reflections. By using them as an organizing principle, the dataset systematically explores a broad range of periodic microstructures with different symmetry properties.

The dataset contains 1,020 microstructures together with their simulated mechanical responses under multiple loading paths in the finite-strain regime, including cases where buckling occurs. Because it combines prescribed symmetry groups with nonlinear mechanical responses, the dataset is useful not only for training surrogate models but also for studying how symmetry influences properties such as anisotropy, auxeticity, and the number and type of admissible buckling modes.

3. How can machine-learning models represent symmetry breaking, where one symmetric input may lead to multiple valid outputs? Buckling creates a fundamental difficulty for deterministic prediction. Before buckling, a structure may be highly symmetric. After buckling, an individual deformation pattern can break this symmetry. However, the full set of possible post-buckling solutions still reflects the symmetry of the original system. This means that the correct object to learn is not always a single deformation, but a distribution over several admissible outcomes.

Standard deterministic machine-learning models are poorly suited to this situation. When several valid solutions exist for the same input, such models tend to average them. For buckling problems, this average may not correspond to any physically valid deformation pattern. A different formulation is therefore needed: instead of predicting one output, the model should predict a probability distribution over the set of possible post-buckling responses, and be able to sample from it.

The third part of this thesis addresses this challenge using equivariant flow matching. Flow matching is a generative modeling approach that learns how to transform simple random samples into samples from a target distribution. By combining it with equivariant neural networks, the learned distribution respects the symmetries of the physical system, while individual samples are still allowed to break those symmetries.

The central contribution of this part is the distinction between symmetry at the level of a single solution and symmetry at the level of the full solution distribution. Individual post-buckling modes may be symmetry-broken, but the collection of all possible modes should transform consistently under the symmetries of the system. This thesis demonstrates, on controlled bifurcation problems, that equivariant generative models can represent such symmetry-breaking behavior without collapsing multiple solutions into a nonphysical average.

4. How can equivariant generative models be scaled to realistic buckling trajectories in mechanical metamaterials? The final part of the thesis brings the previous ideas together for the prediction of full buckling trajectories in graph-structured mechanical metamaterials. This setting is substantially more challenging than low-dimensional bifurcation examples. Realistic metamaterial simulations involve many degrees of freedom, periodic boundary conditions, long-range interactions, and the need for consistency over an entire loading path. During buckling, distant parts of the structure must select compatible deformation patterns. A model based only on local interactions may therefore fail to coordinate the response globally.

To address this, the thesis combines equivariant graph neural networks, flow matching, temporal modeling, and a hierarchical architecture into one probabilistic framework. The contribution is the adaptation and integration of these components for the specific problem of buckling in mechanical metamaterials. The graph representation captures the geometry of the microstructure, the equivariant architecture preserves physical symmetries, the generative formulation represents multiple admissible solutions, and the hierarchical structure helps coordinate local and global deformation patterns.

This framework enables a single model to generate different physically plausible deformation trajectories for the same microstructure and loading path. It is evaluated on both the stacked-holes benchmark that was also used to evaluate the SimEGNN, and the wallpaper-group dataset introduced earlier. The results show that the model can sample competing buckling trajectories and capture important aspects of their spatial and temporal coherence. At the same time, they also reveal current limitations in accuracy, computational cost, and scalability, which remain important directions for future work.

Together, the four parts of this thesis develop a coherent approach to learning the nonlinear and symmetry-governed behavior of buckling mechanical metamaterials, including cases with multiple admissible post-buckling solutions. The thesis shows that machine learning for such systems should not be treated as generic function approximation. Instead, the model architecture, the dataset, and the probabilistic formulation should all reflect the structure of the underlying physics.

The main outcome is an equivariant data-driven framework that combines graph

neural networks, systematic datasets, and generative modeling. This framework enables faster simulation, improved generalization across microstructures, and a systematic treatment of multiple possible post-buckling responses. More broadly, the thesis contributes to the development of machine-learning methods for physical systems in which geometry, symmetry, instability, and multiple possible solutions play a central role.

CHAPTER 1

Introduction

1.1 Mechanical metamaterials and the design problem

Metamaterials are materials whose effective properties are determined not only by their chemical composition, but also by an intentionally designed internal structure. In the case of mechanical metamaterials, the macroscopic mechanical response can be programmed through geometry, by arranging material and voids at a smaller length scale. This principle is used to design mechanical behavior that is rare or even absent in conventional solids, such as negative Poisson's ratio, negative compressibility, and tunable stiffness [1–3]. These properties make mechanical metamaterials attractive for applications including acoustic and elastic wave control [4–6], energy absorption [7], soft robotic actuation [8, 9], programmable shape morphing [10], and mechanical logic [11, 12].

A particularly rich class of mechanical metamaterials exploits elastic instabilities. In these materials, external loading can trigger buckling-induced pattern transformations, where the microstructure switches from one deformation pattern to another [13, 14]. This mechanism is useful because it enables large, reversible, and controllable changes in macroscopic response. At the same time, it makes simulation and design difficult. The response is nonlinear, sensitive to geometry and loading, and may change abruptly once a critical loading condition is reached.

Designing a mechanical metamaterial means choosing a microstructure that produces a prescribed behavior at the macroscopic scale. Typical design targets include a desired stiffness, a negative Poisson's ratio, or a specific buckling response. Because many mechanical metamaterials are periodic, their behavior can often be studied by analysing one representative repeating portion of the material rather than the full material. In computational mechanics, this portion is called a representative volume element (RVE). The effective response of the material is obtained by applying a macroscopic loading to the RVE and solving the corresponding microscale boundary value problem.

Such simulations can be used to evaluate a single proposed design, but they are often required repeatedly during optimization, where the geometry is updated until the target response is achieved [15, 16], as illustrated in Figure 1.1. For buckling metamaterials, this procedure is computationally expensive. Each design evaluation may require a nonlinear simulation involving large deformations and stability analysis. At the same time, the design space is large: the shape, size and number of holes, the symmetry of the microstructure, the shape of the RVE, material parameters, and loading path can all be varied.

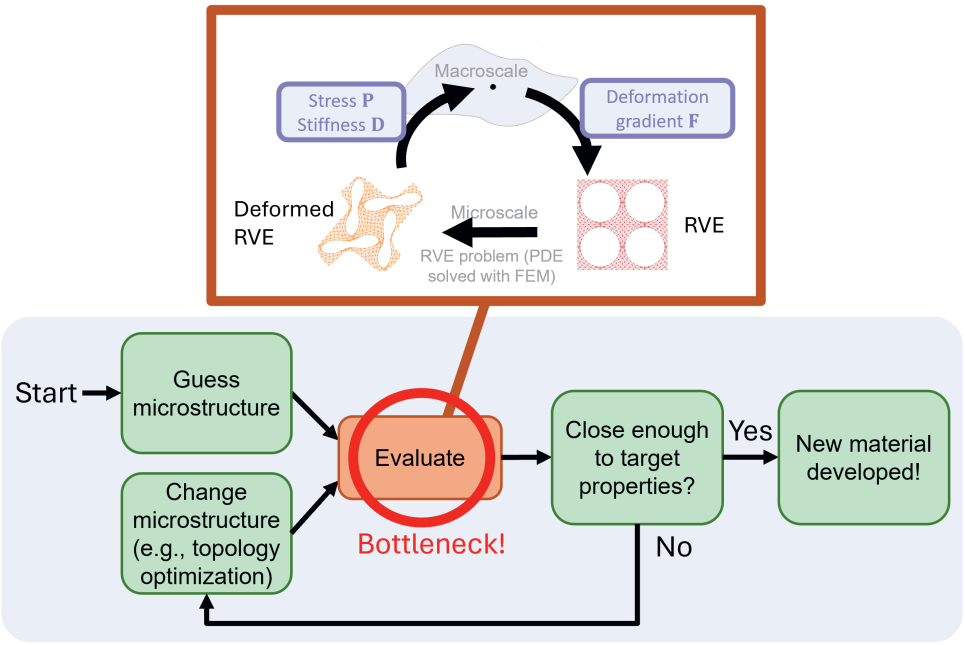


Figure 1.1: Schematic view of the metamaterial design problem. A microstructure is evaluated using the finite element method (FEM) under prescribed macroscopic loading to obtain effective mechanical quantities and deformation fields. These quantities are then used to optimize the design. Fast surrogate models can replace repeated finite-element simulations in this loop.

The ability to manufacture increasingly complex architected materials has expanded the accessible design space [17, 18]. However, as the design space grows, simulations become more complicated and often more computationally expensive. Classical homogenization methods provide the theoretical basis for linking microscale geometry to macroscale response [19,20], and topology optimization has demonstrated that microstructures can be designed for prescribed effective properties [15,16]. Nevertheless, when large deformations and instabilities are involved, repeated nonlinear simulations remain expensive [21, 22]. This motivates the use of machine learning as a fast surrogate for forward simulation, and ultimately as a component in data-driven metamaterial design.

1.2 State of the art: from simulations to learned surrogate models

Machine learning has become a promising tool for accelerating the simulation and design of mechanical systems [23–25]. In the context of metamaterials, learned sur-

rogate models aim to approximate the mapping from a microstructure and loading condition to mechanical quantities such as energy, stress, stiffness, or deformation fields. Once trained, such models can be evaluated much faster than conventional finite-element simulations, making them attractive for design-space exploration.

Several types of surrogate models have been explored. Reduced-order models can be highly efficient when the relevant deformation space is low-dimensional, but they are usually tied to a fixed geometry or a prescribed parameterization [26–29]. Convolutional neural networks can process image-based geometry descriptions and have been used for metamaterial property prediction and design [30, 31]. However, image-based models are naturally tied to regular grids and do not directly represent finite-element connectivity, arbitrary changes in topology, or varying shapes of the RVE. Graph neural networks (GNNs) provide a more flexible alternative because meshes, point clouds, and skeleton-like geometry descriptions can all be represented as graphs. They also naturally respect permutation invariance: the prediction should not depend on the arbitrary ordering of nodes or elements [23].

For physical systems, however, permutation invariance alone is not enough. Mechanical responses are constrained by geometric symmetries. Translating or rotating the coordinate system should not change scalar quantities such as energy, while vector and tensor quantities should transform consistently with the coordinate transformation. This principle is closely related to material objectivity [32]. Equivariant neural networks incorporate such transformation rules directly into the model architecture, instead of requiring the model to learn them from data [33–37]. This provides an important inductive bias: it reduces the number of physically equivalent cases that must be learned separately and improves generalization to transformed configurations.

Despite these developments, standard equivariant architectures do not automatically satisfy all constraints relevant to periodic mechanical metamaterials. In addition to Euclidean symmetries such as translations, rotations, and reflections, the model should handle periodic boundary conditions, changes in the choice of RVE window, scale changes, finite deformations, and multiple types of output quantities. For periodic mechanical metamaterials, a useful surrogate model therefore has to combine geometric flexibility with the relevant physical constraints: it should process arbitrary microstructures, respect the symmetries of the homogenization problem, and remain efficient enough to be used in repeated simulations or design loops.

1.3 Symmetry, buckling, and multimodality

Buckling metamaterials introduce an additional complication that goes beyond fast surrogate modeling. The deformation after buckling is often not unique. Before buckling, the structure may possess a high degree of symmetry. Beyond a critical load, several symmetry-related post-buckling modes can become admissible [38, 39]. For example, a porous periodic structure may buckle clockwise or counterclockwise. The input can therefore correspond to several physically valid outputs. See Figure 1.2 for an example.

As a result, the mapping from microstructure and loading to deformation is fundamentally *one-to-many* rather than deterministic. This is problematic for deterministic

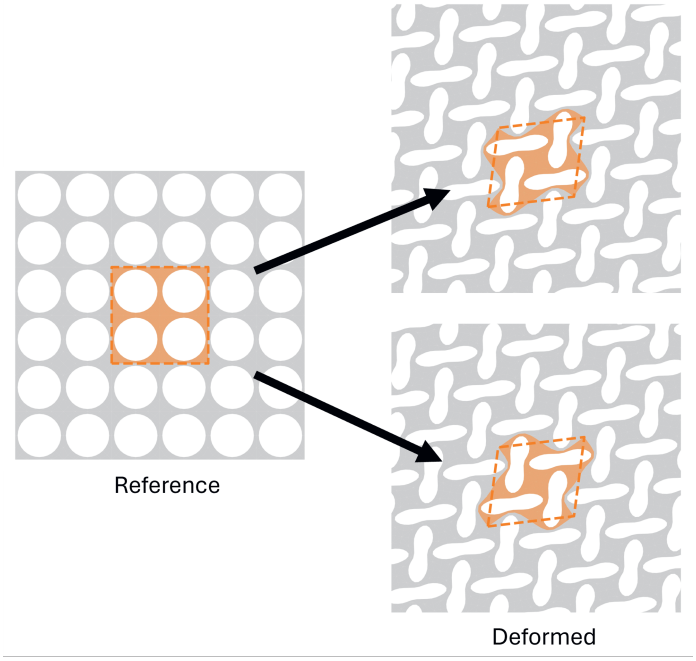


Figure 1.2: Example of symmetry-breaking resulting from buckling. A single symmetric input may admit multiple symmetry-related post-buckling configurations, so the response is better described as a distribution over admissible outcomes than as one deterministic deformation field.

machine learning models. A model trained with a mean-squared-error loss will average over possible outcomes. For buckling, such an average may be nonphysical: it can lie between stable deformation modes rather than corresponding to any admissible post-buckling state [40]. Equivariance by itself does not solve this issue. In fact, a deterministic equivariant model given a symmetric input is constrained to produce a symmetric output, whereas the physically realised post-buckling state may break that symmetry [41].

This issue is not unique to mechanical metamaterials. Symmetry-breaking bifurcations also occur in many other physical and dynamical systems, such as fluid dynamics [42], climate systems [43], and biological processes [44]. In all these cases, the governing equations or input conditions may possess a symmetry that is not shared by individual realised states. The relevant learning problem is therefore not only to predict a single response, but to learn a probability distribution over all possible responses. Such distributions should still respect the symmetries of the underlying physical system, even when individual samples from the distribution break those symmetries. Generative modeling provides a natural framework for this setting, because it can recover multiple admissible outputs for the same input. Continuous-time generative approaches such as flow matching are particularly suitable because they learn

to transport a simple prior distribution to a complex target distribution in multiple steps [45, 46], which makes them particularly effective at modeling multimodal and highly concentrated distributions. Diffusion-based generative modeling has also been explored for nonlinear metamaterial design [47]. However, applying such ideas to mechanical bifurcation problems requires care: the model must distinguish between symmetry at the distribution level and symmetry breaking at the sample level.

A further challenge appears when moving from low-dimensional bifurcation examples to realistic metamaterial simulations. A full deformation trajectory may contain thousands of degrees of freedom, periodic boundary conditions, changing unit cell shape, and long-range mechanical coordination. During buckling, different regions of the structure should not independently choose incompatible modes. A physically meaningful trajectory requires global coordination across the entire graph-structured microstructure. This motivates architectures that combine local message passing with mechanisms for long-range communication, while preserving the relevant equivariances.

1.4 Datasets for metamaterial design

Progress in data-driven modeling also depends on the availability of suitable datasets. Existing datasets for mechanical metamaterials often focus on a restricted family of unit cells, a small set of loading conditions, or small-deformation properties [47, 48]. Such datasets are valuable, but they are not sufficient for studying the combined effects of geometry, loading, symmetry, and buckling across a broad design space.

Although real mechanical metamaterials are three-dimensional, many porous and cellular systems are commonly studied through periodic, two-dimensional, plane-strain or plane-stress models. In this setting, symmetry is especially important. The possible planar periodic symmetry groups are the 17 wallpaper groups [49]. These groups include translations, rotations, reflections, and glide reflections. These symmetries constrain admissible deformation modes and influence mechanical properties such as anisotropy, auxeticity, and buckling behavior [50, 51], however, no systematic categorization of the relationship between symmetry and buckling or mechanical properties has been established yet. A dataset that systematically covers all wallpaper groups can therefore support two goals at once. First, it can provide training data for machine-learning models that must generalise across geometries and symmetries. Second, it can be used to study how symmetry affects mechanical response, including the tendency to buckle and the multiplicity of possible modes.

1.5 Research questions

The preceding discussion identifies several sources of structure that are important in the modeling of buckling mechanical metamaterials: geometry, loading, periodicity, symmetry, scale, graph connectivity, and the possible coexistence of multiple post-buckling responses. These structures provide natural starting points for constructing datasets and machine-learning models. The thesis is organised around the following research questions.

- **RQ1. How can neural-network surrogate models be designed so that they respect the symmetries and physical constraints of mechanical metamaterials?**
  The first challenge is to replace expensive finite-element simulations with a fast surrogate that remains consistent under changes of coordinate system, RVE choice, and scale. This requires a model that can process geometry directly, predict both local deformation and homogenized quantities, and respect the symmetries of the underlying mechanics.
- **RQ2. How can a dataset be constructed to study and learn mechanical responses across a broad range of loading conditions and metamaterial designs, including all periodic planar symmetry groups?**
  General surrogate models require data that are not restricted to one canonical microstructure. The second challenge is therefore to generate a diverse, high-quality dataset of periodic porous geometries, covering all 17 wallpaper groups and including nonlinear large-deformation responses under multiple loading paths.
- **RQ3. How can machine-learning models predict symmetry breaking, where one symmetric input may lead to multiple valid outputs?**
  Buckling creates multiple admissible outcomes for the same input. The third challenge is to move from deterministic prediction to probabilistic modeling, while ensuring that the learned distribution respects the symmetries of the system even though individual samples may break them.
- **RQ4. How can equivariant generative models be scaled to realistic buckling trajectories in mechanical metamaterials?**
  The final challenge is to combine the previous ideas in a realistic setting: arbitrary porous microstructures, loading trajectories, periodic graphs, and globally coordinated buckling modes. This requires architectures that capture both local mechanical interactions and long-range coordination.

1.6 Contributions and thesis outline

This thesis addresses the research questions through four main contributions, presented in Chapters 2–5. Together, these contributions develop an equivariant and probabilistic approach to learning the response of buckling mechanical metamaterials.

Chapter 2 addresses RQ1 by developing a similarity-equivariant graph neural network, SimEGNN, for the homogenization of periodic porous metamaterials. The model predicts the local deformation as well as homogenized quantities such as strain energy density, stress, and stiffness. Starting from an $E(n)$-equivariant graph neural network [52], the architecture is extended to account for periodicity, scale equivariance, and tensorial output quantities. The chapter introduces the graph representation, the relevant invariances and equivariances, and the architectural modifications required to incorporate these physical constraints into a graph-based surrogate model.

Chapter 3 addresses RQ2 by introducing a dataset of two-dimensional wallpaper-group-based mechanical metamaterials. The dataset contains 1,020 microstructures

spanning all 17 wallpaper groups, together with simulated mechanical responses under multiple loading paths in the hyperelastic large-deformation regime. The geometries are generated from periodic graph skeletons and parametrised hole boundaries, with checks for connectivity and ligament thickness. The chapter describes the geometry generation procedure, the simulation setup, the data records, and possible uses of the dataset for learning and analyzing mechanical responses across a large geometry, loading, and symmetry space.

Chapter 4 addresses RQ3 by developing an equivariant flow-matching framework for symmetry-breaking bifurcation problems. The key idea is to model the full distribution of possible outcomes instead of a single deterministic solution. The framework preserves equivariance at the distribution level, while allowing individual samples to select symmetry-broken states. Symmetric coupling is introduced to align group-equivalent targets during training. The chapter formulates the distinction between equivariance of distributions and symmetry breaking of individual samples, and demonstrates the approach on several bifurcation problems, including a buckling beam and phase separation described by the Allen–Cahn equation.

Chapter 5 addresses RQ4 by extending the probabilistic approach to full buckling trajectories in mechanical metamaterials. The model combines equivariant graph neural networks, flow matching, a specialized random-walk prior over entire trajectories, temporal U-Nets, and hierarchical message passing. The hierarchical architecture is introduced to improve global coordination, which is essential when an entire structure must consistently select one buckling mode. The chapter evaluates the framework on both the stacked-holes benchmark and the wallpaper-group metamaterial dataset, thereby scaling equivariant generative modeling from low-dimensional bifurcation examples to high-dimensional graph-structured metamaterial systems.

Chapter 6 concludes the thesis by summarizing the answers to the research questions, discussing the remaining limitations, and outlining future research directions for equivariant generative modeling and data-driven metamaterial design.

1

Bibliography

[1] K. Bertoldi, V. Vitelli, J. Christensen, M. Van Hecke, Flexible mechanical metamaterials (oct 2017). doi:10.1038/natrevmats.2017.66.
URL https://www.nature.com/articles/natrevmats201766

[2] S. Babaee, J. Shim, J. C. Weaver, E. R. Chen, N. Patel, K. Bertoldi, 3D soft metamaterials with negative poisson's ratio, Advanced Materials 25 (36) (2013) 5044–5049. doi:10.1002/adma.201301986.
URL https://onlinelibrary.wiley.com/doi/10.1002/adma.201301986

[3] Z. G. Nicolaou, A. E. Motter, Mechanical metamaterials with negative compressibility transitions, Nature Materials 11 (7) (2012) 608–613. arXiv:1207.2185, doi:10.1038/nmat3331.
URL https://www.nature.com/articles/nmat3331

[4] L. Wu, Y. Wang, K. Chuang, F. Wu, Q. Wang, W. Lin, H. Jiang, A brief review of dynamic mechanical metamaterials for mechanical energy manipulation, Materials Today 44 (April) (2021) 168–193. doi:10.1016/j.mattod.2020.10.006.
URL https://doi.org/10.1016/j.mattod.2020.10.006

[5] S. Ning, D. Chu, H. Jiang, F. Yang, Z. Liu, Z. Zhuang, The role of material and geometric nonlinearities and damping effects in designing mechanically tunable acoustic metamaterials, International Journal of Mechanical Sciences 197 (January) (2021) 106299. doi:10.1016/j.ijmecsci.2021.106299.
URL https://doi.org/10.1016/j.ijmecsci.2021.106299

[6] F. Wang, O. Sigmund, J. S. Jensen, Design of materials with prescribed nonlinear properties, Journal of the Mechanics and Physics of Solids 69 (1) (2014) 156–174. doi:10.1016/j.jmps.2014.05.003.

[7] A. Bekele, M. A. Wadee, A. Phillips, Enhancing energy absorption through sequential instabilities in mechanical metamaterials, Royal Society Open Science 10 (8) (2023).

[8] D. Yang, B. Mosadegh, A. Ainla, B. Lee, F. Khashai, Z. Suo, K. Bertoldi, G. M. Whitesides, Buckling of Elastomeric Beams Enables Actuation of Soft Machines, Advanced Materials 27 (41) (2015) 6323–6327. doi:10.1002/adma.201503188.
URL https://onlinelibrary.wiley.com/doi/full/10.1002/adma.201503188https://onlinelibrary.wiley.com/doi/abs/10.1002/adma.201503188https://onlinelibrary.wiley.com/doi/10.1002/adma.201503188

[9] Y. Kim, G. A. Parada, S. Liu, X. Zhao, Ferromagnetic soft continuum robots, Science Robotics 4 (33) (2019) 7329. doi:10.1126/SCIROBOTICS.AAX7329.
URL https://www.science.org/doi/10.1126/scirobotics.aax7329

[10] K. Bertoldi, P. M. Reis, S. Willshaw, T. Mullin, Negative poisson's ratio behavior induced by an elastic instability, Advanced Materials 22 (3) (2010) 361–366. doi:10.1002/adma.200901956.

[11] C. El Helou, P. R. Buskohl, C. E. Tabor, R. L. Harne, Digital logic gates in soft, conductive mechanical metamaterials, Nature communications 12 (1) (2021) 1633.

[12] J. Byun, A. Pal, J. Ko, M. Sitti, Integrated mechanical computing for autonomous soft machines, Nature Communications 15 (1) (2024) 2933.

[13] K. Bertoldi, M. C. Boyce, S. Deschanel, S. M. Prange, T. Mullin, Mechanics of deformation-triggered pattern transformations and superelastic behavior in periodic elastomeric structures, Journal of the Mechanics and Physics of Solids 56 (8) (2008) 2642–2668. doi:10.1016/j.jmps.2008.03.006.
URL http://arxiv.org/abs/2301.11355

[14] J. T. Overvelde, K. Bertoldi, Relating pore shape to the non-linear response of periodic elastomeric structures, Journal of the Mechanics and Physics of Solids 64 (1) (2014) 351–366. doi:10.1016/j.jmps.2013.11.014.
URL http://dx.doi.org/10.1016/j.jmps.2013.11.014

[15] M. P. Bendsøe, O. Sigmund, Topology optimization: Theory, methods and applications, 2004. URL https://link.springer.com/book/10.1007/978-3-662-05086-6#toc

[16] O. Sigmund, Materials with prescribed constitutive parameters: An inverse homogenization problem, International Journal of Solids and Structures 31 (17) (1994) 2313–2329. doi:10.1016/0020-7683(94)90154-6.

[17] Y. Jiang, Q. Wang, Highly-stretchable 3D-architected Mechanical Metamaterials, Scientific Reports 6 (1) (2016) 1–11. doi:10.1038/srep34147.
URL https://www.nature.com/articles/srep34147

[18] X. Ren, J. Shen, P. Tran, T. D. Ngo, Y. M. Xie, Design and characterisation of a tuneable 3D buckling-induced auxetic metamaterial, Materials and Design 139 (2018) 336–342. doi:10.1016/j.matdes.2017.11.025.

[19] R. Hill, Elastic properties of reinforced solids: Some theoretical principles, Journal of the Mechanics and Physics of Solids 11 (5) (1963) 357–372. doi:10.1016/0022-5096(63)90036-X.

[20] T. Mori, K. Tanaka, Average stress in matrix and average elastic energy of materials with misfitting inclusions, Acta Metallurgica 21 (5) (1973) 571–574. doi:10.1016/0001-6160(73)90064-3.

[21] C. R. Thomsen, F. Wang, O. Sigmund, Buckling strength topology optimization of 2D periodic materials based on linearized bifurcation analysis, Computer Methods in Applied Mechanics and Engineering 339 (2018) 115–136. doi:10.1016/j.cma.2018.04.031.
URL www.sciencedirect.comwww.elsevier.com/locate/cma

[22] F. Wang, O. Sigmund, 3D architected isotropic materials with tunable stiffness and buckling strength, Journal of the Mechanics and Physics of Solids 152 (2021) 104415. arXiv:2012.01359, doi:10.1016/j.jmps.2021.104415.

[23] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, P. W. Battaglia, Learning Mesh-Based Simulation with Graph Networks (oct 2020). arXiv:2010.03409.
URL http://arxiv.org/abs/2010.03409

[24] A. L. Frankel, R. E. Jones, C. Alleman, J. A. Templeton, Predicting the mechanical response of oligocrystals with deep learning, Computational Materials Science 169 (2019). arXiv:1901.10669, doi:10.1016/j.commatsci.2019.109099.
URL www.elsevier.com/locate/commatsci

[25] A. Pandey, R. Pokharel, Machine learning based surrogate modeling approach for mapping crystal deformation in three dimensions, Scripta Materialia 193 (2021) 1–5. doi:10.1016/j.scriptamat.2020.10.028.

[26] G. Kerschen, J.-C. Golinval, A. F. Vakakis, L. A. Bergman, The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview, Tech. rep. (2005).

[27] T. Guo, O. Rokoš, K. Veroy, Learning constitutive models from microstructural simulations via a non-intrusive reduced basis method, Computer Methods in Applied Mechanics and Engineering 384 (2021) 113924.

[28] T. Guo, F. A. Silva, O. Rokoš, K. Veroy, Learning constitutive models from microstructural simulations via a non-intrusive reduced basis method: Extension to geometrical parameterizations, Computer Methods in Applied Mechanics and Engineering 401 (2022) 115636.

[29] T. Guo, O. Rokoš, K. Veroy, A reduced order model for geometrically parameterized two-scale simulations of elasto-plastic microstructures under large deformations, Computer Methods in Applied Mechanics and Engineering 418 (2024) 116467.

[30] J. K. Wilt, C. Yang, G. X. Gu, Accelerating Auxetic Metamaterial Design with Deep Learning, Advanced Engineering Materials 22 (5) (2020) 1901266. doi:10.1002/adem.201901266.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/adem.201901266

[31] J. R. Mianroodi, S. Rezaei, N. H. Siboni, B. X. Xu, D. Raabe, Lossless multi-scale constitutive elastic relations with artificial intelligence, npj Computational Materials 8 (1) (2022) 1–12. arXiv:2108.02837, doi:10.1038/s41524-022-00753-3.

[32] E. B. Tadmor, R. E. Miller, R. S. Elliott, Continuum Mechanics and Thermodynamics, 2012.

[33] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, P. Riley, Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds (feb 2018). arXiv:1802.08219.
URL http://arxiv.org/abs/1802.08219

[34] K. T. Schütt, O. T. Unke, M. Gastegger, S. Schütt, O. T. Unke, M. Gastegger, Equivariant Message Passing for the Prediction of Tensorial Properties and Molecular Spectra, in: Proceedings of Machine Learning Research, Vol. 139, PMLR, 2021, pp. 9377–9388. arXiv:2102.03150.
URL https://proceedings.mlr.press/v139/schutt21a.html

[35] B. Jing, S. Eismann, P. Suriana, R. J. Townshend, R. O. Dror, LEARNING FROM PROTEIN STRUCTURE WITH GEOMETRIC VECTOR PERCEPTRONS, in: ICLR 2021 - 9th International Conference on Learning Representations, International Conference on Learning Representations, ICLR, 2021. arXiv:2009.01411.
URL https://arxiv.org/abs/2009.01411v3

[36] J. Gasteiger, F. Becker, S. Günnemann, GemNet: Universal Directional Graph Neural Networks for Molecules, in: Advances in Neural Information Processing Systems, Vol. 9, 2021, pp. 6790–6802. arXiv:2106.08903.
URL https://www.daml.in.tum.de/gemnet

[37] Y.-L. Liao, A. J. Hoffman, S. C. Shen, A. Duval, S. W. Norwood, T. Smidt, Equiformerv3: Scaling efficient, expressive, and general se (3)-equivariant graph attention transformers, arXiv preprint arXiv:2604.09130 (2026).

[38] Z. P. Bažant, L. Cedolin, Stability of structures: elastic, inelastic, fracture, and damage theories, Courier Corporation, 2003.

[39] R. Azulay, C. Combescure, Predicting the post-bifurcated patterns of architectured materials using group-theoretic tools, Journal of the Mechanics and Physics of Solids 187 (2024) 105631.

[40] X. Zhang, L. Wang, J. Helwig, Y. Luo, C. Fu, Y. Xie, M. Liu, Y. Lin, Z. Xu, K. Yan, et al., Artificial intelligence for science in quantum, atomistic, and continuum systems, Foundations and Trends® in Machine Learning 18 (4) (2025) 385–912.

[41] F. Hendriks, V. Menkovski, M. Doškář, M. G. Geers, O. Rokoš, Similarity equivariant graph neural networks for homogenization of metamaterials, Computer Methods in Applied Mechanics and Engineering 439 (2025) 117867.

[42] D. Ruelle, F. Takens, On the nature of turbulence, Les rencontres physiciens-mathématiciens de Strasbourg-RCP25 12 (1971) 1–44.

[43] W. Steffen, J. Rockström, K. Richardson, T. M. Lenton, C. Folke, D. Liverman, C. P. Summerhayes, A. D. Barnosky, S. E. Cornell, M. Crucifix, et al., Trajectories of the earth system in the anthropocene, Proceedings of the national academy of sciences 115 (33) (2018) 8252–8259.

[44] M. Sadria, T. M. Bury, Fatenet: an integration of dynamical systems and deep learning for cell fate prediction, Bioinformatics 40 (9) (2024) btae525.

[45] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, M. Le, Flow matching for generative modeling, arXiv preprint arXiv:2210.02747 (2022).

[46] M. S. Albergo, E. Vanden-Eijnden, Building normalizing flows with stochastic interpolants, arXiv preprint arXiv:2209.15571 (2022).

[47] J.-H. Bastek, D. M. Kochmann, Inverse design of nonlinear mechanical metamaterials via video denoising diffusion models, Nature Machine Intelligence 5 (12) (2023) 1466–1475.

[48] H. T. Kollmann, D. W. Abueidda, S. Koric, E. Guleryuz, N. A. Sobh, Deep learning for topology optimization of 2D metamaterials, Materials and Design 196 (2020) 109098. doi:10.1016/j.matdes.2020.109098.
URL https://doi.org/10.1016/j.matdes.2020.109098

[49] D. Schattschneider, The plane symmetry groups: their recognition and notation, The American Mathematical Monthly 85 (6) (1978) 439–450.

[50] J. Shim, S. Shan, A. Košmrlj, S. H. Kang, E. R. Chen, J. C. Weaver, K. Bertoldi, Harnessing instabilities for design of soft reconfigurable auxetic/chiral materials, Soft Matter 9 (34) (2013) 8198–8202.

[51] S. Jin, Y. P. Korkolis, Y. Li, Shear resistance of an auxetic chiral mechanical metamaterial, International Journal of Solids and Structures 174 (2019) 28–37.

[52] V. G. Satorras, E. Hoogeboom, M. Welling, E (n) equivariant graph neural networks, in: International conference on machine learning, PMLR, 2021, pp. 9323–9332.

1

CHAPTER 2

Similarity Equivariant Graph Neural Networks for Homogenization of Metamaterials

Reproduced from Hendriks, F., Menkovski, V., Doškář, M., Geers, M. G., & Rokoš, O. (2025). Similarity equivariant graph neural networks for homogenization of metamaterials. Computer Methods in Applied Mechanics and Engineering, 439, 117867. doi:10.1016/j.cma.2025.117867

Soft, porous mechanical metamaterials exhibit pattern transformations that may have important applications in soft robotics, sound reduction and biomedicine. To design these innovative materials, it is important to be able to simulate them accurately and quickly, in order to tune their mechanical properties. Since conventional simulations using the finite element method entail a high computational cost, in this article we aim to develop a machine learning-based approach that scales favorably to serve as a surrogate model. To ensure that the model is also able to handle various microstructures, including those not encountered during training, we include the microstructure as part of the network input. Therefore, we introduce a graph neural network that predicts global quantities (energy, stress, stiffness) as well as the pattern transformations that occur (the kinematics) in hyperelastic, two-dimensional, microporous materials. Predicting these pattern transformations means predicting the displacement field. To make our model as accurate and data-efficient as possible, various symmetries are incorporated into the model. The starting point is an $E(n)$-equivariant graph neural network (which respects translation, rotation and reflection) that has periodic boundary conditions (i.e., it is in-/equivariant with respect to the choice of RVE), is scale in-/equivariant, can simulate large deformations, and can predict scalars, vectors as well as second and fourth order tensors (specifically energy, stress and stiffness). The incorporation of scale equivariance makes the model equivariant with respect to the similarities group, of which the Euclidean group $E(n)$ is a subgroup. We show that this network is more accurate and data-efficient than graph neural networks with fewer symmetries. To create an efficient graph representation of the finite element discretization, we use only the internal geometrical hole boundaries from the finite element mesh to achieve a better speed-up and scaling with the mesh size.

2.1 Introduction

Metamaterials are materials with a microstructure designed to exhibit special properties. For instance, mechanical metamaterials have unusual mechanical properties [1,2],

such as a negative Poisson's ratio [3] or negative compressibility [4]. Here, we are specifically interested in flexible, porous mechanical metamaterials that have a tunable stiffness as a result of a pattern transformation [5, 6] that can be activated, for example, by mechanical loading, pneumatics [7] or magnetic fields [8]. Such pattern transformations can also change the acoustic properties, allowing the design of mechanically tunable acoustic metamaterials [9–12], or they can be used to control the shape of the material, with applications in soft robotics [7, 13, 14]. They also have applications in biomedicine [15].

These kinds of pattern-transforming materials provide a large design space to be explored, because there are a lot of possibilities regarding geometry (shape and number of holes and inclusions), electromagnetic, chemical and mechanical properties of the base material, and different means of loading and activation. Moreover, thanks to the recent advancements in 3D printing [16–21], the broad design space is no longer inherently restricted by manufacturing concerns. Consequently, being able to design mechanical metamaterials for a specific target response (i.e., having a prescribed stiffness and buckling exactly when needed) is highly appealing. This can be achieved with shape [22, 23] and/or topology optimization [24, 25] on the Representative Volume Element (RVE) [26]. Topology optimization can be used to design a microstructure for various homogenized properties [27, 28], including the buckling strength of materials [29–31]. Because topology and/or shape optimization are almost always iterative, being able to rapidly evaluate the performance of new designs using numerical simulations is critical for accelerating the design process.

However, the behavior of the metamaterials of interest is complex, highly nonlinear, and involves large deformations, which requires repeated evaluation of the constitutive law of the matrix material. The buckling behavior that enables pattern transformations makes the system even more complicated to solve (here avoided by using a perturbation). Moreover, to capture all fine microstructural details, the discretization needs to have a sufficiently high resolution. For these reasons, the numerical simulations are typically expensive to evaluate, which makes them too inefficient to explore the large design space. Therefore, the main objective of this article is to build an efficient, accurate and fast-to-evaluate machine learning-based surrogate model for the modeling of periodic elastomeric mechanical metamaterials with a highly non-linear response.

Given the goal is to develop a model enabling the design of new mechanical metamaterials, the model must have the potential to generalize to unseen, arbitrary microstructures, even including topology changes. This implies that the surrogate model should be able to take a description of the full geometry as input. This rules out other approaches, such as analytical (closed-form) homogenization methods (e.g. mixture methods [32] or methods based on Eshelby's [33] analytical result for an ellipsoidal inclusion in infinite matrix [34, 35]), reduced order models [36–44], data-driven approaches [45], self-consistent clustering analysis [46], relatively simple machine learning approaches that map deformation gradient (and optionally some microstructural parameters) to global quantities [47–49], and graph networks (not to be confused with graph neural networks) [50], which model porous mechanical metamaterials as rigid crosses connected by neural network-modeled springs.

These methods are certainly valuable in their developed context, but only work

very well in the linear-elastic regime or only for a fixed geometry and topology. One reason neural networks are promising is the fact that they are universal approximators [51–53], which means they can approximate any arbitrary function and are therefore not constrained to any specific form of macroscopic constitutive law. Specifically, we are interested in deep learning solutions, since deep learning excels at using high-dimensional data [54], which allows us to use a full description of the geometry of the RVE as an input and which also scales favorably (often linearly) with the size of the input and the number of parameters. Especially in 3D, the scaling advantage could be considerable.

However, creating an efficient deep learning model is not trivial. Specifically, because the following properties need to be incorporated into the architecture of the model:

- The model should be able to handle large deformations (to cover buckling), high strains *and* rotations.
- If the input is a mesh, the model should be in-/equivariant under permutation of nodes (the order of nodes should not matter). The concepts of invariance and equivariance are defined and explained in detail in Section 2.3.
- Similarity in-/equivariance ($E(n)$ in-/equivariance, i.e., rotation, translation, reflection symmetry), to satisfy material objectivity. These transformations correspond to choosing a different coordinate system. This is also one of the requirements of the principle of material frame indifference, also called objectivity [55, p.195].[1]
- Scale in-/equivariance; since we limit ourselves to first-order computational homogenization of a hyperelastic material (Bertoldi-Boyce [5], Equation (2.10)), a change in scale of the RVE should not change its behavior.
- Periodicity, which implies in-/equivariance with respect to a shift of the RVE (i.e., RVE window translation) and with respect to merging multiple RVEs into a new, bigger one. This assumes the RVE is already big enough to capture all relevant buckling patterns. If not, a larger RVE could allow for a new buckling pattern, which breaks the in-/equivariance with respect to merging multiple RVEs.
- Prediction of scalars (energy), vectors (displacement), and higher-order tensors (stress and stiffness).

To the best of our knowledge, there are no existing machine-learning based models that satisfy all these constraints. See Table 2.1 for an overview of the constraints respected by different types of models.

The in-/equivariances we mention above are all related to geometric symmetries. However, there are other possible constraints that can be incorporated, such as (poly-)convexity as formulated by Ball [56], symmetry of the stress tensor and the condition that in the reference configuration, the stress tensor is zero. There has been a lot of work in this area already [49, 57–59], specifically concerning polyconvexity, usually implemented using approaches based on input-convex neural networks (ICNNs) by Amos et al [60], such as the approach by Klein et al. [61, 62]. These all concern

[1] Usually, when discussing material frame indifference, reflection symmetry is omitted. However, it still applies, because changing the orientation of the coordinate system does not change the behavior of the material, and that is why, for completeness, we include it here as well.

Table 2.1: Overview of GNN architectures and the constraints they address, along with the presently developed EGNN and similarity-equivariant GNN (SimEGNN).

Architecture	Updates node positions?	Invariances/equivariances					
		Translation	Rotation	Reflection	Scale	Periodicity	
						Shifted RVE	Extended RVE
CNN*	✗	✓	✗	✗	✓	✓	✓
GNN	✓	✓	✗	✗	✗	✓	✓
GNN**	✗	✓	✓	✓	✗	✓	✓
Original EGNN	✓	✓	✓	✓	✗	✗	✗
From the present study:							
GNN	✓	✓	✗	✗	✗	✓	✓
EGNN	✓	✓	✓	✓	✗	✓	✓
SimEGNN	✓	✓	✓	✓	✓	✓	✓

*periodicity and translation in-/equivariance are only exactly respected if there is no pooling or dense layers (which is possible but unusual), otherwise the CNN is only approximately translation in-/equivariant. Periodicity additionally needs circular padding.

**assuming only $E(n)$-invariant attributes such as distances and angles are used.

constitutive relations where the deformation gradient $\mathbf{F}$ or right Cauchy-Green deformation tensor $\mathbf{C} = \mathbf{F}^T \cdot \mathbf{F}$ is mapped to stress or stiffness. These models do not take the full geometry as input, but could possibly be incorporated into a model that does. However, in our case, our potential energy density is not convex, because of the presence of instabilities due to buckling.

The convolutional layers of convolutional neural networks (CNNs) respect translation in-/equivariance and can respect periodicity when used with circular/periodic padding. CNNs applied to 2D or 3D images of microstructures [63–69] therefore approximately respect these symmetries. [2] They also respect scale in-/equivariance, because the input picture does not change in that case. They can be made to respect rotation and reflection in-/equivariance [70], but not for any arbitrary rotation or reflection; only with respect to rotations that are multiples of 90° and reflections that are horizontal or vertical. In addition, they can only predict vectors and higher-order tensors by treating them as an arbitrary list of numbers (that therefore do not necessarily transform as tensors), they cannot track the movement of nodes, and their geometry input is constrained to a square grid.

A promising alternative is a graph neural network (GNN): a type of neural network that takes a graph as input. The mesh representation used for the finite element simulations is well-suited for a graph representation. GNNs also respect permutation in-/equivariance, which is advantageous, because any description of the geometry of

[2] CNN with circular padding will not be exactly in-/equivariant with respect to shifting the RVE, because of the max pooling and the flattening that usually happens at the end of the convolutional layers, which is fed into the dense layers afterwards.

the material microstructure has to respect permutation in-/equivariance with respect to the geometrical features (i.e., the order in which the nodes, elements or entire holes are described should be irrelevant). GNNs have been used before [71–74] for the simulation of materials. However, these GNNs (i) only predict scalars (e.g. energy, mean pressure, deviatoric stress, fracture probability, damage), (ii) operate typically in the small-strain regime, and (iii) are neither $E(n)$-in-/equivariant nor (iv) scale in-/equivariant. This makes them unsuitable for the goal of simulating the specific metamaterials of interest here. MeshGraphNets [74] using translation equivariant GNNs to recreate results from physical simulations *do* allow for large strains, as demonstrated by a simulation of a body made of a hyperelastic material. However, these models are neither $E(n)$-equivariant nor scale in-/equivariant.

$E(n)$-in-/equivariance in GNNs can be achieved simply by using only $E(n)$-invariant features such as distances or angles, as many GNNs for molecular property prediction do [75–80]. In addition, periodicity can be incorporated by merging nodes from one side of the RVE with their corresponding side (left to right, top to bottom), or equivalently, have 'ghost nodes' [81]. Unfortunately, these molecular GNNs tend to be suitable only for graphs with a small graph diameter, whereas a finite element (FE) mesh is usually much larger with a large graph diameter, requiring a high number of message passing steps, which are the iterations used by a GNN to update the graph, see Section 2.4. Moreover, using these $E(n)$-invariant features precludes any dependence on a global coordinate system. Consequently, the node positions cannot be updated directly.

There exist other ways to update node positions in a global coordinate system such that the updates are independent of the global coordinate system, using, e.g., expansions in spherical harmonics. Relying on spherical harmonics allows for updating positions independently of a global coordinate system, albeit at increased computational cost [82, 83]. Alternatively, GVP-GNNs (Geometric Vector Perceptron Graph Neural Networks) [84], PaiNN (polarizable atom interaction neural network) [85] and GemNet (geometric message passing neural network) [86] can predict vector quantities as well, without the use of spherical harmonics, which makes them also suitable to predict deformation. Here, we choose another approach named $E(n)$-equivariant graph neural networks (EGNNs) [87], due to their simplicity, which updates node positions by 'pulling' or 'pushing' them along their edges.

EGNNs currently do not respect periodicity, because the distances are recomputed from the new coordinates of the nodes after each message passing step. This approach does not work for edges that wrap around from one side of the RVE to another (i.e., for periodic edges). We show that it is relatively easy to adjust an EGNN such that it is periodic (referred to as 'EGNN' in this paper). EGNNs are also not scale invariant, because there is a dependence on the distances between nodes. We show this can also be fixed (SimEGNN).

The paper is structured as follows: Section 2.2 covers the essentials of first-order computational homogenization and describes the generation of the training dataset. Section 2.3 describes the concept of in-/equivariance, and how it applies here. Section 2.4 details the developed GNN architecture. Numerical results and discussion are provided in Section 2.5, while Section 2.6 concludes this paper. All the code can be found at: https://github.com/FHendriks11/SimEGNN, and all the data is available

in a Zenodo repository here: https://zenodo.org/records/14229619.

Throughout the paper we use the following notation: scalars (except for the strain energy density $\mathfrak{W}$) are in italics (e.g. r_{ij}), abstract 'vectors' (neural network-generated lists of numbers without a physical meaning), which can have any dimension, are in bold (e.g. $\mathbf{h}_i$), physical/geometric vectors, which are always 2D or 3D, are in italics with an arrow (e.g. $\vec{x}_i$), geometric tensors are in bold capitals (e.g. $\mathbf{P}$), of which the 4^{th}-order tensors have a superscript 4 in front (e.g. ${}^4\mathbf{D}$). We use the following definitions of the tensor product (also called open product or dyadic product)

$$\mathbf{A} = \vec{a} \otimes \vec{b} \quad \Longleftrightarrow \quad A_{ij} = a_i b_j \tag{2.1}$$

$${}^4\mathbf{C} = \mathbf{A} \otimes \mathbf{B} \quad \Longleftrightarrow \quad C_{ijkl} = A_{ij} B_{kl} \tag{2.2}$$

and the dot product

$$\vec{b} = \vec{a} \cdot \mathbf{A} \quad \Longleftrightarrow \quad b_j = \sum_i a_i B_{ij} \tag{2.3}$$

$$\vec{b} = \mathbf{A} \cdot \vec{a} \quad \Longleftrightarrow \quad b_i = \sum_j A_{ij} a_j \tag{2.4}$$

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{B} \quad \Longleftrightarrow \quad C_{ik} = \sum_j A_{ij} B_{jk}, \tag{2.5}$$

as well as the double dot product

$$\mathbf{C} = {}^4\mathbf{A} : \mathbf{B} \quad \Longleftrightarrow \quad C_{ij} = \sum_k \sum_l A_{ijkl} B_{lk}. \tag{2.6}$$

The divergence of a second order tensor field with respect to the reference configuration is defined as

$$\vec{a} = \vec{\nabla}_0 \cdot \mathbf{A} \quad \Longleftrightarrow \quad a_j = \sum_i \frac{\partial A_{ij}}{\partial x_i^{\text{ref}}}. \tag{2.7}$$

Note that for the definitions in the equations above, the indices correspond to the different components of tensors or vectors. In the rest of the paper, however, the subscripts indicate how many nodes are relevant to that quantity, e.g., $\mathbf{h}_i$ is an abstract vector specific to one node i, r_{ij} is a scalar quantity specific to an edge ij between node i and j.

2.2 Dataset Generation

To create a training dataset, finite element simulations are exploited, in combination with the computational homogenization scheme to compute representative volume element (RVE) responses and their effective properties. Specifically, we construct a dataset of the following mapping from the microscopic RVE (geometry and material) and the macroscopic deformation gradient tensor $\mathbf{F}$ to the microscopic deformation

and the macroscopic target variables

$$\left.\begin{matrix}\text{RVE geometry \& material}\\ \mathbf{F}\end{matrix}\right\} \quad \rightarrow \quad \begin{cases}\vec{x}\\ \mathfrak{W}\\ \mathbf{P}\\ {}^4\mathbf{D}\end{cases}, \tag{2.8}$$

where $\mathfrak{W}$ is the effective (macroscopic) strain energy density, $\mathbf{P}$ is the effective first Piola-Kirchhoff stress tensor and ${}^4\mathbf{D}$ is the effective stiffness tensor and $\vec{x}$ is the deformed configuration, meaning the final position of each node. This stiffness is defined through the incremental stiffness relationship

$$\delta\mathbf{P} = {}^4\mathbf{D} : \delta\mathbf{F}. \tag{2.9}$$

Even though $\vec{x}$ is not strictly necessary, we want to obtain it as well, to make the eventual machine learning model more interpretable, and to offer the possibility of observing the various pattern transformations directly. This dataset will later be used to train a machine learning model.

2.2.1 Computational Homogenization

We focus on calculations pertinent to computational homogenization, which is a workhorse in shape or topology optimization methods of materials. It replaces an explicit constitutive law at the macroscale with a boundary value problem posed on an RVE at the microscale, loaded by a deformation gradient $\mathbf{F}$, which is extracted from the macroscale (for an overview of the basics and extensions of computational homogenization, see [88]). The macroscale (global scale) is the scale over which the mechanical loading of the entire system varies, whereas the microscale is the scale of the heterogeneities in the microstructure of the underlying material. In this section, the microscopic quantities will be denoted by the subscript m. See Figure 2.1 for a schematic representation of the computational homogenization procedure.

We focus here on the microscale computation, which is a finite element (FE) simulation of an RVE. Since the microstructural features in our materials are quite fine, the micro-scale FE mesh must be adequately small as well, which makes the micro-scale simulations expensive.

Even though the nature of architectured materials often calls for a generalized continuum at a macroscale (e.g., second-order [89, 90] or micromorphic [91–93]), we limit ourselves here to the standard first-order computational homogenization which leads to a classical Cauchy continuum at the macroscale, assuming separation of scales.

2.2.2 Microstructural RVE problem

The underlying pattern-transforming 2D metamaterial consists of a flexible, hyperelastic matrix with circular (or, in our case, almost circular) holes arranged in a square grid as shown in Figure 2.2a [91, 94]. This material exhibits auxetic behavior after

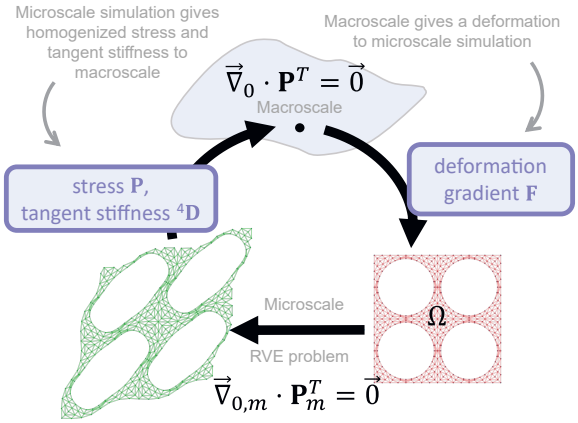
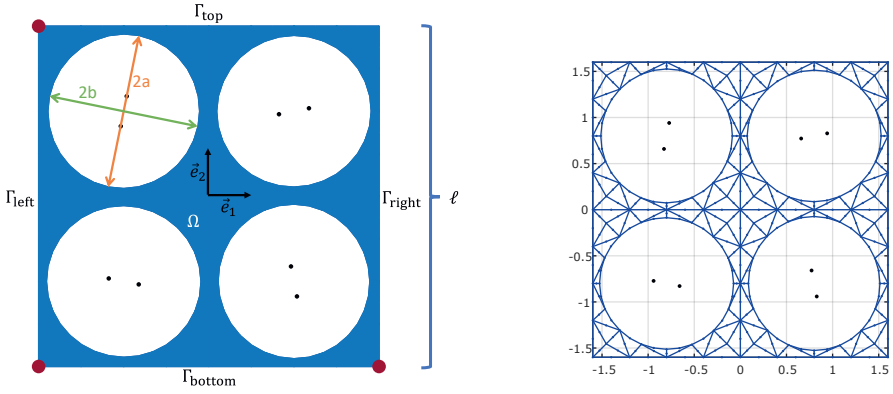


Figure 2.1: Schematic overview of the first-order computational homogenization procedure.



(a) (b)

Figure 2.2: (a) The RVE geometry, with four holes, slightly flattened by 1% into ellipses, with $2a$ the major axis and $2b$ the minor axis. The periodic boundaries are indicated by Γ, the control points by red dots, and the foci of the ellipses by black dots. The length and width of the RVE is ℓ and the domain is Ω. The directions of the basis vectors are indicated by $\vec{e}_1$ and $\vec{e}_2$. (b) The finite element discretization of this geometry.

buckling. We assume a hyperelastic, finite-strain model at the microscale. One RVE has four holes, which has been confirmed to be of a sufficient size to capture the most important pattern transformation when buckling under prescribed macroscopic strain (e.g., [5]), although it should be noted that it is possible for a 2×2 RVE to be inadequate in other cases. For example, Polukhov et al. [95] show that in an electrostatically activated version of the same material sometimes a 2×3 RVE is necessary. Ideally, Bloch analysis should be used to determine the appropriate RVE size [5, 96–98], although this is computationally expensive.

Each hole has a diameter 0.45 times the side length ℓ of the RVE. In the extended dataset, the diameter is also varied; all holes have the same diameter, which can be 0.4ℓ, 0.425ℓ, 0.45ℓ, or 0.475ℓ. This results in ligament thicknesses of 0.1ℓ, 0.075ℓ, 0.05ℓ, and 0.025ℓ, respectively.

The above-described metamaterial constitutes a challenging test case, because of the multiple buckling modes it shows, and the strong dependence of effective properties on the deformation state. Depending on the loading conditions, i.e., prescribed macroscopic $\mathbf{F}$, a bifurcation breaking the RVE's symmetry can occur. There are two bifurcation modes: the microstructure can either (i) buckle clockwise or counterclockwise – referred to in this paper as 'rotational bifurcation' –, or (ii) bulge out left or right (or up or down if $F_{11} < F_{22}$) – 'left/right bifurcation',[3] or even both situations can occur one after another when increasing the loading, see also Figure 2.3. Buckling significantly alters the homogenized stress and stiffness; Figure 2.4 shows how the normal stress P_{11} and stiffness component D_{1111} change under biaxial compression.

We use the Bertoldi-Boyce constitutive law for the polymer, given by the strain energy density function [5]

$$\mathfrak{W}_m(\mathbf{F}_m) = c_1 \left(I_1 - 2\right) + c_2 \left(I_1 - 2\right)^2 - 2c_1 \ln\left(J_m\right) + \frac{K}{2} \left(J_m - 1\right)^2, \tag{2.10}$$

where $J_m = \det \mathbf{F}_m$, $I_1 = \operatorname{tr} \mathbf{C}_m$, $\mathbf{C}_m$ is the microscopic right Cauchy-Green tensor $\mathbf{C}_m = \mathbf{F}_m^T \cdot \mathbf{F}_m$, and c_1, c_2 and K are bulk material parameters, which take the values $c_1 = 0.55\,\mathrm{MPa}$, $c_2 = 0.3\,\mathrm{MPa}$, $K = 55\,\mathrm{MPa}$.[4]

The mechanical behavior is modeled using a standard Cauchy continuum. Assuming no body forces and neglecting inertia, this amounts to solving

$$\vec{\nabla}_{0,m} \cdot \mathbf{P}_m^T(\mathbf{F}_m(\vec{u})) = \vec{0} \quad \forall \vec{x}^{\,\mathrm{ref}} \in \Omega \tag{2.11}$$

for the microscopic displacement field $\vec{u}$, where $\vec{\nabla}_{0,m}$ is the gradient operator with respect to the *initial* microstructural position $\vec{x}^{\,\mathrm{ref}}$ on the domain Ω. This equation and its solutions respect the in- and equivariances briefly mentioned in the Introduction and further discussed in Section 2.3. The new position $\vec{x}$ is decomposed as

$$\vec{x} = \vec{x}^{\,\mathrm{ref}} + \vec{u}(\vec{x}^{\,\mathrm{ref}}) = \mathbf{F} \cdot \vec{x}^{\,\mathrm{ref}} + \vec{w}(\vec{x}^{\,\mathrm{ref}}), \tag{2.12}$$

[3] This left/right buckling mode would actually correspond to a global buckling mode, and the RVE with 2×2 holes we use therefore does not capture it accurately. However, because here the system only serves as a test case for the GNN, we ignore this fact for now.

[4] In 3D, the first two terms would have $I_1 - 3$ instead of $I_1 - 2$.

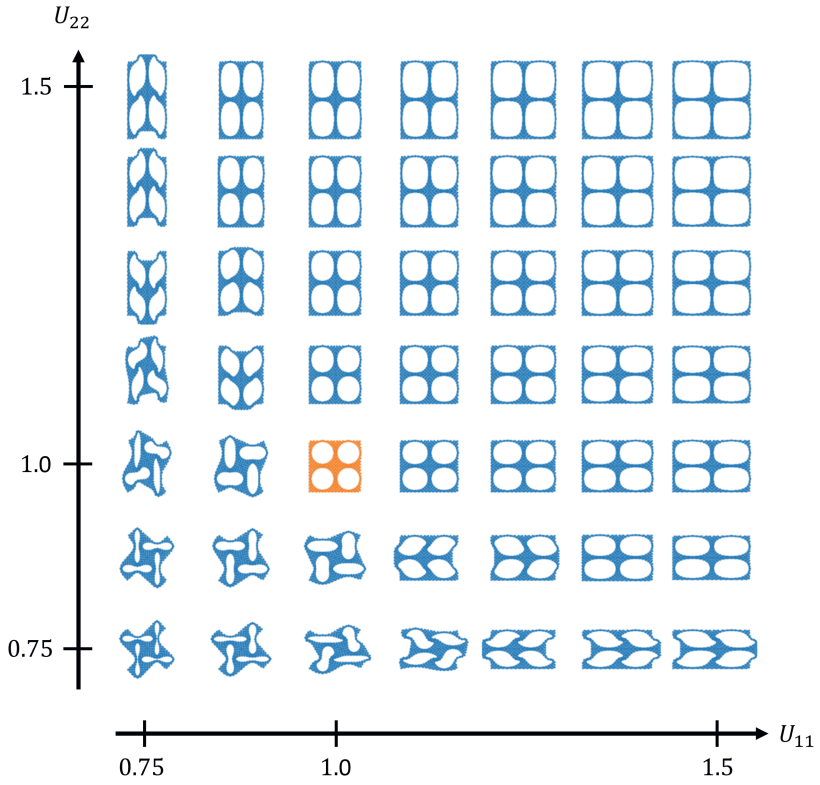
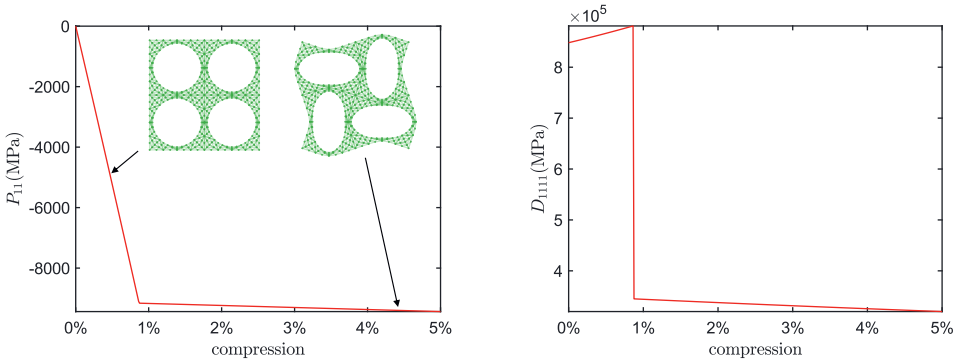


Figure 2.3: The behavior of the metamaterial RVE under various loading conditions. In this plot, U_{ij} are components of the right stretch tensor, see Section 2.2.3. The components U_{11} and U_{22} are varied, while U_{12} is constrained at zero. The reference configuration (i.e., the unloaded state) is plotted in orange.



(a) (b)

Figure 2.4: Change in (a) stress component P_{11} and (b) stiffness component D_{1111} as a function of the biaxial compression.

where $\vec{u}(\vec{x})$ is the displacement field, $\mathbf{F} \cdot \vec{x}^{\,\mathrm{ref}}$ is the affine part dictated by the prescribed macroscopic deformation gradient $\mathbf{F}$ and $\vec{w}$ denotes the fluctuation part of the solution.

From this solution, the homogenized quantities can be obtained as follows:

$$\mathfrak{W} = \frac{1}{\Omega} \int_{\Omega} \mathfrak{W}_m \left(\mathbf{F}_m \left(\vec{u} \right) \right) \mathrm{d}\vec{x}, \tag{2.13}$$

$$\mathbf{P} = \frac{\partial \mathfrak{W}}{\partial \mathbf{F}} = \frac{1}{\Omega} \int_{\Omega} \mathbf{P}_m \left(\mathbf{F}_m \left(\vec{u} \right) \right) \mathrm{d}\vec{x}, \tag{2.14}$$

$$^4\mathbf{D} = \frac{\partial^2 \mathfrak{W}}{\partial \mathbf{F} \partial \mathbf{F}} = \frac{\partial \mathbf{P}}{\partial \mathbf{F}}. \tag{2.15}$$

We do not provide the full definition for $^4\mathbf{D}$, but it can be found in [55, p. 227] and the method of computing it is adopted from [99, §2.4.4], see also [100].

The periodic boundary conditions enforce the top and bottom boundaries of the RVE to deform in the same way, and likewise for the left and right boundaries. This implies that the microfluctuation field $\vec{w}$ (i.e., the displacement field $\vec{u}$ minus its affine part) is equal for corresponding points on these boundaries [101], see Figure 2.2a:

$$\vec{w} \left(\Gamma_{\mathrm{left}} \right) = \vec{w} \left(\Gamma_{\mathrm{right}} \right), \tag{2.16}$$

$$\vec{w} \left(\Gamma_{\mathrm{top}} \right) = \vec{w} \left(\Gamma_{\mathrm{bottom}} \right). \tag{2.17}$$

The macroscale deformation gradient $\mathbf{F}$ is imposed on this RVE by applying it as an affine transformation through the three corner control nodes (the fourth one is dependent on the others due to periodicity) in multiple adaptive load increments.

The RVE's geometry shown in Figure 2.2a is discretized with quadratic triangular elements, using Gmsh (https://gmsh.info/) [102] to create the mesh shown in Figure 2.2b. This mesh is fairly coarse, which means the results are not as accurate as desired for FE^2, but good enough to capture the overall behavior and to quickly gener-

ate a large dataset to test the proposed method. The FE simulations were performed using an in-house Matlab code [92].

Because the neural network cannot handle the ambiguity in buckling (i.e., left vs right, clockwise vs counterclockwise), we ensure that the material always buckles in the same direction, by steering the bifurcation through a minor perturbation of the holes' initial geometry. To this end, the circles are flattened into ellipses by 1% (see Figure 2.2b for the resulting finite element mesh, where the foci of the four ellipses are depicted as black dots). Moreover, the ellipses are tilted such that both lateral and rotational bifurcations are triggered consistently. A stability analysis was performed to verify that this adequately controls the bifurcation ambiguity. Of course, this perturbation is only possible because we already know what modes may emerge. In a real-world application, where one would like to predict the behavior of unseen geometries, this would not be possible and other approaches must be considered.

2.2.3 Sampling and Data Augmentation

For the purpose of training GNNs, a training dataset was generated by uniformly sampling the RVE's response to a prescribed macroscopic deformation gradient $\mathbf{F}$. Using the polar decomposition, $\mathbf{F}$ can be decomposed as $\mathbf{F} = \mathbf{R}\cdot\mathbf{U}$ into the symmetric macroscopic right-stretch tensor $\mathbf{U}$ and a rotation tensor $\mathbf{R}$. Because $\mathbf{R}$ encodes only a rigid-body rotation, which does not meaningfully affect the results, we only sampled $\mathbf{F} = \mathbf{U}$. This simplification left us with only three components U_{11}, U_{12} and U_{22}.

The sampling was carried out according to Figure 2.5, where a visual representation of the sampling is presented. Sampling was not uniform, because each simulation starts from $\mathbf{U} = \mathbf{I}$, and all intermediate configurations on the loading path to the prescribed final $\mathbf{U}$ were also saved and used for training, which is why when sampling the final $\mathbf{U} = \mathbf{U}^{\text{fin}}$, always at least one component was at a minimum or maximum value of the chosen range. For the intermediate values of $\mathbf{U}$, the loading was by default increased in 20 uniform steps from $\mathbf{I}$ to $\mathbf{U}^{\text{fin}}$, but because of the adaptive time stepping, this sometimes differed.

To sample different values of $\mathbf{U}^{\text{fin}}$, in turn one of the three components was kept at its extreme value while the other two were uniformly sampled from a specified range. These ranges were $[0.75, 1.5]$ in steps of 0.05 for U_{11}^{fin} and U_{22}^{fin}, and $[0,\ 0.5]$ in steps of 0.05 for $U_{12}^{\text{fin}} = U_{21}^{\text{fin}}$. This is a wide enough range to capture the different types of behavior. Because of the symmetry of the RVE (disregarding the small perturbation), we only consider $\mathbf{U}^{\text{fin}}$ where $U_{11}^{\text{fin}} > U_{22}^{\text{fin}}$, and $U_{12}^{\text{fin}} \geq 0$.

Whenever a computed response featured overlapping elements, indicating the presence of contact (not considered in the numerical model), the corresponding $\mathbf{U}$ was removed from the dataset. This resulted in a dataset of 446 trajectories, which in total contain 8451 distinct loadcases. As indicated in the data statement at the end of the paper, all training data is included in a Zenodo repository accompanying this paper.

In the results Section 2.5, the newly developed machine learning model that respects all symmetries is compared to two models that have fewer symmetries incorporated. To train models with fewer symmetries to a higher degree of accuracy, a method called data augmentation can be used. Here, one trains the model on trans-

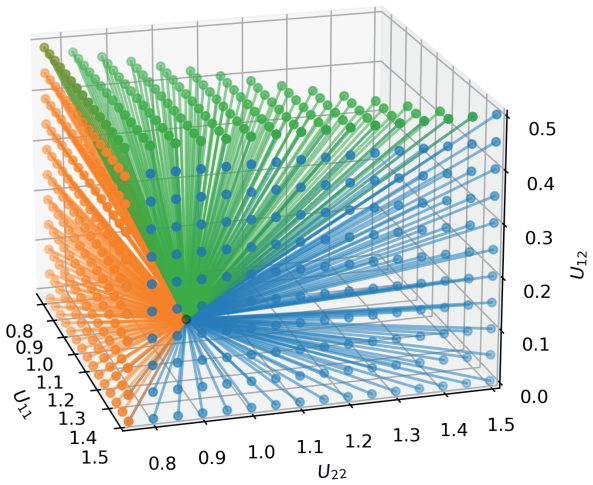


Figure 2.5: The sampled loading paths in U_{11}, U_{22} and U_{12} space. The lines indicate the loading paths, all starting from the black dot which indicates $\mathbf{U} = \mathbf{I}$. The colored dots represent the sampled values of final $\mathbf{U}^{\text{fin}}$, with the blue dots indicating cases with U_{11}^{fin} constant, orange dots indicating U_{22}^{fin} constant and green dots indicating U_{12}^{fin} constant.

formed versions of the data to promote learning of these symmetries (e.g., teaching the model rotation by including rotated versions of each data point). Therefore, to make the comparison more fair, during training, we create rotated, reflected and scaled versions of the data points. To this end, we transformed each training batch, using random uniform sampling for the rotation and scaling, and random booleans for the x-reflection and y-reflection. The bounds on rotation were set between 0 and 2π radians and the scaling factor was chosen between 0.1 and 10 (sampling as 10^x, with linear sampling of $x \in [-1, 1]$).

2.3 In- and Equivariance

Inductive biases are a set of assumptions about the data, built into machine learning models, that do not need to be learned from data. Incorporating domain knowledge into a model can improve the model's efficiency with respect to the amount of training data needed and can improve its generalization capabilities. The symmetries (invariances and equivariances) mentioned in the introduction are examples of such inductive biases [103].

Invariance with respect to a certain transformation T means that when this transformation is applied to the input $\mathbf{x}$, the output $\mathbf{y}$ of the model ϕ stays the same

$$\phi(\mathbf{x}) = \phi(T\mathbf{x}). \tag{2.18}$$

Equivariance with respect to a transformation means that there is a corresponding transformation T' (often $T = T'$) that can be performed on the model's output to get the same result

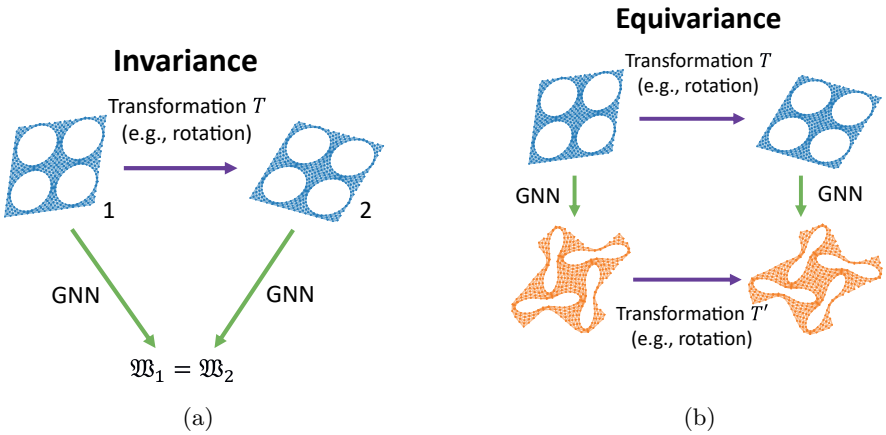
$$T'\phi(\mathbf{x}) = \phi(T\mathbf{x}). \tag{2.19}$$

For example, if the input configuration is rotated (because the coordinate system is chosen differently, or because $\mathbf{F}$ contains a rotational part), then the energy density $\mathfrak{W}$ stays the same (invariance), while the output configuration $\vec{x}$ is also rotated (equivariance), and the homogenized stress and stiffness tensors are transformed as tensors under a rotation of the basis vectors (also equivariance). See Figure 2.6 for a visual explanation of these concepts. So, rather than giving many examples of such transformations in the training data and letting the model learn these invariances and equivariances, we aim to incorporate these symmetries directly into the model architecture and guarantee the model's performance under such transformations.

As mentioned in the introduction, in addition to the node label permutation in-/equivariance that GNNs already respect, we consider similarity in-/equivariance (translation, rotation, reflection, scaling) and RVE in-/equivariance (periodicity). These in-/equivariances result directly from the governing balance equation of the system, recall Equation (2.11).

Figure 2.7 illustrates these equivariances for the deformation, and Table 2.2 indicates which of the two (invariance or equivariance) applies for each type for each target variable. These are the node positions $\vec{x}$, the macroscopic strain energy density $\mathfrak{W}$, the first Piola-Kirchhoff stress tensor $\mathbf{P}$ and the stiffness ${}^4\mathbf{D}$.

Incorporating more in-/equivariances makes it easier for the model to construct and use features in a general way [103]. Otherwise, it would have to learn them



(a) (b)

Figure 2.6: The concepts of (a) invariance and (b) equivariance illustrated, using a GNN predicting the strain energy density and deformation as examples, respectively.

Table 2.2: Overview of the invariance (I) or equivariance (E) of target variables with respect to different types of transformations

Transformation	$\vec{x}$	$\mathfrak{W}$	$\mathbf{P}$	$^4\mathbf{D}$
Translation	E	I	I	I
Rotation	E	I	E	E
Reflection	E	I	E	E
Scaling	E	I	I	I
Shifting RVE	E	I	I	I
Extending RVE	E	I	I	I

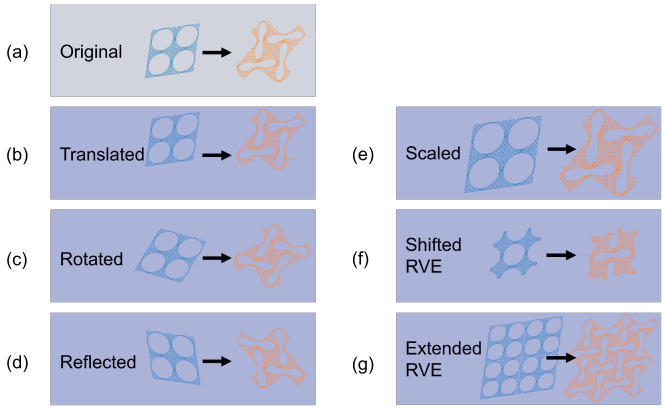


Figure 2.7: Different types of equivariance that should be respected by the model. Each case shows that the output configuration (orange) is transformed similarly to the input (blue), which is the meaning of equivariance.

from the data, which can be encouraged by using data augmentation (see section 2.2.3). However, in general, obtaining the same performance from a model with fewer in-/equivariances will require more data, longer training and/or more parameters.

2.4 GNN Approach

To achieve the goals set out in the introduction, we have developed a graph neural network (GNN)-based approach as detailed in this section. The final model – the SimEGNN – respects all the in-/equivariances discussed in Table 2.3. Respecting these symmetries is important for data and parameter efficiency and generalizability [103].

We first discuss the data structure encoding, i.e., graph encoding, and then present and compare three GNN architectures that were created specifically for metamaterial simulations. Each successive network has more symmetries built in, to quantify the added value of incorporating these symmetries. The first model, discussed in Subsection 2.4.3, is a base GNN model that can work with the data representation, and respects the periodicity (i.e., the 'Shifted RVE' and 'Extended RVE' in-/equivariance in Table 2.2), which is explained in Subsection 2.4.2 as well as translation in-/equivariance. The second architecture, discussed in Subsection 2.4.4, extends this base GNN to also incorporate $E(n)$-in-/equivariance. Finally, the third architecture, discussed in Subsection 2.4.5, is our newly proposed SimEGNN, which also adds scale in-/equivariance. This means it respects all in-/equivariances listed in Table 2.2.

Details on the implementation, including the training process, can be found in 2.B.

2.4.1 Graph Encoding

A graph consists of a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$ connecting pairs of the nodes. When creating a graph that represents the microstructure, we desire the following properties of the graph representation:

- connected, since otherwise the message-passing steps cannot propagate information throughout the entire graph;
- small graph diameter (the maximum degree of separation of any two nodes), to reduce the number of message-passing steps needed;
- a sufficiently accurate representation of the physical structure of the material;
- easy to automate;
- not too large number of nodes and edges.

The element edges and nodes of the FE mesh (the one used for data generation, shown in Figure 2.2b) are an obvious option for the input graph. This graph definitely offers an accurate representation of the physical structure of the material, however, it has a large number of nodes and edges and a large graph diameter. We therefore use it as our starting point, but we only keep the nodes and edges at the boundary of the holes (i.e., we remove all the bulk nodes). Because the deformation of the boundary nodes of the mesh is enough to see how the material deforms, and the rest of the quantities of interest are global ($\mathfrak{W}$, $\mathbf{P}$, ${}^4\mathbf{D}$), we do not require any local fields, which makes the bulk nodes unnecessary in the first place. Although it is possible to use the full FE mesh, our numerical experiments showed that it is a slower and less accurate option, presumably due to the higher number of message passing steps needed, leading to problems such as oversmoothing [104].[5]

When constructing the new graph, we make no distinction between nodes at the corner of the quadratic triangular elements and the mid-edge nodes. The elimination of the bulk nodes leaves us with four disconnected components, each corresponding to one of the four holes and consisting of the hole boundary nodes and edges. To ensure connectivity, we join these components by connecting each node to the closest node on each of the other three hole boundaries (i.e., adding three new edges for each node, and then deduplicating the set of edges). This results in at most $4N$ undirected edges, with N the number of remaining hole boundary nodes, but in practice less due to the deduplication. In the final input graph, each undirected edge is represented by two directed edges. Due to the RVE's periodicity, an edge may start on one side of the RVE and 'wrap around' to the other side of the RVE (i.e., a Pac-Man World geometry). The resulting graph, derived from the FE mesh shown in Figure 2.2b, is shown in Figure 2.8. The original mesh contained 413 nodes and 1056 edges, while the generated input graph has 128 nodes and 480 edges. This fairly simple approach to constructing the graph representation is easily automated and already works adequately, even though it is possible to further optimize the graph representation, especially when the number of nodes and edges is very large, or when the number of holes in the RVE becomes larger. In the latter case, it might be better not to connect each node to every other hole, but to only a subset of the holes. Other options could be a small-world graph or a hierarchical graph for more efficient

[5] When fully detailed local fields *are* of interest, it may be possible to limit the number of necessary message passing steps by using graph pooling strategies.

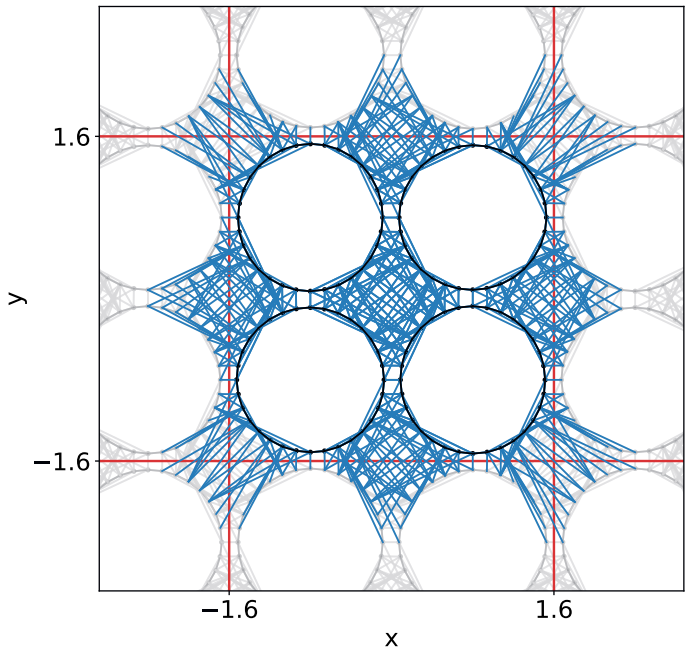


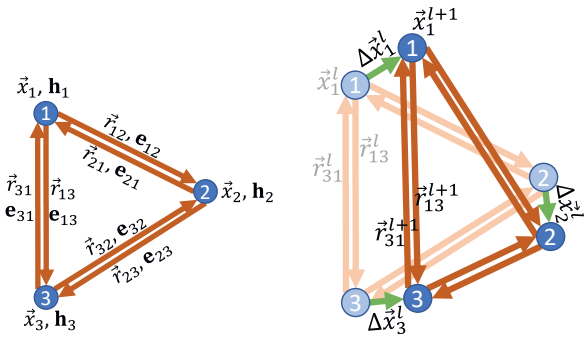
Figure 2.8: The input graph used for all GNNs considered in this paper, shown in blue and black. Periodic copies are shown in gray. The red lines indicate the RVE boundaries. The graph includes only nodes at the boundaries of the holes and edges connecting the nodes to the nearest nodes at the boundaries of the other holes.

transfer of information throughout the graph, but we kept the question of optimizing the graph representation out of our scope for now.

In the constructed input graph, each node i has a position $\vec{x}_i$ (a 2D or 3D vector) and an abstract embedding $\mathbf{h}_i$ associated with it. In addition, every edge ij has a vector $\vec{r}_{ij}$ pointing along it and an abstract embedding $\mathbf{e}_{ij}$; see Figure 2.9 for a visual explanation of these graph quantities. Initially, the edge embedding $\mathbf{e}_{ij}^0$ carries a Boolean, informing whether the edge corresponds to a hole boundary or not. In our setup, we encode this Boolean by assigning $\mathbf{e}_{ij} = -1$ if at a hole boundary and $+1$ otherwise. Swapping these values, or using other distinct values except zero is possible, since this information simply needs to indicate to the network that these edges differ. A transformation to other values would be straightforward for a neural network to learn. We chose -1 and +1 because by default, neural networks are initialized to handle values in this range the best.

The initial node embedding $\mathbf{h}_i^0$ is empty.

Table 2.A.2 in 2.A shows what this graph representation actually looks like in the data.



(a) (b)

Figure 2.9: Definition of the graph quantities: (a) each node $i \in \mathcal{V}$ has a position $\vec{x}_i$ (a 2D or 3D vector) and an abstract embedding $\mathbf{h}_i$ associated with it. Each edge ij has a vector $\vec{r}_{ij}$ pointing along it and an abstract embedding $\mathbf{e}_{ij}$. (b) During a message passing step l, a shift $\Delta\vec{x}_i^{\,l}$ is applied to each node, which moves the node from its old position $\vec{x}_i^{\,l}$ to the new position $\vec{x}_i^{\,l+1} = \vec{x}_i^{\,l} + \Delta\vec{x}_i^{\,l}$. The edge vectors are then updated correspondingly.

2.4.2 General approach

GNNs incorporate permutation symmetry by updating the node and edge attributes by message passing, which is a way of propagating information throughout the graph [105]. For all three models we compare, the message passing steps update the node positions $\vec{x}_i$ and generate abstract node and edge embeddings $\mathbf{h}_i$ and $\mathbf{e}_{ij}$. The node positions after the last message passing steps are the predicted positions.[6]

Message passing involves constructing a vector $\mathbf{m}_{ij}$ (called a 'message') for each edge $ij \in \mathcal{E}$, and then aggregating at each node $i \in \mathcal{V}$ the 'incoming' messages (the messages associated with edges that connect to the node) to update the node embedding $\mathbf{h}_i$. In its most general form, this looks like

$$\mathbf{m}_{ij}^{l} = \phi_m^l \left(\mathbf{h}_i^l, \mathbf{h}_j^l, \mathbf{e}_{ij} \right), \tag{2.20}$$

$$\mathbf{h}_i^{l+1} = \phi_h^l \left(\mathbf{h}_i^l, \square_{j \in \mathcal{N}(i)} \mathbf{m}_{ij}^l \right), \tag{2.21}$$

where $\square$ is the aggregation function (e.g. sum or average), $\mathbf{m}_{ij}^l$ is the message from node j to i in message passing step l, $\mathcal{N}(i)$ is the set of neighbors of node i, and ϕ_m^l and ϕ_h^l are learnable functions (in our case, one linear layer plus an activation function), that are re-used for each node or edge, but which can differ per message passing step l. We also update the edge embeddings from the messages:

$$\mathbf{e}_{ij}^{l+1} = \phi_e^l(\mathbf{m}_{ij}^l), \tag{2.22}$$

[6] Because any predicted displacement that contains a rigid body translation is still a valid solution, we compare the final GNN prediction for $\vec{x}$ to the FEM ground truth after setting the mean position such that it matches the FEM ground truth.

where ϕ_e^l is another learnable function, which is re-used for each edge, but again can differ per message passing step l. A graph neural network generally consists of multiple message passing steps. In our case, this message passing scheme needs to be modified to also take into account geometric quantities like the distances r_{ij} between nodes. The exact form of the resulting message passing scheme differs for each of the three models we developed, but all have in common that they update the position $\vec{x}_i^l$ of the nodes in each message passing step l, by computing for each node a shift in position $\Delta\vec{x}_i^{\,l}$

$$\vec{x}_i^{\,l+1} = \vec{x}_i^{\,l} + \Delta\vec{x}_i^{\,l} \tag{2.23}$$

The way $\Delta\vec{x}_i^{\,l}$ is computed differs per model.

At the end of the messages passing steps, still only local abstract quantities are available: node and edge embeddings, and the messages from the last message passing step. The way these quantities are converted into global predictions of the strain energy density $\mathfrak{W}$, stress $\mathbf{P}$ and stiffness ${}^4\mathbf{D}$ also differs per model, but they all have in common that some local quantity (either an existing one or a newly created one) is being averaged. We use averaging, because it is permutation invariant with respect to the order of the nodes and edges, and is also the most common approach to homogenize local quantities to global ones in computational homogenization. It also possible to do weighted averaging, which essentially means adding an attention layer. This also applies to the message aggregation. However, we chose to keep our model architecture simple.

The mean square error in the prediction of each of the four target quantities ($\vec{w}$, $\mathfrak{W}$, $\mathbf{P}$, ${}^4\mathbf{D}$) is used in the loss. Details about training and balancing these loss terms are given in 2.B.

Periodic Boundary Conditions

In order to properly implement periodic boundary conditions as described in Section 2.2, we need to construct and process the graph in such a way that the 'Shifted RVE' and 'Extended RVE' in-/equivariance shown in Figure 2.7 are respected. For this purpose, when constructing the input graph, we modify the computation of the edge vectors $\vec{r}_{ij}$ that are associated with the 'wraparound' edges. Instead of pointing from $\vec{x}_i$ to $\vec{x}_j$, $\vec{r}_{ij}$ then points from $\vec{x}_i$ to the periodic image $\vec{x}_j$ on the correct side. Because we use only $\vec{r}_{ij}$ and the distance r_{ij}, which are independent of the actual location of the nodes, in updating the node embeddings, node positions and edge embeddings, the graph is updated independently of how the RVE is chosen.

However, these wraparound edges entail a problem if not handled carefully; their edge vectors $\vec{r}_{ij}$ cannot be directly recalculated from the node positions $\vec{x}_i$. These positions $\vec{x}_i$ are updated multiple times during the message passing steps, and in the standard implementations of the MeshGraphNets and EGNNs, the updated $\vec{r}_{ij}$ are then recalculated from $\vec{x}_i$ as $\vec{r}_{ij} = \vec{x}_j - \vec{x}_i$, which will be inaccurate for the wraparound edges. In order to avoid using periodic images all the time, we only calculate $\vec{r}$ once in the beginning, after that, we keep $\vec{x}_i$ and $\vec{r}_{ij}$ separate, and update $\vec{r}_{ij}$ directly using

$$\vec{r}_{ij}^{\,l+1} = \vec{r}_{ij}^{\,l} - \Delta\vec{x}_i^{\,l} + \Delta\vec{x}_j^{\,l}, \tag{2.24}$$

where $\Delta\vec{x}_i$ is the shift in node position $\vec{x}_i^{l+1} = \vec{x}_i^l + \Delta\vec{x}_i$ in message passing step l. This ensures that the updates of $\vec{r}_{ij}$ are still independent of the absolute positions.

To ensure that in-/equivariance with respect to the transformations 'Extended RVE' and 'Shifted RVE' in Figure 2.7 is still preserved after applying $\mathbf{F}$, we apply $\mathbf{F}$ as an affine transformation to all reference node positions $\vec{x}_i^{\text{ref}}$ and edge vectors $\vec{r}_{ij}^{\text{ref}}$ at the same time. This provides the initial positions and edge vectors fed into the GNN:

$$\vec{x}_i^0 = \mathbf{F} \cdot \vec{x}_i^{\text{ref}}, \tag{2.25}$$

$$\vec{r}_{ij}^0 = \mathbf{F} \cdot \vec{r}_{ij}^{\text{ref}}. \tag{2.26}$$

Figure 2.10 shows the general structure of all three models, including the preprocessing. In the following paragraphs, the 'message passing' and 'read-out' blocks of this diagram will be described per model.

2.4.3 Base GNN

For the base GNN used as a baseline in this paper, we use an approach similar to MeshGraphNets [74]. For that, we use the following message computation:

$$\mathbf{m}_{ij}^l = \phi_m^l \left(\mathbf{h}_i^l, \mathbf{h}_j^l, \vec{r}_{ij}^{\text{ref}}, r_{ij}^{\text{ref}}, \vec{r}_{ij}^{\,l}, r_{ij}^l, \mathbf{e}_{ij}^l \right), \tag{2.27}$$

with ϕ_m^l a learnable function (in our case, one linear layer plus an activation function). This means the message computation for edge ij uses both the reference edge length $r_{ij}^{\text{ref}} = \|\vec{r}_{ij}^{\text{ref}}\|_2$ and all components of the edge vector $\vec{r}_{ij}^{\text{ref}}$ (before any deformation) and the current edge length $r_{ij}^l = \|\vec{r}_{ij}^{\,l}\|_2$ and edge vector $\vec{r}_{ij}^{\,l}$ (after affine deformation and the message passing steps up to l), as well as the current node embeddings $\mathbf{h}_i^l, \mathbf{h}_j^l$ of both nodes, and the edge embedding $\mathbf{e}_{ij}^l$.

The messages $\mathbf{m}_{ij}^l$ are then used to update the node and edge embeddings

$$\mathbf{h}_i^{l+1} = \phi_h^l \left(\mathbf{h}_i^l, \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \mathbf{m}_{ij}^l \right), \tag{2.28}$$

$$\mathbf{e}_{ij}^{l+1} = \phi_e^l(\mathbf{m}_{ij}^l), \tag{2.29}$$

where $|\mathcal{N}(i)|$ is the number of neighbors of node i and ϕ_e^l and ϕ_h^l are, again, learnable functions. We chose not to include $\mathbf{e}_{ij}^l$ in the last equation, because it is already included in the message $\mathbf{m}_{ij}^l$.

From the new node embedding $\mathbf{h}_i^{l+1}$, during every message passing step the shift $\Delta\vec{x}_i$ in the position $\vec{x}_i^{\,l}$, as illustrated in Figure 2.9b, is calculated as

$$\Delta\vec{x}_i^{\,l} = \phi_x^l \left(\mathbf{h}_i^{l+1} \right), \tag{2.30}$$

with another learnable function ϕ_x^l. The edge vectors are then updated correspondingly according to Equation (2.24). Figure 2.11a provides a flowchart of the message passing scheme of this base GNN.

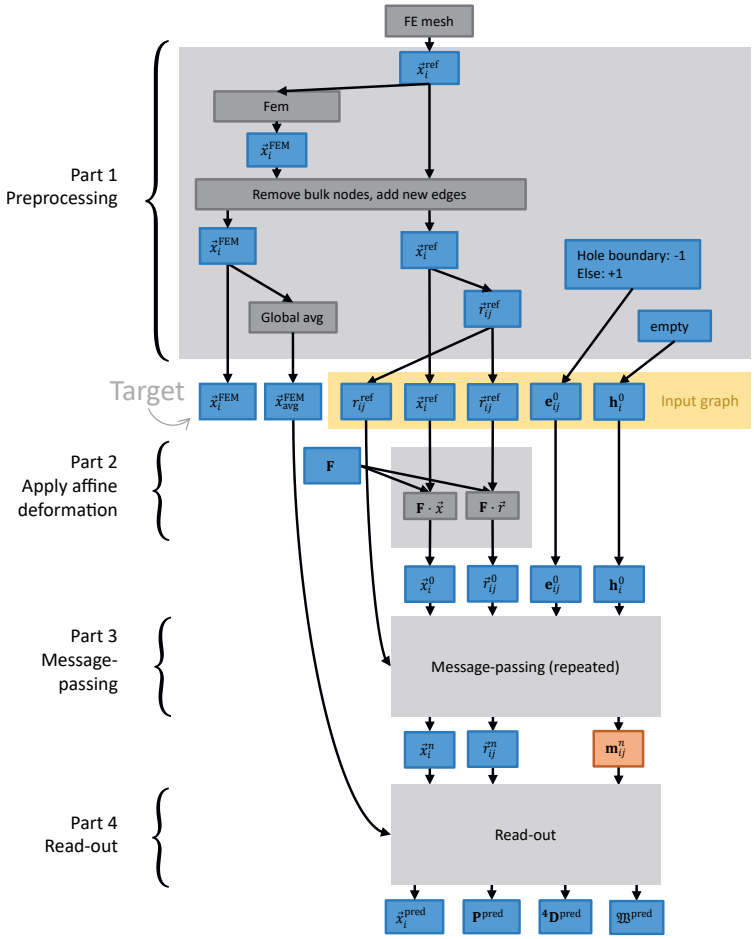


Figure 2.10: Diagram of the structure of all models, including the preprocessing. All blue quantities have a physical meaning, the orange quantity is abstract and generated by the neural network. The dark gray boxes indicate operations.

To predict the global quantities $\mathfrak{W}$, $\mathbf{P}$ and ${}^4\mathbf{D}$, we aggregate the final messages $\mathbf{m}_{ij}^n$ (after all n message passing steps) into a global abstract quantity, to which we then apply three final learnable functions $\phi_{\mathfrak{W}}$, ϕ_P, ϕ_D. Therefore, we predict $\mathfrak{W}$, $\mathbf{P}$ and ${}^4\mathbf{D}$ with

$$\mathfrak{W} = \phi_{\mathfrak{W}} \left(\frac{1}{N_e} \sum_{i,j} \mathbf{m}_{ij}^n \right), \tag{2.31}$$

$$\mathbf{P} = \phi_P \left(\frac{1}{N_e} \sum_{i,j} \mathbf{m}_{ij}^n \right), \tag{2.32}$$

$${}^4\mathbf{D} = \phi_D \left(\frac{1}{N_e} \sum_{i,j} \mathbf{m}_{ij}^n \right), \tag{2.33}$$

where N_e is the total number of edges and $\phi_{\mathfrak{W}}$, ϕ_P and ϕ_D are learnable functions. See Figure 2.11b for a flowchart of this last part of the GNN.

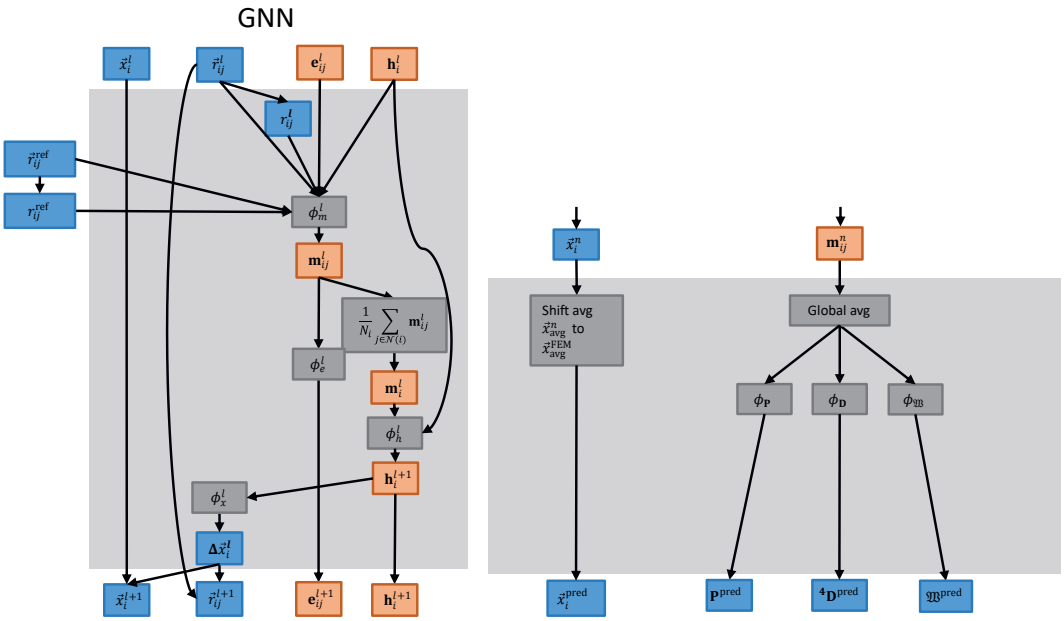
In Equation (2.27), the model treats different components of the edge vectors simply as different inputs. In Equations (2.30), (2.32) and (2.33) different components of $\Delta\vec{x}$, $\mathbf{P}$ and ${}^4\mathbf{D}$ are treated as different outputs. This implies that the network treats these components as unrelated, which means the model does not respect rotation, reflection or scaling in-/equivariance. It respects only translation in-/equivariance, because then all the inputs and outputs stay the same, as well as RVE in-/equivariance because of the periodic boundary conditions. Hence, this model will struggle with generalizing over rotations, reflections and scaling.

2.4.4 EGNN

Even though all positions must be expressed in a coordinate system, the way they evolve is independent of this coordinate system. Therefore, for our second GNN, we aim to update the positions in the chosen coordinate system, in a way that is independent of that coordinate system. To this end, we use $E(n)$-equivariant graph neural networks (EGNNs) by Satorras et al. [87]. [7] This is achieved by pulling or pushing the nodes along their edges, i.e., the shift $\Delta\vec{x}_i$ in the node position $\vec{x}_i$ is computed by using vectors $\vec{r}_{ij}$ as an overcomplete basis set.[8] The message passing

[7] The original paper [87] also includes velocities $\vec{v}_i$. However, these are not relevant for our purposes, because we only consider a quasi-static system.

[8] Meaning, $\vec{x}_i$ is computed as a linear combination of $\vec{r}_{ij}$. This basis set is overcomplete because the number of edge vectors $\vec{r}_{ij}$ connected to node i will always be larger than the number of dimensions.



(a) (b)

Figure 2.11: Diagrams of the base GNN as applied to metamaterial simulations. (a) One message passing step. (b) The read-out part, which computes final predictions for all quantities. The blue quantities have a physical meaning whereas the orange quantities are abstract and generated by the neural network (except for the initial edge and node embeddings $\mathbf{e}_{ij}^0$ and $\mathbf{h}_i^0$, which may also have a physical meaning since they are part of the input). The dark gray boxes indicate operations, either explicitly stated or through an underlying neural network.

scheme is described by the following relations:

$$\mathbf{m}_{ij}^{l} = \phi_m\left(\mathbf{h}_i^l, \mathbf{h}_j^l, r_{ij}^{\ l}, \mathbf{e}_{ij}\right), \tag{2.34}$$

$$\Delta\vec{x}_i = \frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)} \vec{r}_{ij}^{\,l}\tanh\left(\phi_x(\mathbf{m}_{ij}^l)\right), \tag{2.35}$$

$$\mathbf{m}_i^l = \frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)} \mathbf{m}_{ij}^l, \tag{2.36}$$

$$\mathbf{e}_{ij}^{l+1} = \phi_e\left(\mathbf{h}_i^l, \mathbf{m}_i^l\right), \tag{2.37}$$

$$\mathbf{h}_i^{l+1} = \phi_h\left(\mathbf{h}_i^l, \mathbf{m}_i^l\right), \tag{2.38}$$

and summarized in the flowchart of Figure 2.12a. In the expressions above, $\mathbf{m}_i^l$ denotes the aggregated message that node i receives. The hyperbolic tangent in Equation (2.35) stops the network from shifting the nodes over large distances. Without this operator, we observed that for some configurations the distances would grow exponentially with the message-passing steps. This happens because the node movements are based on a linear combination of the edge vectors, making the movements proportional to the distances. As a result, the distances can increase exponentially with the message-passing steps. In this approach, when there is a change in the coordinate system, the edge vectors $\vec{r}_{ij}^{\,l}$ are changed correspondingly. Consequently, since the predicted shifts in nodal positions are linear combinations of these edge vectors and the scalars $\tanh\left(\phi_x^l(\mathbf{m}_{ij}^l)\right)$ are independent of the coordinate system, the shifts $\Delta\vec{x}_i^{\,l}$ also change with the coordinate system, resulting in the desired $E(n)$-equivariance. For a proof of this, see [87].

To predict second- and fourth-order tensors (stress and stiffness, respectively) in an $E(n)$-equivariant way, we take inspiration from the calculation of $\Delta\vec{x}_i^{\,l}$ and use the edge vectors $\vec{r}_{ij}$ as an overcomplete basis set again, although the final ones $\vec{r}_{ij}^{\,n}$ this time. A second-order tensor for each node i can be constructed using dyadic products of $\vec{r}_{ij}^{\,n}$, i.e.,

$$\mathbf{A}_i = \frac{1}{|\mathcal{N}(i)|^2}\sum_{j\in\mathcal{N}(i)}\sum_{k\in\mathcal{N}(i)} c_{i,jk}\vec{r}_{ij}\otimes\vec{r}_{ik}, \tag{2.39}$$

$$c_{i,jk} = \phi_A\left(\mathbf{m}_{ij}^n, \mathbf{m}_{ik}^n\right), \tag{2.40}$$

where $c_{i,jk}$ is a scalar, computed with a learnable layer ϕ_A, for each triplet i, j, k where node j and k are from the neighborhood $\mathcal{N}(i)$ of node i. For the maximum number of neighbors of the graph in Figure 2.8 $|\mathcal{N}(i)| = 8$, Equation (2.39) has $|\mathcal{N}(i)|^2 = 64$ terms, which is reasonable.

Theoretically, we can extend this approach to predict a fourth-order tensor as follows

$${}^4\mathbf{B}_i = \frac{1}{|\mathcal{N}(i)|^4}\sum_{j\in\mathcal{N}(i)}\sum_{k\in\mathcal{N}(i)}\sum_{l\in\mathcal{N}(i)}\sum_{m\in\mathcal{N}(i)} c_{i,jklm}\vec{r}_{ij}\otimes\vec{r}_{ik}\otimes\vec{r}_{il}\otimes\vec{r}_{im}, \tag{2.41}$$

$$c_{i,jklm} = \phi_B\left(\mathbf{m}_{ij}^n, \mathbf{m}_{ik}^n, \mathbf{m}_{il}^n, \mathbf{m}_{im}^n\right) \tag{2.42}$$

where $c_{i,jklm}$ is a scalar, computed with a learnable layer ϕ_B, for each combination of edge vectors. However, for the maximum number of neighbors $|\mathcal{N}(i)| = 8$ this results in $|\mathcal{N}(i)|^4 = 4096$ terms, which becomes computationally too expensive. For that reason, we first predict a second-order tensor $\mathbf{A}'_i$ for each node i using the approach in Equations (2.39) and (2.40), and then use these tensors of neighboring nodes as a basis to predict a fourth-order tensor

$$ {}^4\mathbf{B}_i = \frac{1}{|\mathcal{N}(i)|^2} \sum_{j \in \mathcal{N}(i)} \sum_{k \in \mathcal{N}(i)} c'_{i,jk} \mathbf{A}'_j \otimes \mathbf{A}'_k, \tag{2.43} $$

$$ c'_{i,jk} = \phi_B \left(\mathbf{m}^n_{ij}, \mathbf{m}^n_{ik} \right). \tag{2.44} $$

In this approach, because the predicted second and fourth-order tensors are linear combinations of dyadic/tetradic products of the edge vectors and the scalars $c_{i,jk}$ are independent of the coordinate system, the $\mathbf{A}_i$ and ${}^4\mathbf{B}_i$ are transformed as tensors under a change in the coordinate system, resulting in the desired $E(n)$-equivariance for these tensors as well.

For the final prediction of the global $\mathbf{P}$ and ${}^4\mathbf{D}$, we take a global average of the per-node second- and fourth-order tensors, such that

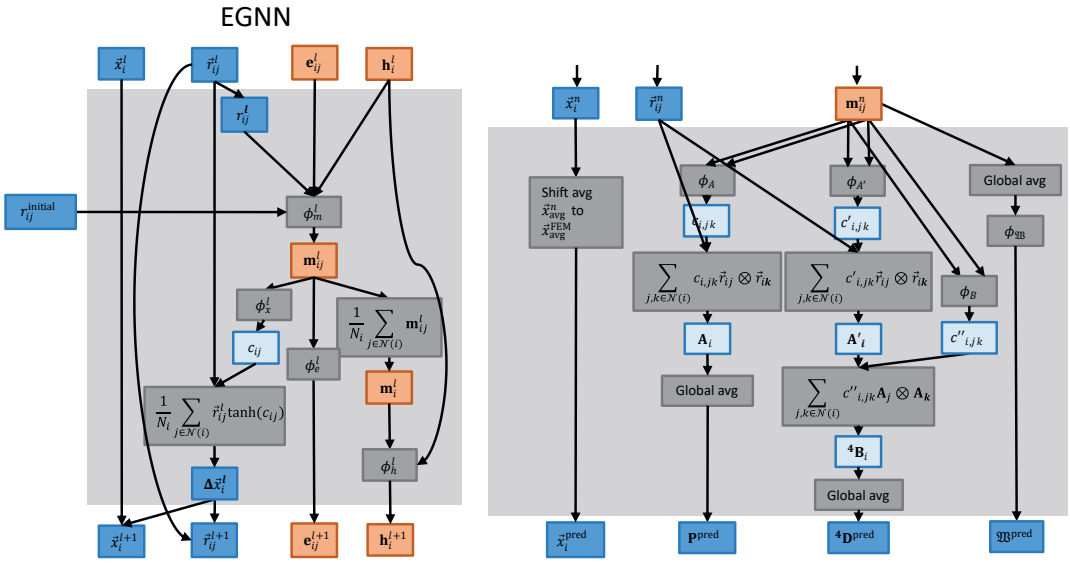
$$ \mathbf{P}_{\text{pred}} = \frac{1}{N} \sum_i^N \mathbf{A}_i \tag{2.45} $$

$$ {}^4\mathbf{D}_{\text{pred}} = \frac{1}{N} \sum_i^N {}^4\mathbf{B}_i. \tag{2.46} $$

The whole read-out part of the EGNN is illustrated with the flowchart in Figure 2.12b.

All the predictions of this model respect $E(n)$-in-/equivariance, as well as RVE in-/equivariance because of the periodic boundary conditions. Only scale in-/equivariance is missing. Hence, this model will struggle with generalizing over scaling.

Since we are modeling a conservative system, an obvious alternative approach to predict tensors would be using Sobolev training [106] (i.e., training not just for a certain output, but also for one or more of its derivatives). This means the network would predict only the predicted energy density $\mathfrak{W}$. The macroscopic first Piola-Kirchhoff stress tensor and the stiffness tensor would then be calculated from $\mathfrak{W}$ according to Equations (2.14) and (2.15). This means calculating $\mathbf{P} = \frac{\partial \mathfrak{W}}{\partial \mathbf{F}}$ and ${}^4\mathbf{D} = \frac{\partial^2 \mathfrak{W}}{\partial \mathbf{F}^2}$ using the autodifferentiation that any package for neural network training such as PyTorch is capable of, and then training for $\mathfrak{W}$ as well as $\mathbf{P}$ and ${}^4\mathbf{D}$. Many molecular GNNs similarly train for the energy as well as the forces [77, 80]. This approach would also guarantee that the differentiability relations in Equations 2.14 and 2.15 are exactly satisfied, which is not the case with the current approach. However, we found that for the second derivative ${}^4\mathbf{D}$ this is prohibitively slow and unstable. Presumably the instability is due to the repeated differentiation, which means a small change in $\mathfrak{W}$ can lead to a large change in ${}^4\mathbf{D}$. For the first derivative $\mathbf{P}$ this approach is nevertheless feasible, but still not as fast or accurate as predicting $\mathbf{P}$ directly.



(a) (b)

Figure 2.12: Diagrams of the EGNN as applied to metamaterial simulations. (a) One message passing step, which iteratively updates the node positions and node and edge embeddings, (b) the read-out part, which computes final predictions for all quantities. Blue quantities have a physical meaning, and the orange quantities are abstract quantities generated by the neural network (except for the initial edge and node embeddings $\mathbf{e}_{ij}^0$ and $\mathbf{h}_i^0$, which may also have physical meaning as they are part of the input); finally, light blue quantities are not abstract but do not have a direct physical interpretation either. The dark gray boxes indicate operations, either explicitly stated or through an underlying neural network.

2.4.5 Similarity-Equivariant GNN (SimEGNN)

Finally, the last architecture developed and tested in this work is the Similarity-Equivariant GNN (SimEGNN), possessing all required in-/equivariances of Table 2.2. The Euclidean group $E(n)$ appearing in the previous architecture includes transformations that preserve Euclidean distances (i.e., rotations, reflections, translations). However, in order to apply it to first-order homogenization problems, we also require scale in-/equivariance. The group equivariance we need is thus with respect to the similarities group S, which also includes isotropic scaling and preserves distance ratios. To achieve this, instead of the distance between nodes r_{ij}^l in the message computation of Equation (2.34), we use the strain measure defined as $\varepsilon_{ij}^l = (r_{ij}^l - r_{ij}^{\text{ref}})/r_{ij}^{\text{ref}}$, as well as a relative distance $r_{ij}^l/\bar{r}_i^{\,l}$, with $\bar{r}_i^{\,l}$ defined as

$$\bar{r}_i^{\,l} = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} r_{ij}^{\,l}. \tag{2.47}$$

This feature means that although each node does not know the absolute lengths of its incoming edges, it does know which incoming edges are longer or shorter than the others. The message computation then looks as follows:

$$\mathbf{m}_{ij}^l = \phi_m^l \left(\mathbf{h}_i^l, \mathbf{h}_j^l, \varepsilon_{ij}^l, r_{ij}^l/\bar{r}_i^{\,l}, \mathbf{e}_{ij}^l \right). \tag{2.48}$$

Strains are also used in the computation of the shift in Equation (2.35), i.e.,

$$\Delta\vec{x}_i^{\,l} = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \vec{r}_{ij}^{\,l} \tanh\left(\varepsilon_{ij}^l \phi_x^l(\mathbf{m}_{ij}^l)\right), \tag{2.49}$$

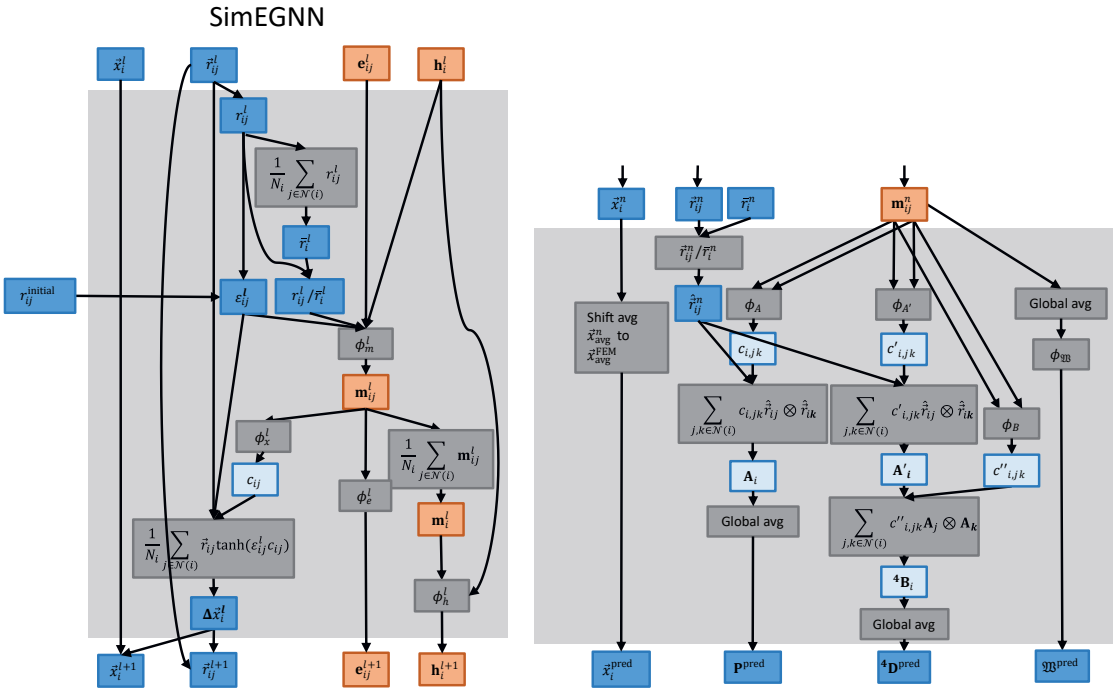
which guarantees that there is no position shift for $\mathbf{F} = \mathbf{I}$, because then all strains ε_{ij}^l will be zero, which will make all $\Delta\vec{x}_i^{\,l}$ equal to zero. The resulting message passing scheme is shown in the flowchart of Figure 2.13a. Because $\mathfrak{W}$, $\mathbf{P}$ and ${}^4\mathbf{D}$ are scale *invariant*, we normalize the final edge vectors $\vec{r}_{ij}^{\,n}$ used as an overcomplete basis set to predict node-wise tensors $\mathbf{A}_i$ (recall Equation (2.39)):

$$\mathbf{A}_i = \frac{1}{|\mathcal{N}(i)|^2} \sum_{j \in \mathcal{N}(i)} \sum_{k \in \mathcal{N}(i)} c_{i,jk} \hat{\vec{r}}_{ij}^{\,n} \otimes \hat{\vec{r}}_{ik}^{\,n}, \tag{2.50}$$

where $\hat{\vec{r}}_{ij}^{\,n} = \vec{r}_{ij}^{\,n}/\bar{r}_i^{\,n}$ are the normalized final edge vectors. This read-out part of the SimEGNN is illustrated with the flowchart in Figure 2.13b.

2.5 Results

The model evaluation consists of 2 parts: The first part in Section 2.5.1 focuses on the effect of embedding the symmetries on test cases that reflect those symmetries. The second part in Section 2.5 focuses on the generalization of the models to different geometries, by testing the models on microstructures with different hole sizes. Finally a discussion on efficiency and scaling is included in Section 2.5.3.



(a) (b)

Figure 2.13: Diagrams of the SimEGNN as applied to metamaterial simulations. (a) One message passing step, which iteratively updates the node positions and node and edge embeddings, (b) the read-out part, which computes final predictions for all quantities. Blue quantities have a physical meaning, and the orange quantities are abstract quantities generated by the neural network (except for the initial edge and node embeddings $\mathbf{e}_{ij}^0$ and $\mathbf{h}_i^0$, which may also have a physical meaning since they are part of the input); finally, light blue quantities are not abstract but do not have a direct physical interpretation either. The dark gray boxes indicate operations, either explicitly stated or through an underlying neural network.

2.5.1 Effect of Embedding Symmetries

We test 7 different versions of the 3 graph neural networks described in Section 2.4, from the smallest to the largest number of symmetries, on 6 different test cases, for 4 target quantities. We test them on the homogenization problem introduced in Section 2.2. We split that dataset by trajectory, such that the training and validation data are from different trajectories, with 80% of the trajectories in the training data and 20% in the validation data. Because we do 5-fold cross-validation, each time we choose a different 20%. The dataset is described in more detail in 2.A. In addition to the three types of GNN architectures, for the GNN and EGNN, we tested augmenting the training data as described in Section 2.2.3, as a different way to enforce the symmetries. The models trained on augmented data are marked 'DA' for 'Data Augmentation'. This resulted in 5 models in total: 2 GNNs, 2 EGNNs and 1 SimEGNN. As mentioned, we perform 5-fold cross-validation for each model, resulting in 5 sets of results per model type. The results cover 6 different test cases:

1. the validation data (labeled 'untransformed', where none of the configurations have been rotated, reflected, translated or scaled),
2. the validation data reflected in the y-axis ('reflected'),
3. rotated by $\pi/4$ radians ('rotated'),
4. with a shifted RVE, which means the material is shifted up and left by one quarter the size of the RVE, such that the RVE is centered on one hole (labeled 'shifted RVE'),
5. with a larger RVE that consists of four original RVEs merged together ('extended RVE'),
6. and an RVE which has been scaled by a factor 1.5 ('scaled').

These test cases represent the various in-/equivariances of Figure 2.7 and Table 2.2 that we built into the model, and were chosen to evaluate the effect of embedding these in-/equivariances. The 4 different target quantities are the microfluctuation field $\vec{w} = \vec{x} - \mathbf{F} \cdot \vec{x}^{\,\mathrm{ref}}$ (see Equation 2.12), homogenized strain energy density $\mathfrak{W}$, homogenized stress $\mathbf{P}$ and tangent stiffness tensor ${}^4\mathbf{D}$.

The spider web charts in Figure 2.14 show the mean fraction of variance unexplained (FVU) obtained by each neural network architecture, averaged over the 5 cross-validation models. Shaded areas indicate the standard error (although it is difficult to see because it is very narrow). The FVU of the prediction of a quantity y is defined as

$$\mathrm{FVU} = \frac{\sum_i^N (y_i - \hat{y}_i)^2}{\sum_i^N (y_i - \bar{y})^2} = \frac{\mathrm{MSE}(y, \hat{y})}{\mathrm{Var}(y)}, \tag{2.51}$$

where $\hat{y}$ is the prediction, $\bar{y}$ is the average value of the target, and $\mathrm{Var}(y)$ is the variance of the target and MSE the mean squared error between the real and predicted values. This means the FVU is the variance in the errors of quantities predicted by the models divided by the variance in the data of the target quantity[9]. Each chart corresponds to one target quantity: $\vec{w}$, $\mathfrak{W}$, $\mathbf{P}$ or $\mathbf{D}$. The investigated models are shown in different line colors and styles, where the color indicates the type of neural network (GNN, EGNN or SimEGNN) and the line style (solid or dashed) indicates

[9] i.e., $\mathrm{FVU} = 1 - R^2$, where R^2 is the coefficient of determination

whether data augmentation was used during training. The different spokes of the spider plots correspond to the six test cases (reference, reflected, rotated, shifted RVE, larger RVE, scaled).

Additionally, for four of these six test cases, we show a comparison of the predicted hole deformations from three of the networks against FEM outputs – considered the ground truth in this work – for biaxial compression in Figure 2.15. We exclude the 'Shifted RVE' and 'Extended RVE' cases, because all models respect periodicity. The Figure only shows the networks trained without data augmentation. For the sake of completeness, we show figures that include all models in 2.C, including the ones trained with data augmentation. These figures also show the results for different $\mathbf{F}$, such that all bifurcation patterns are included. Tables with the data behind Figure 2.14 are also provided in 2.C, as well as tables with the relative error.

The results of Figure 2.14 show that the periodicity is correctly implemented in all models, because all of them can predict the 'shifted RVE' and 'larger RVE' cases with an FVU equal to the untransformed case. By construction, the EGNN indeed additionally respects $E(n)$-in-/equivariance ('reflected' and 'rotated', translated not shown in Figure 2.14), and the SimEGNN additionally respects scale in-/equivariance.[10] The figure also shows that the microfluctuation $\vec{w}$ and stiffness ${}^4\mathbf{D}$ are generally more difficult to predict than the strain energy density $\mathfrak{W}$ and the stress $\mathbf{P}$; in the plots for $\vec{w}$ and ${}^4\mathbf{D}$, the FVU stays above 10^{-4}, whereas in the plots for $\mathfrak{W}$ and $\mathbf{P}$ the FVU gets close to 10^{-6}.

For GNNs (the blue lines in Figure 2.14), augmenting the data with reflected, rotated and scaled cases improves the predictions for those cases, i.e., it makes the result more similar to the 'untransformed' result. However, it worsens the 'untransformed' result at the same time. This indicates that the model is trying to learn all the equivalent cases separately, which effectively reduces the capacity of the network. A similar effect can be seen for the EGNN (orange lines in Figure 2.14) and the 'scaled' case.

Figure 2.14 shows that with all symmetries incorporated directly into the network's architecture, the FVU improves by about an order of magnitude or more, compared to the GNN with 2× data augmentation, on all target quantities. This holds for all test cases, even for the untransformed case, where none of the configurations have been rotated, reflected, translated or scaled. Incorporating more symmetries thus makes it easier for the model to construct and use features in a general way. For example, it can more easily transfer learned features from one edge to predict the behavior of another longer edge with scale in-/equivariance. Embedding the symmetries thus allows the network to focus on constructing actually meaningful features, because it does not first have to filter out the aspects that are irrelevant (scale, rotation, etc.). Consequently, the newly proposed SimEGNN outperforms the other models in all aspects.

[10] The EGNN and SimEGNN still show a small deviation in the FVU for the rotated and reflected cases for the stress $\mathbf{P}$ and stiffness ${}^4\mathbf{D}$; this is because those transformations slightly change the variance in the ground truth data. The mean square error does remain the same, however.

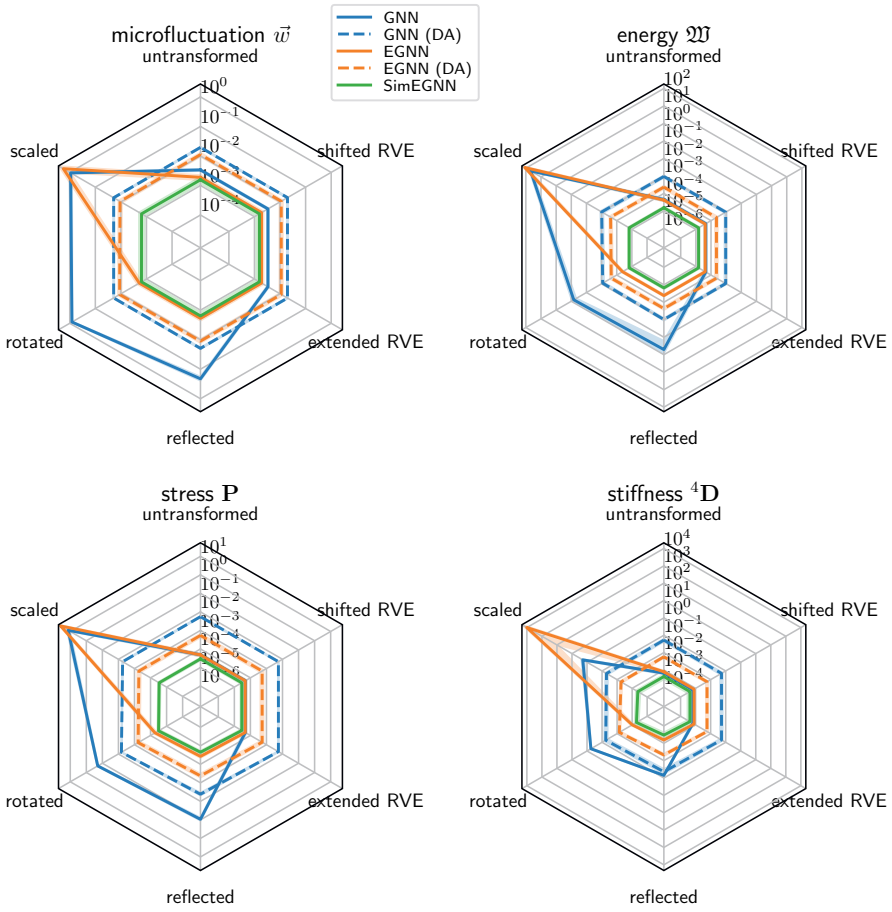


Figure 2.14: Mean fraction of variance unexplained of the four target quantities ($\vec{w}$, $\mathfrak{W}$, $\mathbf{P}$, $\mathbf{D}$) achieved by the investigated models (shown in different line colors and styles) for the six test cases (untransformed, reflected, rotated, shifted RVE, larger RVE, scaled). The mean is calculated over the 5 folds from the cross-validation. The standard error is indicated by a shaded area. However, it is visible mainly for $\mathfrak{W}$ in the reflected case by the GNN and for ${}^4\mathbf{D}$ in the scaled case by the EGNN. The SimEGNN is the only model that respects all symmetries, and it outperforms the other models.

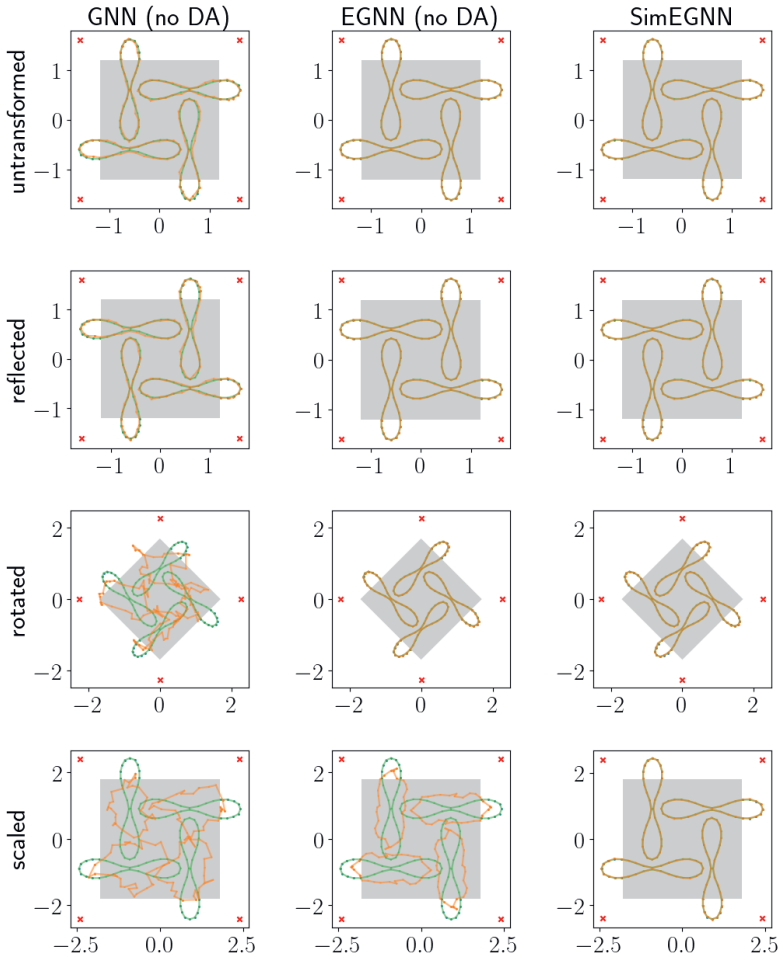


Figure 2.15: Predicted deformation, using the three models without data augmentation, of the hole boundaries (orange) compared to the FEM ground truth (green), for $\mathbf{F} = \begin{bmatrix} 0.75 & 0 \\ 0 & 0.75 \end{bmatrix}$ (biaxial compression, resulting in a rotational pattern).

2.5.2 Generalization to Different Geometries

As a first test of the generalization capabilities of the above-introduced models, we consider microstructures with hole size diameters of 0.4ℓ, 0.425ℓ, 0.45ℓ and 0.475ℓ (corresponding to minimum ligament thicknesses of 0.1ℓ, 0.075ℓ, 0.05ℓ and 0.025ℓ, respectively). We test the three architectures, and additionally train with data augmentation for the GNN and the EGNN.

The results displayed in Figure 2.16 show that on microstructures with 4 different diameters, the models can still get performance close to the performance on the dataset with just one microstructure. It also shows that the SimEGNN is still the only model that performs well on all test cases. The GNN performs really poorly on the scaled, rotated and reflected cases, and the EGNN performs badly on the scaled case. With data augmentation, the performance on these cases can be improved. However, especially for the energy $\mathfrak{W}$ and the stress $\mathbf{P}$, we see the same trade-off for data augmentation as in the previous section: without the data augmentation, the performance is very bad on cases that reflect the symmetries that are not included in the model, but with it, the performance is worse overall. For the microfluctuation $\vec{w}$ the data augmentation also makes it worse overall, but the effect is less dramatic. For the stiffness ${}^4\mathbf{D}$, the data augmentation on average makes it worse, but in the case of the EGNN on the reference case, the standard errors overlap, which means it is unclear if this difference is significant.

2.5.3 Efficiency and Scaling

Generating one prediction with the finite element method takes on average 0.93 seconds on a single-threaded CPU (an Intel(R) Core(TM) i7-9750H), and 0.49 seconds on a 12-threaded CPU. Preprocessing the results for GNNs takes 9.0 milliseconds per graph, of which 5.2 milliseconds are for removing cases with contact, which would be necessary as post-processing for FEM anyway. In inference, the SimEGNN takes 0.13 seconds to generate one prediction on a single-threaded CPU, and 0.06 seconds on a 12-threaded CPU, which gives a speed-up of 7.10 and 7.56 times, respectively. If the SimEGNN is evaluated on a GPU (NVIDIA Quadro P2000 GPU), it takes 3.6 milliseconds to generate one prediction, which is a speed-up of 135 relative to the 12-threaded FEM. However, because the training time is quite long at 11.4 hours, the initial investment into the SimGNN is quite high, and will currently only pay off if many predictions are needed. Fortunately, we expect that this trade-off can still improve in several ways in the future. For example, it is likely that our SimEGNN approach can also be made faster in the most obvious way: by lowering our standards. The current model is remarkably accurate. If we require a maximum relative error of about 5% in the homogenized target quantities (instead of the current $0.31\% - 0.90\%$), a much smaller model will probably also suffice. We also expect our method to scale much better when applied to finer meshes and more complicated or 3D geometries, because we only use the boundary nodes. This means the number of used nodes scales linearly with the mesh size, while the number of nodes in the finite element mesh scales quadratically (in 2D). In 3D the number of boundary nodes scales quadratically and the number of bulk nodes scales to the power 3, hence we expect the SimEGNN to scale much better than the finite element method. The main limitation of the pre-

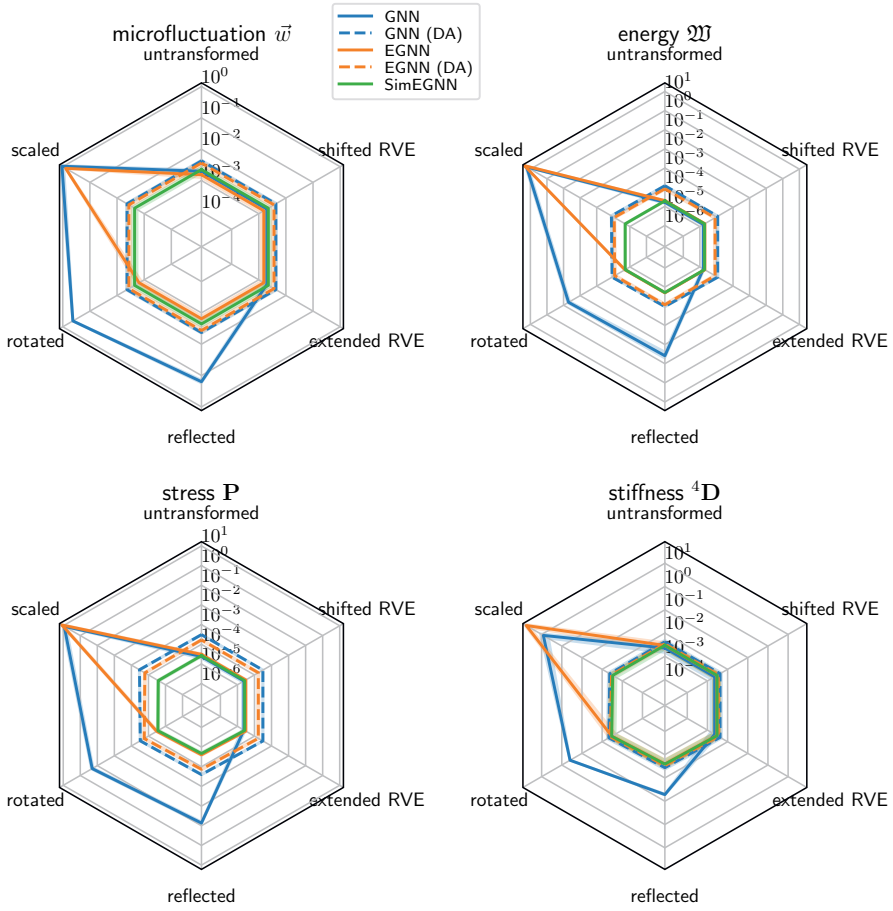


Figure 2.16: Mean fraction of variance unexplained of the four target quantities ($\vec{w}$, $\mathfrak{W}$, $\mathbf{P}$, $\mathbf{D}$) achieved by the investigated models (shown in different line colors and styles) for the six test cases (untransformed, reflected, rotated, shifted RVE, larger RVE, scaled), on the dataset with 4 different diameters. The mean is calculated over the 5 folds from the cross-validation. The standard error is indicated by a shaded area.

sented approach is the need for at least as many message passing steps as the graph diameter, although this can possibly be addressed by, e.g., using pooling strategies and adding long-distance edges to the graph.

2.6 Conclusion

For the development of new hyperelastic, flexible, porous, 2D mechanical metamaterials with complex microstructures, fast simulations of the homogenized response are needed. Conventional finite element simulations are often too slow for this purpose. In this paper, we present a graph neural network architecture called a Similarity-Equivariant Graph Neural Network (SimEGNN) to speed up simulations of these materials. The model is trained on finite element simulations and incorporates all relevant symmetries. The network predicts the final node positions $\vec{x}$ by repositioning the nodes during message passing. From the final graph, the homogenized strain energy density $\mathfrak{W}$, homogenized stress $\mathbf{P}$ and tangent stiffness tensor ${}^4\mathbf{D}$ are predicted, which serve as fast surrogates in computational homogenization schemes.

All necessary in-/equivariances pertinent to computational homogenization are incorporated in the proposed architecture. To achieve this, we use $E(n)$-equivariant graph neural networks [87] as the initial stepping stone and extend them to also (i) respect periodicity (i.e., RVE in-/equivariance), which we achieve by using a periodic graph (i.e., a Pac-Man world geometry) and coordinate-independent updates, (ii) respect scale in-/equivariance, for which we use strains and relative distances instead of distances and normalize the edge vectors $\vec{r}_{ij}$ where necessary, and (iii) equivariantly output higher-order tensors, by using normalized edge vectors $\hat{\vec{r}}_{ij}$ as an overcomplete basis for predicting $\mathbf{P}$ and ${}^4\mathbf{D}$.

We demonstrated that the SimEGNN approximates the results of material simulations (both micro- and macroscopic response) in first-order homogenization with very high accuracy (i.e. FVU $< 10^{-3}$, or $R^2 > 0.999$ in all cases), even for cases with large deformations. Our numerical experiments also confirm that SimEGNN outperforms the architectures with fewer built-in in-/equivariances (base GNN and EGNN). Even though some symmetries can also be trained for in the GNN and EGNN explicitly incorporating them (using data augmentation), training for them is achieved at the expense of a decreased performance, which suggests the only reason the GNN and EGNN manage to get a satisfactory performance at all on the reference case – i.e., relative error $< 10\%$ for all target quantities – is because this dataset can be 'standardized' in the sense one can always choose a unit cell with the same orientation and scale. Such standardization is not possible for more complicated datasets.

This is also the case for varying geometries, as shown by the results on the dataset with 4 different diameters. For a dataset with even more geometries, the SimEGNN should be able to generalize with respect to geometry, i.e., predict the response of unseen geometries. More detailed investigations regarding the generalization capabilities are needed, but are left for future study. The SimEGNN's high accuracy and potential to generalize with respect to geometry make this model highly promising for the future development of new soft porous mechanical metamaterials.

We will also address the main limitation of our current approach, which is the

need for at least as many message passing steps as the graph diameter. The current model can only use one type of vector as input (the edge vectors). To extend the model such that it can handle more vectors and/or tensors as input, we investigate ways to increase model expressiveness, or even higher-order body terms (e.g. [83]), by allowing the features to be similarity-equivariant instead of similarity-invariant (similar to [84, 86]). See [107] for a more in-depth look at graph neural network expressiveness.

We also investigate approaches that can model the bifurcations without needing a perturbation that depends on the buckling mode. The symmetry-breaking involved in the pattern transformation will then cause there to be multiple correct possible outputs, and one cannot simply train on a training dataset of input/output pairs. Therefore, we are investigating approaches based on generative models to predict all possible bifurcations.

2.7 Data Statement

The datasets used in this study are available at https://zenodo.org/records/14229619. This repository contains the raw FEM outputs as well as the preprocessed data in the form of graphs, which can be used as input for the GNNs. We share this data under a CC BY license, such that it can be used for further research.

Appendices

2.A Dataset

Table 2.A.1: Number of trajectories and loadcases in the datasets. The dataset containing only diameter 0.45 is the dataset we do most of the experiments with.

Total	Trajectories	1784
	Load cases	33708
Diameter 0.4	Trajectories	446
	Load cases	8364
Diameter 0.425	Trajectories	446
	Load cases	8404
Diameter 0.45	Trajectories	446
	Load cases	8451
Diameter 0.475	Trajectories	446
	Load cases	8489

Table 2.A.2: Data layout of an input graph for the GNNs. N is the number of nodes, N_e the number of edges.

Quantity	Description	Shape	Data type
Inputs			
x	Node attributes, initially empty	$(N, 0)$	float
edge_index	Indices of nodes that are connected by an edge	$(2, N_e)$	integer
edge_attr	Edge attributes, indicates if edge is a hole boundary edge (-1), or an extra added edge that connects to a different hole (+1)	$(N_e, 1)$	float
pos	Node positions $\vec{x}_i$ in the reference configuration	$(N, 2)$	float
r	Edge vectors $\vec{r}_{ij}$ in the reference configuration	$(N_e, 2)$	float
d	Edge lengths r_{ij} in the reference configuration	$(N_e, 1)$	float
mean_pos	Average position $\vec{x}_{avg}$ of the nodes in the reference configuration	$(1, 2)$	float
F	Applied macroscale deformation gradient tensor $\mathbf{F}$	$(1, 2, 2)$	float
traj	Index indicating to which trajectory this load case belongs	$(1,)$	integer
Target outputs			
y	Final node positions (the target positions)	$(N, 2)$	float
W	Strain energy density $\mathfrak{W}$	$(1,)$	float
P	First Piola-Kirchhoff stress tensor $\mathbf{P}$	$(1, 2, 2)$	float
D	Effective stiffness tensor ${}^4\mathbf{D}$	$(1, 2, 2, 2, 2)$	float

2.B Implementation

All GNN architectures involve message passing steps; recall Figures 2.11a, 2.12a and 2.13a. One can reuse the parameters (i.e., the ones parametrizing ϕ_m, ϕ_e, ϕ_h and ϕ_x in aforementioned Figures) of one message passing step for another step. In other words, one message passing 'layer' can be used for multiple message passing steps. In this work, each network has five distinct message passing layers, out of which the middle three layers are repeated three times each. This provides an adequate balance between the number of parameters and the expressiveness of the network. This results in eleven message passing steps in total. This is close to the graph diameter, which is nine for our input graph, which means that in nine message passing steps, all nodes can communicate with all other nodes in the graph. The message size is set to 64 in all networks and the embedding sizes (edge and node embedding) at 32. ϕ_m, ϕ_e, ϕ_h and ϕ_x each only have one layer and use the softplus activation function. This setup

resulted in the following number of learnable parameters: 76351 for base GNN, 91465 parameters for EGNN, and 91145 for SimEGNN.

The networks are implemented in PyTorch (https://pytorch.org/) and PyTorch Geometric (https://pytorch-geometric.readthedocs.io/en/latest/). For the EGNN and SimEGNN, each message passing step, described by Equations (2.34) to (2.38) and Equations (2.48) and (2.49), can be implemented as two sequential message passing steps. In the first message passing step, node embeddings $\mathbf{h}_i$ are updated using computed messages $\mathbf{m}_{ij}$, which are aggregated into $\mathbf{m}_i$. In the second message passing step, $\vec{x}_i^{\,l}$ is updated using 'messages' $\vec{r}_{ij}^{\,l} \tanh\left(\phi_x^l\left(\mathbf{m}_{ij}^l\right)\right)$ or $\vec{r}_{ij}^{\,l} \tanh\left(\phi_x^l\left(\varepsilon_{ij}\mathbf{m}_{ij}^l\right)\right)$ that are aggregated into $\Delta\vec{x}_i^{\,l}$. Accordingly, the networks are implemented such that each message passing layer contains two different PyTorch Geometric message passing layers that are alternated. (For the GNN, one type of message-passing layer is sufficient.)

For training all learnable parameters, we use the Adam optimizer, with a schedule that reduces the learning rate in steps from 2.5×10^{-4} to 2.5×10^{-6}, see Table 2.B.1. In total, we train for 1620 epochs, with a batch size of 12, which takes 6 to 10 hours per model on an NVIDIA A100-SXM4-40GB GPU. All outputs except the position of the nodes are scaled such that their target values have an average squared value of 1. The mean square error of each of the four outputs (position $\vec{w}$, $\mathfrak{W}$, $\mathbf{P}$ and ${}^4\mathbf{D}$) is used in the loss. The loss terms are weighted proportionally to the inverse of the square value of the target values (which for all loss terms except $\vec{w}$ was 1, because of the aforementioned scaling). We apply gradient clipping at 0.5 to prevent exploding gradients. See Figure 2.B.1 for a plot of the validation loss during training for each model.

Table 2.B.1: Learning rate schedule for training the models.

Epochs	Learning rate
0-119	2.5×10^{-4}
120-719	1.0×10^{-4}
720-1079	5.0×10^{-5}
1080-1439	2.5×10^{-5}
1440-1499	1.0×10^{-5}
1500-1559	5.0×10^{-6}
1560-1619	2.5×10^{-6}

2.C Full results

Tables 2.C.1, 2.C.2, 2.C.3, 2.C.4 show, for each target quantity respectively, the fraction of variance unexplained (FVU) obtained by each neural network architecture for each test case. Tables 2.C.5, 2.C.6, 2.C.7, 2.C.8 show the relative error, which we define as the mean Frobenius norm of the error divided by the mean Frobenius norm

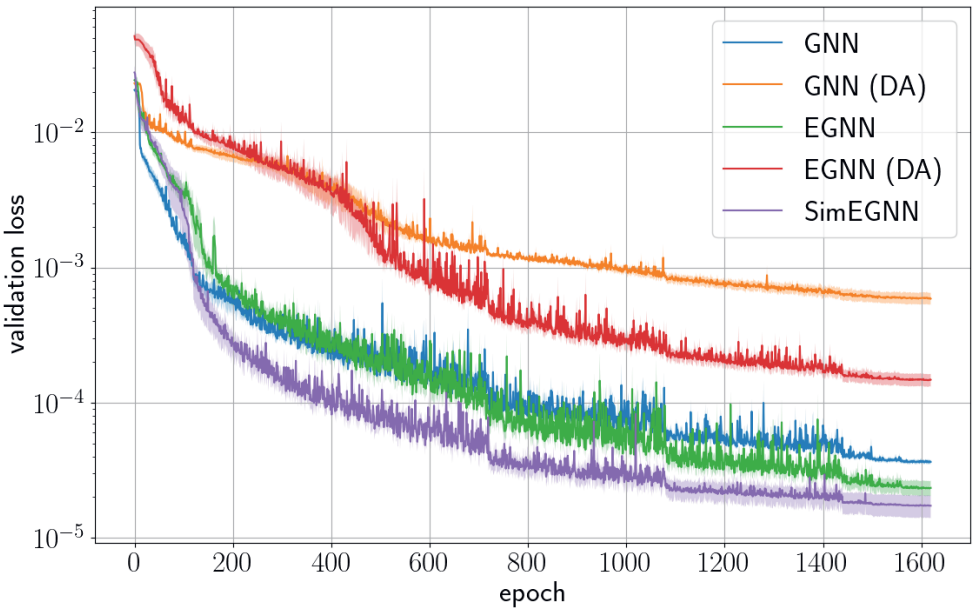


Figure 2.B.1: The mean validation loss during training for each architecture. The mean is calculated over the 5 models resulting from the cross-validation. The standard error is indicated with a shaded area.

of the target

$$\frac{\frac{1}{N}\sum_i^N \left\|\mathbf{A}_{\text{pred},i} - \mathbf{A}_{\text{target},i}\right\|_F}{\frac{1}{N}\sum_i^N \left\|\mathbf{A}_{\text{target},i}\right\|_F} \times 100\%, \tag{2.52}$$

where $\mathbf{A}_{\text{pred},i}$ is some predicted tensor, $\mathbf{A}_{\text{target},i}$ is the ground truth value of this tensor, $\|...\|_F$ is the Frobenius norm and N is the number of targets in the validation data, which for the global quantities $\mathfrak{W}$, $\mathbf{P}$, ${}^4\mathbf{D}$ is equal to the number of graphs, but for the local quantity $\vec{w}$ is equal to the total number of nodes (number of graphs $\times$ number of nodes per graph). For a scalar quantity such as $\mathfrak{W}$, this reduces to the root mean squared error divided by the root mean square target value.

In addition to Figure 2.15 in Section 2.5, Figures 2.C.1, 2.C.2, 2.C.3, 2.C.4 and 2.C.5 show for all test cases a comparison of predicted deformations from all of the networks against FEM outputs. Each figure shows a different loading $\mathbf{F}$, such that all bifurcation patterns are included.

Table 2.C.1: Mean FVU of the microfluctuation $\vec{w}$

	GNN	GNN (DA)	EGNN	EGNN (DA)	SimEGNN
untransformed	0.0034 ± 0.0003	0.020 ± 0.001	0.0019 ± 0.0004	0.011 ± 0.002	$\mathbf{0.0016 \pm 0.0004}$
shifted RVE	0.0034 ± 0.0003	0.020 ± 0.001	0.0019 ± 0.0004	0.011 ± 0.002	$\mathbf{0.0016 \pm 0.0004}$
extended RVE	0.0034 ± 0.0003	0.020 ± 0.001	0.0019 ± 0.0004	0.011 ± 0.002	$\mathbf{0.0016 \pm 0.0004}$
reflected	0.21 ± 0.03	0.020 ± 0.001	0.0019 ± 0.0004	0.011 ± 0.002	$\mathbf{0.0016 \pm 0.0004}$
rotated	0.82 ± 0.04	0.019 ± 0.001	0.0019 ± 0.0004	0.011 ± 0.002	$\mathbf{0.0016 \pm 0.0004}$
scaled	0.93 ± 0.11	0.019 ± 0.001	1.8 ± 0.8	0.011 ± 0.002	$\mathbf{0.0016 \pm 0.0004}$

Table 2.C.2: Mean FVU of the strain energy density $\mathfrak{W}$

	GNN	GNN (DA) ($\times 10^{-4}$)	EGNN	EGNN (DA) ($\times 10^{-4}$)	SimEGNN ($\times 10^{-5}$)
untransformed	$(4.50 \pm 0.87)\ 10^{-5}$	9.34 ± 1.44	$(4.16 \pm 0.57)\ 10^{-5}$	2.30 ± 0.46	$\mathbf{1.51 \pm 0.25}$
shifted RVE	$(4.50 \pm 0.87)\ 10^{-5}$	9.34 ± 1.44	$(4.16 \pm 0.57)\ 10^{-5}$	2.30 ± 0.46	$\mathbf{1.51 \pm 0.25}$
extended RVE	$(4.50 \pm 0.87)\ 10^{-5}$	9.34 ± 1.44	$(4.16 \pm 0.57)\ 10^{-5}$	2.30 ± 0.46	$\mathbf{1.51 \pm 0.25}$
reflected	0.051 ± 0.035	9.70 ± 1.48	$(4.16 \pm 0.57)\ 10^{-5}$	2.30 ± 0.46	$\mathbf{1.51 \pm 0.25}$
rotated	0.069 ± 0.021	8.45 ± 1.40	$(4.16 \pm 0.57)\ 10^{-5}$	2.30 ± 0.46	$\mathbf{1.51 \pm 0.25}$
scaled	48 ± 11	9.87 ± 2.12	$(1.2 \pm 0.5)\ 10^{2}$	2.55 ± 0.48	$\mathbf{1.51 \pm 0.25}$

Table 2.C.3: Mean FVU of the first Piola-Kirchhoff stress tensor $\mathbf{P}$

	GNN	GNN (DA) ($\times 10^{-3}$)	EGNN	EGNN (DA) ($\times 10^{-4}$)	SimEGNN ($\times 10^{-5}$)
untransformed	$(4.85 \pm 0.79) \times 10^{-5}$	5.5 ± 0.3	$(4.79 \pm 0.87) \times 10^{-5}$	5.39 ± 1.17	$\mathbf{2.90 \pm 0.48}$
shifted RVE	$(4.85 \pm 0.79) \times 10^{-5}$	5.5 ± 0.3	$(4.79 \pm 0.87) \times 10^{-5}$	5.39 ± 1.17	$\mathbf{2.90 \pm 0.48}$
extended RVE	$(4.85 \pm 0.79) \times 10^{-5}$	5.5 ± 0.3	$(4.79 \pm 0.87) \times 10^{-5}$	5.39 ± 1.17	$\mathbf{2.90 \pm 0.48}$
reflected	0.093 ± 0.016	4.1 ± 0.7	$(3.74 \pm 0.74) \times 10^{-5}$	4.22 ± 0.99	$\mathbf{2.24 \pm 0.37}$
rotated	0.19 ± 0.01	6.5 ± 0.2	$(5.24 \pm 0.99) \times 10^{-5}$	5.90 ± 1.31	$\mathbf{3.16 \pm 0.52}$
scaled	14 ± 3	5.5 ± 0.3	40 ± 11	5.33 ± 1.16	$\mathbf{2.90 \pm 0.48}$

Table 2.C.4: Mean FVU of the stiffness tensor $\mathbf{D}$

	GNN	GNN (DA) ($\times 10^{-2}$)	EGNN	EGNN (DA) ($\times 10^{-3}$)	SimEGNN ($\times 10^{-4}$)
untransformed	$(6.60 \pm 0.89) \times 10^{-4}$	5.3 ± 1.0	$(7.91 \pm 1.15) \times 10^{-4}$	5.7 ± 0.8	$\mathbf{4.19 \pm 1.27}$
shifted RVE	$(6.60 \pm 0.89) \times 10^{-4}$	5.3 ± 1.0	$(7.91 \pm 1.15) \times 10^{-4}$	5.7 ± 0.8	$\mathbf{4.19 \pm 1.27}$
extended RVE	$(6.60 \pm 0.89) \times 10^{-4}$	5.3 ± 1.0	$(7.91 \pm 1.15) \times 10^{-4}$	5.7 ± 0.8	$\mathbf{4.19 \pm 1.27}$
reflected	0.071 ± 0.006	4.4 ± 1.3	$(6.45 \pm 1.00) \times 10^{-4}$	4.7 ± 0.7	$\mathbf{3.38 \pm 1.00}$
rotated	0.55 ± 0.03	5.7 ± 1.4	$(9.80 \pm 1.53) \times 10^{-4}$	7.1 ± 1.1	$\mathbf{5.15 \pm 1.53}$
scaled	1.9 ± 0.4	5.1 ± 0.9	$(1.02 \pm 1.01)\ 10^{4}$	5.5 ± 0.8	$\mathbf{4.19 \pm 1.27}$

Table 2.C.5: Mean relative error in percentage of the microfluctuation $\vec{w}$

	GNN	GNN (DA)	EGNN	EGNN (DA)	SimEGNN
untransformed	5.4 ± 0.2	14 ± 0	3.7 ± 0.4	10 ± 1	$\mathbf{3.3 \pm 0.4}$
shifted RVE	5.4 ± 0.2	14 ± 0	3.7 ± 0.4	10 ± 1	$\mathbf{3.3 \pm 0.4}$
extended RVE	5.4 ± 0.2	14 ± 0	3.7 ± 0.4	10 ± 1	$\mathbf{3.3 \pm 0.4}$
reflected	37 ± 2	14 ± 0	3.7 ± 0.4	10 ± 1	$\mathbf{3.3 \pm 0.4}$
rotated	90 ± 3	14 ± 0	3.9 ± 0.4	11 ± 1	$\mathbf{3.4 \pm 0.4}$
scaled	$(1.0 \pm 0.1) \times 10^2$	14 ± 0	$(1.4 \pm 0.3) \times 10^2$	9.9 ± 0.6	$\mathbf{3.3 \pm 0.4}$

Table 2.C.6: Mean relative error in percentage of the strain energy density $\mathfrak{W}$

	GNN	GNN (DA)	EGNN	EGNN (DA)	SimEGNN
untransformed	0.58 ± 0.04	3.0 ± 0.3	0.60 ± 0.05	1.5 ± 0.2	$\mathbf{0.36 \pm 0.04}$
shifted RVE	0.58 ± 0.04	3.0 ± 0.3	0.60 ± 0.05	1.5 ± 0.2	$\mathbf{0.36 \pm 0.04}$
extended RVE	0.58 ± 0.04	3.0 ± 0.3	0.60 ± 0.05	1.5 ± 0.2	$\mathbf{0.36 \pm 0.04}$
reflected	14 ± 5	3.2 ± 0.4	0.60 ± 0.05	1.5 ± 0.2	$\mathbf{0.36 \pm 0.04}$
rotated	24 ± 3	3.0 ± 0.4	0.60 ± 0.05	1.5 ± 0.2	$\mathbf{0.36 \pm 0.04}$
scaled	$(7.4 \pm 0.9) \times 10^2$	3.0 ± 0.4	$(1.10 \pm 0.32) \times 10^3$	1.6 ± 0.2	$\mathbf{0.36 \pm 0.04}$

Table 2.C.7: Mean relative error in percentage of the first Piola-Kirchhoff stress tensor $\mathbf{P}$

	GNN	GNN (DA)	EGNN	EGNN (DA)	SimEGNN
untransformed	0.64 ± 0.04	7.3 ± 0.3	0.65 ± 0.07	2.3 ± 0.3	$\mathbf{0.48 \pm 0.04}$
shifted RVE	0.64 ± 0.04	7.3 ± 0.3	0.65 ± 0.07	2.3 ± 0.3	$\mathbf{0.48 \pm 0.04}$
extended RVE	0.64 ± 0.04	7.3 ± 0.3	0.65 ± 0.07	2.3 ± 0.3	$\mathbf{0.48 \pm 0.04}$
reflected	23 ± 2	7.0 ± 0.8	0.65 ± 0.07	2.3 ± 0.3	$\mathbf{0.48 \pm 0.04}$
rotated	43 ± 2	7.5 ± 0.3	0.65 ± 0.07	2.3 ± 0.3	$\mathbf{0.48 \pm 0.04}$
scaled	$(3.8 \pm 0.4) \times 10^2$	7.2 ± 0.3	$(5.6 \pm 0.7) \times 10^2$	2.2 ± 0.3	$\mathbf{0.48 \pm 0.04}$

Table 2.C.8: Mean relative error in percentage of the stiffness tensor $\mathbf{D}$

	GNN	GNN (DA)	EGNN	EGNN (DA)	SimEGNN
untransformed	1.3 ± 0.1	19 ± 2	1.5 ± 0.1	5.0 ± 0.4	$\mathbf{0.97 \pm 0.10}$
shifted RVE	1.3 ± 0.1	19 ± 2	1.5 ± 0.1	5.0 ± 0.4	$\mathbf{0.97 \pm 0.10}$
extended RVE	1.3 ± 0.1	19 ± 2	1.5 ± 0.1	5.0 ± 0.4	$\mathbf{0.97 \pm 0.10}$
reflected	17 ± 1	18 ± 3	1.5 ± 0.1	5.0 ± 0.4	$\mathbf{0.97 \pm 0.10}$
rotated	58 ± 2	17 ± 2	1.5 ± 0.1	5.0 ± 0.4	$\mathbf{0.97 \pm 0.10}$
scaled	$(1.2 \pm 0.1) \times 10^2$	18 ± 2	$(1.36 \pm 1.08) \times 10^3$	4.9 ± 0.4	$\mathbf{0.97 \pm 0.10}$

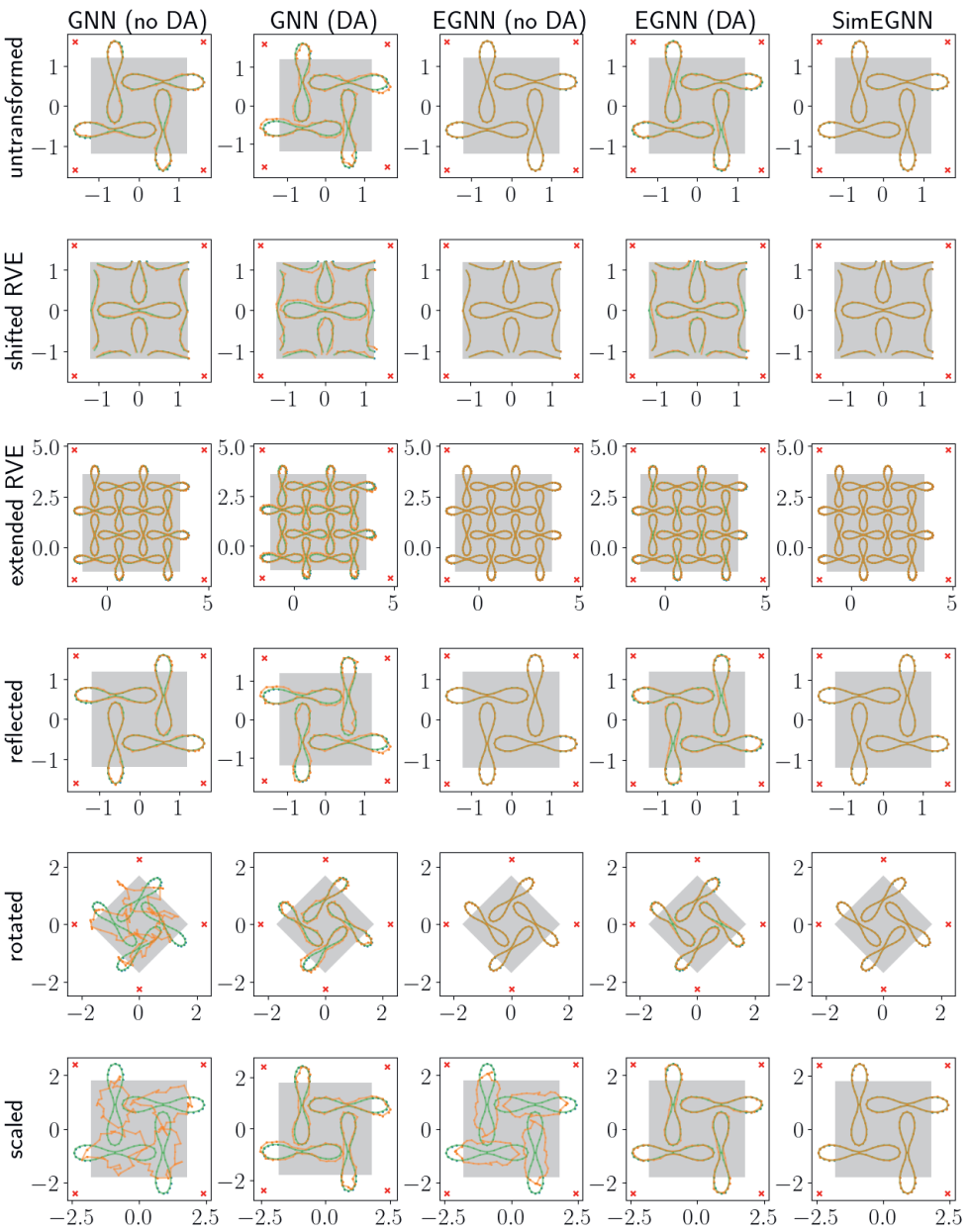


Figure 2.C.1: Predicted deformation of the hole boundaries (orange) compared to the FEM ground truth (green), for $\mathbf{F} = \begin{bmatrix} 0.75 & 0 \\ 0 & 0.75 \end{bmatrix}$ (biaxial compression, resulting in a rotational pattern). The gray square indicates how $\mathbf{F}$ deforms the square RVE through an affine transformation. The red crosses indicate the original positions of the corners.

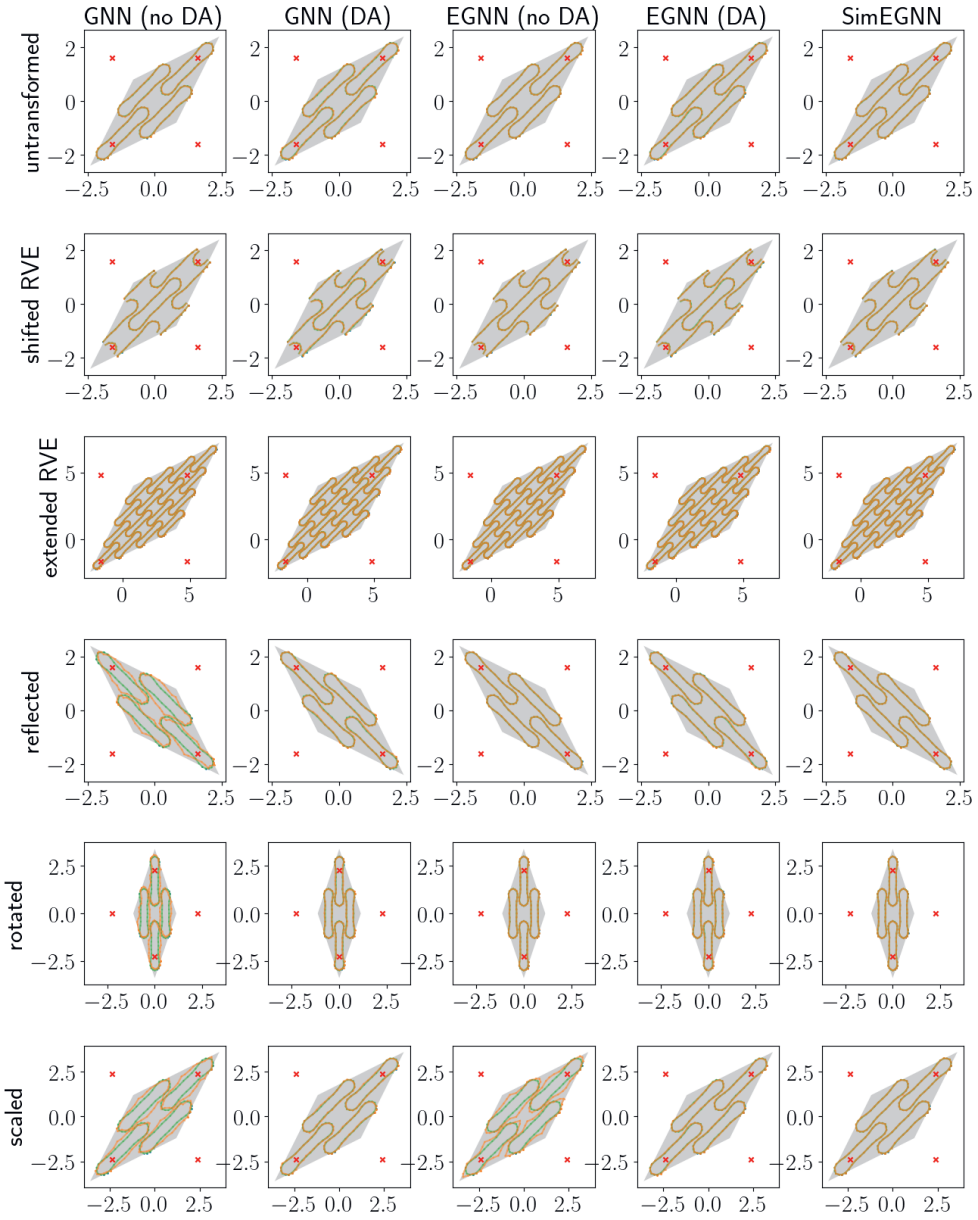


Figure 2.C.2: Model-predicted deformation (orange) compared to the FEM ground truth (green), for $\mathbf{F} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$ (pure shear). The gray square indicates how $\mathbf{F}$ deforms the square RVE through an affine transformation. The red crosses indicate the original positions of the corners.

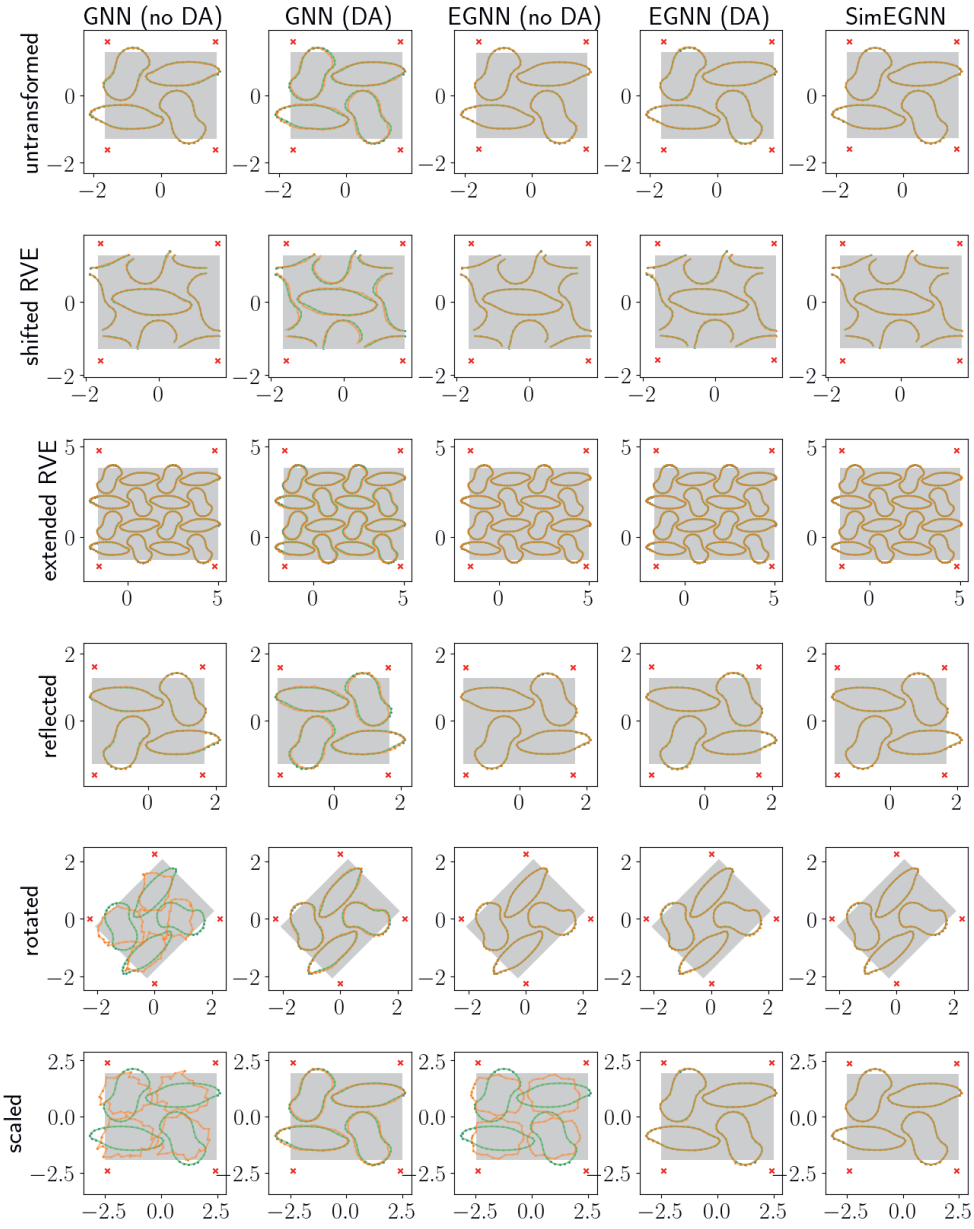


Figure 2.C.3: Model-predicted deformation (orange) compared to the FEM ground truth (green), for $\mathbf{F} = \begin{bmatrix} 1.04 & 0 \\ 0 & 0.8 \end{bmatrix}$ (combination of tension and compression, resulting in left/right bifurcation as well as rotational bifurcation). The gray square indicates how $\mathbf{F}$ deforms the square RVE through an affine transformation. The red crosses indicate the original positions of the corners.

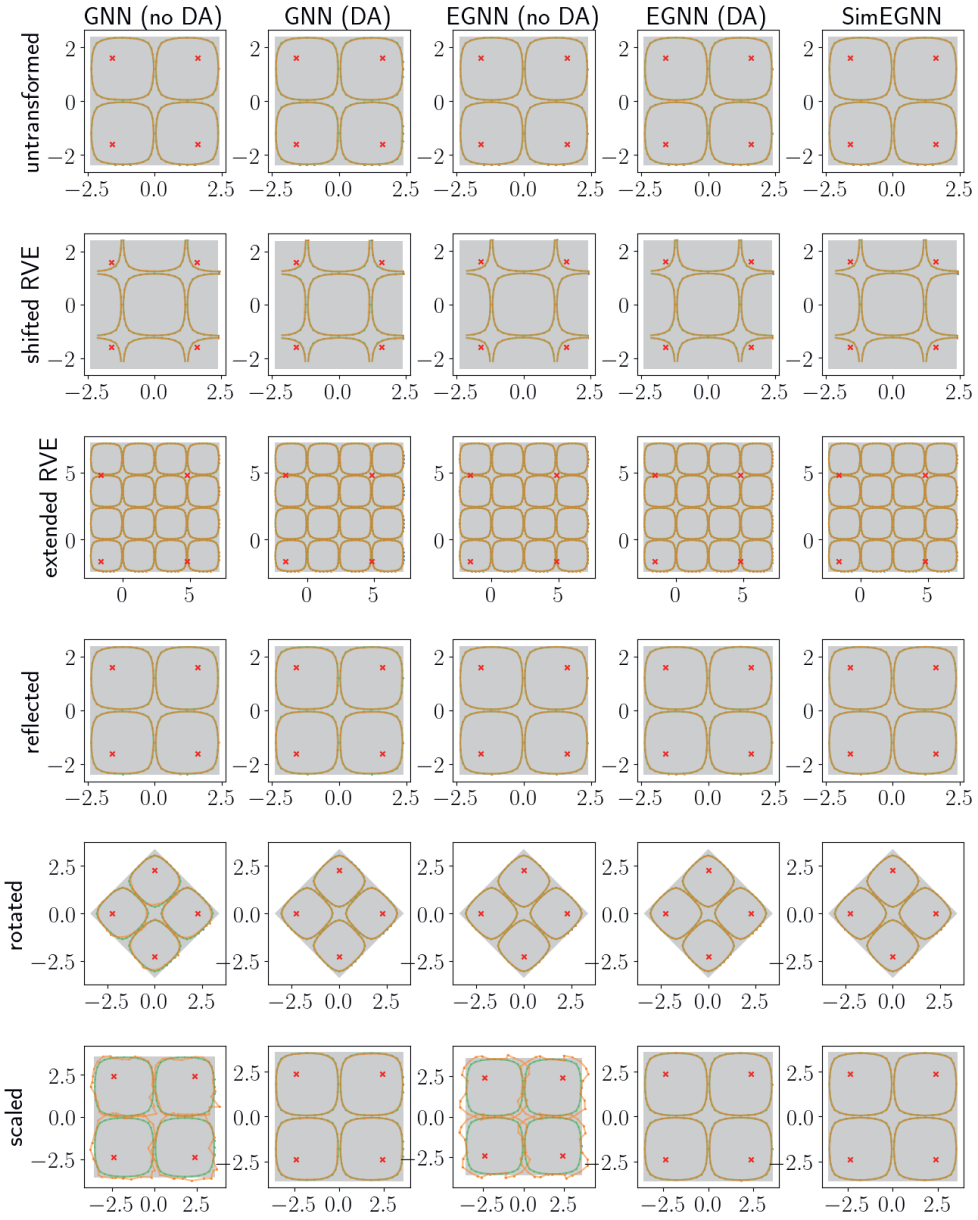


Figure 2.C.4: Model-predicted deformation (orange) compared to the FEM ground truth (green), for $\mathbf{F} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}$ (biaxial tension). The gray square indicates how $\mathbf{F}$ deforms the square RVE through an affine transformation. The red crosses indicate the original positions of the corners.

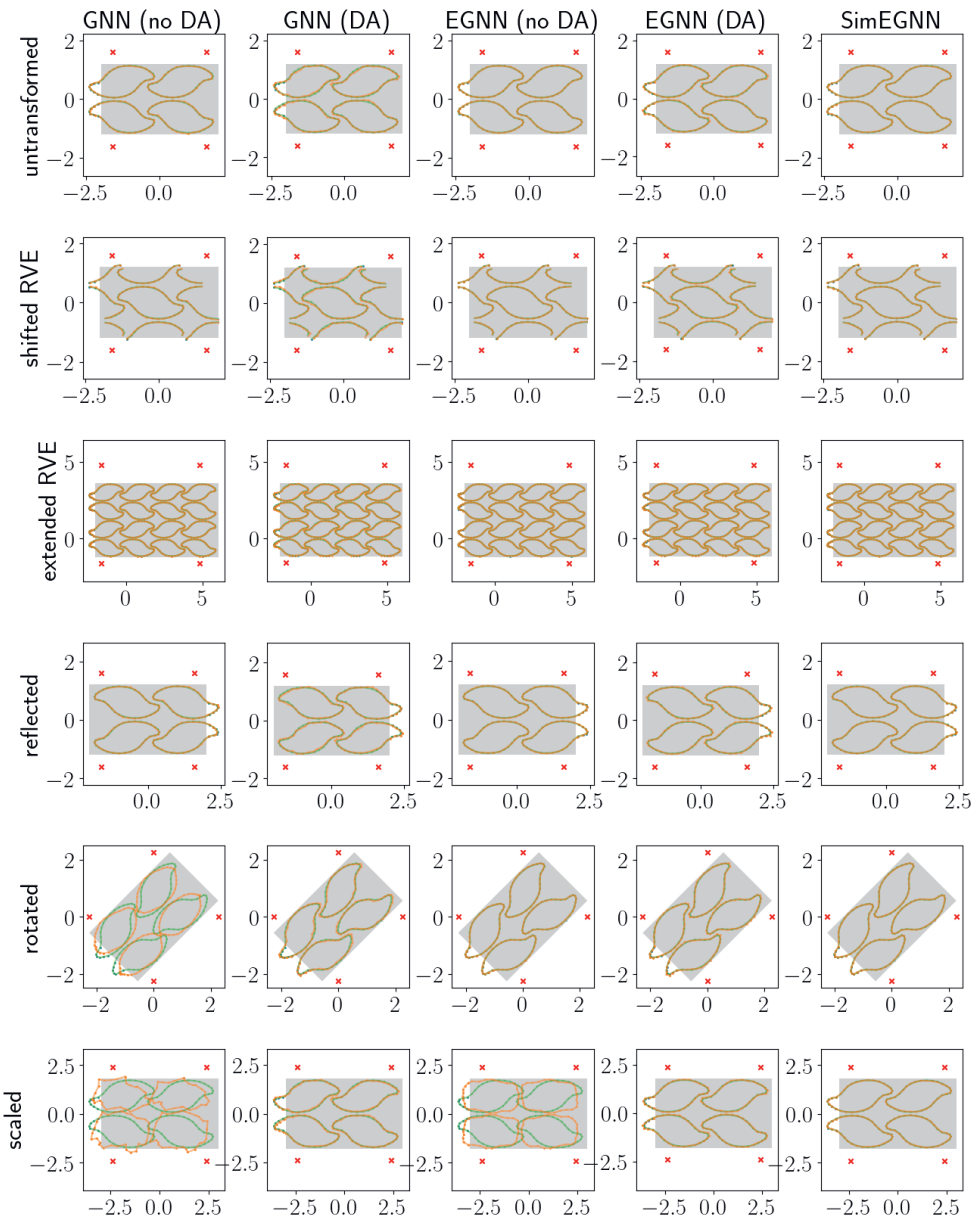


Figure 2.C.5: Model-predicted deformation (orange) compared to the FEM ground truth (green), for $\mathbf{F} = \begin{bmatrix} 1.25 & 0 \\ 0 & 0.75 \end{bmatrix}$ (combination of tension and compression, resulting in left/right bifurcation). The gray square indicates how $\mathbf{F}$ deforms the square RVE through an affine transformation. The red crosses indicate the original positions of the corners.

Bibliography

[1] K. Bertoldi, V. Vitelli, J. Christensen, M. Van Hecke, Flexible mechanical metamaterials (oct 2017). doi:10.1038/natrevmats.2017.66.
URL https://www.nature.com/articles/natrevmats201766

[2] X. Yu, J. Zhou, H. Liang, Z. Jiang, L. Wu, Mechanical metamaterials associated with stiffness, rigidity and compressibility: A brief review, Progress in Materials Science 94 (2018) 114–173. doi:10.1016/j.pmatsci.2017.12.003.
URL https://doi.org/10.1016/j.pmatsci.2017.12.003

[3] S. Babaee, J. Shim, J. C. Weaver, E. R. Chen, N. Patel, K. Bertoldi, 3D soft metamaterials with negative poisson's ratio, Advanced Materials 25 (36) (2013) 5044–5049. doi:10.1002/adma.201301986.
URL https://onlinelibrary.wiley.com/doi/10.1002/adma.201301986

[4] Z. G. Nicolaou, A. E. Motter, Mechanical metamaterials with negative compressibility transitions, Nature Materials 11 (7) (2012) 608–613. arXiv:1207.2185, doi:10.1038/nmat3331.
URL https://www.nature.com/articles/nmat3331

[5] K. Bertoldi, M. C. Boyce, S. Deschanel, S. M. Prange, T. Mullin, Mechanics of deformation-triggered pattern transformations and superelastic behavior in periodic elastomeric structures, Journal of the Mechanics and Physics of Solids 56 (8) (2008) 2642–2668. doi:10.1016/j.jmps.2008.03.006.
URL http://arxiv.org/abs/2301.11355

[6] J. T. Overvelde, K. Bertoldi, Relating pore shape to the non-linear response of periodic elastomeric structures, Journal of the Mechanics and Physics of Solids 64 (1) (2014) 351–366. doi:10.1016/j.jmps.2013.11.014.
URL http://dx.doi.org/10.1016/j.jmps.2013.11.014

[7] D. Yang, B. Mosadegh, A. Ainla, B. Lee, F. Khashai, Z. Suo, K. Bertoldi, G. M. Whitesides, Buckling of Elastomeric Beams Enables Actuation of Soft Machines, Advanced Materials 27 (41) (2015) 6323–6327. doi:10.1002/adma.201503188.
URL https://onlinelibrary.wiley.com/doi/full/10.1002/adma.201503188https://onlinelibrary.wiley.com/doi/abs/10.1002/adma.201503188https://onlinelibrary.wiley.com/doi/10.1002/adma.201503188

[8] Y. Kim, H. Yuk, R. Zhao, S. A. Chester, X. Zhao, Printing ferromagnetic domains for untethered fast-transforming soft materials, Nature 558 (7709) (2018) 274–279. doi:10.1038/s41586-018-0185-0.

[9] A. Guell Izard, L. Valdevit, Magnetoelastic Metamaterials for Energy Dissipation and Wave Filtering, Advanced Engineering Materials 22 (2) (2020) 1901019. doi:10.1002/adem.201901019.
URL https://onlinelibrary.wiley.com/doi/full/10.1002/adem.201901019https://onlinelibrary.wiley.com/doi/abs/10.1002/adem.201901019https://onlinelibrary.wiley.com/doi/10.1002/adem.201901019

[10] S. Ning, D. Chu, H. Jiang, F. Yang, Z. Liu, Z. Zhuang, The role of material and geometric nonlinearities and damping effects in designing mechanically tunable acoustic metamaterials, International Journal of Mechanical Sciences 197 (January) (2021) 106299. doi:10.1016/j.ijmecsci.2021.106299.
URL https://doi.org/10.1016/j.ijmecsci.2021.106299

[11] S. M. Montgomery, S. Wu, X. Kuang, C. D. Armstrong, C. Zemelka, Q. Ze, R. Zhang, R. Zhao, H. J. Qi, Magneto-Mechanical Metamaterials with Widely Tunable Mechanical Properties and Acoustic Bandgaps, Advanced Functional Materials 31 (3) (2021) 2005319. doi:10.1002/adfm.202005319.
URL https://onlinelibrary.wiley.com/doi/full/10.1002/adfm.202005319https://onlinelibrary.wiley.com/doi/abs/10.1002/adfm.202005319https://onlinelibrary.wiley.com/doi/10.1002/adfm.202005319

[12] L. Wu, Y. Wang, K. Chuang, F. Wu, Q. Wang, W. Lin, H. Jiang, A brief review of dynamic mechanical metamaterials for mechanical energy manipulation, Materials Today 44 (April) (2021) 168–193. doi:10.1016/j.mattod.2020.10.006.
URL https://doi.org/10.1016/j.mattod.2020.10.006

[13] S. Terryn, J. Brancart, D. Lefeber, G. Van Assche, B. Vanderborght, Self-healing soft pneumatic robots, Science Robotics 2 (9) (2017) 16. doi:10.1126/scirobotics.aan4268.
URL https://www.science.org/doi/10.1126/scirobotics.aan4268

[14] Y. Kim, G. A. Parada, S. Liu, X. Zhao, Ferromagnetic soft continuum robots, Science Robotics 4 (33) (2019) 7329. doi:10.1126/SCIROBOTICS.AAX7329.
URL https://www.science.org/doi/10.1126/scirobotics.aax7329

[15] U. Veerabagu, H. Palza, F. Quero, Review: Auxetic Polymer-Based Mechanical Metamaterials for Biomedical Applications (jul 2022). doi:10.1021/acsbiomaterials.2c00109.
URL https://pubs.acs.org/doi/full/10.1021/acsbiomaterials.2c00109

[16] Y. Jiang, Q. Wang, Highly-stretchable 3D-architected Mechanical Metamaterials, Scientific Reports 6 (1) (2016) 1–11. doi:10.1038/srep34147.
URL https://www.nature.com/articles/srep34147

[17] L. D'Alessandro, E. Belloni, R. Ardito, A. Corigliano, F. Braghin, Modeling and experimental verification of an ultra-wide bandgap in 3D phononic crystal, Applied Physics Letters 109 (22) (2016) 221907. doi:10.1063/1.4971290.
URL http://dx.doi.org/10.1063/1.4971290http://aip.scitation.org/toc/apl/109/22

[18] Q. Shi, K. Yu, X. Kuang, X. Mu, C. K. Dunn, M. L. Dunn, T. Wang, H. Jerry Qi, Recyclable 3D printing of vitrimer epoxy, Materials Horizons 4 (4) (2017) 598–607. doi:10.1039/c7mh00043j.
URL https://pubs.rsc.org/en/content/articlehtml/2017/mh/c7mh00043jhttps://pubs.rsc.org/en/content/articlelanding/2017/mh/c7mh00043j

[19] X. Ren, J. Shen, P. Tran, T. D. Ngo, Y. M. Xie, Design and characterisation of a tuneable 3D buckling-induced auxetic metamaterial, Materials and Design 139 (2018) 336–342. doi:10.1016/j.matdes.2017.11.025.

[20] S. M. Montgomery, X. Kuang, C. D. Armstrong, H. J. Qi, Recent advances in additive manufacturing of active mechanical metamaterials, Current Opinion in Solid State and Materials Science 24 (5) (2020) 100869. doi:10.1016/j.cossms.2020.100869.

[21] M. Bodaghi, A. Serjouei, A. Zolfagharian, M. Fotouhi, H. Rahman, D. Durand, Reversible energy absorbing meta-sandwiches by FDM 4D printing, International Journal of Mechanical Sciences 173 (2020) 105451. doi:10.1016/j.ijmecsci.2020.105451.

[22] Z. P. Wang, L. H. Poh, J. Dirrenberger, Y. Zhu, S. Forest, Isogeometric shape optimization of smoothed petal auxetic structures via computational periodic homogenization, Computer Methods in Applied Mechanics and Engineering 323 (2017) 250–271. doi:10.1016/j.cma.2017.05.013.
URL www.sciencedirect.comwww.elsevier.com/locate/cma

[23] E. Medina, C. H. Rycroft, K. Bertoldi, Nonlinear shape optimization of flexible mechanical metamaterials, Extreme Mechanics Letters 61 (2023) 102015. doi:10.1016/j.eml.2023.102015.
URL https://doi.org/10.1016/j.eml.2023.102015

[24] M. P. Bendsøe, N. Kikuchi, Generating optimal topologies in structural design using a homogenization method, Computer Methods in Applied Mechanics and Engineering 71 (2) (1988) 197–224. doi:10.1016/0045-7825(88)90086-2.

[25] M. P. Bendsøe, O. Sigmund, Topology optimization: Theory, methods and applications, 2004.
URL https://link.springer.com/book/10.1007/978-3-662-05086-6#toc

[26] O. Sigmund, Materials with prescribed constitutive parameters: An inverse homogenization problem, International Journal of Solids and Structures 31 (17) (1994) 2313–2329. doi:10.1016/0020-7683(94)90154-6.

[27] E. Andreassen, B. S. Lazarov, O. Sigmund, Design of manufacturable 3D extremal elastic microstructure, Mechanics of Materials 69 (1) (2014) 1–10. doi:10.1016/j.mechmat.2013.09.018.

[28] Q. Chen, X. Zhang, B. Zhu, Design of buckling-induced mechanical metamaterials for energy absorption using topology optimization, Structural and Multidisciplinary Optimization 58 (4) (2018) 1395–1410. doi:10.1007/s00158-018-1970-y.
URL https://link.springer.com/article/10.1007/s00158-018-1970-y

[29] C. R. Thomsen, F. Wang, O. Sigmund, Buckling strength topology optimization of 2D periodic materials based on linearized bifurcation analysis, Computer Methods in Applied Mechanics and Engineering 339 (2018) 115–136. doi:10.1016/j.cma.2018.04.031.
URL www.sciencedirect.comwww.elsevier.com/locate/cma

[30] F. Wang, O. Sigmund, 3D architected isotropic materials with tunable stiffness and buckling strength, Journal of the Mechanics and Physics of Solids 152 (2021) 104415. arXiv:2012.01359, doi:10.1016/j.jmps.2021.104415.

[31] T. Xue, S. Mao, Mapped shape optimization method for the rational design of cellular mechanical metamaterials under large deformation, International Journal for Numerical Methods in Engineering 123 (10) (2022) 2357–2380. doi:10.1002/nme.6941.
URL https://onlinelibrary.wiley.com/doi/full/10.1002/nme.6941https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.6941https://onlinelibrary.wiley.com/doi/10.1002/nme.6941

[32] R. Hill, Elastic properties of reinforced solids: Some theoretical principles, Journal of the Mechanics and Physics of Solids 11 (5) (1963) 357–372. doi:10.1016/0022-5096(63)90036-X.

[33] L. J. Walpole, The determination of the elastic field of an ellipsoidal inclusion in an anisotronic medium, Mathematical Proceedings of the Cambridge Philosophical Society 81 (2) (1977) 283–289. doi:10.1017/S0305004100053366.
URL https://www.jstor.org/stable/100095

[34] R. Hill, A self-consistent mechanics of composite materials, Journal of the Mechanics and Physics of Solids 13 (4) (1965) 213–222. doi:10.1016/0022-5096(65)90010-4.

[35] T. Mori, K. Tanaka, Average stress in matrix and average elastic energy of materials with misfitting inclusions, Acta Metallurgica 21 (5) (1973) 571–574. doi:10.1016/0001-6160(73)90064-3.

[36] D. J. Lucia, P. S. Beran, W. A. Silva, Reduced-order modeling: New approaches for computational physics, Progress in Aerospace Sciences 40 (1-2) (2004) 51–117. doi:10.1016/j.paerosci.2003.12.001.

[37] G. Kerschen, J.-C. Golinval, A. F. Vakakis, L. A. Bergman, The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview, Tech. rep. (2005).

[38] F. El Halabi, D. González, A. Chico, M. Doblaré, FE2 multiscale in linear elasticity based on parametrized microscale models using proper generalized decomposition, Computer Methods in Applied Mechanics and Engineering 257 (2013) 183–202. doi:10.1016/j.cma.2013.01.011.
URL http://dx.doi.org/10.1016/j.cma.2013.01.011

[39] J. Yvonnet, Q. C. He, The reduced model multiscale method (R3M) for the non-linear homogenization of hyperelastic media at finite strains, Journal of Computational Physics 223 (1) (2007) 341–368. doi:10.1016/j.jcp.2006.09.019.

[40] J. A. Hernández, J. Oliver, A. E. Huespe, M. A. Caicedo, J. C. Cante, High-performance model reduction techniques in computational multiscale homogenization, Computer Methods in Applied Mechanics and Engineering 276 (2014) 149–189. doi:10.1016/j.cma.2014.03.011.
URL http://dx.doi.org/10.1016/j.cma.2014.03.011www.sciencedirect.comwww.elsevier.com/locate/cma

[41] C. Gogu, Improving the efficiency of large scale topology optimization through on-the-fly reduced order model construction, International Journal for Numerical Methods in Engineering 101 (4) (2014) 281–304. doi:10.1002/nme.4797.
URL https://onlinelibrary.wiley.com/doi/full/10.1002/nme.4797https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.4797https://onlinelibrary.wiley.com/doi/10.1002/nme.4797

[42] M. Guo, J. S. Hesthaven, Reduced order modeling for nonlinear structural analysis using Gaussian process regression, Computer Methods in Applied Mechanics and Engineering 341 (2018) 807–826. doi:10.1016/j.cma.2018.07.017.

[43] T. Guo, O. Rokoš, K. Veroy, Learning constitutive models from microstructural simulations via a non-intrusive reduced basis method, Computer Methods in Applied Mechanics and Engineering 384 (2021) 113924. doi:10.1016/j.cma.2021.113924.
URL https://linkinghub.elsevier.com/retrieve/pii/S0045782521002619

[44] T. Guo, F. A. Silva, O. Rokoš, K. Veroy, Learning constitutive models from microstructural simulations via a non-intrusive reduced basis method: Extension to geometrical parameterizations, Computer Methods in Applied Mechanics and Engineering 401 (2022) 115636. doi:10.1016/j.cma.2022.115636.

[45] T. Kirchdoerfer, M. Ortiz, Data-driven computational mechanics, Computer Methods in Applied Mechanics and Engineering 304 (2016) 81–101. arXiv:1510.04232, doi:10.1016/j.cma.2016.02.001.

[46] Z. Liu, M. A. Bessa, W. K. Liu, Self-consistent clustering analysis: An efficient multi-scale scheme for inelastic heterogeneous materials, Computer Methods in Applied Mechanics and Engineering 306 (2016) 319–341. doi:10.1016/j.cma.2016.04.004.
URL http://dx.doi.org/10.1016/j.cma.2016.04.004

[47] G. Liang, K. Chandrashekhara, Neural network based constitutive model for elastomeric foams, Engineering Structures 30 (7) (2008) 2002–2011. doi:10.1016/j.engstruct.2007.12.021.

[48] B. A. Le, J. Yvonnet, Q. He, Computational homogenization of nonlinear elastic materials using neural networks, International Journal for Numerical Methods in Engineering 104 (12) (2015) 1061–1084. doi:10.1002/nme.4953.
URL http://onlinelibrary.wiley.com/doi/10.1002/nme.3279/fullhttps://onlinelibrary.wiley.com/doi/10.1002/nme.4953

[49] K. Linka, M. Hillgärtner, K. P. Abdolazizi, R. C. Aydin, M. Itskov, C. J. Cyron, Constitutive artificial neural networks: A fast and general approach to predictive data-driven constitutive modeling by deep learning, Journal of Computational Physics 429 (2021) 110010. doi:10.1016/j.jcp.2020.110010.
URL https://doi.org/10.1016/j.jcp.2020.110010https://linkinghub.elsevier.com/retrieve/pii/S0021999120307841

[50] T. Xue, S. Adriaenssens, S. Mao, Learning the nonlinear dynamics of soft mechanical metamaterials with graph networks (2022). arXiv:2202.13775.
URL https://www.researchgate.net/publication/358885333http://arxiv.org/abs/2202.13775

[51] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control, Signals, and Systems 2 (4) (1989) 303–314. doi:10.1007/BF02551274.
URL https://link.springer.com/article/10.1007/BF02551274

[52] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks 4 (2) (1991) 251–257. doi:10.1016/0893-6080(91)90009-T.

[53] M. Leshno, V. Y. Lin, A. Pinkus, S. Schocken, Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, Neural Networks 6 (6) (1993) 861–867. doi:10.1016/S0893-6080(05)80131-5.
URL http://archive.nyu.edu/handle/2451/14329

[54] G. C. Peng, M. Alber, A. Buganza Tepole, W. R. Cannon, S. De, S. Dura-Bernal, K. Garikipati, G. Karniadakis, W. W. Lytton, P. Perdikaris, L. Petzold, E. Kuhl, Multiscale Modeling Meets Machine Learning: What Can We Learn?, Archives of Computational Methods in Engineering 28 (3) (2021) 1017–1037. arXiv:1911.11958, doi:10.1007/s11831-020-09405-5.
URL https://link.springer.com/article/10.1007/s11831-020-09405-5

[55] E. B. Tadmor, R. E. Miller, R. S. Elliott, Continuum Mechanics and Thermodynamics, 2012.

[56] J. M. Ball, Convexity conditions and existence theorems in nonlinear elasticity, Archive for Rational Mechanics and Analysis 63 (4) (1976) 337–403. doi:10.1007/BF00279992.

[57] M. Fernández, M. Jamshidian, T. Böhlke, K. Kersting, O. Weeger, Anisotropic hyperelastic constitutive models for finite deformations combining material theory and data-driven approaches with application to cubic lattice metamaterials, Computational Mechanics 67 (2) (2021) 653–677. doi:10.1007/s00466-020-01954-7.
URL https://doi.org/10.1007/s00466-020-01954-7

[58] L. Linden, D. K. Klein, K. A. Kalina, J. Brummund, O. Weeger, M. Kästner, Neural networks meet hyperelasticity: A guide to enforcing physics, Journal of the Mechanics and Physics of Solids 179 (2023). arXiv:2302.02403, doi:10.1016/j.jmps.2023.105363.
URL www.elsevier.com/locate/jmps

[59] K. Linka, E. Kuhl, A new family of Constitutive Artificial Neural Networks towards automated model discovery, Computer Methods in Applied Mechanics and Engineering 403 (2023) 115731. arXiv:2210.02202, doi:10.1016/j.cma.2022.115731.

[60] B. Amos, L. Xu, J. Z. Kolter, Input Convex Neural Networks, in: 34th International Conference on Machine Learning, ICML 2017, Vol. 1, 2017, pp. 146–155. arXiv:1609.07152.

[61] D. K. Klein, M. Fernández, R. J. Martin, P. Neff, O. Weeger, Polyconvex anisotropic hyperelasticity with neural networks, Journal of the Mechanics and Physics of Solids 159 (2022). arXiv:2106.14623, doi:10.1016/j.jmps.2021.104703.
URL www.elsevier.com/locate/jmps

[62] D. K. Klein, F. J. Roth, I. Valizadeh, O. Weeger, Parametrized polyconvex hyperelasticity with physics-augmented neural networks, Data-Centric Engineering 4 (2) (2023). arXiv:2307.03463, doi:10.1017/dce.2023.21.
URL https://doi.org/10.1017/dce.2023.21

[63] A. L. Frankel, R. E. Jones, C. Alleman, J. A. Templeton, Predicting the mechanical response of oligocrystals with deep learning, Computational Materials Science 169 (2019). arXiv:1901.10669, doi:10.1016/j.commatsci.2019.109099.
URL www.elsevier.com/locate/commatsci

[64] J. K. Wilt, C. Yang, G. X. Gu, Accelerating Auxetic Metamaterial Design with Deep Learning, Advanced Engineering Materials 22 (5) (2020) 1901266. doi:10.1002/adem.201901266.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/adem.201901266

[65] A. Pandey, R. Pokharel, Machine learning based surrogate modeling approach for mapping crystal deformation in three dimensions, Scripta Materialia 193 (2021) 1–5. doi:10.1016/j.scriptamat.2020.10.028.

[66] Z. Yang, C. H. Yu, M. J. Buehler, Deep learning model to predict complex stress and strain fields in hierarchical composites, Science Advances 7 (15) (apr 2021). doi:10.1126/SCIADV.ABD7416.
URL https://www.science.org/doi/10.1126/sciadv.abd7416

[67] J. R. Mianroodi, N. H. Siboni, D. Raabe, Teaching solid mechanics to artificial intelligence—a fast solver for heterogeneous materials, npj Computational Materials 7 (1) (2021). arXiv:2103.09147, doi:10.1038/s41524-021-00571-z.
URL https://doi.org/10.1038/s41524-021-00571-z

[68] J. R. Mianroodi, S. Rezaei, N. H. Siboni, B. X. Xu, D. Raabe, Lossless multi-scale constitutive elastic relations with artificial intelligence, npj Computational Materials 8 (1) (2022) 1–12. arXiv:2108.02837, doi:10.1038/s41524-022-00753-3.

[69] M. S. Khorrami, J. R. Mianroodi, N. H. Siboni, P. Goyal, B. Svendsen, P. Benner, D. Raabe, An artificial neural network for surrogate modeling of stress fields in viscoplastic polycrystalline materials, npj Computational Materials 9 (1) (2023) 1–10. arXiv:2208.13490, doi:10.1038/s41524-023-00991-z.
URL https://www.nature.com/articles/s41524-023-00991-z

[70] T. S. Cohen, M. Welling, Group equivariant convolutional networks, in: 33rd International Conference on Machine Learning, ICML 2016, Vol. 6, 2016, pp. 4375–4386. arXiv:1602.07576.

[71] N. N. Vlassis, R. Ma, W. Sun, Geometric deep learning for computational mechanics Part I: anisotropic hyperelasticity, Computer Methods in Applied Mechanics and Engineering 371 (2020) 113299. arXiv:2001.04292, doi:10.1016/j.cma.2020.113299.
URL https://doi.org/10.1016/j.cma.2020.113299www.sciencedirect.comwww.elsevier.com/locate/cmahttps://linkinghub.elsevier.com/retrieve/pii/S0045782520304849

[72] A. Thomas, A. Durmaz, M. Alam, P. Gumbsch, H. Sack, C. Eberl, Materials fatigue prediction using graph neural networks on microstructure representations, Scientific Reports 2023 13:1 13 (1) (2023) 1–16. doi:10.1038/s41598-023-39400-2.
URL https://www.nature.com/articles/s41598-023-39400-2

[73] K. Karapiperis, D. M. Kochmann, Prediction and control of fracture paths in disordered architected materials using graph neural networks, Communications Engineering 2 (1) (2023). doi:10.1038/s44172-023-00085-0.
URL www.ae108.ethz.ch

[74] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, P. W. Battaglia, Learning Mesh-Based Simulation with Graph Networks (oct 2020). arXiv:2010.03409.
URL http://arxiv.org/abs/2010.03409

[75] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, A. Tkatchenko, Quantum-chemical insights from deep tensor neural networks, Nature Communications 8 (2017). arXiv:1609.08259, doi:10.1038/ncomms13890.
URL www.nature.com/naturecommunications

[76] K. T. Schütt, P. J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko, K. R. Müller, SchNet: A continuous-filter convolutional neural network for modeling quantum interactions, in: Advances in Neural Information Processing Systems, Vol. 2017-Decem, 2017, pp. 992–1002. arXiv:1706.08566.
URL www.quantum-machine.org.

[77] K. T. Schütt, H. E. Sauceda, P. J. Kindermans, A. Tkatchenko, K. R. Müller, SchNet - A deep learning architecture for molecules and materials, Journal of Chemical Physics 148 (24) (2018). arXiv:1712.06113, doi:10.1063/1.5019779.

[78] N. Lubbers, J. S. Smith, K. Barros, Hierarchical modeling of molecular energies using a deep neural network, Journal of Chemical Physics 148 (24) (2018) 241715. arXiv:1710.00017, doi:10.1063/1.5011181.
URL https://doi.org/10.1063/1.5011181

[79] O. T. Unke, M. Meuwly, PhysNet: A Neural Network for Predicting Energies, Forces, Dipole Moments, and Partial Charges, Journal of Chemical Theory and Computation 15 (6) (2019) 3678–3693. arXiv:1902.08408, doi:10.1021/acs.jctc.9b00181.
URL https://pubs.acs.org/sharingguidelines

[80] J. Gasteiger, J. Groß, S. Günnemann, Directional Message Passing for Molecular Graphs (2020). arXiv:2003.03123.
URL https://www.daml.in.tum.de/dimenethttp://arxiv.org/abs/2003.03123

[81] T. Xie, J. C. Grossman, Crystal Graph Convolutional Neural Networks for an Accurate and Interpretable Prediction of Material Properties, Physical Review Letters 120 (14) (2018). arXiv:1710.10324, doi:10.1103/PhysRevLett.120.145301.

[82] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, P. Riley, Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds (feb 2018). arXiv:1802.08219.
URL http://arxiv.org/abs/1802.08219

[83] I. Batatia, D. P. Kovács, G. N. Simm, C. Ortner, G. Csányi, MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields, in: Advances in Neural Information Processing Systems, Vol. 35, 2022, pp. 11423–11436. arXiv:2206.07697.

[84] B. Jing, S. Eismann, P. Suriana, R. J. Townshend, R. O. Dror, LEARNING FROM PROTEIN STRUCTURE WITH GEOMETRIC VECTOR PERCEPTRONS, in: ICLR 2021 - 9th International Conference on Learning Representations, International Conference on Learning Representations, ICLR, 2021. arXiv:2009.01411.
URL https://arxiv.org/abs/2009.01411v3

[85] K. T. Schütt, O. T. Unke, M. Gastegger, S. Schütt, O. T. Unke, M. Gastegger, Equivariant Message Passing for the Prediction of Tensorial Properties and Molecular Spectra, in: Proceedings of Machine Learning Research, Vol. 139, PMLR, 2021, pp. 9377–9388. arXiv:2102.03150.
URL https://proceedings.mlr.press/v139/schutt21a.html

[86] J. Gasteiger, F. Becker, S. Günnemann, GemNet: Universal Directional Graph Neural Networks for Molecules, in: Advances in Neural Information Processing Systems, Vol. 9, 2021, pp. 6790–6802. arXiv:2106.08903.
URL https://www.daml.in.tum.de/gemnet

[87] V. G. Satorras, E. Hoogeboom, M. Welling, E(n) Equivariant Graph Neural Networks, in: Proceedings of Machine Learning Research, Vol. 139, 2021, pp. 9323–9332. arXiv:2102.09844.

[88] M. G. D. Geers, V. G. Kouznetsova, W. A. M. Brekelmans, Multi-scale computational homogenization: Trends and challenges, in: Journal of Computational and Applied Mathematics, Vol. 234, 2010, pp. 2175–2182. doi:10.1016/j.cam.2009.08.077.

[89] V. G. Kouznetsova, M. G. D. Geers, W. A. M. Brekelmans, Multi-scale second-order computational homogenization of multi-phase materials: A nested finite element solution strategy, Computer Methods in Applied Mechanics and Engineering 193 (48-51) (2004) 5525–5550. doi:10.1016/j.cma.2003.12.073.

[90] Ł. Kaczmarczyk, C. J. Pearce, N. Bićanić, Scale transition and enforcement of RVE boundary conditions in second-order computational homogenization, International Journal for Numerical Methods in Engineering 74 (3) (2008) 506–522. doi:10.1002/nme.2188.
URL https://onlinelibrary.wiley.com/doi/full/10.1002/nme.2188https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.2188https://onlinelibrary.wiley.com/doi/10.1002/nme.2188

[91] O. Rokoš, M. M. Ameen, R. H. J. Peerlings, M. G. D. Geers, Micromorphic computational homogenization for mechanical metamaterials with patterning fluctuation fields, Journal of the Mechanics and Physics of Solids 123 (2019) 119–137. doi:10.1016/j.jmps.2018.08.019.

[92] S. E. H. M. van Bree, O. Rokoš, R. H. J. Peerlings, M. Doškář, M. G. D. Geers, A Newton solver for micromorphic computational homogenization enabling multiscale buckling analysis of pattern-transforming metamaterials, Computer Methods in Applied Mechanics and Engineering 372 (dec 2020). arXiv:2008.12850, doi:10.1016/j.cma.2020.113333.

[93] O. Rokoš, M. M. Ameen, R. H. J. Peerlings, M. G. D. Geers, Extended micromorphic computational homogenization for mechanical metamaterials exhibiting multiple geometric pattern transformations, Extreme Mechanics Letters 37 (may 2020). arXiv:2004.05226, doi:10.1016/j.eml.2020.100708.

[94] K. Bertoldi, P. M. Reis, S. Willshaw, T. Mullin, Negative poisson's ratio behavior induced by an elastic instability, Advanced Materials 22 (3) (2010) 361–366. doi:10.1002/adma.200901956.

[95] E. Polukhov, D. Vallicotti, M. A. Keip, Computational stability analysis of periodic electroactive polymer composites across scales, Computer Methods in Applied Mechanics and Engineering 337 (2018) 165–197. doi:10.1016/j.cma.2018.01.020.
URL www.sciencedirect.comwww.elsevier.com/locate/cma

[96] G. Geymonat, S. Müller, N. Triantafyllidis, Homogenization of nonlinearly elastic materials, microscopic bifurcation and macroscopic loss of rank-one convexity, Archive for Rational Mechanics and Analysis 122 (3) (1993) 231–290. doi:10.1007/BF00380256.

[97] N. Triantafyllidis, M. D. Nestorović, M. W. Schraad, Failure surfaces for finitely strained two-phase periodic solids under general in-plane loading, Journal of Applied Mechanics, Transactions ASME 73 (3) (2006) 505–515. doi:10.1115/1.2126695.

[98] G. Zhang, N. Feng, K. Khandelwal, A computational framework for homogenization and multiscale stability analyses of nonlinear periodic materials, International Journal for Numerical Methods in Engineering 122 (22) (2021) 6527–6575. doi:10.1002/nme.6802.

[99] V. G. Kouznetsova, Computational homogenization for the multi-scale analysis of multi-phase materials, Ph.D. thesis (2002). doi:10.6100/IR560009.
URL http://www.narcis.nl/publication/RecordID/oai:library.tue.nl:560009

[100] C. Miehe, Computational micro-to-macro transitions for discretized micro-structures of heterogeneous materials at finite strains based on the minimization of averaged incremental energy, Computer Methods in Applied Mechanics and Engineering 192 (5-6) (2003) 559–591. doi:10.1016/S0045-7825(02)00564-9.

[101] C. Miehe, Strain-driven homogenization of inelastic microstructures and composites based on an incremental variational formulation, International Journal for Numerical Methods in Engineering 55 (11) (2002) 1285–1322. doi:10.1002/nme.515.
URL https://onlinelibrary.wiley.com/doi/10.1002/nme.515

[102] C. Geuzaine, J. F. Remacle, Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities, International Journal for Numerical Methods in Engineering 79 (11) (2009) 1309–1331. doi:10.1002/nme.2579.

[103] M. M. Bronstein, J. Bruna, T. Cohen, P. Veličković, Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges (apr 2021). arXiv:2104.13478.
URL http://arxiv.org/abs/2104.13478

[104] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, X. Sun, Measuring and relieving the over-smoothing problem for graph neural networks from the topological view, AAAI 2020 - 34th AAAI Conference on Artificial Intelligence (2020) 3438–3445arXiv:1909.03211, doi:10.1609/aaai.v34i04.5747.

[105] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, G. E. Dahl, Neural message passing for quantum chemistry, in: 34th International Conference on Machine Learning, ICML 2017, Vol. 3, 2017, pp. 2053–2070. arXiv:1704.01212.

[106] W. M. Czarnecki, S. Osindero, M. Jaderberg, G. Swirszcz, R. Pascanu, Sobolev training for neural networks, in: Advances in Neural Information Processing Systems, Vol. 2017-Decem, 2017, pp. 4279–4288. arXiv:1706.04859.

[107] C. K. Joshi, C. Bodnar, S. V. Mathis, T. Cohen, P. Liò, On the Expressive Power of Geometric Graph Neural Networks (2023). arXiv:2301.09308.
URL http://arxiv.org/abs/2301.09308

CHAPTER 3

Wallpaper Group-Based Mechanical Metamaterials: Dataset Including Mechanical Responses

Reproduced from Hendriks, F., Menkovski, V., Doškář, M., Geers, M. G., Verbeek, K., & Rokoš, O. (2025). Wallpaper Group-Based Mechanical Metamaterials: Dataset Including Mechanical Responses. Scientific Data. doi: 10.1038/s41597-025-06150-x

Mechanical metamaterials often exhibit pattern transformations through instabilities, enabling applications in, e.g., soft robotics, sound reduction, and biomedicine. These transformations and their resulting mechanical properties are closely tied to the symmetries in these metamaterials' microstructures, which remain under-explored. Designing such materials is challenging due to the unbounded design space, and while machine learning offers promising tools, they require extensive training data. Here, we present a large dataset of 2D microstructures and their macroscopic mechanical responses in the hyperelastic, finite-strain regime, including buckling. The microstructures are generated using a novel method, which covers all 17 wallpaper symmetry groups and employs Bézier curves for a rich parametric space. Mechanical responses are obtained through finite element-based computational homogenization. The dataset includes 1,020 distinct geometries, each subjected to 12 loading trajectories, totaling 12,240 trajectories. Our dataset supports the development and benchmarking of surrogate models, facilitates the study of symmetry-property relationships, and enables investigations into symmetry-breaking during pattern transformations, potentially revealing emergent behavior in mechanical metamaterials.

3.1 Background & Summary

Mechanical metamaterials feature architected microstructures that are engineered to exhibit specific effective mechanical properties. To achieve this, designers often leverage buckling phenomena. Buckling causes sudden pattern transformations that result in an abrupt change in macroscopic behavior. Consequently, a metamaterial can switch between multiple distinct modes. These transformations enable tuning of acoustic metamaterials [1–4], and can control the shape of a material, with applications in soft robotics [5–7] and biomedicine [8]. These metamaterials can also exhibit a tunable negative Poisson's ratio [9] or negative compressibility [10].

Here, we focus on flexible, porous 2D metamaterials revealing buckling-induced pattern transformations under mechanical loading [11, 12] (neglecting other options

such as pneumatics [5], or magnetic activation [13]). These materials consist of a polymer base with holes of various shapes and sizes, typically arranged symmetrically to promote buckling. The geometry of these holes and their distribution controls mechanical properties such as auxeticity, anisotropy, and the bulk and shear moduli. Importantly, during buckling, microstructure symmetries are broken, making symmetry analysis essential to understand and predict deformation modes and their multiplicity [14–16]. In 2D, these symmetries are described by the 17 wallpaper groups, which (in short IUCr notation) are *p1*, *p2*, *pm*, *pg*, *cm*, *pmm*, *pmg*, *pgg*, *cmm*, *p4*, *p4m*, *p4g*, *p3*, *p3m1*, *p31m*, *p6m* and *p6* [17,18]. The symmetries include translations, rotations (specified by a number that indicates the rotation order), reflections (indicated with m), and glide reflections (indicated with g). See Schattschneider [18] for a detailed overview.

However, microstructural symmetry effects extend beyond buckling metamaterials, for example, chiral geometries are often exploited in auxetic materials [19]. Note that although chiral metamaterials have proven their value in achieving auxeticity, chirality is neither necessary nor sufficient to achieve auxeticity (e.g., [20]). In the same spirit, a mirror or glide mirror line ensures orthotropy of the effective behavior. The symmetries are also relevant when analyzing band gaps (Bragg diffraction), since they determine the shape of the irreducible Brillouin zone [21] and the location of the band-gap extremum within it [22]. They are also relevant for tailoring the dispersion of modes in acoustic metamaterials [23] and for designing origami-based metamaterials [24,25].

Currently, metamaterial datasets are scarce and limited in microstructure geometries, compromising the development and benchmarking of different design or analysis methods. Many datasets are limited to geometries based on periodic unit cells within the wallpaper groups *pmm* [26–29] or *p4m* [12], which inherently lack chirality and consistently result in orthotropic mechanical responses. Stochastic methods such as random distributions of ellipses, spinodal decomposition, Voronoi microstructures, or fractal noise [30] only generate geometries within the group *p1*. This misses highly symmetric geometries, such as *p6m*, which can exhibit interesting multiplicities of buckling modes [31]. Moreover, these methods are often based on a limited number of building blocks with limited freedom (e.g. trusses) [32] or are pixel-based [26,28–30], with low resolution and often resulting in thin ligaments or disconnected structures, unless specifically constrained [33]. Therefore, sometimes the material volume fraction is bounded from below by 0.7 [27], leading to dense materials that do not buckle. Moreover, most datasets do not cover large deformations, as they only simulate one loading case in the reference configuration without a trajectory. Those that do [28,34] are typically limited to uniaxial compression, which does not allow, e.g., analyzing of multiplicity of buckling modes as in honeycomb structures [31].

The prime motivation behind our dataset of mechanical metamaterials with a large variety of geometries, is to facilitate the development of machine learning models, which are currently of significant interest [28, 35–47]. Currently, topology optimization is a staple for creating microstructures with specific mechanical responses [48–53]. However, this method is computationally expensive, especially when large deformations and buckling are considered, and typically ends up in a local optimum. Moreover, it needs predefined domain boundaries of the unit cell (e.g. hexagonal, square,

parallelogram). Machine learning-based methods hold promise as a faster alternative for simulation and generation of new designs [28, 34, 44], and could rapidly generate a diverse set of potential solutions or provide probabilistic insights into the design space, increasing the chances of identifying a near-optimal or globally optimal configuration. However, they require a proper training dataset of microstructures with their corresponding mechanical responses.

This contribution addresses this need by providing a dataset that encompasses:

- A large variety of geometries covering a broad design space of mechanical metamaterials, including all symmetry groups, based on a periodic graph 'skeleton'.
- Well-structured, high-quality geometries: high resolution using Bézier curves, no too thin ligaments, fully connected (e.g., no laminates, no checkerboards), periodic, not too high volume fraction, and without sharp corners (to prevent stress concentrations).
- High-quality simulations covering large deformation responses, including buckling, to a variety of loading conditions.

Figure 3.1 gives a schematic overview of the creation of the dataset, and Table 3.1 summarizes the input and output data. The dataset is available at https://zenodo.org/records/15849549 [54]; the Python code used to generate the geometries is available at https://github.com/FHendriks11/wallpaper_microstructures, and the MATLAB code used for the finite element simulations is available at https://github.com/FHendriks11/mechmetamat_homogenization.

Input	**Description**	**Shape, data type**
Geometry	Description of a mesh representing the geometry and of its periodicity	-
$\mathbf{F}$	Deformation gradient tensor	$(2, 2)$ float
Output	**Description**	**Shape, data type**
$\vec{x}$	Deformed configuration	$(N, 2)$ float
$\mathfrak{W}$	Strain energy density	$(1,)$ float
$\mathbf{P}$	First Piola-Kirchhoff stress tensor	$(2, 2)$ float
$\mathbf{D}$	Tangent stiffness tensor	$(2, 2, 2, 2)$ float

Table 3.1: The main inputs and outputs of the dataset.

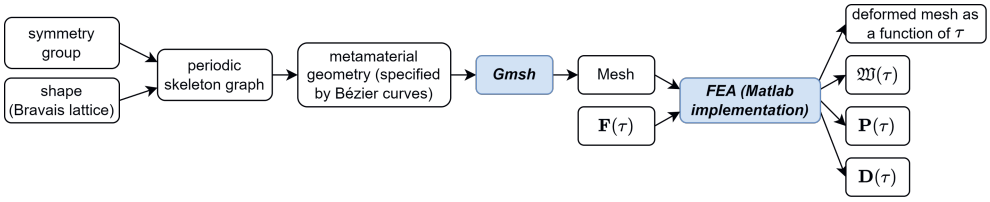


Figure 3.1: Schematic overview of the creation process of the dataset.

3.2 Methods

In Figure 3.1 it can be seen that our dataset was created in two steps: first, we generated the microstructure geometries, and then we computed their mechanical response for prescribed mascroscopic loading cases. We will cover the steps of microstructure generation, sampling of the loading and simulation of the mechanical response in the sections below.

Generating new microstructures

All 2D periodic structures belong to one of 17 wallpaper groups [17, 18], which categorize all possible ways to tile a 2D plane based on their symmetries, which include translations, rotations, reflections, and glide reflections. Each group represents a unique combination of these symmetries. In this framework, the unit cell is the smallest repeating unit that can create a periodic structure using only translations. The unit cell further consists of one or more copies of the fundamental domain, which tiles the space using not just translations, but also reflections, rotations, and/or glide reflections. The lattice vectors that define the unit cell can correspond to any of the 5 Bravais lattices in 2D. The Bravais lattices characterize the symmetries of a lattice, generated only by translating a point along the lattice vectors. In 2D, there are 5 Bravais lattice types: square, rectangular, hexagonal, centered rectangular (rhombic), and oblique, although not all Bravais lattice types are possible for all wallpaper groups. See Figure 3.2 for an example showing the relevant geometric quantities, and Figure 3.3, 3.4, 3.5 and 3.6 for examples of each wallpaper group and its possible Bravais lattices. Note that for the wallpapergroup *cm*, there are two options to use a hexagonal Bravais lattice: one with the mirror line bisecting the unit cell along the short diagonal and the other along the long diagonal. We denote these two options 'hexagonal1' and 'hexagonal2', respectively.

In our method, a new geometry was generated using the following steps, shown in Figure 3.7:

- Choose a wallpaper group and a Bravais lattice type.
- Generate the parallelogram or triangle that represents the shape of the fundamental domain.
- Generate a random periodic graph: the 'skeleton graph'. This geometric graph should be planar and connected, and respect the chosen wallpaper group.
- Tile this graph into a graph covering the unit cell.
- Assume at first that all graph faces are filled with bulk material. Then create a hole on each face of the graph; each hole is described by a béziergon, i.e., a closed path composed of Bézier curves.
- Discretize only one fundamental domain, respecting compatibility requirements, then tile this to obtain the unit cell discretization.
- Validate the obtained geometry throughout (e.g., check if the volume fraction is not too high, and that ligaments are not too thin) and discard invalid geometries.

This method of generating new microstructures is based on a periodic graph 'skeleton', which roughly defines the connectivity of the microstructure. This graph skeleton also offers a versatile backbone that can be used by machine learning methods

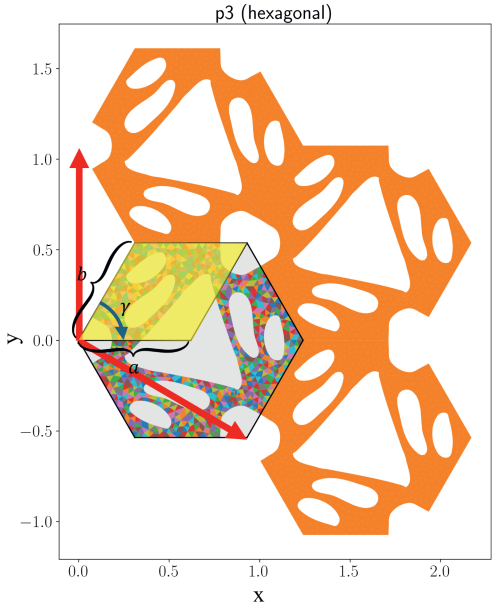


Figure 3.2: Annotated example of a geometry with *p3* symmetry. The fundamental domain is highlighted in yellow, the lattice vectors are shown as red arrows, and the side lengths a and b and the lower left corner γ are indicated. The unit cell mesh is shown with a colorful mesh, and the rest of the entire RVE is orange.

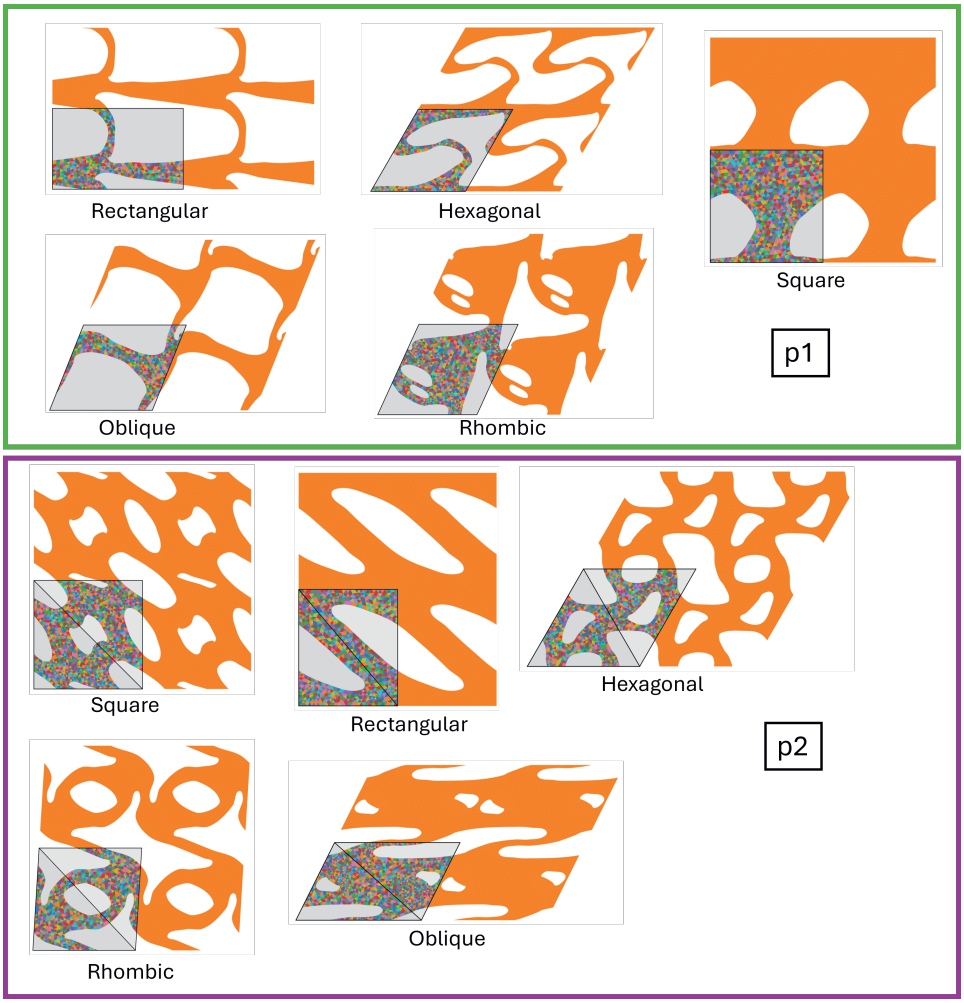


Figure 3.3: Examples of geometries corresponding to the wallpaper groups *p1* and *p2*. Each entire structure shown is one RVE, of which one unit cell is indicated by thick black lines and a light grey background. Within the unit cell, one fundamental domain is indicated, also with thick black lines and with a slightly darker grey background. Different Bravais lattices for the same wallpaper group are grouped together using colored boxes. The text in the white boxes indicates the wallpaper group; the other text indicates the Bravais lattice.

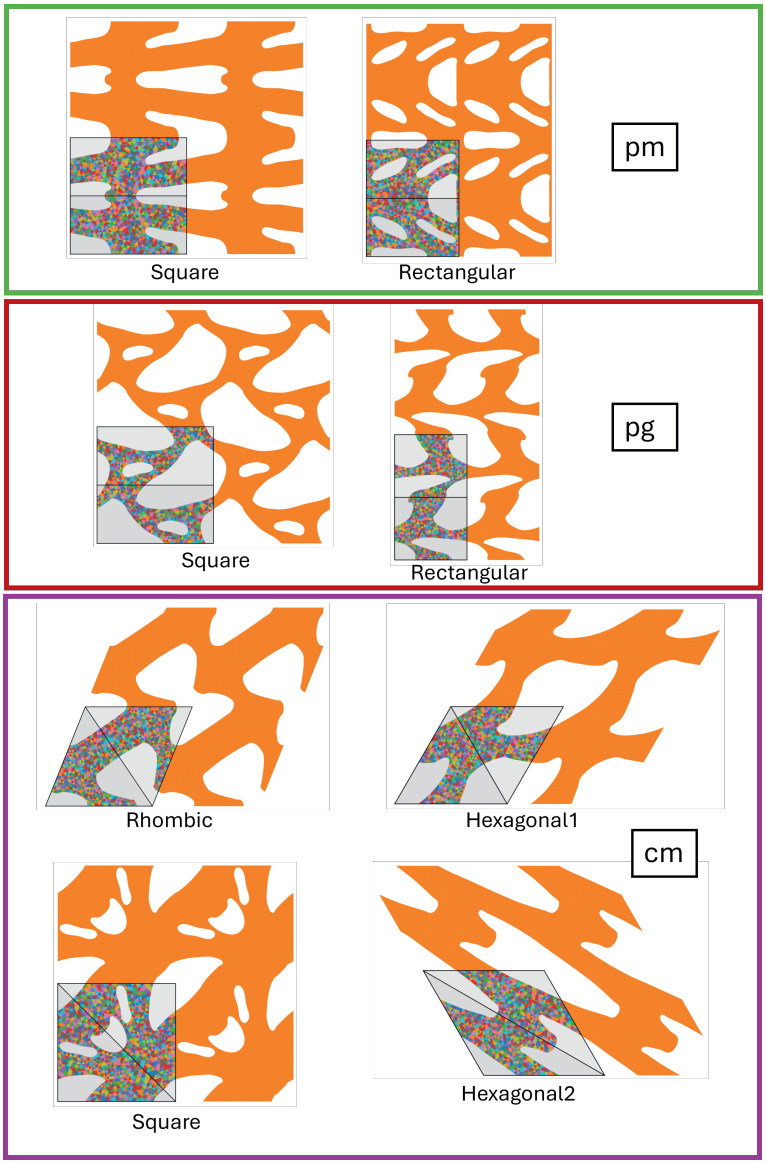


Figure 3.4: Examples of geometries corresponding to the wallpaper groups *pm*, *pg*, and *cm*. Each entire structure shown is one RVE, of which one unit cell is indicated by thick black lines and a light grey background. Within the unit cell, one fundamental domain is indicated, also with thick black lines and with a slightly darker grey background. Different Bravais lattices for the same wallpaper group are grouped together using colored boxes. The text in the white boxes indicates the wallpaper group; the other text indicates the Bravais lattice.

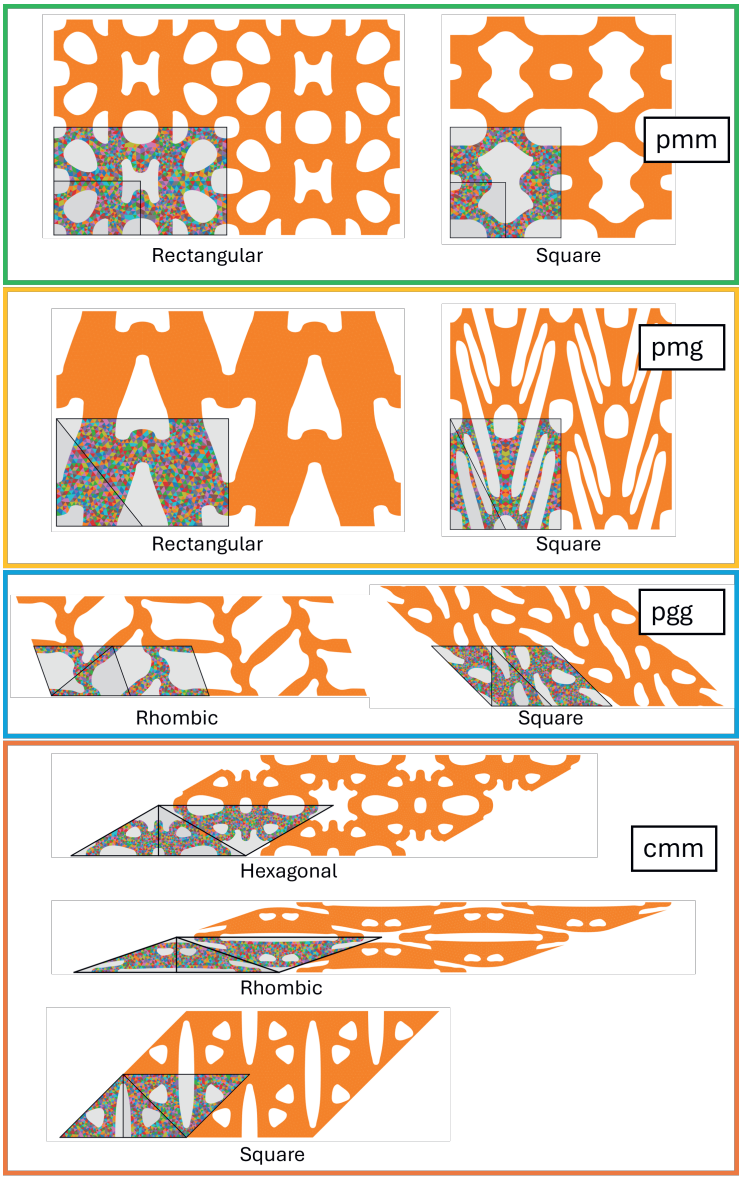


Figure 3.5: Examples of geometries corresponding to the wallpaper groups *pmm*, *pmg*, *pgg*, and *cmm*. Each entire structure shown is one RVE, of which one unit cell is indicated by thick black lines and a light grey background. Within the unit cell, one fundamental domain is indicated, also with thick black lines and with a slightly darker grey background. Different Bravais lattices for the same wallpaper group are grouped together using colored boxes. The text in the white boxes indicates the wallpaper group; the other text indicates the Bravais lattice.

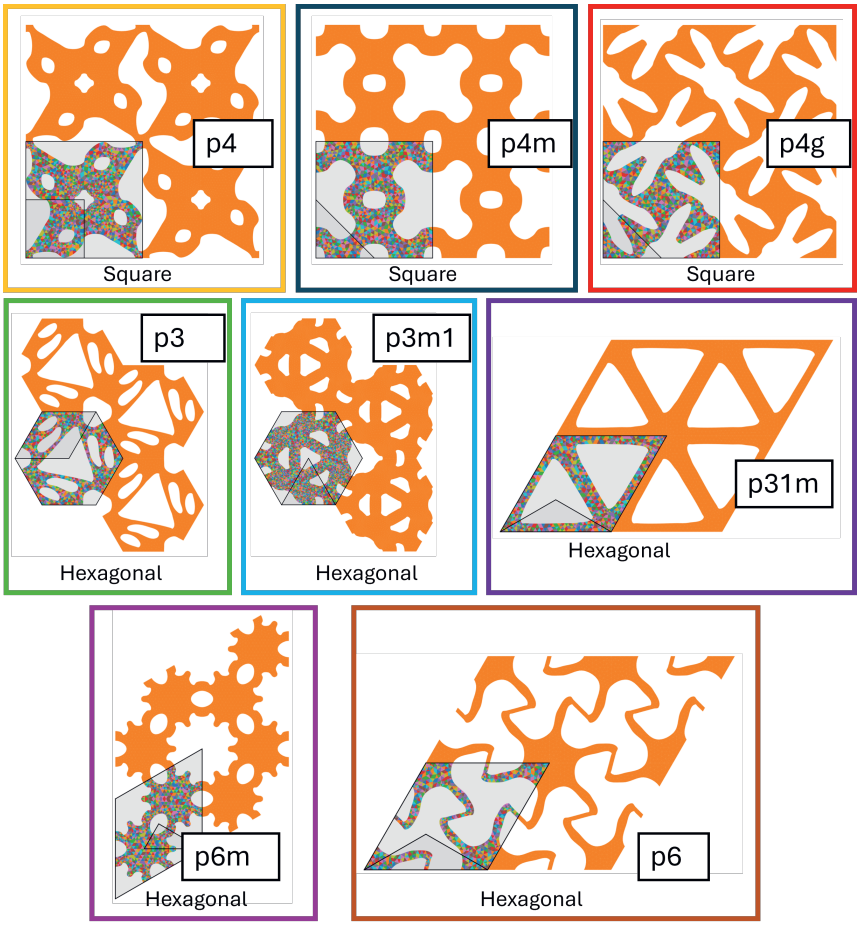


Figure 3.6: Examples of geometries corresponding to the wallpaper groups *p4*, *p4m*, *p4g*, *p3*, *p3m1*, *p31m*, *p6m*, and *p6*. Each entire structure shown is one RVE, of which one unit cell is indicated by thick black lines and a light grey background. Within the unit cell, one fundamental domain is indicated, also with thick black lines and with a slightly darker grey background. The text in the white boxes indicates the wallpaper group; the other text indicates the Bravais lattice.

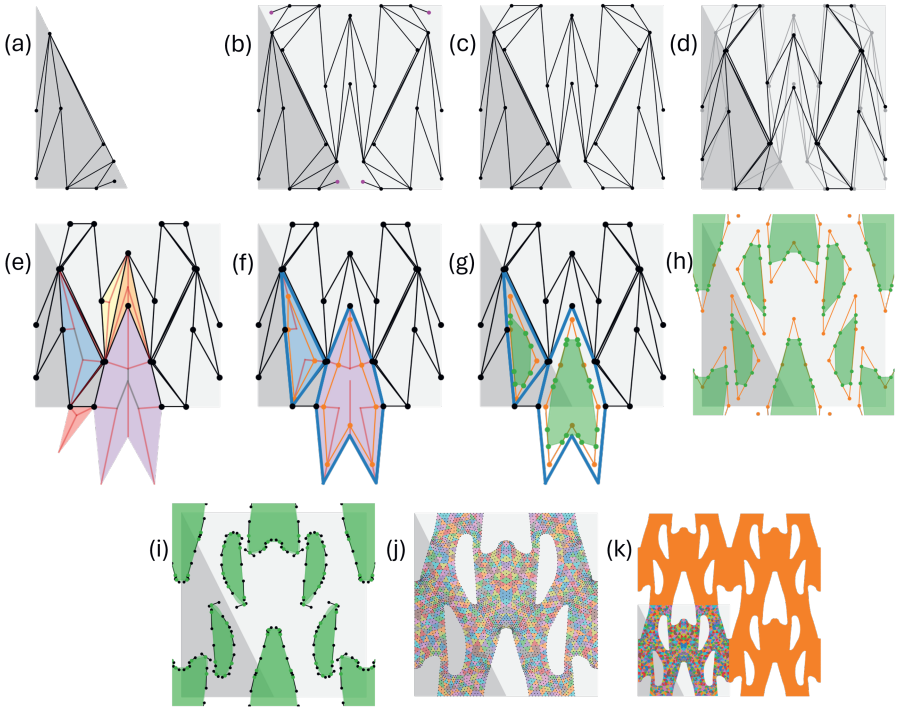


Figure 3.7: The various steps in the generation of the microstructures. (a) Skeleton graph on the fundamental domain, (b) fundamental domain tiled into a unit cell, with leaves identified in magenta, (c) leaves removed, (d) nodes shifted to mitigate sharp angles, (e) unique faces (initial face polygons) with their straight skeleton, (f) secondary face polygon in orange, (g) tertiary face polygons approximating hole shape in green, (h) polygons copied around, (i) beziergons, (j) mesh of the unit cell, (k) final RVE.

such as graph neural networks [47]. This method overcomes the limitations of previous approaches mentioned in the previous section by generating microstructures for any of the 17 wallpaper groups, with varying ligament thicknesses and hole shapes. These 2D materials can be turned into 3D materials by extruding them along the out-of-plane direction.

Other works have used periodic graphs to generate metamaterials, but these are limited to simple building blocks, like trusses, shells or cubes [34, 44, 55, 56]. In contrast, our geometries are specified by Bézier curves, making them smooth and highly resolved, allowing any arbitrary shape. The method also ensures that the microstructures are connected and periodic, and that the ligaments are unlikely to be thinner than a specified threshold (and the cases that do have too thin ligaments are rare and easily filtered out). The fact that other works have used graph-based approaches to generate 3D metamaterials, albeit in simpler forms, suggests that our method could be extended to 3D as well.

The microstructures were generated using Python code which is available at https://github.com/FHendriks11/wallpaper_microstructures.

We discuss the microstructure generation steps in order below.

Generating the fundamental domain shape

Depending on the chosen wallpaper group we used either a parallelogram or a triangle as the fundamental domain shape. Without loss of generality, we fixed the orientation of the fundamental domain by aligning its bottom edge along the x-axis, and then set the length of this edge to 1. We then chose the length of the left side b (which is not necessarily vertical), as well as the lower-left corner angle γ. Depending on the wallpaper group and the Bravais lattice, b and/or γ could be fixed already, e.g., for a square Bravais lattice, b was also 1 and γ was 90 degrees. If they were not fixed, we picked them from a uniform range of values, specified for each combination of wallpaper group and Bravais lattice. Different values of b and γ lead to different geometries and therefore different material properties. See Figure 3.2, where a, b and γ are indicated for the wallpaper group *p3*. We then immediately rescaled a and b such that the fundamental domain has an area of 1/(number of fundamental domains in the unit cell), to ensure the unit cell would have unit area.

Periodic skeleton graph generation

For our skeleton graph, we desire a random geometric graph that allows for a lot of freedom in the shape of the faces (unlike, e.g., a Voronoi graph, which can only have convex faces), but still respects the right symmetries. To this end, we started with the fundamental domain, see Figure 3.7(a), and first randomly distributed points on its boundary. We made sure each edge of the boundary contained at least one point (a point on a vertex of the polygon involving that edge also counts). Next, we randomly generated points inside it. Edges of the skeleton graph were also created stochastically: we randomly picked two points and checked whether an edge connecting them intersected already defined edges. If not, we added the new edge to the graph. We repeated this process until all points were connected to each other. Checking

the connectivity of a periodic graph is not as simple as checking connectivity of the fundamental domain graph; we explain our connectivity check in Appendix 3.A. By rotating, reflecting and translating the graph defined on the fundamental domain, we were able to create the skeleton graph of the unit cell, see Figure 3.7(b). We further removed leaves (nodes of degree 1), and shifted some of the nodes of the graph to attempt to reduce the number of really sharp angles ($< \pi/7$), see Figures 3.7(c) and (d), respectively.

Creating the holes

To distribute holes in a microstructure, we treated each face of the graph as a potential location for a hole. To this end, we followed this procedure, which we illustrate in Figure 3.7:

- *Construct primary polygons.* We first identified all the unique faces of the unit cell skeleton graph, i.e. those that are not transformed copies of each other, see Figure 3.7(e). This ensured that the transformed copies got the same hole shape. We call the shapes of the graph faces the primary polygons.
- *Randomly select faces for hole creation* For each face, we randomly decided to either attempt to create a hole in it (with probability 98%), or fill it with bulk material (probability 2%).
- *Construct secondary polygons: approximate the thickness of the ligaments (orange lines in Figure 3.7(f)).* To approximately determine the thickness of the ligaments, we created a secondary polygon on each graph face marked for hole generation, outlined with orange lines in Figure 3.7(f). To this end, we picked a point on each bisector of the corners of the primary polygon. To ensure that the secondary polygon did not self-overlap, we only allowed picking points on a certain segment of the bisector. To determine this segment, we used the straight skeleton, which is a well-known method of representing a polygon in computational geometry, for which efficient algorithms exist to compute it. This straight skeleton represents a continuous shrinking inwards of the polygon edges. We generated the straight skeleton of each face using the scikit-geometry interface to the CGAL library [57], see Figure 3.7(e). We used the segments of the bisectors that are part of the straight skeleton (the red lines in Figure 3.7(e) and (f)) as our range to pick the new points, which ensured that the new polygon did not overlap itself. For each new point, we made sure it had a minimum distance l_{min} to the corresponding vertex of the face. This minimum distance was calculated as

$$d_{\min} = \begin{cases} \texttt{MIN\_THICKNESS} / \sin\left(\dfrac{\theta^{\text{prim}}}{2}\right), & \text{if the polygon is convex} \\ & (0 < \theta^{\text{prim}} \leq \pi) \text{ at this corner,} \\ \texttt{MIN\_THICKNESS} / \cos\left(\dfrac{\theta^{\text{prim}} - \pi}{2}\right), & \text{if the polygon is concave} \\ & (\theta^{\text{prim}} > \pi) \text{ at this corner,} \end{cases} \tag{3.1}$$

where `MIN_THICKNESS` is the minimum thickness to add and θ^{prim} is the angle of the primary polygon corner. This ensured that the ligament thickness is at least

2MIN_THICKNESS, because the neighboring face can also add the same thickness. In other words, MIN_THICKNESS is not the minimum thickness of the ligament itself, but of the ligament inside one face. The maximum distance d_{max} was MAX_REL_THICKNESS= 0.7 times the length of the bisector. If no point satisfies the minimum and maximum distance because $d_{min} > d_{max}$, then no hole was added to the face and the face was filled with bulk material. We also checked if this polygon has an area larger than the parameter MIN_AREA. If the area was smaller, the face was, again, filled with bulk material.

- *Construct tertiary polygons: add material in the corners of the secondary polygon (green lines in Figure 3.7(g)).* So far, all the polygons had very sharp corners where ligaments come together. Therefore, we created a tertiary polygon inside the secondary one, which added material in the corners. This tertiary polygon is shown in green in Figure 3.7(g). We created it by randomly choosing two points on each edge of the secondary polygon which were (1) a minimum distance away from the corners, and (2) a minimum distance apart from each other. The minimum distance (1) was computed

$$\text{Minimum distance} = \max\left(\frac{\texttt{MIN\_RADIUS}}{\tan\left(\theta^{\text{sec}}/2\right)}, \texttt{MIN\_D\_REL} \times \text{length of edge}\right) \quad (3.2)$$

where MIN_RADIUS and MIN_D_REL are parameters, which are the minimum radius of curvature, and a minimum distance relative to the length of the edge. θ^{sec} is the angle of the secondary polygon corner, in the interval $[0, \pi]$ (i.e., no distinction between convex and concave corners). The minimum distance apart from each other (2) was computed as:

$$\text{Minimum separation} = \max\left(\texttt{MIN\_SEP\_ABS}, \texttt{MIN\_SEP\_REL} \times \text{length of edge}\right) \quad (3.3)$$

where MIN_SEP_ABS and MIN_SEP_REL are parameters, denoting the minimum absolute and relative distance between the two points on the edge, respectively. In creating the secondary and tertiary polygons, we respected the symmetry of the holes, by using the same randomly generated numbers for corresponding corners and edges.
- *Replicate hole shapes across identical faces (Figure 3.7(h)).* After copying the tertiary polygons to all equivalent faces of the unit cell skeleton graph.
- *Create beziergon to define the final hole shape (Figure 3.7(i)).* A closed composite cubic Bézier curve (a so-called beziergon) was drawn through all vertices of each tertiary polygon, rendering a smooth, continuous path defining the hole shape, see Figure 3.7(i). The corners of the tertiary polygon were used as the start and end points of the Bézier segments. Per segment, two extra control points were generated, which defined the direction of the tangents to the Bézier curve segment. These control points were chosen such that (1) control points of two neighbouring curves around the joint between these curves are collinear with this joint (see for example control points T_1 and T_2 and the joint at point B in Figure 3.8; T_1, T_2 and B are collinear), (2) the line connecting them is perpendicular to the bisector of the corner of the tertiary polygon (see the line T_1T_2 and the red bisector in Figure 3.8) and (3) the distance between the joint

and the new control points is equal to 1/3 times the distance between the end points of the Bézier segment (e.g., $|T_1B| = |AB|/3$, $|BT_2| = |BC|/3$ in Figure 3.8).

- *Filter out invalid geometries.* The cases resulting in too high a volume fraction or unacceptable geometries were excluded from the dataset. See the validation section for details.

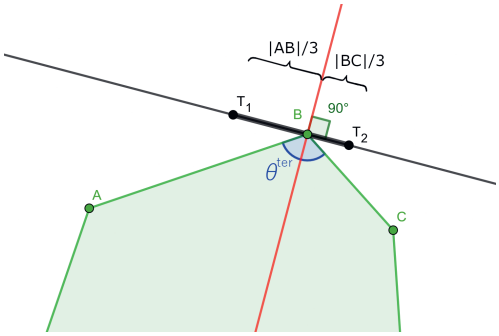


Figure 3.8: Illustration showing how the control points T_1 and T_2 controlling the tangents of the Bézier curves at point B are determined. The bisector of the tertiary polygon corner with angle θ^{ter} is shown in red.

Table 3.2 summarizes all parameters and their values used in the generation of the microstructures.

Creating the meshes

The final generated Bézier curves serve to define the geometry of the fundamental domain (see Figure 3.7i), which was then discretized using Gmsh [58]. For this dataset, we used quadratic triangular elements with a maximum element size of 0.0408, although Gmsh automatically used smaller elements near fine geometrical features. This size is relative to the area of the unit cell, which is always 1. The resulting mesh of the fundamental domain was then tiled into the unit cell (see Figure 3.7(j)).

We further assumed that a representative volume element (RVE) consisting of 2x2 unit cells would be sufficient to capture the most important pattern transformations (see Figure 3.7(k)). In literature, and from experimentation on our part, this does seem to be the case most of the time, although it should be noted that it is possible for a 2×2 RVE to be inadequate in certain load scenarios. For example, Polukhov et al. [59] show that in a electrostatically activated version of the same material a 2×3 RVE is necessary. A global buckling mode may also occur; however, in such case, any RVE size is insufficient, as the deformation wavelength becomes infinite relative to the RVE size and hence full-system (finite-size) analysis would have to be performed. Ideally, Bloch analysis should be used to determine the appropriate RVE size [11, 60–62]. Due to its high computational cost, we kept the Bloch analysis outside the scope of our contribution.

Parameter Name	Meaning	Value
Secondary polygon parameters		
MIN_AREA	Minimum area of secondary polygon	0.0167
MIN_THICKNESS	Minimum ligament thickness per face	0.012
MAX_REL_THICKNESS	Maximum ligament thickness per face relative to bisector from straight skeleton	0.7
PROB_RANDOM_FILL	Probability of bulk material	0.02
Tertiary polygon parameters		
MIN_RADIUS	Minimum corner radius	0.02
MIN_D_REL	Minimum distance of tertiary polygon vertex to secondary polygon vertex, relative to secondary polygon side length	0.2
MIN_SEP_ABS	Minimum separation between two tertiary polygon vertices on the same secondary polygon edge	0.02
MIN_SEP_REL	Minimum separation between two tertiary polygon vertices on the same secondary polygon edge, relative to secondary polygon side length	0.2
MIN_AREA2	Minimum area of tertiary polygon	0.0083
Mesh parameters		
clmax	Maximum characteristic element size	0.041

Table 3.2: Various parameters and their values used in the generation of microstructures.

Simulating mechanical response

Each microstructure in the dataset is accompanied by the results of computational homogenization, characterizing the macroscopic response of the entry. For the simulations of the response, we sampled a variety of loading conditions for each geometry, then performed a simulation that captures the large deformation and buckling behavior of the material. To simulate the response of the material, we performed computational homogenization [63] using finite element analysis (FEA), taking the applied macroscopic deformation gradient $\mathbf{F}_{\text{final}}$ as an input along with the RVE mesh. The calculation yielded the deformed shape of the RVE and effective macroscopic quantities: the homogenized strain energy density $\mathfrak{W}$, homogenized 1^{st} Piola-Kirchhoff stress tensor $\mathbf{P}$ and homogenized 4^{th}-order tangent stiffness tensor $\mathbf{D}$.

Given the microstructures' characteristics, the simulations are highly nonlinear as they cover large deformations and especially buckling. Therefore, we applied the prescribed deformation $\mathbf{F}$ in a series of small steps, such that the deformation gradient $\mathbf{F}(\tau)$ follows a trajectory parameterized by the pseudo-time τ, which attains values from 0 to 1, with $\mathbf{F}(0) = \mathbf{I}$ and $\mathbf{F}(1) = \mathbf{F}_{\text{final}}$. We adaptively changed the step length

to facilitate robust simulations. Consequently, the effective quantities are then also functions of τ, i.e., $\mathfrak{W}(\tau)$, $\mathbf{P}(\tau)$ and $\mathbf{D}(\tau)$. Through this type of displacement control, we can handle snap-through, but not snap-back [64, §5.1.5, p.165].

For the bulk material, we used the hyperelastic Bertoldi-Boyce constitutive model under the plane strain assumption [11], which is defined by the following microscopic strain energy density $\mathfrak{W}_\mathrm{m}(\mathbf{F}_\mathrm{m})$:

$$\mathfrak{W}_\mathrm{m}(\mathbf{F}_\mathrm{m}) = c_1 \left(I_1 - 2\right) + c_2 \left(I_1 - 2\right)^2 - 2c_1 \ln\left(J_\mathrm{m}\right) + \frac{K}{2}\left(J_\mathrm{m} - 1\right)^2 . \tag{3.4}$$

Here, $J_\mathrm{m} = \det \mathbf{F}_\mathrm{m}$ and $I_1 = \operatorname{tr} \mathbf{C}_\mathrm{m}$ is the first invariant of the right Cauchy-Green tensor $\mathbf{C}_\mathrm{m} = \mathbf{F}_\mathrm{m}^T \cdot \mathbf{F}_\mathrm{m}$. Values of the material parameters were chosen in accordance with Boyce & Bertoldi [11] as $c_1 = 0.55\,\mathrm{MPa}$, $c_2 = 0.3\,\mathrm{MPa}$, $K = 55\,\mathrm{MPa}$. In the 3D case, $I_1 - 3$ in place of $I_1 - 2$ has to be used, as in the original Bertoldi-Boyce paper.

First-order Computational homogenization

The balance equation, neglecting any inertia effects, reads as

$$\vec{\nabla}_{0,\mathrm{m}} \cdot \mathbf{P}_\mathrm{m}^T(\mathbf{F}_\mathrm{m}(\vec{X})) = \vec{0} \quad \forall \vec{X} \in \Omega_0, \tag{3.5}$$

which was solved for the deformed microscopic positions $\vec{x}$. Here, Ω_0 denotes the RVE domain, and $\vec{\nabla}_{0,\mathrm{m}}\cdot$ denotes the divergence of the second order tensor field $\mathbf{P}_\mathrm{m}$ with respect to the initial microstructural position $\vec{x}_0$. We use the following definition of the $\vec{\nabla}_0$ operator:

$$\vec{a} = \vec{\nabla}_0 \cdot \mathbf{A} \quad \Longleftrightarrow \quad a_j = \sum_i \frac{\partial A_{ij}}{\partial x_{0,i}}. \tag{3.6}$$

We adopted the classical kinematical ansatz of first-order computational homogenization, expressed as:

$$\vec{x}(\vec{x}_0) = \mathbf{F} \cdot \vec{x}_0 + \vec{w}(\vec{x}_0), \quad \forall \vec{x}_0 \in \Omega_0, \tag{3.7}$$

where $\vec{w}(\vec{x}_0)$ is a periodic fluctuation field, which sufficed to ensure

$$\mathbf{F} = \frac{1}{|\Omega_0|} \int_{\Omega_0} (\vec{\nabla}_0 \vec{x})^T \,\mathrm{d}\vec{x}_0 \,. \tag{3.8}$$

The fluctuation field $\vec{w}(\vec{x}_0)$ was obtained by minimizing the strain energy of the RVE, which corresponds to ensuring the microscale equilibrium equation given by Equation (3.5) was satisfied in a weak sense.

In the numerical implementation, $\mathbf{F}$ acted as a directly prescribed load. The periodicity of $\vec{w}(\vec{x}_0)$ was enforced using a periodic projection of unique degrees of freedom, i.e., the image-source approach. In addition, we fixed one boundary node in place to eliminate rigid-body modes.

We solved for $\vec{w}$ incrementally, using the standard Newton-Raphson method combined with a line search. After each load increment, we checked the stability of a converged state by evaluating the eigenvalues of the tangent stiffness matrix. If instability was identified, we went back to the previous time step and tried again with

a smaller load increment, i.e., a smaller $\Delta\tau$. If $\Delta\tau$ fell below a predefined threshold (0.005, sometimes lower for the more tricky simulations), we assumed that the system was close to the critical point. We then perturbed the current state in a direction of the eigenmode corresponding to the smallest eigenvalue and let the system relax.

Once a stable solution for $\vec{w}$ was obtained, the homogenized $\mathfrak{W}$ and $\mathbf{P}$ were obtained as volume averages of their microscopic counterparts, i.e.,

$$\mathfrak{W} = \frac{1}{|\Omega_0|}\int_{\Omega_0} \mathfrak{W}_\mathrm{m}\left(\mathbf{F}_\mathrm{m}\left(\vec{x}\right)\right) \mathrm{d}\vec{x}_0, \tag{3.9}$$

$$\mathbf{P} = \frac{\partial \mathfrak{W}}{\partial \mathbf{F}} = \frac{1}{|\Omega_0|}\int_{\Omega_0} \mathbf{P}_\mathrm{m}\left(\mathbf{F}_\mathrm{m}\left(\vec{x}\right)\right) \mathrm{d}\vec{x}_0, \tag{3.10}$$

where $|\Omega_0|$ denotes the volume of the RVE in the reference configuration. The consistent homogenized stiffness $\mathbf{D}$ is defined as [65, p. 227]:

$${}^4\mathbf{D} = \frac{\partial^2 \mathfrak{W}}{\partial \mathbf{F} \partial \mathbf{F}} = \frac{\partial \mathbf{P}}{\partial \mathbf{F}}, \tag{3.11}$$

or equivalently

$$\delta\mathbf{P} = {}^4\mathbf{D} : \delta\mathbf{F}, \tag{3.12}$$

where the double-dot product between a 4^{th} and 2^{nd} order tensor is defined as

$$\mathbf{C} = {}^4\mathbf{A} : \mathbf{B} \quad \Leftrightarrow \quad C_{ij} = \sum_{k,l} A_{ijkl} B_{lk}. \tag{3.13}$$

Tensor ${}^4\mathbf{D}$ was obtained via static condensation via static condensation of the fluctuation degrees of freedom from an extended tangent stiffness matrix, which incorporates both the fluctuation field and the generalized degrees of freedom associated with the prescribed macroscopic deformation gradient $\mathbf{F}$. This procedure follows the approaches of, e.g., Kouznetsova, 2002 [66, §2.4.4] and Miehe, 2003 [67], simplified in our case by the explicit separation of degrees of freedom pertinent to the kinematic ansatz in Equation (3.7).

The FE simulations were performed using an in-house MATLAB code which is available at https://github.com/FHendriks11/mechmetamat_homogenization.

Sampling of the Deformation Gradient

Per geometry we sampled the RVE's response to 12 different prescribed loading paths, each defined by a different prescribed final macroscopic deformation gradient $\mathbf{F}$. Using the right polar decomposition, $\mathbf{F}$ can be decomposed into the symmetric macroscopic right-stretch tensor $\mathbf{U}$ and a rotation tensor $\mathbf{R}$, i.e., $\mathbf{F} = \mathbf{R} \cdot \mathbf{U}$. Because $\mathbf{R}$ encodes rotation, which does not meaningfully affect the results, we only sampled symmetric $\mathbf{F}$, which corresponds to prescribing the right-stretch tensor $\mathbf{U}$.

The sampling was carried out using the strategy introduced by Kunc and Fritzen [68], which samples a large variety of symmetric positive definite tensors in 2D. Each sample *btU* is defined by three independent parameters: volumetric deformation given

by $J = \det \mathbf{U}$, the orientation of the principal strains parameterized by the angle ϕ, and the deviatoric magnitude t of the deformation.

The right stretch tensor $\mathbf{U}$ (and from it $\mathbf{F}$) in its matrix-representation $\underline{\underline{U}}$ was then constructed as follows:

$$\underline{\underline{U}} = J^{1/2} \exp\left(t\left(\alpha \underline{\underline{Y}}^{(1)} + \beta \underline{\underline{Y}}^{(2)}\right)\right), \tag{3.14}$$

with

$$\underline{\underline{Y}}^{(1)} = \sqrt{\frac{1}{2}} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \text{and} \quad \underline{\underline{Y}}^{(2)} = \sqrt{\frac{1}{2}} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{3.15}$$

where $\alpha = \sin\phi$ and $\beta = \cos\phi$. For our sampling, we set $J \in [J_{\min}, J_{\max}]$, $t \in [0, t_{\max}]$ and $\phi \in [0, 2\pi]$. This sampling results in an isotropic distribution of principal directions.

Since all simulations were performed in load increments anyway, we sampled only the extreme values; i.e., $J = J_{min}$, $J = J_{max}$ or $t = t_{max}$. Because we performed 12 simulations for each geometry, we sampled four values of $\mathbf{F}$ with $J = J_{\min}$, four with $J = J_{\max}$ and four with $t = t_{\max}$. We chose $J_{\min}$ equal to the volume fraction of the geometry whose response we were computing, because at that point, virtually all of the deformed configurations include self-contact. Continuing beyond that point is not physically meaningful, since our simulations did not include contact. For $J_{\max}$, we chose 1.5, and $t_{\max}$ was set to 0.5. We then sampled four combinations of the other two parameters (either t and ϕ if J was at an extremum, or J and ϕ if t was at its maximum), by dividing the range of these parameters into two equal intervals and sampling with uniform probability from each combination of these intervals.

An example of the resulting set of deformations at the end of all twelve simulations for one geometry is provided in Figure 3.9, and examples of the resulting stress over the course of four different trajectories (each of a different geometry) are given in Figure 3.10.

3.3 Data Records

The dataset, available in a Zenodo repository (https://zenodo.org/records/15849549), contains 1,020 microstructural geometries in total: 60 distinct ones for each of the 17 wallpaper groups [54]. The dataset therefore consists of 1,020 .pkl and 1,020 .png files, which are placed into the subdirectories `pickle_files\` and `images\`. Each .pkl file contains data describing the geometry of one microstructure and the results of the 12 simulations of this geometry. Each .png file then contains a visualization of the geometry. The total size of the .pkl files is 47.2 GB, and 87.6 MB of .png files, making the total size of the dataset 47.3 GB.

Since we performed 12 simulations for each geometry, our dataset contains $12 \times 60 \times 17 = 12,240$ trajectories, each of which has on average 11.1 pseudo-time steps. Considering each pseudo-time step to be a separate load case, the dataset contains 135,947 microstructural responses to a prescribed $\mathbf{F}$ in total.

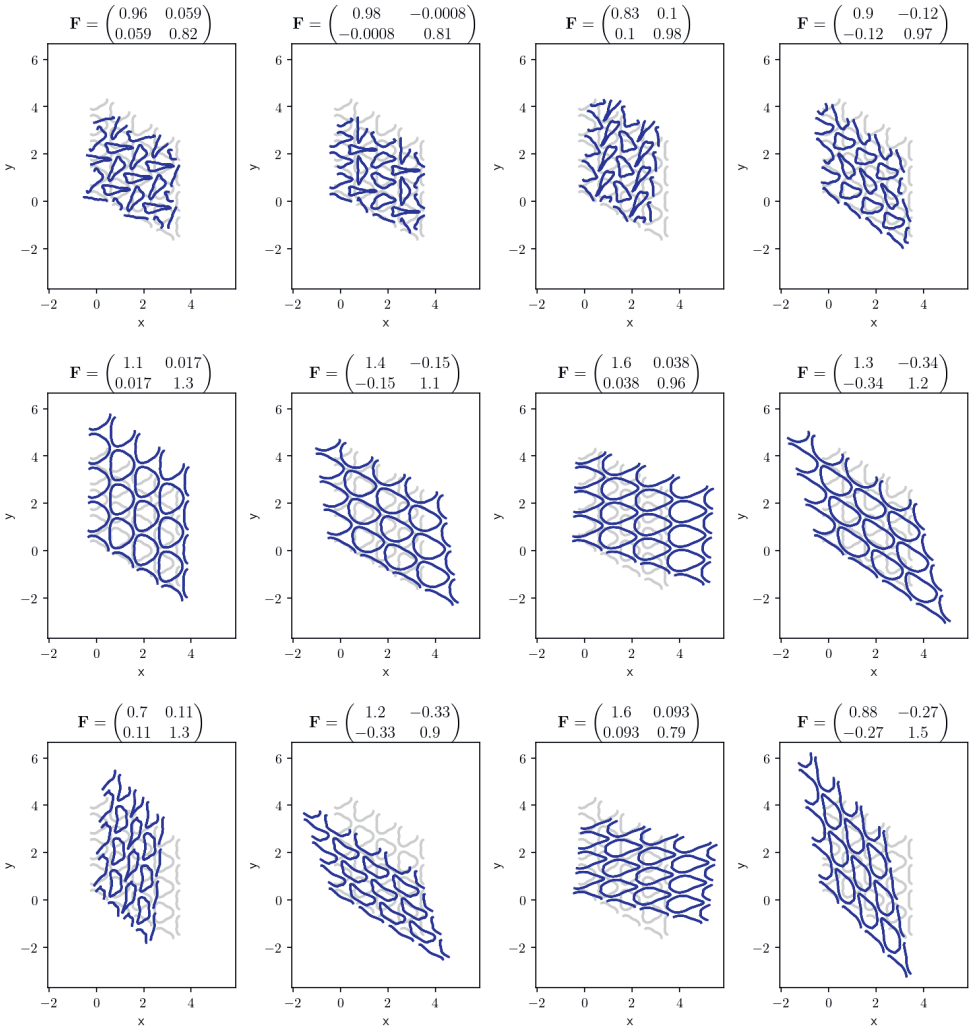


Figure 3.9: Deformed configurations (in blue) at the end of the 12 simulations of the same geometry with *p3m1* symmetry. The undeformed geometry is shown in grey. To allow for better understanding of the pattern transformation, the entire RVE is plotted 4 times per figure (2 repetitions along each lattice vector). The title of each subfigure indicates the deformation gradient **F** at the end of the simulation.

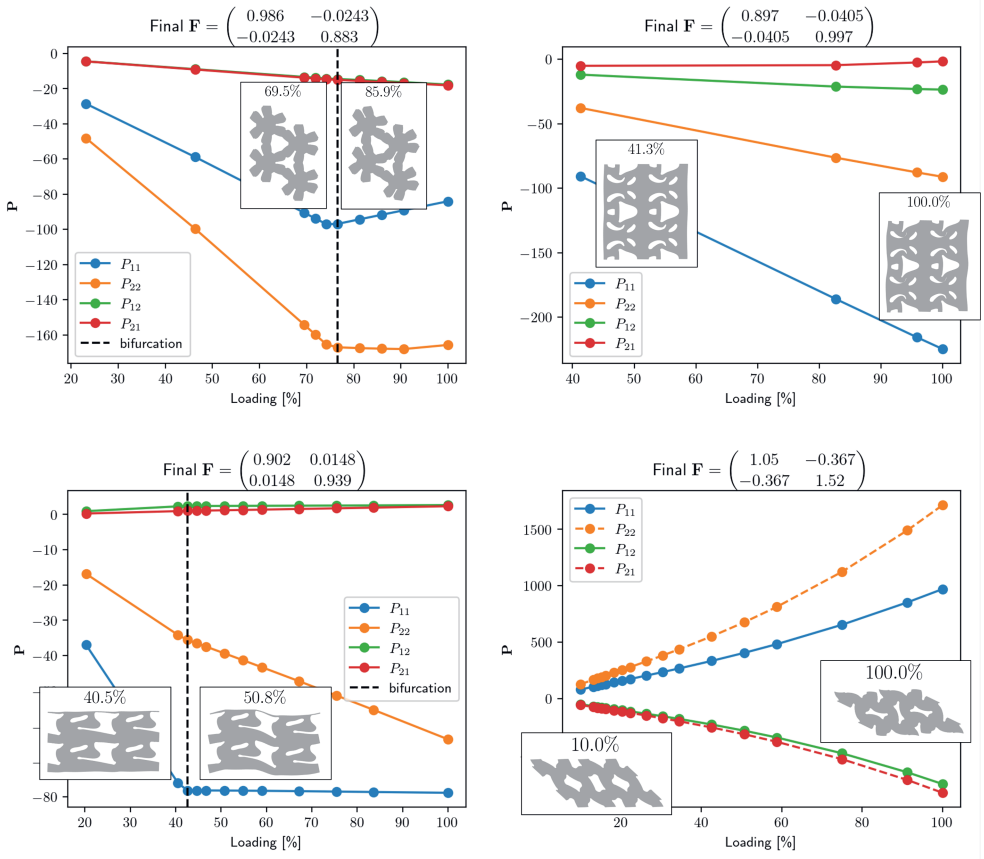


Figure 3.10: 1st Piola-Kirchhoff stress tensor components vs percentage loading over the course of a trajectory. Each figure represents the stress response of one material under the loading condition applied to it.

.pkl file

Each .pkl file contains a Python dictionary (dict), which contains multiple keys, whose values are dicts themselves. Figure 3.11 shows the structure of the dicts inside the .pkl file. The README markdown file in the Zenodo contains an exhaustive list of all the quantities in this dict and their meanings. The most important quantities are the inputs and outputs of the simulations, which are listed here.
Inputs:

- `data['simulations']['time_steps']['F']`: a numpy array of float64 values of shape (number of time steps, 2, 2), containing the deformation gradient $\mathbf{F}$ at each time step.
- `data['mesh']['RVE']['p']`: a numpy array of float64 values of shape (number of nodes, 2), containing the x, y-coordinates of the nodes of the RVE mesh.
- `data['mesh']['RVE']['t']`: a numpy array of int32 values of shape (number of elements, 6), containing the indices of the nodes of each triangular quadratic element.
- `data['mesh']['RVE']['source_nodes']`: list of uint16 values of indices of the boundary nodes that the `periodic_image_nodes` are dependent on.
- `data['mesh']['RVE']['image_nodes']`: list of uint16 values of indices of boundary nodes whose displacements are not solved for but instead calculated from their corresponding source node, in order to ensure periodic boundary conditions.

Outputs:

- `data['simulations']['time_steps']['W']`: a numpy array of float64 values of shape (number of time steps,), containing the strain energy density $\mathfrak{W}$ at each time step.
- `data['simulations']['time_steps']['P']`: a numpy array of float64 values of shape (number of time steps, 2, 2), containing the first Piola-Kirchhoff stress tensor $\mathbf{P}$ at each time step.
- `data['simulations']['time_steps']['D']`: a numpy array of float64 values of shape (number of time steps, 2, 2, 2, 2), containing the tangent stiffness $\mathbf{D}$ at each time step.
- `data['simulations']['D_ref']`: a numpy array of float64 values of shape (2, 2, 2, 2), containing the tangent stiffness $\mathbf{D}$ in the reference (undeformed) configuration.
- `data['simulations']['time_steps']['x']`: a numpy array of float64 values of shape (number of time steps, number of nodes, 2), containing the x, y-coordinates of the nodes of the RVE at each time step.

.png file

Each .png file contains a visualization of the geometry. The image always captures the whole RVE, of which one unit cell is indicated by thick black lines and a light grey background. In this unit cell, the mesh is shown as triangles of various colors. Within the unit cell, one fundamental domain is indicated, also with thick black lines and with a slightly darker grey background. Examples from all 17 wallpaper groups

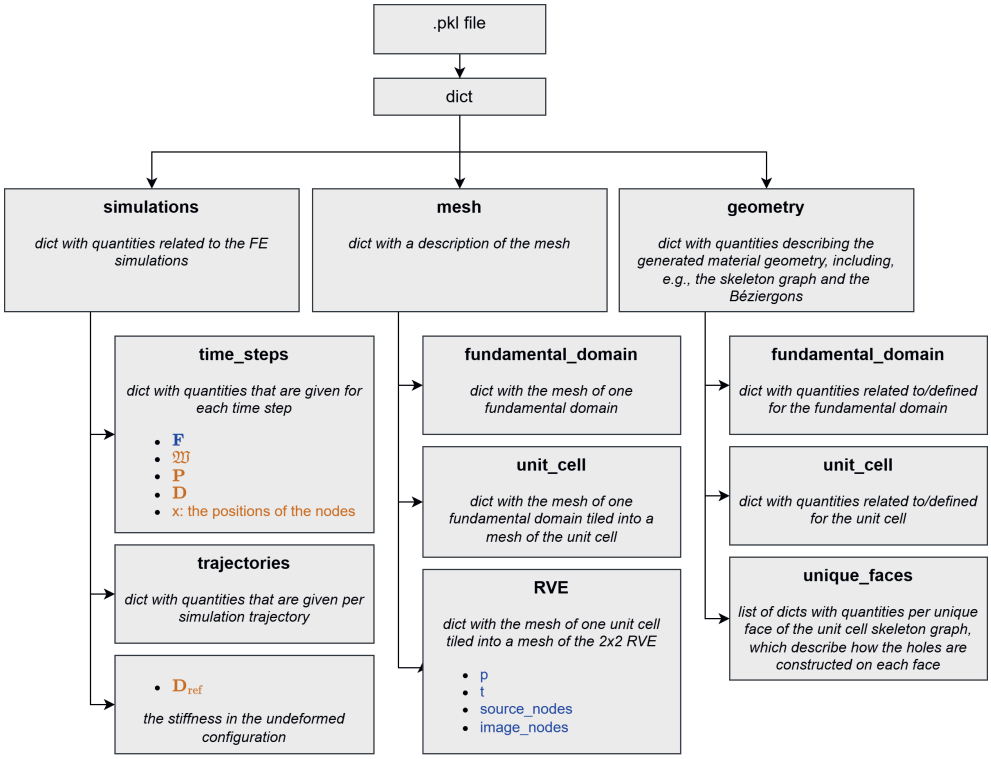


Figure 3.11: Hierarchical representation of the structure of the .pkl files. Quantities in blue are the inputs to the simulations, and quantities in orange are the outputs.

are shown in Figures 3.3, 3.4, 3.5 and 3.6.

3.4 Technical Validation

We thoroughly validated the microstructure generation algorithm, the discretization (checking both the validity of the mesh as well as the mesh refinement) and the simulations. The microstructure generation algorithm produces connected and periodic geometries. Both features are rigorously tested in the subsequent steps: periodicity is checked as a side effect when stitching together multiple copies of the fundamental domain to create the unit cell, and later 4 copies of the unit cell to create the RVE; if periodicity were not respected, this stitching together triggers an error. Similarly, if the geometry were not connected, the numerical simulations would result in an error.

Sometimes the generation algorithm fills too many graph faces, resulting in a very high volume fraction, which hampers buckling. For this reason, we discard geometries with a volume fraction above 0.75.

We prescribe the minimum thickness of the ligaments by controlling the distance between corners of the initial and secondary face polygon. However, since the final Bézier curves that specify the ligaments can 'bulge' out beyond the boundaries of the secondary face polygon, we added an extra check in the code that computes the distance between Bézier curves and discards the geometry if this distance is too small, ensuring the minimum thickness of the ligaments.

Very rarely, Gmsh generates invalid meshes with slightly overlapping elements. Therefore, the fundamental domain mesh was validated by checking for elements overlapping other elements or themselves. This was done by checking for crossing element edges. If any were found, the geometry was discarded.

To assess the necessary level of mesh refinement, we perform a mesh convergence study on 4 different geometries, containing a variety of features, including very thin ligaments and multiple holes. We then assess the accuracy of the calculations and their convergence.

For each geometry, we use a different applied deformation gradient $\mathbf{F}$ with $J = 0.75$, i.e., with significant volumetric compression, which are the most difficult type of simulations prone to involve buckling. We then calculate the relative error ϵ_P in the 1st Piola-Kirchhoff stress tensor $\mathbf{P}$ and ϵ_D in the tangent stiffness $\mathbf{D}$, compared to the mesh with the smallest element size. ϵ_P and ϵ_D are calculated as

$$\epsilon_P = \frac{\|\mathbf{P}^\tau - \mathbf{P}^\tau_{\text{fine}}\|}{\|\mathbf{P}^\tau_{\text{fine}}\|} \tag{3.16}$$

and

$$\epsilon_D = \frac{\|\mathbf{D}^\tau - \mathbf{D}^\tau_{\text{fine}}\|}{\|\mathbf{D}^0_{\text{fine}}\|}, \tag{3.17}$$

where $\|\cdot\|$ is the Frobenius norm, and τ is the pseudo-time. The subscript 'fine' indicates it is the finest mesh, for which the maximum characteristic element size $cl_{\max} = 0.0078$. Note that for ϵ_P, we normalize against $\|\mathbf{D}^0_{\text{fine}}\|$ instead of $\|\mathbf{D}^\tau_{\text{fine}}\|$, because the latter can attain very small values or even zero at the buckling point. The convergence study demonstrates that, for a mesh size of `cl_max` $= 0.044$, both

ϵ_P and ϵ_D are below 5% most of the time, even for these most challenging cases, and consistently below 10%, with the sole exception of ϵ_D close to the buckling point. At that point, there is a discontinuity in $\mathbf{D}$, and its location can shift with varying mesh sizes. Since a 5% to 10% accuracy is good enough for our purposes (neural network training and properly identifying the qualitative response of the material), we deem `cl_max` $= 0.044$ or lower acceptable. Hence, the adopted discretisation represents an acceptable compromise between the accuracy of the reported responses and the computational cost of generating the dataset. See Figure 3.12 for the relative error in $\mathbf{P}$ and $\mathbf{D}$ for one of the test geometries, over the course of the simulation.

Technical validation of the simulation code itself was carried out by comparing its results to the literature for two common geometries; the square stacked holes and the hexagonally stacked holes. Our code correctly reproduces the same pattern transformations as seen in the literature [69].

Since our simulation code does not take into account contact, we check for contact in the results during postprocessing, by checking for overlapping element edges. Cases with overlapping elements are removed, such that only the part of the trajectory before contact remains in the dataset.

3

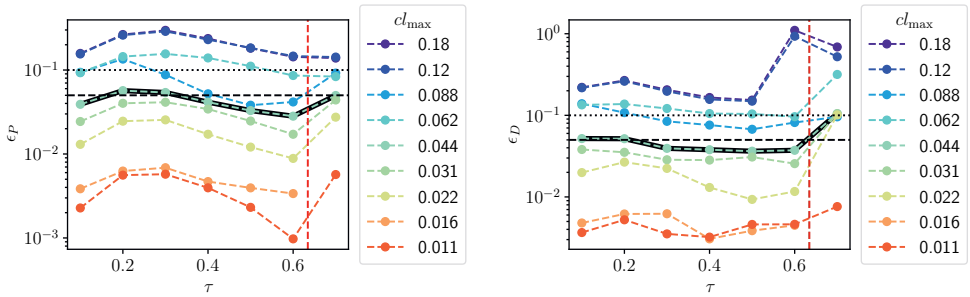
3.5 Usage Notes

A Jupyter notebook titled `Usage_notes.ipynb`, which is provided in the Zenodo repository, demonstrates use cases of the dataset. It shows how to load the data, how to access the different quantities in the data, and how to visualize the geometries. For example, it visualizes the mesh before and after deformation, and plots the homogenized quantities $\mathbf{P}$ and $\mathbf{D}$ as a function of the pseudo-time τ.

The envisioned use cases for the dataset are at least twofold: training machine learning models and studying symmetries in mechanical metamaterials.

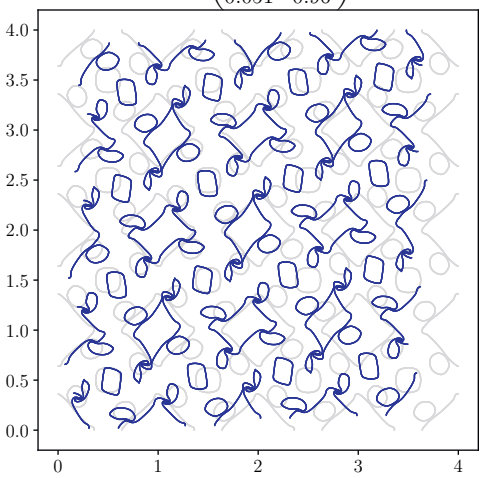
For the purpose of training machine learning models, we suggest using the data to create a graph representation suitable for a graph neural network, by removing at least the bulk nodes from the mesh and creating a graph from the remaining boundary nodes, as suggested by Hendriks et al [47], or from the Bézier control points. Additionally, the graph skeleton used to generate the geometry can be used in the graph representation.

For the purpose of studying how symmetries in a material relate to its mechanical properties, one can investigate the homogenized quantities for each symmetry in search for patterns in the dataset. For example, one can examine the auxeticity per wallpaper group. Auxeticity refers to a negative Poisson's ratio, where a material expands perpendicular to the direction of stretching and contracts perpendicular to the direction of compression. The Poisson's ratio can be calculated from the tangent stiffness tensor $\mathbf{D}$. In our dataset, we find that in the reference configuration, 306 out of the 1,020 geometries are auxetic in at least one direction, of which 110 geometries are auxetic in all directions. Code to plot the Poisson's ratio as a function of direction is provided in the `Usage_notes.ipynb` notebook, where the anisotropic Poisson's



(a) (b)

$$\mathbf{F} = \begin{pmatrix} 0.86 & 0.051 \\ 0.051 & 0.96 \end{pmatrix}$$



(c)

Figure 3.12: Relative error in (a) the homogenized 1st Piola-Kirchhoff stress tensor $\mathbf{P}$ and (b) tangent stiffness $\mathbf{D}$ for a geometry with *p4* symmetry, over the course of the simulation until contact, i.e., for $\tau \in [0, 0.70]$, for varying maximum characteristic element sizes $cl_{\max}$. The $cl_{\max} = 0.044$ closest to our final chosen maximum characteristic element size of 0.0408 is additionally indicated with a black line. Subfigure (c) shows the geometry of the test RVE, with the undeformed geometry in grey and the deformed geometry at the end of the simulation in blue. To allow for better understanding of the pattern transformation, the entire RVE is plotted 4 times (2 repetitions along each lattice vector). The title of subfigure (c) indicates the deformation gradient $\mathbf{F}$ at the end of the simulation when contact happens (at $\tau = 0.70$).

ratio is calculated as [70]:

$$\nu(\vec{n}, \vec{m}) = -\frac{1}{E(\vec{n})} s_{ij\alpha\beta} n_i n_j m_\alpha m_\beta, \tag{3.18}$$

$$E(\vec{n}) = s_{ij\alpha\beta} n_i n_j n_\alpha n_\beta, \tag{3.19}$$

with $\vec{n}$ a unit vector in the loading direction and $\vec{m}$ a unit vector perpendicular to $\vec{n}$ (only two directions possible in 2D, which will give the same result), and $s_{ij\alpha\beta}$ the components of the compliance tensor $\mathbf{S}$, which is the pseudo-inverse of the stiffness tensor $\mathbf{D}$.

One can also look at the tendency of each wallpaper group to generate materials that buckle, and which buckling modes show up. For example, Figure 3.13 shows the fraction of geometries that buckled for each wallpaper group, indicating large differences between the wallpaper groups in their tendency to buckle.

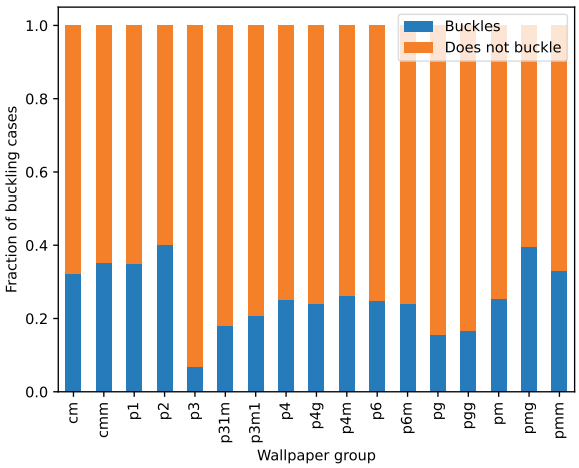


Figure 3.13: Fraction of trajectories that buckled for each wallpaper group, calculated as the number of trajectories that buckled divided by the total number of trajectories for that wallpaper group.

3.6 Code availability

The code used to generate the geometries is available at https://github.com/FHendriks11/wallpaper_microstructures. The code used to simulate the mechanical response is available at https://github.com/FHendriks11/mechmetamat_homogenization. Both repositories are available under the CC-BY 4.0 license.

We used Python 3.9.18, with the following package versions: scipy 1.13.1, scikit-learn 1.3.0 (solely for the `radius_neighbors_graph` for fast deduplication of points and checking of mesh or Bézier curve overlap), scikit-geometry 0.1.2 (solely for the straight skeleton), NumPy 1.26.4, meshio 5.3.4 (solely for reading gmsh mesh files)

and Matplotlib 3.9.2. We used Gmsh 4.12.2 to generate the meshes, and MATLAB R2021a to perform the simulations. These packages and software can be found at the following URLs:

- Python: https://www.python.org/
- SciPy: https://scipy.org/
- scikit-learn, `radius_neighbors_graph`: https://scikit-learn.org/1.3/modules/generated/sklearn.neighbors.radius_neighbors_graph.html
- scikit-geometry, straight skeleton: https://scikit-geometry.github.io/scikit-geometry/skeleton.html
- NumPy: https://numpy.org/
- meshio: https://github.com/nschloe/meshio
- Matplotlib: https://matplotlib.org/
- Gmsh: https://gmsh.info/
- MATLAB: https://www.mathworks.com/products/matlab.html

Bibliography

[1] Guell Izard, A. & Valdevit, L. Magnetoelastic Metamaterials for Energy Dissipation and Wave Filtering. *Advanced Engineering Materials* **22**, 1901019, https://doi.org/10.1002/adem.201901019 (2020).

[2] Ning, S. *et al.* The role of material and geometric nonlinearities and damping effects in designing mechanically tunable acoustic metamaterials. *International Journal of Mechanical Sciences* **197**, 106299, https://doi.org/10.1016/j.ijmecsci.2021.106299 (2021).

[3] Montgomery, S. M. *et al.* Magneto-Mechanical Metamaterials with Widely Tunable Mechanical Properties and Acoustic Bandgaps. *Advanced Functional Materials* **31**, 2005319, https://doi.org/10.1002/adfm.202005319 (2021).

[4] Wu, L. *et al.* A brief review of dynamic mechanical metamaterials for mechanical energy manipulation. *Materials Today* **44**, 168–193, https://doi.org/10.1016/j.mattod.2020.10.006 (2021).

[5] Yang, D. *et al.* Buckling of Elastomeric Beams Enables Actuation of Soft Machines. *Advanced Materials* **27**, 6323–6327, https://doi.org/10.1002/adma.201503188 (2015).

[6] Terryn, S., Brancart, J., Lefeber, D., Van Assche, G. & Vanderborght, B. Self-healing soft pneumatic robots. *Science Robotics* **2**, 16, https://doi.org/10.1126/scirobotics.aan4268 (2017).

[7] Kim, Y., Parada, G. A., Liu, S. & Zhao, X. Ferromagnetic soft continuum robots. *Science Robotics* **4**, 7329, https://doi.org/10.1126/SCIROBOTICS.AAX7329 (2019).

[8] Veerabagu, U., Palza, H. & Quero, F. Auxetic polymer-based mechanical metamaterials for biomedical applications. *ACS Biomaterials Science & Engineering* **8**, 2798–2824, https://doi.org/10.1021/acsbiomaterials.2c00109 (2022).

[9] Babaee, S. *et al.* 3D soft metamaterials with negative poisson's ratio. *Advanced Materials* **25**, 5044–5049, https://doi.org/10.1002/adma.201301986 (2013).

[10] Nicolaou, Z. G. & Motter, A. E. Mechanical metamaterials with negative compressibility transitions. *Nature Materials* **11**, 608–613, https://doi.org/10.1038/nmat3331 (2012). 1207.2185.

[11] Bertoldi, K., Boyce, M. C., Deschanel, S., Prange, S. M. & Mullin, T. Mechanics of deformation-triggered pattern transformations and superelastic behavior in periodic elastomeric structures. *Journal of the Mechanics and Physics of Solids* **56**, 2642–2668, https://doi.org/10.1016/j.jmps.2008.03.006 (2008).

[12] Overvelde, J. T. & Bertoldi, K. Relating pore shape to the non-linear response of periodic elastomeric structures. *Journal of the Mechanics and Physics of Solids* **64**, 351–366, https://doi.org/10.1016/j.jmps.2013.11.014 (2014).

[13] Kim, Y., Yuk, H., Zhao, R., Chester, S. A. & Zhao, X. Printing ferromagnetic domains for untethered fast-transforming soft materials. *Nature* **558**, 274–279, https://doi.org/10.1038/s41586-018-0185-0 (2018).

[14] Shim, J. *et al.* Harnessing instabilities for design of soft reconfigurable auxetic/chiral materials. *Soft Matter* **9**, 8198–8202, https://doi.org/10.1039/c3sm51

148k (2013).

[15] Azulay, R., Combescure, C. & Dirrenberger, J. Instability-induced pattern generation in architectured materials—a review of methods. *International Journal of Solids and Structures* **274**, 112240, https://doi.org/10.1016/j.ijsolstr.2023.112240 (2023).

[16] Azulay, R. & Combescure, C. Predicting the post-bifurcated patterns of architectured materials using group-theoretic tools. *Journal of the Mechanics and Physics of Solids* **187**, 105631 (2024).

[17] International Union of Crystallography. International tables for crystallography, index of the 17 plane groups. In *International Tables for Crystallography*, chap. 2.2, 175–192, https://doi.org/10.1107/9780955360206000092 7 (International Union of Crystallography, 2016).

[18] Schattschneider, D. The plane symmetry groups: their recognition and notation. *The American Mathematical Monthly* **85**, 439–450 (1978).

[19] Jin, S., Korkolis, Y. P. & Li, Y. Shear resistance of an auxetic chiral mechanical metamaterial. *International Journal of Solids and Structures* **174-175**, 28–37, https://doi.org/10.1016/j.ijsolstr.2019.06.005 (2019).

[20] Han, D. *et al.* Lightweight auxetic metamaterials: Design and characteristic study. *Composite Structures* **293**, 115706 (2022).

[21] Kittel, C. & McEuen, P. *Introduction to solid state physics* (John Wiley & Sons, 2018).

[22] Maurin, F., Claeys, C., Deckers, E. & Desmet, W. Probability that a band-gap extremum is located on the irreducible Brillouin-zone contour for the 17 different plane crystallographic lattices. *International Journal of Solids and Structures* **135**, 26–36, https://doi.org/10.1016/j.ijsolstr.2017.11.006 (2018).

[23] Moore, D. B., Starkey, T. A. & Chaplain, G. J. Acoustic metasurfaces with Frieze symmetries. *The Journal of the Acoustical Society of America* **155**, 568–574, https://doi.org/10.1121/10.0024359 (2024).

[24] Liu, H., Plucinsky, P., Feng, F. & James, R. D. Origami and materials science. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **379**, https://doi.org/10.1098/rsta.2020.0113 (2021). 2008.06026.

[25] Liu, H. & James, R. D. Design of origami structures with curved tiles between the creases. *Journal of the Mechanics and Physics of Solids* **185**, 105559, https://doi.org/10.1016/j.jmps.2024.105559 (2024).

[26] Wang, L., Chan, Y. C., Liu, Z., Zhu, P. & Chen, W. Data-driven metamaterial design with Laplace-Beltrami spectrum as "shape-DNA". *Structural and Multidisciplinary Optimization* **61**, 2613–2628, https://doi.org/10.1007/s00158-020-02523-5 (2020).

[27] Chan, Y. C., Ahmed, F., Wang, L. & Chen, W. METASET: Exploring shape and property spaces for data-driven metamaterials design. *Journal of Mechanical Design* **143**, https://doi.org/10.1115/1.4048629 (2021). 2006.02142.

[28] Bastek, J. H. & Kochmann, D. M. Inverse design of nonlinear mechanical metamaterials via video denoising diffusion models. *Nature Machine Intelligence* **5**, 1466–1475, https://doi.org/10.1038/s42256-023-00762-x (2023). 2305.19836.

[29] Kollmann, H. T., Abueidda, D. W., Koric, S., Guleryuz, E. & Sobh, N. A. Deep

learning for topology optimization of 2D metamaterials. *Materials and Design* **196**, 109098, https://doi.org/10.1016/j.matdes.2020.109098 (2020).

[30] Lyu, X. & Ren, X. Microstructure reconstruction of 2D/3D random materials via diffusion-based deep generative models. *Scientific Reports* **14**, 5041, https://doi.org/10.1038/s41598-024-54861-9 (2024). 2311.17319.

[31] Ohno, N., Okumura, D. & Noguchi, H. Microscopic symmetric bifurcation condition of cellular solids based on a homogenization theory of finite deformation. *Journal of the Mechanics and Physics of Solids* **50**, 1125–1153, https://doi.org/10.1016/S0022-5096(01)00106-5 (2002).

[32] Vangelatos, Z., Komvopoulos, K. & Grigoropoulos, C. Regulating the mechanical behavior of metamaterial microlattices by tactical structure modification. *Journal of the Mechanics and Physics of Solids* **144**, 104112 (2020).

[33] Tyburec, M., Doškář, M., Somr, M., Kružík, M. & Zeman, J. Modular-topology optimization for additive manufacturing of reusable mechanisms. *Computers & Structures* **307**, 107630, https://doi.org/10.1016/j.compstruc.2024.107630 (2025).

[34] Zheng, L., Karapiperis, K., Kumar, S. & Kochmann, D. M. Unifying the design space and optimizing linear and nonlinear truss metamaterials by generative modeling. *Nature Communications* **14**, 1–14, https://doi.org/10.1038/s41467-023-42068-x (2023).

[35] Frankel, A. L., Jones, R. E., Alleman, C. & Templeton, J. A. Predicting the mechanical response of oligocrystals with deep learning. *Computational Materials Science* **169**, https://doi.org/10.1016/j.commatsci.2019.109099 (2019). 1901.10669.

[36] Wilt, J. K., Yang, C. & Gu, G. X. Accelerating Auxetic Metamaterial Design with Deep Learning. *Advanced Engineering Materials* **22**, 1901266, https://doi.org/10.1002/adem.201901266 (2020).

[37] Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A. & Battaglia, P. Learning mesh-based simulation with graph networks. In *International conference on learning representations* (2020).

[38] Vlassis, N. N., Ma, R. & Sun, W. Geometric deep learning for computational mechanics Part I: anisotropic hyperelasticity. *Computer Methods in Applied Mechanics and Engineering* **371**, 113299, https://doi.org/10.1016/j.cma.2020.113299 (2020). 2001.04292.

[39] Pandey, A. & Pokharel, R. Machine learning based surrogate modeling approach for mapping crystal deformation in three dimensions. *Scripta Materialia* **193**, 1–5, https://doi.org/10.1016/j.scriptamat.2020.10.028 (2021).

[40] Yang, Z., Yu, C. H. & Buehler, M. J. Deep learning model to predict complex stress and strain fields in hierarchical composites. *Science Advances* **7**, https://doi.org/10.1126/SCIADV.ABD7416 (2021).

[41] Mianroodi, J. R., H. Siboni, N. & Raabe, D. Teaching solid mechanics to artificial intelligence—a fast solver for heterogeneous materials. *npj Computational Materials* **7**, https://doi.org/10.1038/s41524-021-00571-z (2021). 2103.09147.

[42] Mianroodi, J. R., Rezaei, S., Siboni, N. H., Xu, B. X. & Raabe, D. Lossless multi-scale constitutive elastic relations with artificial intelligence. *npj Computational Materials* **8**, 1–12, https://doi.org/10.1038/s41524-022-00753-3 (2022). 2108.0

2837.

[43] Khorrami, M. S. *et al.* An artificial neural network for surrogate modeling of stress fields in viscoplastic polycrystalline materials. *npj Computational Materials* **9**, 1–10, https://doi.org/10.1038/s41524-023-00991-z (2023). 2208.13490.

[44] Bastek, J. H., Kumar, S., Telgen, B., Glaesener, R. N. & Kochmann, D. M. Inverting the structure–property map of truss metamaterials by deep learning. *Proceedings of the National Academy of Sciences of the United States of America* **119**, e2111505119, https://doi.org/10.1073/pnas.2111505119 (2022).

[45] Thomas, A. *et al.* Materials fatigue prediction using graph neural networks on microstructure representations. *Scientific Reports 2023 13:1* **13**, 1–16, https://doi.org/10.1038/s41598-023-39400-2 (2023).

[46] Karapiperis, K. & Kochmann, D. M. Prediction and control of fracture paths in disordered architected materials using graph neural networks. *Communications Engineering* **2**, https://doi.org/10.1038/s44172-023-00085-0 (2023).

[47] Hendriks, F., Menkovski, V., Doškář, M., Geers, M. G. & Rokoš, O. Similarity equivariant graph neural networks for homogenization of metamaterials. *Computer Methods in Applied Mechanics and Engineering* **439**, 117867, https://doi.org/10.1016/j.cma.2025.117867 (2025). arXiv:2404.17365.

[48] Sigmund, O. Materials with prescribed constitutive parameters: An inverse homogenization problem. *International Journal of Solids and Structures* **31**, 2313–2329, https://doi.org/10.1016/0020-7683(94)90154-6 (1994).

[49] Andreassen, E., Lazarov, B. S. & Sigmund, O. Design of manufacturable 3D extremal elastic microstructure. *Mechanics of Materials* **69**, 1–10, https://doi.org/10.1016/j.mechmat.2013.09.018 (2014).

[50] Chen, Q., Zhang, X. & Zhu, B. Design of buckling-induced mechanical metamaterials for energy absorption using topology optimization. *Structural and Multidisciplinary Optimization* **58**, 1395–1410, https://doi.org/10.1007/s00158-018-1970-y (2018).

[51] Thomsen, C. R., Wang, F. & Sigmund, O. Buckling strength topology optimization of 2D periodic materials based on linearized bifurcation analysis. *Computer Methods in Applied Mechanics and Engineering* **339**, 115–136, https://doi.org/10.1016/j.cma.2018.04.031 (2018).

[52] Wang, F. & Sigmund, O. 3D architected isotropic materials with tunable stiffness and buckling strength. *Journal of the Mechanics and Physics of Solids* **152**, 104415, https://doi.org/10.1016/j.jmps.2021.104415 (2021).

[53] Xue, T. & Mao, S. Mapped shape optimization method for the rational design of cellular mechanical metamaterials under large deformation. *International Journal for Numerical Methods in Engineering* **123**, 2357–2380, https://doi.org/10.1002/nme.6941 (2022).

[54] Hendriks, F. *et al.* Wallpaper group-based mechanical metamaterials and their mechanical responses. *Zenodo* https://zenodo.org/records/15849549, (2025).

[55] Abu-Mualla, M. & Huang, J. A Dataset Generation Framework for Symmetry-Induced Mechanical Metamaterials. *Journal of Mechanical Design* 1–24, https://doi.org/10.1115/1.4066169 (2024).

[56] Karimi Mahabadi, R. *et al.* Graph-based design of irregular metamaterials. *International Journal of Mechanical Sciences* **295**, 110203, https://doi.org/10.101

6/j.ijmecsci.2025.110203 (2025).

[57] Cacciola, F., Loriot, S. & Rouxel-Labbé, M. 2D straight skeleton and polygon offsetting. In *CGAL User and Reference Manual* (CGAL Editorial Board, 2024), 6.0.1 edn.

[58] Geuzaine, C. & Remacle, J. F. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering* **79**, 1309–1331, https://doi.org/10.1002/nme.2579 (2009).

[59] Polukhov, E., Vallicotti, D. & Keip, M. A. Computational stability analysis of periodic electroactive polymer composites across scales. *Computer Methods in Applied Mechanics and Engineering* **337**, 165–197, https://doi.org/10.1016/j.cma.2018.01.020 (2018).

[60] Geymonat, G., Müller, S. & Triantafyllidis, N. Homogenization of nonlinearly elastic materials, microscopic bifurcation and macroscopic loss of rank-one convexity. *Archive for Rational Mechanics and Analysis* **122**, 231–290, https://doi.org/10.1007/BF00380256 (1993).

[61] Triantafyllidis, N., Nestorović, M. D. & Schraad, M. W. Failure surfaces for finitely strained two-phase periodic solids under general in-plane loading. *Journal of Applied Mechanics, Transactions ASME* **73**, 505–515, https://doi.org/10.1115/1.2126695 (2006).

[62] Zhang, G., Feng, N. & Khandelwal, K. A computational framework for homogenization and multiscale stability analyses of nonlinear periodic materials. *International Journal for Numerical Methods in Engineering* **122**, 6527–6575, https://doi.org/10.1002/nme.6802 (2021).

[63] Geers, M. G. D., Kouznetsova, V. G. & Brekelmans, W. A. M. Multi-scale computational homogenization: Trends and challenges. *Journal of Computational and Applied Mathematics* **234**, 2175–2182, https://doi.org/10.1016/j.cam.2009.08.077 (2010).

[64] Wriggers, P. *Nonlinear finite element methods* (Springer Science & Business Media, 2008).

[65] Tadmor, E. B., Miller, R. E. & Elliott, R. S. *Continuum Mechanics and Thermodynamics* (Cambridge University Press, 2012).

[66] Kouznetsova, V. G. *Computational homogenization for the multi-scale analysis of multi-phase materials*. Ph.D. thesis, Eindhoven University of Technology (2002). https://doi.org/10.6100/IR560009.

[67] Miehe, C. Computational micro-to-macro transitions for discretized microstructures of heterogeneous materials at finite strains based on the minimization of averaged incremental energy. *Computer Methods in Applied Mechanics and Engineering* **192**, 559–591, https://doi.org/10.1016/S0045-7825(02)00564-9 (2003).

[68] Kunc, O. & Fritzen, F. Finite Strain Homogenization Using a Reduced Basis and Efficient Sampling. *Mathematical and Computational Applications* **24**, 56, https://doi.org/10.3390/mca24020056 (2019). 1904.01521.

[69] Rokoš, O., Ameen, M. M., Peerlings, R. H. J. & Geers, M. G. D. Extended micromorphic computational homogenization for mechanical metamaterials exhibiting multiple geometric pattern transformations. *Extreme Mechanics Letters*

37, https://doi.org/10.1016/j.eml.2020.100708 (2020). 2004.05226.

[70] Gorodtsov, V. A. & Lisovenko, D. S. Extreme values of young's modulus and poisson's ratio of hexagonal crystals. *Mechanics of Materials* **134**, 1–8 (2019).

Appendices

3.A Periodic graph connectivity check

Checking the connectivity of a periodic graph is not as simple as checking connectivity of the fundamental domain graph; a connected fundamental domain graph is neither necessary nor sufficient for a connected infinitely tiled graph. For example, Figure 3.A.1 shows a periodic graph which is not connected inside the fundamental domain, but its infinitely tiled graph *is* connected. To assess the connectivity of the infinite graph we do the following:

1. Make sure the fundamental domain shape has the smallest number of vertices possible. For example, the symmetry group *pmg* allows for either a triangular or a rectangular fundamental domain, so we opt for the triangle. If a rectangular fundamental domain was used when a triangular one was possible, convert it to a triangular one.
2. Check if every boundary has at least one point on it which has at least one edge connected to it. If not, the graph is not connected.
3. Check if the fundamental domain graph is connected.
4. If yes, the graph is connected.
5. If yes to (1), but no to (2), we check if there are equivalent pairs of nodes that are connected on one boundary but not on another. For example, if there are two nodes A and B on the same fundamental domain boundary, and this boundary is equivalent to another boundary because of the symmetry of the tiling, then on that equivalent boundary there must be an equivalent pair of nodes A' and B' (see Figure 3.A.1). Then if A is connected to B, but A' is not connected to B' on this fundamental domain, then in the infinitely tiled graph, A' and B' will also be connected, but through another copy of the fundamental domain.
6. If we find such a pair of nodes A' and B', add an edge between them, and check again if the fundamental domain is connected.
7. Repeat from (2) until there are no such pairs.
8. If the fundamental domain graph is still not connected, the periodic graph is not connected.

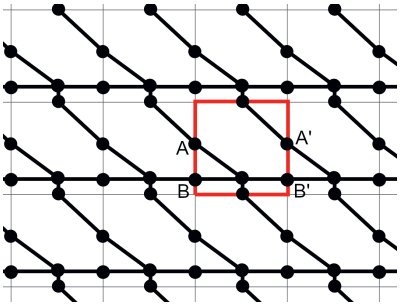


Figure 3.A.1: A periodic graph of wallpaper group *p1*, with the fundamental domain (equal to the unit cell in this case) outlined in red. Points A and B are connected by the graph inside the fundamental domain, but their periodic equivalent points A' and B' are not.

CHAPTER 4

Equivariant Flow Matching for Symmetry-Breaking Bifurcation Problems

Reproduced from Hendriks, F., Rokoš, O., Doškář, M., Geers, M. G., & Menkovski, V. (2025). Equivariant Flow Matching for Symmetry-Breaking Bifurcation Problems. Neurips workshop Machine Learning and the Physical Sciences.

Bifurcation phenomena in nonlinear dynamical systems often lead to multiple coexisting stable solutions, particularly in the presence of symmetry breaking. Deterministic machine learning models are unable to capture this multiplicity, averaging over solutions and failing to represent lower-symmetry outcomes. In this work, we formalize the use of generative AI, specifically flow matching, as a principled way to model the full probability distribution over bifurcation outcomes. Our approach builds on existing techniques by combining flow matching with equivariant architectures and an optimal-transport-based coupling mechanism. We generalize equivariant flow matching to a symmetric coupling strategy that aligns predicted and target outputs under group actions, allowing accurate learning in equivariant settings. We validate our approach on a range of systems, from simple conceptual systems to physical problems such as buckling beams and the Allen–Cahn equation. The results demonstrate that the approach accurately captures multimodal distributions and symmetry-breaking bifurcations. Moreover, our results demonstrate that flow matching significantly outperforms non-probabilistic and variational methods. This offers a principled and scalable solution for modeling multistability in high-dimensional systems.

4.1 Introduction

Machine learning enables efficient data-driven modeling of complex systems, especially when traditional methods impose high complexity or are incomplete [7,8,13,22,25,32]. Many such systems exhibit bifurcations that mark sudden changes in system behavior due to a small change in a control parameter. Bifurcations are crucial in fields such as fluid dynamics [28], climate science [2,27,34,38], biology [29], and crowd dynamics [12,24,37], where predicting transitions is essential. Many bifurcating systems exhibit multistability, where multiple distinct stable states coexist under the same input parameters, and the number of these states can change as the parameters vary. This phenomenon often arises due to symmetry breaking, where a system's state after the bifurcation possesses fewer symmetries than before. This results in a multiplicity of possible solutions, all of which are equally valid.

For example, a fluid flow at high Reynolds number can start oscillating, break-

ing time-translation symmetry [28]. A symmetric structure under compression can buckle in multiple directions, breaking rotation and reflection symmetry [4]. An even mixture of phases can separate into distinct regions, breaking translational and reflection symmetry. Symmetry breaking is also central to magnetism, superconductivity, and particle physics [5, 35], as well as the formation of enantiomers [18]. In mechanical metamaterials, buckling is a design feature that enables programmable shape transformations and tunable mechanical responses through controlled symmetry breaking [3, 16].

Simulating such systems with machine learning is nontrivial and remains an open problem [40, §2.9.1]. Deterministic machine learning models can only produce the average of multiple solutions, resulting in nonphysical predictions that do not represent any true solution. Geometric deep learning approaches that rely on equivariant models preserve the system's symmetry but cannot select asymmetric outcomes, limiting their ability to capture bifurcations. Approaches such as canonicalization [20], input perturbation [13], model ensembles [41], deterministic machine learning models combined with pseudo-arclength continuation [9, 11], and sampling group operations to artificially break symmetry [20, 39] offer partial solutions but often lack generalizability or precision.

We propose modeling the full output distribution using generative modeling. Generative models aim to learn mappings from simple base distributions (e.g., Gaussian noise) to complex, high-dimensional target distributions. We use generative modeling to parameterize the distribution of possible bifurcation outcomes given the input parameters.

However, the probability distributions we require are singular: the set of allowed values (the support) lies on a subspace that has lower dimensionality than the input space, e.g., multiple Dirac deltas in 1D, or a circle in 2D space. This means the probability mass must be concentrated as sharply as possible around the allowed values. When the target distribution is singular, the necessary mapping becomes highly nonlinear: nearby points in the source distribution may map to distant points in the target space. Capturing such mappings is challenging for neural networks, as it requires representing high-frequency functions [26]. This is a fundamental limitation of direct generative approaches, which must learn a single, highly nonlinear transformation. For example, variational autoencoders (VAE) have a tendency to create 'bridges' between modes, which is a common failure mode when modeling highly concentrated, multimodal distributions [30]. For this reason, they also tend to generate blurry images [6, 10, 19].

In contrast, iterative methods such as diffusion models, denoising processes, and flow matching approximate this complex mapping as a sequence of small integration steps. This iterative structure distributes the required nonlinearity across many stages, making the learning problem more tractable, as each step only needs to model a small, smooth transformation. As a result, these methods are better suited for (i) modeling singular distributions, where the probability mass is concentrated on a low-dimensional manifold, and (ii) multimodal distributions, where the probability mass must split far apart [30].

Therefore, to address the challenge of modeling multistability, we use flow matching [21] to train a stochastic interpolant [1] to model the full output distribution. We

show that flow matching excels at approximating distributions whose probability mass is highly concentrated in a low-dimensional subspace (e.g., very close to two Dirac deltas). We further systematically integrate flow matching with equivariant modeling to tackle symmetry-breaking. Towards this end, we develop *symmetric coupling*, an approach to determine the optimal training target, similar to minibatch optimal transport [36] and equivariant flow matching [17, 33], that exploits the self-similarity of the input. This approach substantially improves the training and the quality of generated flow paths. We demonstrate our method on abstract and physical systems, including coin flips, buckling beams, and phase separation via the Allen–Cahn equation. Our approach effectively captures bifurcations and multistability, with and without symmetry breaking.

The code is available at [githublinkwillbeadded] [see supplementary material for anonymous repo].

4.2 Equivariance and Symmetry Breaking

Equivariance A map $f : \mathcal{X} \to \mathcal{Y}$ is equivariant with respect to a group G if:

$$g \cdot y = f(g \cdot x), \quad \forall x \in \mathcal{X}, \forall y \in \mathcal{Y}, \forall g \in G. \tag{4.1}$$

If an input x is invariant (i.e., self-similar) under a subset of the group actions (i.e., $g_x \cdot x = x$ for all $g_x \in G_x$, with $G_x \subseteq G$), then equivariance implies that output y is also invariant (i.e., $g_x \cdot y = y$). However, this preservation of symmetry inherently prevents a model from expressing symmetry-breaking behavior. To capture bifurcations and transitions into asymmetric states, equivariant models must be adapted to allow for symmetry-breaking solutions, which means the traditional equivariance condition in Eq. (4.1) cannot hold for individual outputs after the bifurcation.

If there is symmetry-breaking in an equivariant system, there are multiple coexisting solutions for the same input. Meaning, if y is a solution, then so are all $g_x \cdot y$, $\forall g_x \in G_x$ (i.e., the orbit of y under G_x).

The equivariance in Eq. (4.1) then still holds for the set of solutions $\{g_x \cdot y | g_x \in G_x\}$. This is similar to the definition of "relaxed equivariance" by Kaba et al. [14, 15] in that both account for input symmetries via stabilizer subgroups (i.e., groups G_x whose actions leave the input x invariant), but it differs by preserving equivariance over sets of outputs (i.e., orbits of solutions), whereas relaxed equivariance modifies the equivariance condition itself, by allowing a correction with an element from the input's stabilizer subgroup.

As a probability distribution When using a generative model, we consider the set of solutions $\{y\}$ as a singular probability distribution $p(y|x)$, where the support of the distribution is the set of allowed solutions. The equivariance condition in Eq. (4.1) then translates to:

$$p(y|x) = p(g \cdot y | g \cdot x) \quad \forall x \in \mathcal{X}, y \in \mathcal{Y}, g \in G. \tag{4.2}$$

To ensure that the predictions by the model satisfy Eq. (4.2), we use (i) a model that is equivariant under G to parametrize the probability distribution and (ii) a

prior $p(y_0)$ that is invariant under G. This ensures that the generated probability distribution respects the symmetry of the problem [23].

Flow matching To model this probability distribution, we use flow matching [21], which learns a vector field $u(y_t, t, x)$ that transforms samples from an uninformed prior $p(y_0)$ into samples y_1 from the target distribution $p(y|x)$ over a continuous pseudo-time variable $t \in [0, 1]$. The vector field is learned by minimizing the expected squared error between the model prediction and the target vector field along linear interpolation paths between prior and target samples.

Symmetric coupling Additionally, we leverage the fact that there are multiple equivalent outputs to improve our predictions, using *symmetric coupling*. During training, for each latent sample y_0 and target y_1, we find the closest equivalent under G_x to the current prediction, i.e., find

$$y_1' = \tilde{g}_x \cdot y_1 \tag{4.3}$$

$$\tilde{g}_x = \text{argmin}_{g_x \in G_x} c(y_0, g_x \cdot y_1), \tag{4.4}$$

where c is the cost function, for which the Euclidean distance squared $c = ||y_0 - y_1||^2$ is a possible choice.

This is a way of straightening the flow paths similar to minibatch optimal transport [36], but instead of finding the optimal coupling between a minibatch of the prior samples and target samples, we find the optimal coupling between one prior sample y_0 and all symmetric equivalents $g_x \cdot y_1, g_x \in G_x$ of its corresponding target sample y_1. Our approach is similar to equivariant flow matching as described by Klein [17] and Song [33], except we only consider the group G_x that leaves the input x invariant. If this group contains permutations or rotations, we can therefore follow their lead and use the Hungarian algorithm for permutations or the Kabsch algorithm for rotations. Other options are checking all possible reflections of the target or using FFT cross-correlation to determine the optimal periodic translation of the target.

See Fig. 4.1(a) for a visualization of the concepts of equivariance and self-similarity in the context of a buckling beam problem.

4.3 Experiments and Results

Here, we showcase the performance of the proposed approach with four toy problems and two physically grounded examples.

All models were trained with the Adam optimizer.

4.3.1 Gaussian to 2 Dirac Deltas

This toy problem demonstrates the ability of flow matching to capture multimodal distributions with very sharp peaks. There is no fixed dataset size because, during training, we continuously sample new data points. The input x_i is sampled from

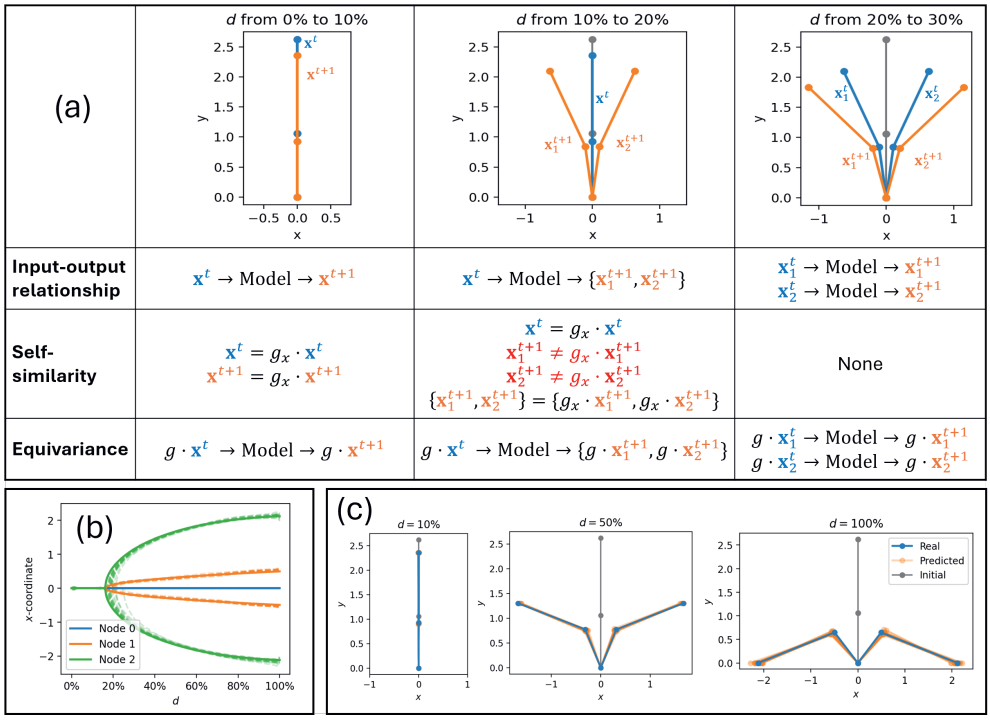


Figure 4.1: (a) Self-similarity and equivariance illustrated with the buckling beam problem, in which the input is the state at the current time step t and the target output is the next time step $t+1$. d is the vertical downward displacement of the beam tip relative to the total length of the beam, which is increased over time. The relevant symmetry group G here consists of only two elements: identity and reflection in the y-axis. In this case, the self-similarity of the beam before buckling happens to be described by the same group, i.e., here $G_x = G$. (b) The x-coordinates of each node over the course of a trajectory, showing 50 predictions of a trained flow matching model (dashed lines) compared to the two possible ground truths (solid lines). (c) The same 50 predictions compared to the two possible ground truths, showing the beam deformation.

a Gaussian distribution $\mathcal{N}(0, 1)$; the output y_i is sampled from $\{-1, 1\}$ with equal probability, or equivalently from $p(y_i) = 0.5\delta(y_i - 1) + 0.5\delta(y_i + 1)$.

For the VAE, we used an MLP with 2 hidden layers of 64 neurons per layer as an encoder and another MLP as a decoder, with a latent dimension of 16. For the flow model, we used an MLP with 3 hidden layers and 64 neurons per layer. Both models were trained for 10000 epochs with a batch size of 256. We used a learning rate of 10^{-3} for flow matching and 10^{-2} for the VAE.

Figure 4.2 compares predictions from the VAE and flow matching, showing that the VAE gives a less sharp distribution and creates a 'bridge' between the solutions, while flow matching gives a much sharper distribution that is more concentrated around the two peaks. The figure also visualizes the learned flow field with and without symmetric coupling, illustrating how symmetric coupling straightens the flow paths.

4.3.2 Coin Flip

To model a coin flip, we take as input the amount of money one bets (positive for heads, negative for tails), and then predict the winnings as output. This means the outcome is either the same as the input (winning the bet amount) or the negative of the input (losing the bet amount). Our dataset consists of 1000 inputs x_i sampled uniformly from the range $[-100, 100]$, with corresponding outputs $y_i \in \{x_i, -x_i\}$ with equal probability. We use 800 of these data points for training and 200 for testing.

As a non-probabilistic baseline, we used an MLP with 2 hidden layers of 32 neurons per layer, trained for 52 epochs (early stopping), with batch size 64 and learning rate 10^{-4}. For the VAE, we used an MLP encoder-decoder with 2 hidden layers of 32 neurons, latent dimension 16, and an additional MLP for conditioning on the bet amount. The VAE was trained for 1000 epochs with learning rate 10^{-3}. For flow matching, we used a similar MLP to predict the flow field with Gaussian prior noise of standard deviation 1.0.

Even though the model is not equivariant, we still test symmetric coupling in the form of picking -1 or $+1$ depending on which is closer to the prior sample.

Figure 4.3 compares deterministic, VAE, and flow matching predictions, showing that the deterministic model fails to capture the bimodality, while the VAE captures it but with a blurry distribution that is not concentrated around the two solutions. In contrast, the flow matching model captures the bimodality with a much sharper distribution.

4.3.3 Three Roads

This test system models bifurcation in a situation where there are two entities (can be nodes in a graph or something else) that need to coordinate. One can imagine that these are two people that are in a store with very narrow aisles. Both of them are standing right in front of an obstacle separating the aisles, and want to avoid this obstacle by either swerving left or right. However, they don't want to bump into each other, so if one of them chooses the middle aisle (e.g., going left for the right person, going right for the left person), the other will not pick the same aisle. See Fig. 4.4 for an illustration. This is important in prediction of, e.g., pedestrian dynamics, where 1)

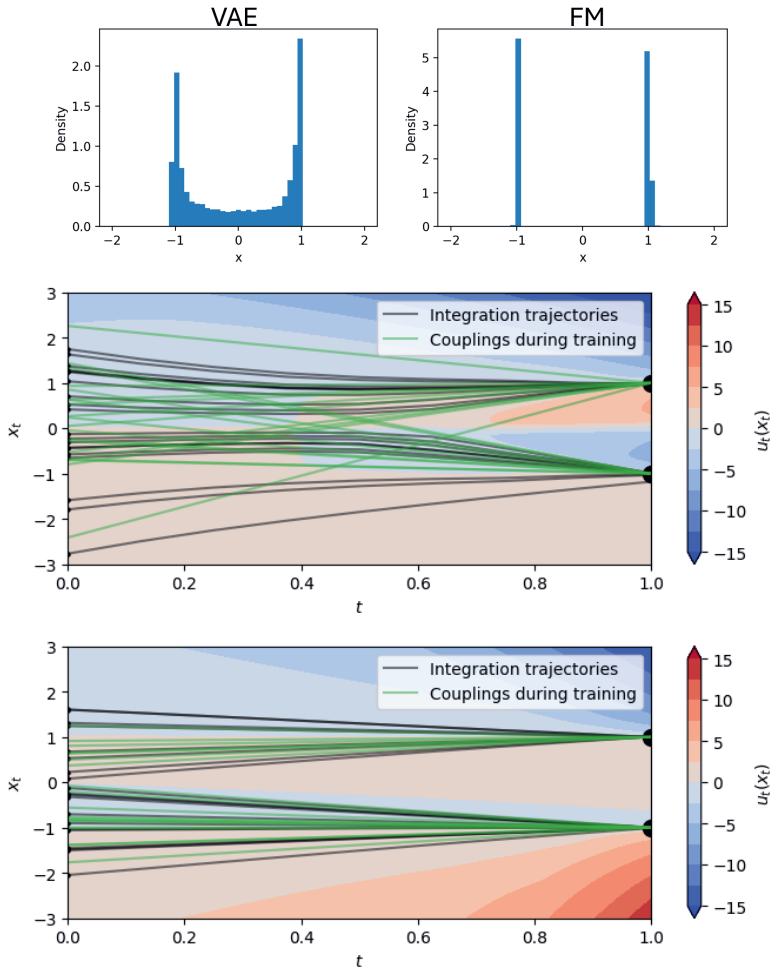


Figure 4.2: Top: the results of predicting a probability distribution consisting of two Dirac deltas, using a VAE and flow matching. Middle/bottom: the learned flow field by the trained flow matching model (contour plot), showing with green lines the coupling during training between prior and targets (middle: without symmetric matching, bottom: with) and how the coupling samples from the Gaussian prior are pushed towards the two delta peaks (black lines).

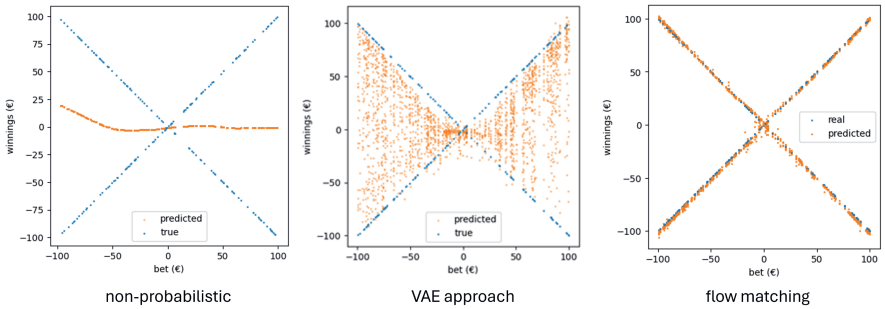


Figure 4.3: The results of predicting outcomes of a coin flip using three different methods: (a) a regular neural network, (b) a conditional VAE, and (c) flow matching. Each method makes 10 predictions per test data point.

people coordinate to avoid bumping into each other, and 2) the distributions of their possible paths are highly multimodal when obstacles are involved; there are multiple ways to go around an obstacle, but none to go through it [7, 22]. We choose to model this as a very simple system with two input features and two output features. The input and output features $[x_i^{(1)}, x_i^{(2)}]$ and $[y_i^{(1)}, y_i^{(2)}]$ are two numbers that indicate the horizontal position of both entities before and after deciding their path, respectively. The possible values of $[y_i^{(1)}, y_i^{(2)}]$ are

$$[y_i^{(1)}, y_i^{(2)}] \in \begin{cases} [x_i^{(1)} - d/2, x_i^{(2)} + d/2] \\ [x_i^{(1)} - d/2, x_i^{(2)} - d/2] \\ [x_i^{(1)} + d/2, x_i^{(2)} + d/2] \end{cases} \tag{4.5}$$

with $d = x_i^{(2)} - x_i^{(1)}$.

For example, for the input $[x_i^{(1)}, x_i^{(2)}] = [1.0, 2.0]$, the output could be $[0.5, 2.5]$, $[0.5, 1.5]$, or $[1.5, 2.5]$. In this test system, we have chosen to consider all these possibilities equally likely. Our dataset consists of 2000 data points, with $[x_i^{(1)}, x_i^{(2)}]$ sampled uniformly from the range $[-50, 50]$. Of the 2000 data points, 1600 are used for training and 400 for testing.

We built all approaches (non-probabilistic, VAE, flow matching) on a set model that applies MLPs to each element and each ordered pair of elements, with permutation-equivariant aggregation. As a prior for the flow matching model, we used Gaussian noise with standard deviation 1.0.

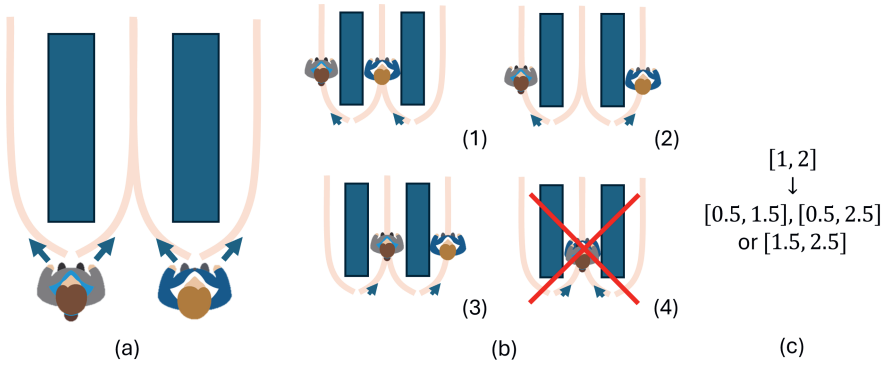


Figure 4.4: Illustration of the 3 roads problem. (a) Input, (b) possible outputs, (c) what that looks like in the actual data.

4.3.4 Four Node Graph

For this problem, the input is a graph consisting of 4 nodes, connected in a square. Each node has the same value x_i, as a node embedding, which is sampled uniformly from the range $[-50, 50]$. The output is an embedding for each node which is equal to $x_i \pm 5$, with connected nodes choosing the opposite sign. See Fig. 4.5 for a visualization. This system is equivariant with respect to permutation. Additionally, the input is self-similar under some permutations, whereas each individual output is self-similar under fewer permutations, which means there is symmetry-breaking. Our dataset contains 2000 data points, of which 70% are used for training, 30% for testing.

As a non-probabilistic model, we used a message-passing graph neural network with 3 message-passing layers. For the VAE, we used such a message-passing GNN as encoder and another as decoder, and another to condition on the input graph. For flow matching, we used a similar message-passing GNN with the input value plus Gaussian noise (standard deviation 1.0) as a prior. Symmetric coupling is implemented by choosing the target permutation with minimum Euclidean distance squared to the noise sample. Only two permutations are relevant here: identity, and the swap of nodes 0 with 2 and 1 with 3.

4.3.5 Buckling Beam

For this problem, we consider a beam that is attached to a base on the bottom, and then prescribe a vertical displacement to the top of the beam, while leaving the horizontal displacement free. For a small displacement, the beam will compress but stay straight, but when the displacement reaches a critical value, the system becomes unstable and the beam will buckle, which will make it bend to either the left or right side, with either option equally likely. We model the beam as a series of n connected segments, which connect $n + 1$ nodes, where node 0 is attached to the base. We consider n ranging from 2 to 10. Each segment has a length L_i and an axial stiffness C_i. Individual segments can rotate relative to each other, reflected by a rotational

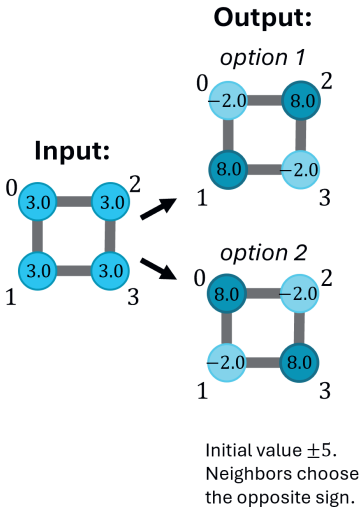


Figure 4.5: Illustration of the four node graph problem.

stiffness K_i. The input d is the vertical displacement applied to the top node. We solve the deformation of the beam by finding the Hencky strains ϵ_i and angles q_i that minimize the total energy:

$$U(\mathbf{q}, \epsilon) = \frac{1}{2}\left[K_1 q_1^2 + \sum_{i=2}^{n} K_i (q_i - q_{i-1})^2 + \sum_{i=1}^{n} L_i C_i \epsilon_i^2\right] \tag{4.6}$$

subject to the constraint that the top of the beam is at the right displacement d:

$$\sum_{i=1}^{n} L_i \left[1 - \exp(\epsilon_i) \cos q_i\right] = d. \tag{4.7}$$

We solve this using Lagrange multipliers, combined with stability analysis. The solution is perturbed using the lowest eigenmode of the stiffness matrix to break symmetry when the system becomes unstable. In total, we generate 1000 beams. We sample n from the integers 2 to 11, and sample L_i, C_i and K_i from the log-uniform distribution between 0.5 and 2. For each beam we solve for its trajectory as d is increased from zero. We choose the range of d such that at the end of a trajectory the top node is at the ground, and we divide this range into 200 time steps. 70% of the resulting trajectories are used for training, 30% for testing.

For this system, we used a flow matching model with a random walk prior similar to [7], with an EGNN [31] to predict interactions between nodes and UNets for time evolution. Symmetric coupling selects the closest reflected target trajectory.

Figure 4.1(b,c) shows representative predictions against both valid ground truths, showing that the model can accurately capture both solution branches.

4.3.6 Allen–Cahn

The Allen–Cahn equation is a reaction-diffusion equation and describes the time evolution of a binary mixture of phases, e.g., ice and water at zero degrees Celsius. It is given by

$$\frac{\partial u}{\partial t} = \epsilon^2 \frac{\partial^2 u}{\partial x^2} - (u^3 - \mu u), \tag{4.8}$$

where u is the phase-field variable, representing the concentration of one of the phases. ϵ is a parameter that controls the interface width between phases, and $u^3 - \mu u$ is the derivative of a double-well potential with minima at $\pm\sqrt{\mu}$ if $\mu > 0$; for $\mu \leq 0$, there is only one minimum at 0. We solve this equation on a 1D domain of length 1.0 with periodic boundary conditions, from $t = 0$ to $t = 100$, using a finite difference scheme with 200 spatial points and a time step of 0.1. Each trajectory starts from the homogeneous state $u = 0$, however, because this is an unstable state when $\mu > 0$, we add a small random perturbation sampled from $\mathcal{N}(0, 0.001)$ to each spatial point. We generate 400 trajectories in total, of which 300 are used for training and 100 for testing. For each trajectory, ϵ is randomly sampled from a log-uniform distribution between 0.001 and 0.1, and μ from $\mathcal{U}(-0.1, 1.0)$.

We used a random walk prior (cumulative scaled Gaussian noise) and a 2D UNet to predict the flow field from the entire random-walk trajectory to the entire final trajectory, with periodicity only in the spatial dimension. The UNet has 4 downsampling steps and 32 channels. Symmetric coupling combines the closest circular shift (via FFT cross-correlation), reflection, and sign flip, which improves performance and enables recovery of the pitchfork bifurcation behavior.

See Figure 4.6 for several predictions by the trained flow model, showing that even though the solutions are not the same as the ground truth, they do reproduce the same qualitative behavior. This is also illustrated by Figure 4.7, which shows that the trained model can reproduce bifurcation diagrams of the system.

4.3.7 Overall quantitative comparison

Table 4.1 reports the mean Wasserstein distance between the predicted and actual distribution of allowed outcomes for each test data point (except for Allen–Cahn, where we report the governing equation residual). For the 2-Delta Peaks dataset and the 4-node graph, we also compare to symmetric coupling. In all cases, the flow matching outperforms the non-probabilistic model and the VAE, and symmetric coupling further improves the result. For the 2-Delta Peaks, even though the model is not equivariant, symmetric coupling still improves performance because the targets and the prior are symmetric to a sign flip.

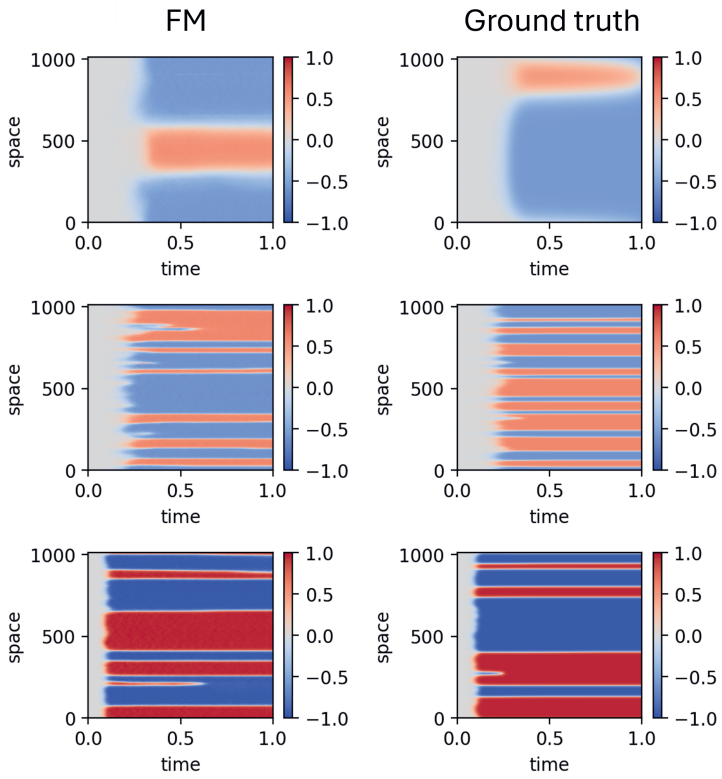


Figure 4.6: Comparison of predicted Allen–Cahn trajectories by the flow model with symmetric coupling with examples of possible ground truth trajectories.

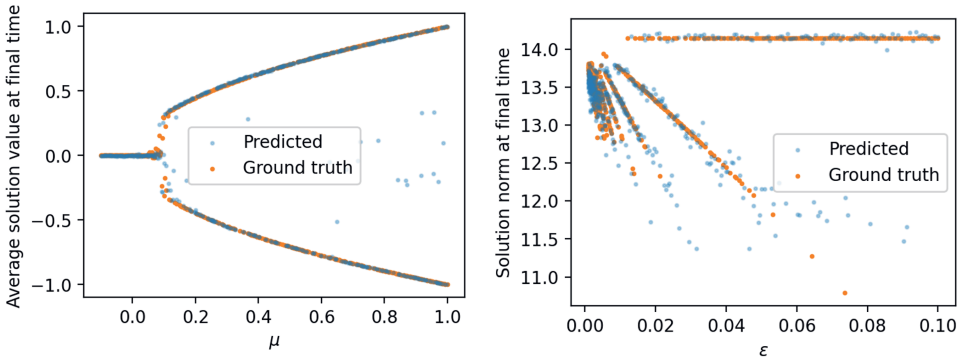


Figure 4.7: Bifurcation diagrams of the Allen–Cahn equation. Left: the parameter μ is varied, keeping ϵ constant at 0.1. The trained model is able to model the pitchfork bifurcation. Right: the parameter ϵ is varied, keeping μ constant at 1.0. The model is able to capture the addition of new stable states as ϵ decreases.

Table 4.1: Performance of various approaches for the different test systems. All metrics except for the Allen–Cahn are the Wasserstein distance of a set of 100 predictions to the actual distribution of allowed outcomes for each test data point, averaged over all test data points. For the Allen–Cahn system, we use the residual of the governing equation as a performance metric, acknowledging its limitations due to sensitivity to small fluctuations in the prediction, particularly in the Laplacian term.

Test System	Non-prob.	VAE	FM	FM*
Two Delta Peaks	1.0**	0.25	0.091	**0.0041**
Heads or Tails	56.2	33.0	**8.30**	-
Three Roads	21.4	17.3	**3.88**	-
Four Node Graph	10.0	9.89	2.02	**1.19**
Buckling Beam	-	-	23.1	**9.6**
Allen–Cahn	-	-	255	**244**

*With symmetric coupling.

**Theoretical result, a non-probabilistic model would always predict the mean 0.

4.4 Conclusion

We introduced a probabilistic, equivariant flow matching framework to address the challenge of modeling multiple coexisting solutions in symmetry-breaking bifurcation problems. By leveraging generative modeling and symmetric coupling, our approach captures multimodal output distributions while respecting system symmetries. Through experiments on both abstract and physically grounded systems, we demonstrated that our method outperforms traditional and variational models in accurately representing bifurcation behavior. This work opens new avenues for integrating symmetry-aware generative models into the analysis of complex dynamical systems.

Bibliography

[1] Albergo, M.S., Vanden-Eijnden, E.: Building normalizing flows with stochastic interpolants. arXiv preprint arXiv:2209.15571 (2022)

[2] Armstrong McKay, D.I., Staal, A., Abrams, J.F., Winkelmann, R., Sakschewski, B., Loriani, S., Fetzer, I., Cornell, S.E., Rockström, J., Lenton, T.M.: Exceeding 1.5 c global warming could trigger multiple climate tipping points. Science **377**(6611), eabn7950 (2022)

[3] Azulay, R., Combescure, C.: Predicting the post-bifurcated patterns of architectured materials using group-theoretic tools. Journal of the Mechanics and Physics of Solids **187**, 105631 (2024)

[4] Bažant, Z.P., Cedolin, L.: Stability of structures: elastic, inelastic, fracture, and damage theories. Courier Corporation (2003)

[5] Beekman, A., Rademaker, L., Van Wezel, J.: An introduction to spontaneous symmetry breaking. SciPost Physics Lecture Notes p. 011 (2019)

[6] Bousquet, O., Gelly, S., Tolstikhin, I., Simon-Gabriel, C.J., Schoelkopf, B.: From optimal transport to generative modeling: the vegan cookbook. arXiv preprint arXiv:1705.07642 (2017)

[7] Brinke, K.B.t., Minartz, K., Menkovski, V.: Flow matching for geometric trajectory simulation. arXiv preprint arXiv:2505.18647 (2025)

[8] Brunton, S.L., Noack, B.R., Koumoutsakos, P.: Machine learning for fluid mechanics. Annual review of fluid mechanics **52**(1), 477–508 (2020)

[9] Della Pia, A., Patsatzis, D.G., Rozza, G., Russo, L., Siettos, C.: Surrogate normal-forms for the numerical bifurcation and stability analysis of navier-stokes flows via machine learning. arXiv preprint arXiv:2506.21275 (2025)

[10] Dosovitskiy, A., Brox, T.: Generating images with perceptual similarity metrics based on deep networks. Advances in neural information processing systems **29** (2016)

[11] Fabiani, G., Vandecasteele, H., Goswami, S., Siettos, C., Kevrekidis, I.G.: Enabling local neural operators to perform equation-free system-level analysis. arXiv preprint arXiv:2505.02308 (2025)

[12] Gu, F., Guiselin, B., Bain, N., Zuriguel, I., Bartolo, D.: Emergence of collective oscillations in massive human crowds. Nature **638**(8049), 112–119 (2025)

[13] Hendriks, F., Menkovski, V., Doškář, M., Geers, M.G., Rokoš, O.: Similarity equivariant graph neural networks for homogenization of metamaterials. Computer Methods in Applied Mechanics and Engineering **439**, 117867 (2025)

[14] Kaba, S.O., Mondal, A.K., Zhang, Y., Bengio, Y., Ravanbakhsh, S.: Equivariance with learned canonicalization functions. In: International Conference on Machine Learning. pp. 15546–15566. PMLR (2023)

[15] Kaba, S.O., Ravanbakhsh, S.: Symmetry breaking and equivariant neural networks. arXiv preprint arXiv:2312.09016 (2023)

[16] Kang, S.H., Shan, S., Noorduin, W.L., Khan, M., Aizenberg, J., Bertoldi, K.: Buckling-induced reversible symmetry breaking and amplification of chirality using supported cellular structures. Adv. Mater **25**(24), 3380–3385 (2013)

[17] Klein, L., Krämer, A., Noé, F.: Equivariant flow matching. Advances in Neural Information Processing Systems **36**, 59886–59910 (2023)
[18] Kondepudi, D.K., Asakura, K.: Chiral autocatalysis, spontaneous symmetry breaking, and stochastic behavior. Accounts of chemical research **34**(12), 946–954 (2001)
[19] Larsen, A.B.L., Sønderby, S.K., Larochelle, H., Winther, O.: Autoencoding beyond pixels using a learned similarity metric. In: International conference on machine learning. pp. 1558–1566. PMLR (2016)
[20] Lawrence, H., Portilheiro, V., Zhang, Y., Kaba, S.O.: Improving equivariant networks with probabilistic symmetry breaking. arXiv preprint arXiv:2503.21985 (2025)
[21] Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 (2022)
[22] Minartz, K., Hendriks, F., Koop, S.M., Corbetta, A., Menkovski, V.: Discovering interaction mechanisms in crowds via deep generative surrogate experiments. Scientific Reports **15**(1), 10385 (2025)
[23] Minartz, K., Poels, Y., Koop, S., Menkovski, V.: Equivariant neural simulators for stochastic spatiotemporal dynamics. Advances in Neural Information Processing Systems **36**, 38930–38957 (2023)
[24] Muramatsu, M., Irie, T., Nagatani, T.: Jamming transition in pedestrian counter flow. Physica A: Statistical Mechanics and its Applications **267**(3-4), 487–498 (1999)
[25] Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., Battaglia, P.: Learning mesh-based simulation with graph networks. In: International conference on learning representations (2020)
[26] Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., Courville, A.: On the spectral bias of neural networks. In: International conference on machine learning. pp. 5301–5310. PMLR (2019)
[27] Rietkerk, M., Bastiaansen, R., Banerjee, S., van de Koppel, J., Baudena, M., Doelman, A.: Evasion of tipping in complex systems through spatial pattern formation. Science **374**(6564), eabj0359 (2021)
[28] Ruelle, D., Takens, F.: On the nature of turbulence. Les rencontres physiciens-mathématiciens de Strasbourg-RCP25 **12**, 1–44 (1971)
[29] Sadria, M., Bury, T.M.: Fatenet: an integration of dynamical systems and deep learning for cell fate prediction. Bioinformatics **40**(9), btae525 (2024)
[30] Salmona, A., De Bortoli, V., Delon, J., Desolneux, A.: Can push-forward generative models fit multimodal distributions? Advances in Neural Information Processing Systems **35**, 10766–10779 (2022)
[31] Satorras, V.G., Hoogeboom, E., Welling, M.: E (n) equivariant graph neural networks. In: International conference on machine learning. pp. 9323–9332. PMLR (2021)
[32] Senior, A.W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Žídek, A., Nelson, A.W., Bridgland, A., et al.: Improved protein structure prediction using potentials from deep learning. Nature **577**(7792), 706–710 (2020)
[33] Song, Y., Gong, J., Xu, M., Cao, Z., Lan, Y., Ermon, S., Zhou, H., Ma, W.Y.: Equivariant flow matching with hybrid probability transport for 3d molecule

generation. Advances in Neural Information Processing Systems **36**, 549–568 (2023)
[34] Steffen, W., Rockström, J., Richardson, K., Lenton, T.M., Folke, C., Liverman, D., Summerhayes, C.P., Barnosky, A.D., Cornell, S.E., Crucifix, M., et al.: Trajectories of the earth system in the anthropocene. Proceedings of the national academy of sciences **115**(33), 8252–8259 (2018)
[35] Strocchi, F.: Symmetry breaking, vol. 643. Springer (2005)
[36] Tong, A., Fatras, K., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Wolf, G., Bengio, Y.: Improving and generalizing flow-based generative models with minibatch optimal transport. arXiv preprint arXiv:2302.00482 (2023)
[37] Warren, W., Wirth, T.: A bifurcation in visually-guided behavior when following a crowd. Journal of Vision **22**(14), 4317–4317 (2022)
[38] Willcock, S., Cooper, G.S., Addy, J., Dearing, J.A.: Earlier collapse of anthropocene ecosystems driven by multiple faster and noisier drivers. Nature Sustainability **6**(11), 1331–1342 (2023)
[39] Xie, Y.: Equivariant symmetry breaking sets. Massachusetts Institute of Technology (2024)
[40] Zhang, X., Wang, L., Helwig, J., Luo, Y., Fu, C., Xie, Y., Liu, M., Lin, Y., Xu, Z., Yan, K., et al.: Artificial intelligence for science in quantum, atomistic, and continuum systems. Foundations and Trends® in Machine Learning **18**(4), 385–912 (2025)
[41] Zou, Z., Wang, Z., Karniadakis, G.E.: Learning and discovering multiple solutions using physics-informed neural networks with random initialization and deep ensemble. arXiv preprint arXiv:2503.06320 (2025)

CHAPTER 5

Probabilistic Modeling of Buckling in Mechanical Metamaterials

Reproduced from Fleur Hendriks, Vlado Menkovski, Martin Doškář, Marc G.D. Geers, and Ondřej Rokoš (2026). Probabilistic Prediction of Buckling Trajectories with Hierarchical Equivariant Flow Matching (in preparation)

Predicting buckling and post-buckling behavior can be intrinsically multimodal: after a symmetry-breaking bifurcation, a single geometry and loading condition may admit several physically valid post-buckling trajectories. This is especially challenging for deterministic machine-learning models, which tend to average over competing modes and may therefore predict deformations that are not physically realized. In this work, we formulate buckling prediction as a probabilistic learning problem. We model the distribution over admissible deformation trajectories using equivariant flow matching, so that individual samples may break symmetry while the learned distribution remains consistent with the symmetries of the underlying mechanical system. The main contribution is to scale this probabilistic formulation from previous low-dimensional bifurcation examples to high-dimensional graph-based representations of mechanical metamaterials. To this end, we combine equivariant graph neural networks, symmetric coupling, trajectory-level temporal modeling, and a hierarchical architecture that enables global coordination across nodes. We evaluate the approach on a stacked-holes benchmark problem and on a large-scale wallpaper-group dataset containing 1,020 geometries under 12 loading paths each. The results show that a single trained model can sample multiple competing buckling trajectories across diverse geometries and loading conditions, demonstrating the potential of equivariant generative models for multimodal structural mechanics.

5.1 Introduction

Buckling is an instability in which a structure under increasing load suddenly changes its deformation pattern. In symmetric systems, this often occurs through a symmetry-breaking bifurcation: several symmetry-related post-buckling states become admissible beyond a critical load. The mapping from geometry and loading to deformation is therefore one-to-many, making buckling prediction fundamentally multimodal. This multimodality is challenging for both simulations and surrogate models based on machine learning. Finite-element simulations can compute individual post-buckling paths, but doing so is computationally expensive and exploring multiple competing solutions typically requires repeated simulations with different perturbations, imper-

fections, or continuation strategies. Deterministic machine-learning models instead tend to average over admissible solutions, producing deformations that may not be physically valid. Equivariance alone does not solve this: for symmetric inputs, a deterministic equivariant model is constrained toward symmetric outputs, even when the physical response breaks symmetry [1]. Our contribution is a probabilistic model for buckling in complex systems. By modeling the multimodal distribution over deformation trajectories, the model can generate multiple admissible post-buckling trajectories for the same geometry and loading condition. Specifically, we apply this approach to buckling in mechanical metamaterials.

Multiplicity of solutions Below a critical load, the structure is highly symmetric. Above it, several symmetry-related deformation states become admissible. Consequently, the mapping from geometry and loading to deformation is one-to-many rather than one-to-one. For a structure with a perfectly symmetric microstructure (e.g., a square array of circular holes), the group of stabilizer symmetries acts on the solution set: every buckling mode obtained by applying a symmetry operation to one valid solution is itself valid. This makes post-buckling prediction fundamentally a multimodal learning problem, where the goal is to model a distribution over admissible deformation trajectories rather than a single deterministic outcome.

Mechanical metamaterials Mechanical metamaterials are engineered materials with architected microstructures that produce special mechanical properties usually not found in nature [2]. Many such materials are designed to exploit elastic instabilities: under sufficient loading, porous cellular structures undergo sudden pattern transformations governed by buckling, leading to interesting properties such as auxeticity [3], and enabling programmable shape change [4, 5], improved energy absorption [6], tunable wave propagation [7–9], controllable encapsulation [10], soft-robotic actuation [11–14] and even logic gates [15, 16]. Buckling instabilities thus drive the behavior of mechanical metamaterials [7, 17–22], and these instabilities can be understood in terms of symmetry-breaking bifurcations [23].

Limitations of existing approaches Finite-element simulations can compute individual post-buckling branches, but recovering the set of admissible solutions generally requires dedicated bifurcation analysis, including path-following, pinpointing of singular points, branch-switching or eigenmode-based perturbations, and subsequent continuation of alternative branches [24–26]. On the other hand, deterministic machine learning models tend to predict an average that corresponds to no physical state. Even worse, in the case of equivariant models, symmetry makes the model produce invariant outputs, which *forces* the model to predict non-physical averages. One can circumvent this by introducing a small geometrical imperfection, which breaks the symmetry and forces a unique solution [1], but this limits applicability; symmetry plays a crucial role in the design and behavior of metamaterials [27–32] and these models with built-in imperfections cannot represent the multiplicity of admissible post-buckling states that is central to these metamaterials.

Our solution To address this challenge, we formulate buckling prediction as the prediction of a probability distribution over admissible deformation trajectories. In our previous proof-of-concept work [33] we showed that flow matching combined with an equivariant model can be used to capture a probability distribution over possible outcomes in symmetry-breaking bifurcations. Flow matching [34, 35] is a generative modeling framework which learns a continuous-time vector field transporting samples from a tractable prior to the data distribution. We showed strong performance of flow matching for multi-modal distributions with sharp, well-separated modes. In this previous work, we also introduced symmetric coupling, a strategy for training flow matching models on symmetric data that finds the nearest symmetry-equivalent target before computing the loss, which straightens the flow and improves convergence.

Challenges However, this framework of equivariant flow matching for bifurcation has so far only been tested on small toy problems, whereas buckling in mechanical metamaterials is much more complex, with a large number of degrees of freedom. Therefore, the main objective of this paper is to combine equivariant GNNs with flow matching and symmetric coupling, in order to learn the full distribution over buckling trajectories, and address the challenges that arise when scaling this approach up to large, complex systems. The most important challenges are:

- A key challenge in learning buckling behavior is not only the presence of multiple valid deformation modes, but the requirement that the entire structure must consistently realize one of these modes. In graph-based models, predictions are constructed from local interactions; as a result, different regions of the structure may implicitly favor different modes, leading to spatially inconsistent "patchwork" solutions. We identify this as a global coordination problem: predicting buckling requires coherent selection of a mode across all degrees of freedom. We therefore adapt hierarchical GNN concepts to the specific requirements of equivariant, periodic structural mechanics, where global coordination across nodes is essential. This involves coarsening the graph and performing global interactions at the coarsened level.
- Because the simulations are performed on RVEs with periodic boundary conditions, we need periodic boundaries, with a *changing* shape of the unit cell. Essentially, the allowed space in which our predictions live changes over the course of our trajectory. The pooling and unpooling operations that make the network hierarchical, as well as the attention mechanism must respect this periodicity too.
- Auto-regressive models suffer from error accumulation. Moreover, only very few time steps show a bifurcation where the solution splits into multiple branches. For these reasons, we adapt trajectory-level prediction via temporal U-Nets to our setting, following recent work in generative spatiotemporal modeling [36,37].
- Even though individual samples from the distribution of buckling trajectories break symmetry and therefore equivariance, the distribution itself still possesses various equivariances. These need to be respected by the network, including its hierarchical and time series components.

Contributions The main contributions of this article are:

1. An equivariant generative formulation of buckling prediction that learns distributions over admissible deformation trajectories rather than single solutions. Our formulation separates symmetry at the level of distributions from symmetry breaking at the level of individual samples, allowing the model to remain equivariant while generating distinct physical modes.
2. A temporal extension that generates complete deformation trajectories, rather than generating only the end state.
3. A hierarchical equivariant GNN that captures long-range interactions efficiently.
4. A framework that operates directly on arbitrary porous microstructures and loading trajectories, avoiding restrictions to specific unit-cell topologies or loading protocols.

This demonstrates that equivariant generative modeling, previously restricted to low-dimensional bifurcation problems [33], can be scaled to high-dimensional graph-structured systems with thousands of degrees of freedom.

Validation We validate our approach on the deformation of a circular stacked-hole metamaterial [1] under various loadings, and on a large-scale *wallpaper group dataset* (Sec. 5.2.2) comprising 1,020 geometries spanning all 17 two-dimensional crystallographic symmetry groups under 12 loading trajectories each [26]. To the best of our knowledge, this is the first study to evaluate such a model on these datasets.

5.1.1 Notation

We distinguish between two notions of 'time': the loading time that parametrizes the deformation, which we denote as t, and the flow matching time that parametrizes the interpolation between prior and target distributions, which we denote as τ. Both are pseudotimes, i.e., they are not physical, and only serve for parameterization. Rescaling either does not lead to any changes.

Scalars are denoted by lowercase letters, abstract, neural network-generated 'vectors' are denoted by bold lowercase letters (e.g., $\mathbf{h}$), physical vectors are denoted by lowercase letters with an arrow (e.g., $\vec{x}$), and higher-order physical tensors are denoted by bold uppercase letters (e.g., $\mathbf{F}$), and so are multidimensional arrays (e.g., $\mathbf{X}$). Learnable parameters of the model are collectively denoted by θ, and functions parametrized by them are written as $\mathcal{F}(x;\theta)$.

The divergence of a second-order tensor field is denoted as $\vec{\nabla}\cdot\mathbf{A}$, which is defined as

$$\vec{a} = \vec{\nabla}\cdot\mathbf{A} \leftrightarrow a_j = \sum_i \frac{\partial A_{ij}}{\partial x_i}, \tag{5.1}$$

where x_i are the spatial coordinates. The dot product between a second-order tensor and a vector is defined as

$$\vec{c} = \mathbf{A}\cdot\vec{b} \leftrightarrow c_i = A_{ij}b_j, \tag{5.2}$$

5.1.2 Paper Organization

The remainder of the paper is organized as follows: section 5.2 contains the problem formulation; it formalizes the buckling prediction task and introduces the datasets. Section 5.3 outlines the key components of our approach: it discusses the main challenges of the problem, how these challenges have or have not been addressed in existing literature and how we choose to tackle them, emphasizing the novelties with respect to the state of the art. Section 5.4 explains the exact implementation of all building blocks and how they are integrated in the three model architectures proposed in this work, along with discussion on their training and evaluation. Section 5.5 presents the results, and Section 5.6 lists the conclusions.

5.2 Problem Formulation

When predicting the effective mechanical behavior of a material, one can simulate a small sample of the material that is assumed to be representative of the whole, called a representative volume element (RVE). Often, the RVE is simulated under the assumption of periodic boundary conditions. Using homogenization, the microscopic response of the RVE can be averaged to obtain effective macroscopic quantities, linking the local (microscopic) deformation and stresses to the global (macroscopic) behavior of the material. In this way, the RVE serves as a bridge between the heterogeneous microstructure and an equivalent homogeneous continuum [38, 39]. Hence, scale separation is assumed, and global buckling is neglected.

In our quasi-static setting, a porous metamaterial is subjected to a given loading trajectory described by the macroscopic deformation gradient $\mathbf{F}_t$, a second order tensor, represented by a 2×2 matrix with respect to a Cartesian basis in our 2D setting. It describes the overall (macroscopic/global) deformation of the RVE at each loading time step t. This pseudotime t runs from 0 to 1 and its only function is to linearly interpolate between the reference state $\mathbf{F}_0 = \mathbf{I}$ and the prescribed final state $\mathbf{F}_1$.

The problem we attempt to solve is therefore: given the RVE geometry and a *macroscopic* deformation in the form of $\mathbf{F}_t$, what will the *microscopic* deformation of that RVE look like?

Using first-principles models, and assuming a hyperelastic material, this requires solving the microstructural equilibrium equation

$$\vec{\nabla}^{\text{ref}} \cdot \mathbf{P}_{\text{micro}}^T(\mathbf{F}_{t,\text{micro}}) = \vec{0}, \quad \text{in } \Omega, \quad t \in [0, 1], \tag{5.3}$$

where $\mathbf{P}_{\text{micro}}$ is the first Piola-Kirchhoff stress tensor derived from a strain energy density $\mathfrak{W}(\mathbf{F}_t^{\text{micro}})$, Ω is the RVE domain, and $\mathbf{F}_{t,\text{micro}}$ is the *microscopic* deformation gradient (which depends on the deformed configuration), and the superscript ref on $\vec{\nabla}^{\text{ref}}$ indicates we take the divergence with respect to the reference configuration. See [26] for more details.

Using a finite element discretization, the solution is represented by the deformed positions of the nodes $\vec{x}_{i,t}$, where i is the node index and t is the loading time. Under the assumption of first-order homogenization, the node positions can be decomposed

as:

$$\vec{x}_{i,t} = \vec{x}^{\text{aff}}_{i,t} + \vec{w}_{i,t}, \quad \vec{x}^{\text{aff}}_{i,t} = \mathbf{F}_t \cdot \vec{x}^{\text{ref}}_i, \tag{5.4}$$

where $\vec{w}_{i,t}$ is the *microfluctuation*, capturing deviations from the affine (macroscopic) deformation $\vec{x}^{\text{aff}}_{i,t}$, caused by the heterogeneous nature of the RVE (in our case the shape, size and distribution of holes). The microfluctuations $\vec{w}_{i,t}$ are obtained by minimizing the total elastic energy of the RVE using the finite element method [26, 40, 41].

For both datasets used here, the RVE consists of 2×2 unit cells, which is assumed to be sufficiently large to capture the dominant local buckling behavior of the periodic structures. This is a common modeling choice [42, 43], but known to be insufficient in certain cases [44]; ideally the correct RVE size should be verified by Bloch analysis [45–48].

As a result, the RVE initially possesses four symmetries corresponding to periodic shifts by $\vec{0}, \vec{\ell}_0, \vec{\ell}_1, \vec{\ell}_0 + \vec{\ell}_1$, where $\vec{\ell}_0$ and $\vec{\ell}_1$ are the lattice vectors spanning the unit cell, i.e., half the RVE lattice vectors. See Figure 5.1a for an illustration of the periodic shift by $\vec{\ell}_1$ for a particular choice of square stacking of holes. The RVE may possess additional point-group symmetries such as reflection and rotation, but most of these are broken once a generic macroscopic deformation gradient $\mathbf{F}_t$ is prescribed[1]. We therefore restrict the analysis to the periodic-shift subgroup.

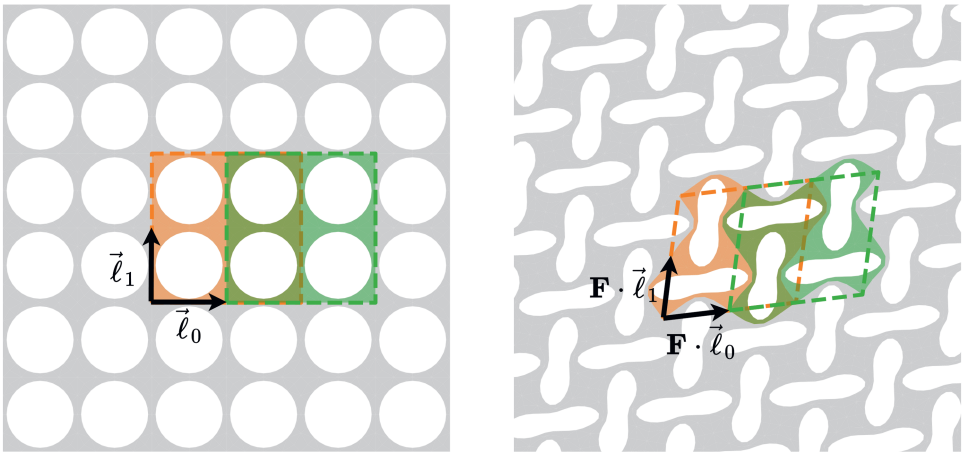
Under prescribed deformation $\mathbf{F}$, there is a certain critical load $\mathbf{F}_c$ at which the system becomes unstable and buckles, which breaks some of the symmetries. However, since applying any of the periodic shifts from $\mathcal{G}_{\text{input}}$ to the input leads to the exact same input, applying the same periodic shift to the output yields another valid buckled configuration. Therefore, when buckling happens, multiple solutions for $[\vec{x}_{i,t}]_{i \in \{0,\dots,N-1\}}$ can satisfy the equilibrium equations for the same $\mathbf{F}_t$. This means the solution set is degenerate, and all solutions are equally probable; see Figure 5.1b.

The deformation trajectory consists of the deformed nodal positions $\vec{x}_{i,t}$ for all values of the loading time t and all nodes $i \in \{0, \dots, N-1\}$, with N the number of nodes. Since t is a pseudotime, which only serves to parametrize the loading and is discretized into T time steps, we use it as an index for the time series from now on. The solution is therefore the set of time series

$$\mathbf{X} = [\vec{x}_{i,t}]^{t \in \{0,\dots,T-1\}}_{i \in \{0,\dots,N-1\}} \in \mathbb{R}^{N \times 2 \times T}. \tag{5.5}$$

The question is now how to obtain $\mathbf{X}$ for a given geometry and loading trajectory $[F_t]^{t \in \{0,\dots,T-1\}}$. Traditionally, this is done through numerical methods, most commonly the finite element method (FEM). For buckling, FEM can become computationally expensive: it requires repeated, carefully designed perturbations with costly eigenmode analyses and subsequent continuation techniques to explore trajectories of alternative post-buckling solutions. Therefore, we aim to learn a surrogate model.

[1] A two-fold rotation (i.e., turning the RVE upside-down) may remain a symmetry for arbitrary $\mathbf{F}_t$, but it is not used in the symmetric coupling or evaluation considered here.



(a) Reference (undeformed) (b) Deformed

Figure 5.1: Illustration of the symmetry-breaking; if we shift the 'window' of the RVE by one unit cell lattice vector $\vec{\ell}_0$ (from orange to green), then the input (a) does not change, but the solution (b) does.

This model should approximate the deformed nodal positions $\mathbf{X}$ as a function of the input geometry and loading:

$$\mathbf{X} = f\left(\left[\mathbf{F}_t\right]^{t\in\{0,\dots,T-1\}}, G\right), \tag{5.6}$$

where G is a periodic graph that represents the domain Ω, and includes the reference positions of the nodes $\left[\vec{x}_i^{\text{ref}}\right]_{i\in\{0,\dots,N-1\}}$ as well as the edge connectivity.

In this work, we focus on two datasets of 2D metamaterials made of one hyperelastic base material with specifically designed holes. These datasets are described below.

5.2.1 Stacked-Holes Dataset

In order to gradually build up complexity, we first train on a dataset that only includes a single microstructure which consists of a square array of circular holes in a matrix of a hyperelastic material. This microstructure is subjected to various loadings, and all time steps of the trajectories are treated as separate data points; one can view the loading $\mathbf{F}_t$ as a trajectory consisting of only one time step. For this, we use a variant of the stacked-holes dataset introduced in our previous work; for details, we refer the reader to [1].

The only difference is that, compared to the dataset used in [1], the present dataset does not rely on an initial geometric imperfection to remove the bifurcation ambiguity. Previously, the circular holes were replaced by slightly elliptical and tilted holes, which explicitly broke the symmetry of the square microstructure and biased the

simulation towards one selected post-buckling branch. Here, the holes remain circular and are placed in an exact square arrangement. As a result, the symmetry-breaking bifurcations are retained, and several symmetry-related post-buckling configurations are physically admissible.

The dataset contains all possible buckling configurations, which are obtained by applying the four periodic shifts of $\mathcal{G}_{\text{input}}$ to the buckled configuration resulting from the finite element simulation.

Dataset statistics: Total number of configurations 8451; train/validation split: 70%/30% by trajectory, that is, all load cases of a trajectory are assigned to the same split to prevent data leakage.

5.2.2 Wallpaper Group Dataset

The second dataset represents a more realistic application; we use a previously published dataset of wallpaper-group based metamaterials [26, 49]. It contains a wide variety of 1020 geometries (60 geometries per wallpaper group), with 12 loading trajectories each, giving 12,232 trajectories in total. Therefore, the model must be able to handle generalization to unseen geometries, as well as predict time series of deformations. Similarly to the previous dataset, we apply periodic shifts to the buckled configurations to obtain the full set of possible solutions.

For this dataset, we *do* train on full trajectories, so the model predicts all loading time steps jointly. Therefore, this dataset serves to test both the ability of the model to predict time series of deformations and its ability to handle generalization to unseen geometries.

Interpolation and masking: For easier handling of the data, we use an interpolated version of the dataset, where all trajectories are uniformly sampled to 32 loading time steps between $t = 0$ and $t = 1$ (when using t as indices: from $t = 0$ to $t = 31$). The original dataset does not include time steps after contact, but if we were to interpolate only the valid time steps (such that the last step is at the contact point), we would be leaking information to the network, because the point at which contact happens is not a priori known. Therefore, we interpolate the entire range from $t = 0$ to $t = 1$, resulting in NaNs for the time steps after contact, and use a mask to indicate which time steps are valid for training.

Split: Train/val/test at 70%/15%/15% by geometry, meaning all 12 trajectories of a geometry are assigned to the same split, preventing geometry leakage.

5.3 Architectural Building Blocks

Because the models combine several components, here we first explain the essentials of each of the building blocks. The description each block contains the rationale behind it (i.e., the problem it tries to solve or the aspects it captures), a summary of the existing methods and related work, and novel developments for the purpose of this work. We do this without going into the exact implementation details, which are described in section 5.4.

In summary:

- we use a graph neural network to handle irregular (non-grid) geometric data;
- we make it equivariant to exploit the symmetries of the problem;
- we ensure that the network respects the periodicity of the problem;
- we use a probabilistic model in the form of flow matching to capture the distribution over possible solutions (combined with symmetric coupling for better training);
- we use time bundling to predict an entire trajectory at once;
- we create a suitable prior distribution for the flow matching that incorporates characteristics of the problem;
- we use a hierarchical architecture to efficiently capture long-range interactions;
- we create an efficient heuristic graph representation;
- we ensure that the model respects physical constraints on the microfluctuation fields;
- and we exploit angles as node features.

5.3.1 Graph Neural Networks

Graph neural networks (GNNs) are a class of neural networks designed to operate on graph-structured data, where the data is represented as a set of nodes and edges. They are often used for geometric data without a regular structure (i.e., not an equispaced grid) such as molecules and, importantly for our application, meshes. Their key property is permutation in-/equivariance; i.e., reordering the input nodes either leaves the output unchanged (invariance) or permutes the output correspondingly (equivariance).

GNNs for structural mechanics Related work on GNN-based mechanical simulators includes Pfaff et al. [50] and Sanchez-Gonzalez et al. [51], who demonstrated mesh-based simulation with graph networks for fluid and solid mechanics. Wang et al. [52] applied deep generative models to metamaterial design, while Kollmann et al. [53] used convolutional networks; neither approach accounted for symmetry or bifurcations. Storm et al. [54] used graph neural networks to predict local stress and strain fields, Jakobsen et al. [55] applied them to predict plastic strain localization, and Patrignani et al. [56] combined a graph neural network with a global attention layer to predict strain fields.

GNN basics Here we briefly introduce GNNs and the notation used in this chapter. The input graph has N nodes, each with a feature vector $\mathbf{h}_i$, and E edges, each with a feature vector $\mathbf{e}_{ij}$. The idea behind a GNN is to iteratively update the representation $\mathbf{h}_i$ of each node i, using information aggregated from its neighboring nodes and edges, thereby capturing the structure of the graph. This happens through message-passing, where each node i sends a message $\mathbf{m}_{ij}$ to each neighbor j:

$$\mathbf{m}_{ij}^l = \mathcal{F}_m\big(\mathbf{h}_i^l, \mathbf{h}_j^l, \mathbf{e}_{ij}; \theta\big), \tag{5.7}$$

$$\mathbf{h}_i^{l+1} = \mathcal{F}_h\left(\mathbf{h}_i^l, \square_{j\in\mathcal{N}(i)}\mathbf{m}_{ij}^l; \theta\right), \tag{5.8}$$

where $\mathbf{m}_{ij}^l$ is the message from node i to node j in message-passing step l, $\mathcal{F}_h$ and $\mathcal{F}_m$ are learnable functions with learnable parameters θ, $\mathcal{N}(i)$ is the neighborhood of node i, $\square$ denotes a permutation invariant aggregation (usually: mean), and $\mathbf{e}_{ij}$ are edge features. See [57] for more detail.

Optionally, the edge features can be updated as well:

$$\mathbf{e}_{ij}^{l+1} = \mathcal{F}_e\left(\mathbf{h}_i^l, \mathbf{h}_j^l, \mathbf{e}_{ij}^l; \theta\right), \tag{5.9}$$

where $\mathcal{F}_e$ is another (possibly learnable) function. However, since both the message-computation and the edge update are edge-based, we combine them such that the messages become the new edge features to save computation:

$$\mathbf{e}_{ij}^{l+1} = \mathbf{m}_{ij}^l. \tag{5.10}$$

By repeating this process for L layers, the node representations can capture information from L hops away in the graph.

5.3.2 Equivariance

The mechanical response of an RVE should not depend on arbitrary choices of coordinate system. When an RVE (or, equivalently, the coordinate system used to describe it) is arbitrarily rotated or reflected, the resulting RVE does not have fundamentally different properties than the original one. In fact, if the entire boundary-value problem (reference RVE geometry, lattice vectors, and the prescribed loading $\mathbf{F}$) is rotated or reflected, then the resulting deformation should only be rotated or reflected in the same way. If one can build a model that respects this, then training on one RVE will automatically generalize to, e.g., rotated or reflected RVEs. This is equivalent to objectivity or frame indifference in continuum mechanics.

More precisely, a function $f : \mathcal{X} \to \mathcal{Y}$ (not to be confused with the function f in Equation (5.6)) is *equivariant* under a group $\mathcal{G}$ if

$$f(g \cdot x) = g \cdot f(x) \quad \text{for all } g \in \mathcal{G}. \tag{5.11}$$

In other words, applying a symmetry transformation to the input results in the same transformation applied to the output[2].

The equivariances that apply to the problem of predicting buckling in metamaterials include translation, rotation, reflection, scaling, periodic shifts and tiling the RVE (the latter two are equivalent to enforcing periodicity). See Hendriks et al. [1, Fig. 7] for an illustrated overview. Illustration specific to the periodic shift equivariance, specifically, is also provided in Figure 5.2.

In the context of machine learning, enforcing equivariance means that a model trained on one RVE will automatically generalize to e.g., that same RVE rotated or reflected, without needing to see those transformations during training. Due to this, equivariance is a powerful inductive bias that can improve generalization and data efficiency by encoding known symmetries of the problem.

[2]Note that the dot $\cdot$ here indicates a very general 'apply this group transformation in a form suitable to the data type', not a specific operation such as a dot product.

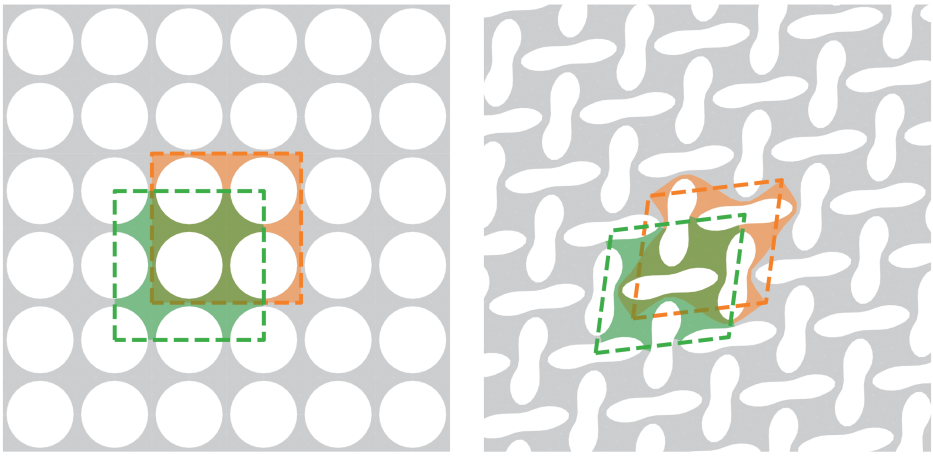


Figure 5.2: Illustration of the periodic shift equivariance that applies to metamaterials; if we shift the 'window' of the RVE (from orange to green, left figure), the solution (right figure) does not change except for the corresponding window shift that needs to be applied.

Equivariant GNNs Various $SO(3)$, $SE(3)$, $E(3)$ or $E(n)$-equivariant GNNs exist for molecular and physical systems [58–61]. In previous work [1] we also introduced similarity-equivariant GNNs for metamaterial homogenization, showing how geometric symmetries and periodicity can be incorporated into graph-based surrogate models for mechanical metamaterials.

For simplicity, here we choose to focus on the $E(n)$-equivariant GNNs (EGNNs) as described by Satorras et al. [62] and adapted by us previously to metamaterials [1]. EGNNs enforce three of these equivariances: rotations, reflections, and translations. These together are the Euclidean symmetries in an arbitrary number of dimensions, hence the $E(n)$ in the name. EGNNs i) modify the message computation in Equation (5.7) to take into account distances and ii) update positions in addition to node features $\mathbf{h}_i$:

$$\mathbf{m}_{ij} = \mathcal{F}_m\left(\mathbf{h}_i, \mathbf{h}_j, \|\vec{x}_j^l - \vec{x}_i^l\|^2, \mathbf{e}_{ij}; \theta\right), \tag{5.12}$$

$$c_{ij} = \mathcal{F}_x(\mathbf{m}_{ij}; \theta), \tag{5.13}$$

$$\vec{x}_i^{l+1} = \vec{x}_i^l + \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} c_{ij}\left(\vec{x}_j^l - \vec{x}_i^l\right), \tag{5.14}$$

where $\mathcal{F}_x$ is a learnable function, $\mathbf{m}_{ij}$ is the message, and the scalar coefficient c_{ij} is invariant. In this framework, no information tied to the absolute coordinates $\vec{x}_i^l, \vec{x}_j^l$ enters the learned functions, only relative geometric information such as the distance $\|\vec{x}_j^l - \vec{x}_i^l\|$. Because the update direction is along $\vec{x}_j^l - \vec{x}_i^l$ (which co-rotates with the geometry), the position update is automatically equivariant with respect to the Euclidean group (rotations, reflections, and translations).

Because the node shift coefficient c_{ij} is edge-wise, just as the messages, we can combine the message computation and the position update into one step to save computation, by simply picking one channel of the message as the shift coefficient:

$$c_{ij} = \mathcal{F}_x(\mathbf{m}_{ij}; \theta) = \mathbf{m}_{ij}[0]. \tag{5.15}$$

The node update from Equation (5.8), which uses the aggregated messages to update the node embedding, stays the same.

In previous work [1], we adapted the EGNN for simulation of metamaterials, by using the affine deformation gradient $\mathbf{F}$ to compute the initial input, as well as the deformed space that the nodes live on, thus enforcing periodicity. Additionally, we introduced similarity-equivariance to capture the effects of uniform scaling, and introduced higher-order tensor outputs. We further made sure the message computation incorporates not just the current distance (taking into account periodicity), but also the reference distance as well as the strain computed from them.

In this work, we mostly follow the same approach for the EGNN, but we retain only the components that are needed for deformation prediction on RVEs represented as periodic graphs. The model remains based on periodic equivariant message passing, but we do not use the read-out branches for the higher-order tensor outputs (stress, stiffness) since we focus on predicting the multimodal deformation only. Similarity-equivariance is also not used in the final model: it was tested in an initial comparison, but did not improve performance in the present setting. We wish to explore the reasons for this in future work.

This EGNN is then used as the backbone for the probabilistic model developed in this work. The following subsections describe how this backbone is embedded in a flow matching framework, extended to predict a time series representing complete loading trajectories, and combined with a hierarchical architecture to improve global coordination of buckling modes.

5.3.3 Periodicity

Periodicity can be viewed as a special case of equivariance, consisting of periodic shifts and tiling the lattice. In principle, our graph representation operates under a PacMan geometry; that is, the graph is defined on a torus, and edges can wrap around the periodic boundaries. However, this requires careful handling of edge vectors and distances, because the naive difference $\vec{x}_j - \vec{x}_i$ does not capture the correct periodic edge vector $\vec{r}_{ij}$ when edges cross the boundaries.

Periodic edge vectors under deformation are computed as

$$\vec{r}_{ij,t} = \vec{x}_{j,t} - \vec{x}_{i,t} + \mathbf{F}_t \cdot \vec{r}_{ij}^{\,\mathrm{per}}, \tag{5.16}$$

where $\vec{r}_{ij}^{\,\mathrm{per}}$ is a periodic correction offset for the edge $i \to j$, determined in the reference configuration. The unit-cell lattice matrix is

$$\mathbf{L} = \begin{bmatrix} \vec{\ell}_0 & \vec{\ell}_1 \end{bmatrix} \in \mathbb{R}^{2\times 2},$$

which stores the unit-cell lattice vectors as columns. Since the RVE consists of 2×2 unit cells, its periodicity vectors are $2\vec{\ell}_0$ and $2\vec{\ell}_1$. The periodic correction is therefore

a matrix-vector multiplication

$$\vec{r}_{ij}^{\,\text{per}} = 2\mathbf{L}T_{ij}, \tag{5.17}$$

where $T_{ij} \in \mathbb{Z}^2$ is an integer lattice-shift vector for the edge $i \to j$, which is selected in the reference configuration using the minimum-image convention. Using the minimum image convention for a lattice with lattice vectors that are not orthonormal requires checking the target node and all neighboring periodic images of it, which means nine total displacements to consider. Since this is expensive to compute on the fly, we precompute the T_{ij} lattice shift indices for all edges in the mesh and store them as edge attributes.

5.3.4 Flow Matching for Symmetry Breaking

As explained in Section 5.2, buckling leads to multiple possible solutions. A deterministic neural network will end up averaging these results, leading to non-physical predictions [33]. To avoid this, we build a probabilistic model.

Related work Machine learning models for bifurcation phenomena have been explored in various contexts, but are i) usually limited to either small systems [33, 63], ii) do not actually learn a distribution over solutions [64], iii) need classical methods such as arc-length continuation for the discovery of bifurcation paths [65, 66], or iv) need to train one network for each solution of each data point [67]. Lawrence et al. [68] used a sample from the relevant symmetry group as an additional input; however, in our case, it is non-obvious how to provide a periodic shift to the network as an additional input, and how to canonicalize the targets to allow this network to train, since all the microstructures buckle differently. Guilhoto et al. [63] explicitly assumed the solution is a multimodal probability distribution with a limited, discrete number of modes. This is also the case in our setting, so their approach might be another option to consider, however, it does not take advantage of the symmetries of the problem, and it is not obvious how to extend this approach to a high-dimensional problem where thousand of nodes need to coordinate over multiple time steps to form a coherent mode.

Solution set As explained in Section 5.2, the input geometry Ω possesses symmetries in the form of periodic translations. In particular, there exists a finite set of shifts $\mathcal{G}_{\text{input}}$ that leave the input unchanged. For our RVE, these correspond to shifts by one unit cell in the lattice directions.[3] A key property of the buckling problem is that these symmetries act *trivially* on the input but *non-trivially* on the output. That is:

- applying a symmetry transformation $g \in \mathcal{G}_{\text{input}}$ to the input leaves it unchanged,

[3] Additionally, a subset of the geometries have a rotation symmetry of an order that is a multiple of 2, which is the only rotation symmetry that is preserved under loading of arbitrary $\mathbf{F}$. This means these geometries have an additional possible symmetry that could break. We did not include this transformation in our set of ground truth solutions, because it was a rare occurrence in our dataset. However, it is possible that the network predicts a correct solution that will get 'counted' wrong because of this.

- but applying the same transformation to a valid solution produces a *different*, yet equally valid, solution, recall Figure 5.1.

In other words, if $\mathbf{X}$ is a valid deformation trajectory, then $g \cdot \mathbf{X}$ is also a valid solution for every $g \in \mathcal{G}_{\text{input}}$. I.e., for a given input $(\Omega, \{\mathbf{F}_t\})$, there is not a single correct output, but a *set* of symmetry-related solutions.

Why standard equivariant models fail Equivariant models are typically designed to satisfy the equivariance condition in Equation (5.11), which ensures that symmetry transformations of the input are reflected in the output. This is also the case for the periodic shifts, since they are a combination of translation equivariance (Subsection 5.3.2) and periodicity (Section 5.3.3). See Figure 5.2 for an illustration of this periodic shift equivariance.

However, in our setting the input is invariant under $\mathcal{G}_{\text{input}}$, i.e.,

$$g \cdot x = x \quad \forall g \in \mathcal{G}_{\text{input}}. \tag{5.18}$$

Substituting this into Equation (5.11) yields

$$f(x) = g \cdot f(x) \quad \forall g \in \mathcal{G}_{\text{input}}, \tag{5.19}$$

which implies that the output must itself be invariant under all symmetry transformations. In practice, this forces the model to predict an average of all valid solutions, which does not correspond to a physically meaningful deformation, as also discussed in [33].

Modeling a distribution over solutions. To address this issue, we follow our previous work [33] and model the output not as a deterministic function, but as a *probability distribution* over all valid solutions:

$$p(\mathbf{X} \mid \Omega, \{\mathbf{F}_t\}). \tag{5.20}$$

This allows the model to represent multiple distinct deformation trajectories for the same input.

The appropriate symmetry condition in this setting is that the *distribution* is equivariant:

$$p(g \cdot \mathbf{X} \mid g \cdot \Omega, g \cdot \{\mathbf{F}_t\}) = p(\mathbf{X} \mid \Omega, \{\mathbf{F}_t\}), \quad \forall g \in \mathcal{G}, \tag{5.21}$$

meaning that symmetry-related outputs must have identical probability.

Our approach We model this distribution using an equivariant generative model based on flow matching. The model learns to map samples from a simple prior distribution to physically valid deformation trajectories.

Two key properties ensure that the learned distribution respects the symmetries of the problem:

- the generative model is equivariant,
- the conditional prior distribution is equivariant.

This combination guarantees that the resulting distribution over solutions is equivariant, while still allowing individual samples to break the symmetry and capture distinct buckling modes.

In contrast to deterministic approaches, this formulation enables the model to generate multiple physically consistent deformation trajectories for the same input, thereby capturing the intrinsic multiplicity of symmetry-breaking bifurcations.

Flow matching Flow matching [34,35] is a framework for generative modeling that learns how to transform samples from a tractable prior distribution into samples from a target distribution. It is closely related to diffusion models, but instead of simulating a stochastic process, it learns a deterministic flow described by an ordinary differential equation (ODE).

More concretely, the model learns a vectors field $\mathbf{U}(\tau, \mathbf{X}; \theta)$ that transports samples from a prior $p_0(\mathbf{X})$ to a target distribution $p_1(\mathbf{X})$ as the flow matching time τ evolves from 0 to 1. Given this vector field, samples are generated by solving the ODE

$$\frac{d\mathbf{X}}{d\tau} = \mathbf{U}(\tau, \mathbf{X}; \theta), \tag{5.22}$$

starting from $\mathbf{X}^0 \sim p_0$, using a numerical integrator such as the midpoint method.

The key idea in flow matching is to train the vector field by defining simple interpolation paths between samples from the prior and the data. Given a prior sample $\mathbf{X}^0 \sim p_0$ and a target sample $\mathbf{X}^1 \sim p_1$, we define a linear interpolation:

$$\mathbf{X}^\tau = (1-\tau)\,\mathbf{X}^0 + \tau\,\mathbf{X}^1, \quad \tau \in [0,1], \tag{5.23}$$

with the superscript indicating the flow matching time τ, and a corresponding target velocity

$$\mathbf{U}^* = \mathbf{X}^1 - \mathbf{X}^0. \tag{5.24}$$

The model is trained to match this target velocity along the interpolation path, resulting in the loss function

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{\tau,\,\mathbf{X}^0,\,\mathbf{X}^1}\left[\|\mathbf{U}(\tau, \mathbf{X}^\tau; \theta) - (\mathbf{X}^1 - \mathbf{X}^0)\|^2\right]. \tag{5.25}$$

In our setting, the target distribution $p_1(\mathbf{X})$ corresponds to the distribution of physically valid deformation trajectories. Following [33], we additionally require that the learned transport respects the symmetries of the problem. This is achieved by using an equivariant model and an equivariant conditional prior, which ensures that the resulting distribution is equivariant [69].

Flow Matching with an EGNN All of our models are EGNN-based, which iteratively update the positions of the nodes. For this reason, they are better suited for predicting the deformed positions rather than a vector field such as the flow $\mathbf{U}(\tau, \mathbf{X})$.

We therefore use an equivalent formulation in which the network directly predicts the endpoint of the transport, i.e., the trajectory $\mathbf{X}^1$ at $\tau = 1$. The actual map that

is predicted by the network is thus still similar to the map in Equation (5.6), but with an added dependency on τ and the interpolation $\mathbf{X}^\tau$:

$$\mathbf{X}^1 = f\left([\mathbf{F}_t]^{t\in\{0,\dots,T-1\}}, G, \mathbf{X}^\tau, \tau\right), \tag{5.26}$$

where G encodes the geometry and $\{\mathbf{F}_t\}$ is the loading trajectory.

During integration, the corresponding flow field is then obtained from the prediction $\hat{\mathbf{X}}^1$ as

$$\mathbf{U}^\tau = \frac{\hat{\mathbf{X}}^1 - \mathbf{X}^\tau}{1-\tau}. \tag{5.27}$$

This formulation is fully equivalent to standard flow matching, but better aligned with the structure of our model. In particular, it allows the EGNN to focus on predicting physically consistent deformation trajectories, while still benefiting from the probabilistic formulation of flow matching.

Symmetric Coupling In flow matching, training requires pairing a prior sample $\mathbf{X}^0$ with a target sample $\mathbf{X}^1$. A naive strategy is to pair them arbitrarily, that is, sample $\mathbf{X}^0 \sim p_0$ and $\mathbf{X}^1 \sim p_1$ independently. This is known to be suboptimal in flow matching, as it leads to unnecessarily complex transport paths which need more integration steps, and slows down training [70].

To address this, we use a coupling strategy inspired by optimal transport [70] and equivariant flow matching methods [71, 72]. Specifically, we use the *symmetric coupling* introduced in our previous work [33], which is tailored to symmetry-breaking problems and approximates optimal transport coupling under group action.

The idea is to exploit the fact that all symmetry-related targets are equivalent solutions. Instead of using $\mathbf{X}^1$ directly, we select the symmetry-transformed version that is closest to the prior sample, in terms of the Euclidean distance. Formally, given a prior sample $\mathbf{X}^0$, a target $\mathbf{X}^1$, and the stabilizer group $\mathcal{G}_{\text{input}}$, we compute:

$$\tilde{g} = \arg\min_{g\in\mathcal{G}_{\text{input}}} \|\mathbf{X}^0 - g\cdot\mathbf{X}^1\|^2, \quad \tilde{\mathbf{X}}^1 = \tilde{g}\cdot\mathbf{X}^1. \tag{5.28}$$

Training then proceeds using $\tilde{\mathbf{X}}^1$ as the target instead of $\mathbf{X}^1$. For a finite group, this requires evaluating $|\mathcal{G}_{\text{input}}|$ cost values, which is tractable for small groups like the four periodic shifts.

5.3.5 Time Bundling using Temporal U-Nets

We predict the full loading trajectory jointly rather than using an autoregressive time-stepping model. This is a design choice motivated by the structure of the buckling problem. Only a few time steps exhibit a bifurcation where the solution splits into multiple branches. In an autoregressive setting, the model would need to predict just a single solution at most time steps, which is ill-suited for a probabilistic model. A trajectory-level model can condition all time steps on each other and can therefore represent a coherent buckling path as a single sample from the learned distribution.

Related work Recently, research has moved away from the autoregressive paradigm, which is known to suffer from error accumulation, by exploring methods that can predict multiple time steps at once in parallel [36,37,73–77]. Some methods [73] directly predict several future steps at once from a history of past observations. In contrast, probabilistic methods based on diffusion or flow matching model a distribution over entire trajectories given the past. These methods start from a noisy trajectory (the prior sample) and then use diffusion or flow matching to map this into a coherent prediction [36,37,74–77], conditioned on the history. Both approaches turn the problem into a sequence-to-sequence mapping problem. Since the entire trajectory must be consistent over time, all time steps need to be modeled jointly. This can be done using a U-Net or a Transformer, which allows information to be exchanged between time steps.

Method This means flow matching defines a path from a prior trajectory to a target trajectory, and the model predicts all deformed positions $\vec{x}_{i,t}$ for all loading time steps t jointly, i.e., both input and output are a *time series of graphs*. Since the number of nodes and the edge connectivity are fixed across time steps, we can also view the trajectory as one graph whose node and edge features are time series. This allows us to use a single model to predict the entire trajectory, and to share information across time steps within the model. This is important for ensuring the consistency of the predicted trajectory; all time steps after buckling must agree on which way to buckle. The key architectural addition are 1D U-Nets (along the loading time axis t) used as the message function $\mathcal{F}_m$ and node update function $\mathcal{F}_h$ in Equations (5.7) and (5.8). The U-Nets process features for all time steps jointly, allowing information to propagate across time within a single message-passing step. Given a time-series of concatenated generalized edge features $\mathbf{e'}_{ij,t} = [\mathbf{h}_{i,t}, \mathbf{h}_{j,t}, \phi_{ij,t}, \mathbf{e}_{ij}, \tau]$, the first U-Net computes:

$$[\mathbf{m}_{ij,t}]_{t=0}^{T-1} = \mathrm{UNet1D}\left([\mathbf{e'}_{ij,t}]_{t=0}^{T-1}\right), \tag{5.29}$$

where we ensure dependence on the flow matching time τ as described by Equation (5.26) through concatenating τ to the edge vectors, although there are more sophisticated ways such as FiLM [78]. The node update U-Net then computes:

$$\left[\mathbf{h}_{i,t}^{l+1}\right]_{t=0}^{T-1} = \mathrm{UNet1D}\left(\left[\mathbf{h}_{i,t}^{l}, \square_{j\in\mathcal{N}(i)}\mathbf{m}_{ij,t}\right]_{t=0}^{T-1}\right). \tag{5.30}$$

Microfluctuation increments The time-bundled setting additionally enables an equivariant update direction that is not available in static graph formulations: the *increment* in microfluctuation between consecutive loading time steps:

$$\delta\vec{w}_{i,t} = \vec{w}_{i,t+1} - \vec{w}_{i,t}. \tag{5.31}$$

This vector captures the local direction of motion of a node i along the loading trajectory (i.e., essentially a discrete velocity). We use $\delta\vec{w}_{i,t}$ in two complementary ways.

First, the norm $\|\delta\vec{w}_{i,t}\|$ is appended to the node embedding as an $E(n)$-invariant scalar feature, informing the network about the local rate of deformation change

between time steps. In particular, it helps identify regions undergoing rapid changes, such as the onset of buckling, or time steps where the noise is still too high and needs to be smoothed out.

Second, the standard EGNN position update in Equation (5.14) is augmented with additional equivariant shifts along the normalized increment directions immediately before and after time step t:

$$\vec{x}_{i,t}^{\,l+1} = \vec{x}_{i,t}^{\,l} + \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} c_{ij} \left(\vec{x}_{j,t}^{\,l} - \vec{x}_{i,t}^{\,l} \right) + a_{i,t}\, \delta\hat{\vec{w}}_{i,t-1} + b_{i,t}\, \delta\hat{\vec{w}}_{i,t}, \tag{5.32}$$

where $\delta\hat{\vec{w}}_{i,t} = \delta\vec{w}_{i,t}/\|\delta\vec{w}_{i,t}\|$ is the normalized increment direction, and $a_{i,t}, b_{i,t}$ are scalar coefficients predicted from the node embedding.

The two terms correspond to information from the *incoming* and *outgoing* time directions:

- $\delta\hat{\vec{w}}_{i,t-1}$ encodes the direction from which the node arrived,
- $\delta\hat{\vec{w}}_{i,t}$ encodes the direction in which the node is evolving.

Since we start from a noisy prior, these can be substantially different. Because the increment vectors are physical vectors that transform consistently under rigid motions, these additional terms are $E(n)$-equivariant by construction.

The rationale is twofold;

1. The graph at a single time step is invariant to global rigid motions. As a result, a node cannot infer how much it has moved between time steps using only local geometric information (i.e., distances to neighbors).[4] The increment $\delta\vec{w}_{i,t}$ provides this missing temporal information directly.
2. In quasi-static buckling trajectories, the direction of motion between consecutive time steps is strongly correlated. By explicitly providing this directional information and using these extra update directions, the network can let nodes decide how much they want to be influenced by their neighbors versus how much they want to follow the trajectory direction. This allows the network to coordinate the time steps and create a more coherent trajectory.

See Figure 5.3; in addition to the standard edge vectors (black), the network can also use the temporal increment directions (green) to guide position updates.

5

5.3.6 Prior Distribution

One advantage of flow matching is that it allows for flexibility in the choice of the prior distribution. Bennema ten Brinke et al. [37] showed that the random walk prior is an excellent choice for flow matching of spatiotemporal data. We adopt this approach here, by applying a random-walk prior over microfluctuations on mesh nodes that models the trajectory as a cumulative sum of Gaussian noise increments. This is motivated by the fact that it yields temporally correlated samples and better matches the structure of quasi-static trajectories than independent Gaussian noise.

[4] This is mitigated by the fact that the mean microfluctuation is always $\vec{0}$, so there is no rigid translation, but it still means that a node can only know its own motion by gathering information from *all* other nodes.

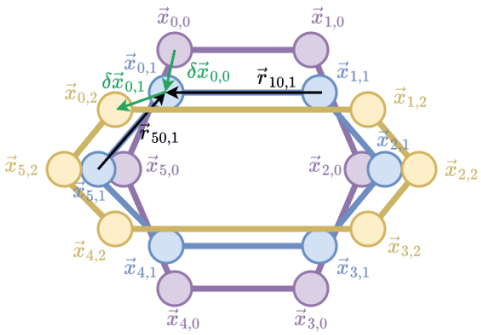


Figure 5.3: An illustration showing a geometric graph deforming over three time steps (purple to blue to yellow). During a regular message-passing update, node 0 uses only the norms of the edge vectors $\|\vec{r}_{ij,t}\|$ (in black) to its neighbors as input, and the edge vectors $\vec{r}_{ij,t}$ themselves to push and pull the node positions. In our time-bundled setting, we additionally allow the network to use $\delta\vec{w}_{i,t-1}$ and $\delta\vec{w}_{i,t}$, here equivalent to $\delta\vec{x}_{i,t-1}$ and $\delta\vec{x}_{i,t}$ (in green) since this figure does not show the affine deformation.

In our implementation, the random walk is added on top of the affine deformation, so the prior samples are of the form

$$\vec{x}^0_{i,t} = \vec{x}^{\text{aff}}_{i,t} + \vec{w}^0_{i,t}, \tag{5.33}$$

For each mesh node i in graph G, we sample i.i.d. Gaussian increments $\boldsymbol{\xi}_{i,t} \sim \mathcal{N}(\vec{0}, \mathbf{I})$ take their cumulative sum to create a random walk:

$$\vec{w}^0_{i,t} \leftarrow \sum_{s=1}^{t} \boldsymbol{\xi}_{i,s}, \qquad \vec{w}^0_{i,0} = \vec{0}. \tag{5.34}$$

The first step is pinned to zero to satisfy the reference-state condition.

To remove rigid translation modes, we enforce zero-mean microfluctuation per graph and loading time:

$$\vec{w}^0_{i,t} \leftarrow \vec{w}^0_{i,t} - \frac{1}{N}\sum_{j=0}^{N-1} \vec{w}^0_{j,t}. \tag{5.35}$$

Finally, we normalize the amplitude of $\vec{w}^0_{i,t}$. For each loading time t, we define a deformation magnitude

$$\gamma_t = \|\mathbf{F}_t - \mathbf{I}\|_F\,, \tag{5.36}$$

where $\|...\|_F$ is the Frobenius norm, and prescribe a target prior magnitude β_t that is proportional to the deformation magnitude:

$$\beta_t = c_w\,\gamma_t, \tag{5.37}$$

with the coefficient c_w calibrated from the training data.

Finally, we normalize amplitudes so that the graph-level mean node magnitude matches β_t (using a small ε for stability when the average microfluctuation norm is close to zero):

$$\vec{w}^0_{i,t} \leftarrow \vec{w}^0_{i,t} \frac{\beta_t}{\frac{1}{N}\sum_{j=0}^{N-1} \|\vec{w}^0_{j,t}\| + \varepsilon}, \quad \varepsilon = 10^{-8}. \tag{5.38}$$

The resulting prior is thus: (i) temporally correlated (random walk), (ii) proportional to the amount of deformation through $\|\mathbf{F} - \mathbf{I}\|_F$, (iii) consistent with the physical constraints (zero-mean over mesh nodes, no microfluctuation at $t = 0$), and (iv) consistent with the deformed tiling through $\vec{x}^{\text{aff}}_{i,t}$, which incorporates $\mathbf{F}$ and periodicity.

See Figure 5.4 for an example of a sample from the prior, and its transformation into a prediction.

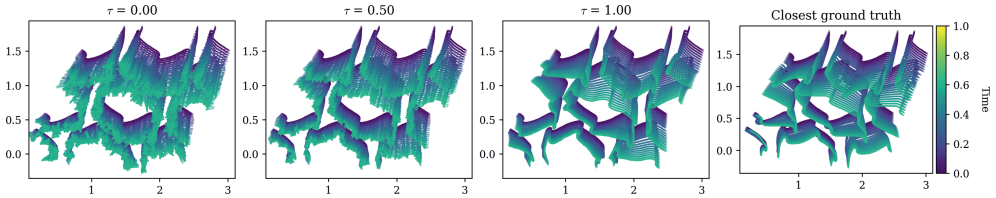


Figure 5.4: An example of a sample of the random walk prior that is transformed into a prediction. Each figure shows all the loading time steps of the trajectory, with color indicating the loading time t. The trajectory ends before $t = 1$ because the ground truth trajectory is cut at contact. 32 integration steps.

5.3.7 Graph Representation

A metamaterial unit cell is represented as a graph $G = (\mathcal{V}, \mathcal{E})$, derived from its FE discretization. We remove all bulk nodes; i.e., we only keep nodes on the hole boundaries. The graph is thus composed of:

- **Nodes** $i \in \mathcal{V}$: mesh points on pore boundaries with associated reference positions $\vec{x}^{\text{ref}}_i \in \mathbb{R}^2$.
- **Edges** $(i, j) \in \mathcal{E}$: mesh edges that are part of the hole boundaries with associated edge attributes: reference vector $\vec{r}^{\text{ref}}_{ij} = \vec{x}^{\text{ref}}_j - \vec{x}^{\text{ref}}_i$ and reference length d^{ref}_{ij} (both corrected for periodicity using minimum image convention, recall Subsection 5.3.3), and integer lattice-shift vector $T_{ij} \in \mathbb{Z}^2$ encoding periodicity.

Auxiliary Nodes & Edges Since this results in a disconnected graph (each hole is its own connected component), we add auxiliary nodes and edges to connect the graph. For the square stacked holes dataset, we re-use the graph representation as described in the original paper [33], which for each node, adds an edge to the closest node on each other hole. For the wallpaper group-based metamaterials based dataset, we add both auxiliary nodes and edges:

- To connect the holes, we use ray tracing; at each node, we start a ray normal to the hole boundary, going inwards into the bulk, and add an edge to the closest node belonging to the first edge that is intersected by the ray. This ensures the edges lie in the bulk, and therefore represent possible paths of load transfer through the bulk.
- We add edges between equivalent nodes (i.e., nodes that are periodic copies of each other), since identical nodes must do something different from each other when buckling. This way, each node has direct access to symmetry-breaking information concerning itself.[5]
- We also incorporate information from the geometry generation process into the graph representation; we add nodes and edges from the graph skeleton and the primary, secondary, and tertiary polygons as described in the wallpaper dataset paper [26].
- To improve propagation of information, we connect each mesh node to the closest Bézier point (i.e., a point from the tertiary polygons).

These different types of edges and nodes are indicated by a one-hot encoding, which is converted into an embedding by the models.

See Figure 5.5 for an example of all the different types of nodes and edges in the graph representation.

5.3.8 Hierarchical Graph Neural Networks

When applying GNNs to physical systems, one can often simply assume a finite cutoff radius beyond which interactions are negligible, as is commonly done in atomistic machine learning models [58, 59, 79, 80]. However, this assumption does not hold in structural mechanics, particularly for quasi-static problems, where the deformation at a point depends on the global state of the structure. In such systems, load transfer is effectively instantaneous, and especially phenomena such as buckling require coherent global coordination. While each local region of the structure may be compatible with multiple deformation patterns, physical solutions require that all regions collectively agree on a single solution. Consequently, models require a global receptive field, i.e., all nodes must be allowed to interact with each other.

Even though GNNs can in principle capture long-range dependencies by stacking many layers, in practice they suffer from oversmoothing and oversquashing, limiting their effectiveness on graphs with a large diameter (i.e., a large maximum degree of separation) [81–84]. All-to-all attention mechanisms can mitigate this issue [56], but they scale quadratically with the number of nodes, which is prohibitive for large graphs.

This motivates architectures that explicitly propagate information across multiple scales. The resulting design is conceptually related to multigrid methods in the finite element literature, where coarse representations enable efficient propagation of global information [85–87].

[5] Because of the periodicity, this results in multiple edges between the same pair of nodes, since a node can be connected to its periodic copy in multiple ways, e.g., to the left or to the right are both the same under the minimum image convention.

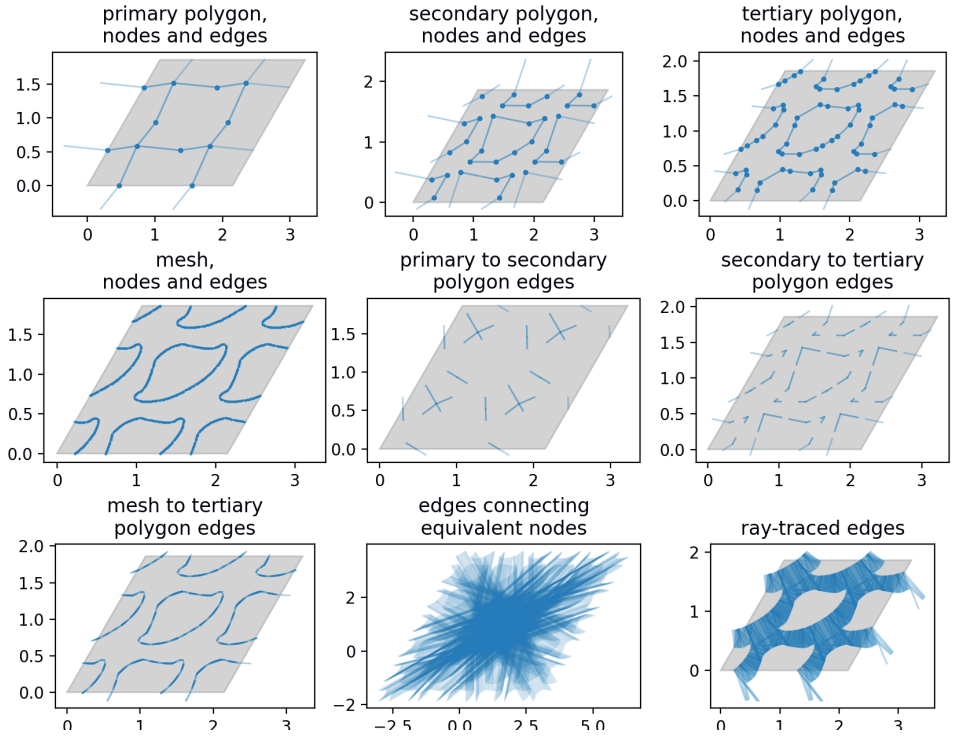


Figure 5.5: The different types of nodes and edges in the graph representation.

Related work Various hierarchical approaches have been proposed to capture long-range interactions in graphs. They all construct some variant of a coarsened representation, or multiple levels of coarsened representations. Roughly, there are three main approaches to create the coarse representation:

1. Similarity-based methods group nodes according to feature similarity or learned representations. For example, DiffPool [88] learns soft cluster assignments, allowing nodes with similar embeddings to be aggregated even if they are far apart in space. EGHNs [89] can even do this in an $E(n)$-equivariant way.
2. Proximity-based methods group nodes based on metric distances or graph connectivity. This includes clustering approaches (e.g., community detection, k-nearest neighbors), hierarchical constructions based on spatial organization such as tree-based methods (e.g., Erwin [90]), and using predefined meshes of varying resolutions as in MultiScale MeshGraphNets [91]. These approaches ensure that grouped nodes are physically related, which is desirable in geometric and physical systems.
3. Selection-based methods do not explicitly form clusters, but instead select a subset of nodes based on fixed rules (e.g., Farthest Point Sampling as in PointNet++ [92], randomly sampling in UPT [93]), or learned scores (e.g., SAG-

Pool [94] or Graph U-Nets [95]). In practice, these methods often still aggregate information from local neighborhoods, e.g., PointNet++ and UPT aggregate information from non-selected nodes into a group around selected nodes, and can therefore be interpreted as implicitly defining proximity-based regions, although without an explicit clustering structure.

These approaches all use pooling and unpooling operations (sometimes termed 'coarsening' and 'broadcasting') to transmit information from one level of the hierarchy to another.

Method We implement a hierarchical extension of the $E(n)$-equivariant graph neural network (EGNN) that introduces multiple levels of graph coarsening. In contrast to similarity-based pooling, nodes are grouped based on spatial proximity and graph connectivity, which is more appropriate for continuum mechanics settings where spatial locality defines coarse degrees of freedom. This avoids grouping physically unrelated nodes that may share similar features (e.g., due to periodicity), and allows us to assign a position to each supernode representing the mean position of the cluster, while pooling based on similarity would assign a location that is meaningless since it is an average of unrelated locations.

Coarse representation We use a simple pooling strategy based on fixed clustering using the Louvain community detection algorithm [96],[6] which clusters nodes based on connectivity, which in our case is in turn based on proximity. For this, we use the mesh nodes and edges as well as the auxiliary rays described in the previous Subsection 5.3.7. This yields near-linear complexity while correctly handling interactions across periodic boundaries, since the graph is already periodic. As an extra motivation, we would like to point out that the graph representation connects neighboring nodes around the holes (through the mesh edges), as well as nodes on opposite sides of ligaments (through the rays), which are exactly the nodes that are most likely to move together. Since in our case, the material connectivity is fixed and the reference configuration is always the stress-free configuration in our elastic setting, the original graph remains physically meaningful throughout the deformation. We therefore hypothesize that reclustering as the material deforms is of limited value. At the coarse level, each cluster is represented by a supernode, whose initial position is the mean of the initial positions of the cluster's member nodes.[7] To connect the supernodes into a new, coarser graph, we use the 3 nearest neighbors of each supernode, under the minimum image convention. See Figure 5.6 for a figure of the clusters and the corresponding coarse graph. This may result in disconnected components in the coarse graph. In our implementation, the hierarchy currently consists of only two levels, but can be extended to arbitrary levels in the future.

[6] Note that the Louvain community detection algorithm is non-deterministic; running the algorithm multiple times may result in slightly different clusterings. We do not consider this to be a problem, the model should be able to accommodate this.

[7] Taking the mean of a set of positions in a periodic space is non-trivial; we create an initial guess for the mean position based on the mid-point of the longest edge between two cluster member nodes, and refine it by making sure the average vector from supernode to member node under the minimum image convention is $\vec{0}$.

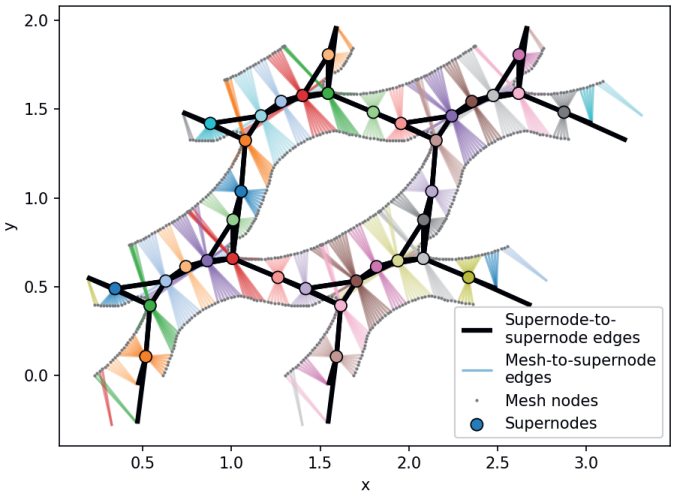


Figure 5.6: Clusters and resulting coarse graph.

Implementation The prior is only applied to the finest level, and the positions of the coarser levels are initialized as the mean of the affine positions of the nodes in the finer level in that cluster.

At each coarsening level, a supernode embedding is computed as a weighted aggregation of its constituent node embeddings, where weights are predicted by the network and depend on relative positions. Similarly, supernode positions are updated using a weighted aggregation of nodal displacements (microfluctuations), ensuring consistency with $E(n)$-equivariance.

During unpooling, finer node features are updated by combining their local embeddings with the corresponding supernode embedding through a learnable function. Node positions are updated by incorporating coarse-level displacements, modulated by learned coefficients that determine how strongly each node follows the motion of its associated supernode.

At the coarsest level, the number of nodes is sufficiently small to enable all-to-all message passing (or attention), which allows for global interactions to be captured effectively. Accordingly, we do not consider the possibility of the coarse graph being disconnected to be a big problem, but in the future optimizations of the graph representation could be done to ensure connectivity.

Figure 5.7 shows a schematic overview of the hierarchical EGNN.

Novelty The proposed approach does not introduce a fundamentally new hierarchical GNN paradigm; rather, it adapts established ideas from hierarchical and multiscale GNNs to the specific requirements of equivariant modeling in periodic structural me-

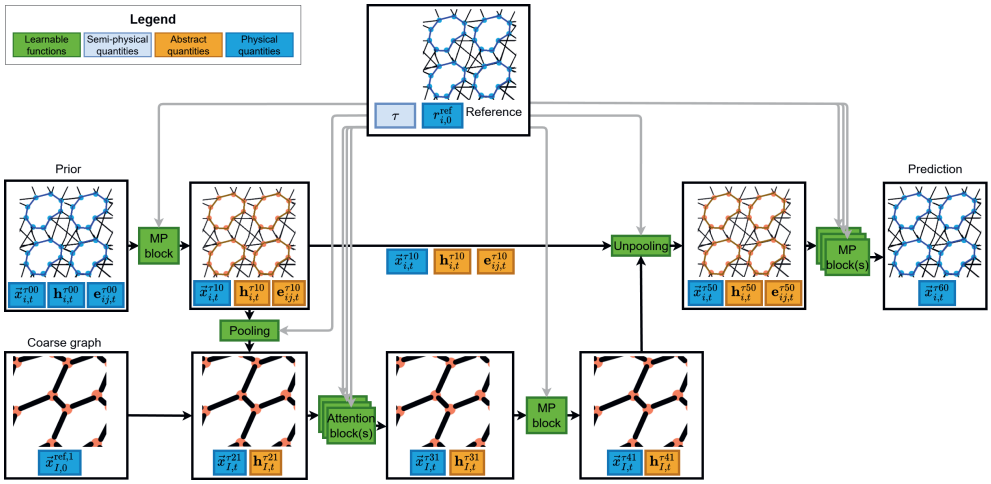


Figure 5.7: Schematic overview of the hierarchical EGNN. For visual clarity, we show only a single loading time step.

chanics. The main contribution is therefore the *integration* of hierarchical coarsening with EGNN-style equivariant updates in a setting where global interactions are essential, together with a physically motivated choice of pooling strategy. This contrasts with most prior EGNN applications, which rely on local neighborhoods and do not explicitly incorporate a multilevel structure.

5.3.9 Physical Constraints

The microfluctuation field $\vec{w}_{i,t}$ captures deviations from the affine deformation. As described in Subsection 5.3.6 on the prior sampling, it therefore needs to satisfy certain constraints to ensure physical validity of the solution. This does not apply just to the prior, but also to the prediction. The model could learn this from data, but we choose to enforce it explicitly to improve learning efficiency and convergence. After every position update, the microfluctuation field is adjusted to satisfy two constraints: (1) zero-mean constraint to remove rigid-body translation mode, such that $\frac{1}{N}\sum_i \vec{w}_{i,t} = \vec{0}$, and (2) no microfluctuation in the reference configuration (i.e., $\vec{w}_{i,0} = \vec{0}$).

We enforce these constraints as follows: 1) we subtract the mean microfluctuation from $\vec{x}_{i,t}$ at each loading time step,

$$\vec{x}_{i,t} \leftarrow \vec{x}_{i,t} - \frac{1}{N}\sum_i \vec{w}_{i,t}, \tag{5.39}$$

and 2) we replace the positions at $t = 0$ with the reference positions to ensure $\vec{w}_{i,0} = \vec{0}$.

5.3.10 Angles

Angles improve expressivity of graph neural networks [97]. Physically speaking, large changes in angles compared to the reference configuration will lead to large mechanical energy, so the angles are an important feature in finding a solution.

Other graph neural networks have incorporated angles into their architecture, e.g., DimeNet [98], DimeNet++ [99], GemNet [100] and PaiNN [60]. However, this comes at a cost of increased model complexity and computational time, because the angles are computed for each *pair* of adjacent edges per node, which can scale poorly depending on the average node degree.

Since we include only the hole boundaries of the mesh, each mesh node (i.e., the finest level of the hierarchy, excluding the auxiliary nodes) is part of exactly two mesh edges, so the angle, denoted $\phi_{i,t}$, between these edges can be used as a node feature. This is advantageous because it allows us to use it as an additional description of the current configuration, without needing to make changes to the architecture and with a negligible increase in model complexity.

5.4 Implementation

We train and compare three models of increasing complexity to isolate the contribution of the different building blocks introduced in Section 5.3. This staged comparison follows the same logic as an ablation study, but in the opposite direction: rather than removing components from the full model one by one, we add them progressively, which makes it possible to assess the effect of each extension. The models progress from a static formulation to a full spatio-temporal hierarchical architecture:

- a **static model**, which predicts a single deformation state,
- a **time-dependent model**, which predicts complete deformation trajectories,
- a **hierarchical time-dependent model**, which improves the handling of long-range interactions with a hierarchical architecture.

Table 5.1 summarizes the main design choices related to each building block of Section 5.3; detailed descriptions follow below.

5

5.4.1 Handling invalid time steps

Some trajectories contain invalid time steps due to contact. These steps are excluded from the loss, but still need to appear in the network input due to temporal convolutions which mix all time steps together.

The prior sample $\mathbf{X}^0$ is defined for all time steps, since the prior does not encode contact information. In contrast, the target trajectory $\mathbf{X}^1$ may contain invalid values after contact occurs. However, during training the input to the flow matching GNN is an interpolation between $\mathbf{X}^0$ and $\mathbf{X}^1$, so target values are needed at every time step. We therefore replace invalid target values by the last valid microfluctuation:

$$\vec{w}_{i,t} = \vec{w}_{i,T_{\text{valid}}-1} \quad \text{for } t \geq T_{\text{valid}}, \tag{5.40}$$

where T_{valid} denotes the number of valid time steps in the sample.

Table 5.1: Summary of the three models and their components.

Model →	Static Flow Matching GNN	Non-hierarchical time-dependent flow matching GNN	Hierarchical time-dependent flow matching GNN
Attributes ↓	Section 5.4.3	Section 5.4.4	Section 5.4.5
Dataset	Square stacked holes dataset	Wallpaper dataset (overfit on one trajectory)	Wallpaper dataset (overfit on full dataset)
GNN	GNN, EGNN, SimEGNN (Hendriks et al. [1])	EGNN (Hendriks et al. [1])	EGNN (Hendriks et al. [1])
Architecture	11 message-passing steps	9 or 14 message-passing steps	2-level hierarchy, 3 attention blocks, 5 MP steps
Physical edge features	SimEGNN: Strain, relative current distance. EGNN: $r_{ij,t}$, r_{ij}^{ref}. GNN: $\vec{r}_{ij,t}$, $r_{ij,t}$, $\vec{r}_{ij}^{\text{ref}}$, r_{ij}^{ref}	Strain, $\vec{r}_{ij}^{\text{ref}}$, r_{ij}^{ref}	Strain, $\vec{r}_{ij}^{\text{ref}}$, r_{ij}^{ref}
Physical node features	–	Angle $\phi_{i,t}$, microfluctuation increment norm $\|\delta\vec{w}_{i,t}\|$	Angle $\phi_{i,t}$, microfluctuation increment norm $\|\delta\vec{w}_{i,t}\|$
Periodicity	Yes	Yes	Yes
Time bundling	N/A	Temporal U-Nets	Temporal U-Nets
Prior	Affine deformation + Gaussian noise	Affine deformation + random walk	Affine deformation + random walk
Hierarchical	No	No	Yes
Graph representation	See Hendriks et al. [1]	See Subsection 5.3.7	See Subsection 5.3.7
Constraints	Mean microfluctuation = $\vec{0}$, enforced after every GNN call	No deformation at $t = 0$; mean microfluctuation = $\vec{0}$, enforced after every update of the positions during GNN call	No deformation at $t = 0$; mean microfluctuation = $\vec{0}$, enforced after every update of the positions during GNN call
Angles	No	Yes	Yes
Symmetric coupling	Yes	Yes	Yes

This corresponds to assuming that deformation becomes purely affine after contact. The goal is to make these regions as uninformative as possible so that the network focuses on physically meaningful (pre-contact) dynamics.

5.4.2 Training

Training follows the standard flow matching paradigm (see Subsubsection 5.3.4), but with three important modifications:

1. some time steps are masked (as explained in the previous Section 5.4.1)
2. the network predicts *endpoint positions* instead of velocities (as explained in Section 5.3.4 – Flow Matching, Equation (5.26))
3. targets also depend on the input loading and graph representation $\left([\mathbf{F}_t]_{t=0}^{T}, G\right)$,
4. as explained in Section 5.3.4, symmetric coupling is applied to remove ambiguity in the targets (Equation (5.28)).

Formal objective Given a prior sample $\mathbf{X}^0$ and a target trajectory $\mathbf{X}^1$ with invalid time steps replaced, we construct the interpolated input:

$$\mathbf{X}^\tau = (1-\tau)\mathbf{X}^0 + \tau\mathbf{X}^1. \tag{5.41}$$

The network predicts the endpoint $\hat{\mathbf{X}}^1$, and the loss is computed only on valid time steps:

$$\mathcal{L} = \mathbb{E}_{\tau,\mathbf{X}^0,\mathbf{X}^1}\left[\left\|\hat{\mathbf{X}}^1_{\text{valid}}\left(\tau, \mathbf{X}^\tau, [\mathbf{F}_t]_{t=0}^{T}, G\right) - \mathbf{X}^{1,*}_{\text{valid}}\right\|^2\right], \tag{5.42}$$

where $\mathbf{X}^{1,*}$ is the symmetry-aligned target obtained via symmetric coupling, where only the valid time steps are used to select a target.

Oversampling Buckling trajectories are both more difficult to model and more relevant for the present problem. We therefore oversample:

- all buckling trajectories, which are seen a total of 3 times per epoch,
- trajectories with ≥ 4 post-buckling steps, which are seen a total of 7 times per epoch.

In practice, this corresponds to one pass over the full training dataset (8560 trajectories), followed by two additional passes over all buckling trajectories (1995 trajectories), and four additional passes over the subset of trajectories with extended post-buckling regimes (1109 trajectories, which are a subset of the buckling trajectories).

5.4.3 Static Flow Matching GNN (Stacked-Hole Model)

This model serves as a baseline to evaluate whether flow matching alone, combined with an equivariant architecture, is sufficient to model solution multiplicity.

It predicts a single microfluctuation field:

$$\vec{w} \in \mathbb{R}^{N\times 2} \tag{5.43}$$

for a given deformation gradient $\mathbf{F}$.

Architecture We use three variants adapted to metamaterial deformation from [1]:
- GNN (translation and permutation equivariant, periodicity equivariant),
- EGNN (additionally $E(n)$-equivariant),
- SimEGNN (additionally similarity-equivariant).

This comparison isolates the effect of different equivariance assumptions. The models use 11 message-passing steps. The flow matching time τ is concatenated to edge features. It does not yet exploit angles or a hierarchy. These models still contain four learnable functions per message-passing step (message computation, node update, edge update, position shift computation), unlike the time-dependent models, in which some functions have been merged (see Equation (5.10) and (5.15)).

Following our previous work [1], we normalize the edge vectors $\vec{r}_{ij} = \vec{x}_j^l - \vec{x}_i^l$ and use a hyperbolic tangent to restrict the range the nodes can move in one step to improve stability, which modifies Equation (5.14) to

$$\vec{x}_i^{l+1} = \vec{x}_i^l + \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \tanh\left(c_{ij}\right) \frac{\vec{r}_{ij}}{\|\vec{r}_{ij}\|}, \tag{5.44}$$

where l is the number of node updates so far.

Prior Since this model does not yet predict entire trajectories, the prior from Section 5.3.6 is reduced to an affine deformation plus Gaussian noise:

$$\vec{x}_i^0 = \mathbf{F}\vec{x}_i^{\text{ref}} + \sigma_w \boldsymbol{\xi}_i, \quad \boldsymbol{\xi}_i \sim \mathcal{N}(\vec{0}, \mathbf{I}). \tag{5.45}$$

with the scaling σ_w determined from the training data statistics.

5.4.4 Non-hierarchical Time-Dependent Model

This model predicts full deformation trajectories, to test if flow matching can create consistent trajectories in a multi-solution setting, and to serve as a stepping stone towards the hierarchical model.

It predicts the entire trajectory

$$\mathbf{X} \in \mathbb{R}^{N \times 2 \times T} \tag{5.46}$$

of a microstructure, given the full loading path $[\mathbf{F}_t]^{t \in \{0,\dots,T-1\}} \in \mathbb{R}^{2\times 2\times T}$.

Architecture All time steps are processed jointly using temporal convolutions as explained in Subsection 5.3.5, allowing the model to capture correlations across the loading path. The model is an EGNN augmented with temporal U-Nets and convolutional blocks:
- 3 layers (first, last, middle) use U-Nets for global temporal context,
- intermediate layers use shallow convNets for efficiency and are repeated more times,

- total depth: 9 or 14 message passing steps.

The choice of the depth depends on the maximum graph diameter in the training data; for the full dataset, this is 14, but for a single trajectory used for an overfitting experiment, this is 9.

The angle $\phi_{i,t}^{\tau l}$ (as discussed in Subsection 5.3.10) and the norm of the microfluctuation increment $\|\delta\vec{w}_{i,t}^{\tau l}\|$ (as discussed in Subsection 5.3.5) are concatenated to the node embedding $\mathbf{h}_{i,t}^{\tau l}$. Following the EGNN implementation described in [1], we concatenate physical scalars – the strain $\varepsilon_{ij,t}^{\tau l}$, the current distance $r_{ij,t}^{\tau l}$, and the reference distance $r_{ij,t}^{\text{ref}}$ – as well as the flow-matching time τ to the edge embedding. See Figure 5.8 for a schematic overview of the message-passing step.

5.4.5 Hierarchical Time-Dependent Flow Matching GNN

This model addresses a key limitation of deep message passing: long-range interactions are difficult to capture due to oversmoothing, oversquashing, and computational cost. As explained in Subsection 5.3.8, we implement a two-level hierarchy that enables global information exchange while maintaining local resolution.

Architecture summary We now describe the forward pass of the hierarchical model. The goal of the hierarchy is to combine local interactions at the fine scale and global interactions at a coarse scale, in order to coordinate all the nodes into a buckling trajectory that is coherent both in time and space. The model follows these steps:

1. **Local message passing (fine graph)** captures short-range interactions,
2. **Pooling to coarse graph** aggregates local information into cluster nodes,
3. **Global attention at coarse level** enables long-range interactions between clusters,
4. **Message passing at coarse level** incorporates local information at the coarse level,
5. **Unpooling to fine graph** distributes global information back to nodes,
6. **Local refinement** using message-passing refines predictions using combined information.

Recall also Figure 5.7 for a schematic overview. Currently, our data representation, and therefore our model, only has two levels in the hierarchy ($p = 0$, fine and $p = 1$, coarse, not to be confused with any probability). However, it is possible to have more levels, in which case Stage 1 and 2 are repeated more than once, alternating, as well as Stage 4 and 5.

Stage 1 — Local message passing at $\mathbf{p = 0}$
At the fine level, we apply local message-passing identical to the non-hierarchical model (Section 5.4.4, Figure 5.8), updating node features and positions using local neighborhood information.

Stage 2 — Pooling: $\mathbf{p \to p + 1}$
We aggregate fine nodes into coarse cluster nodes. For each fine-to-coarse edge ($i \to$

J), we construct edge features:

$$\mathbf{e}_{iJ,t}^{\tau lp\to p+1} = \left[\mathbf{h}_{i,t}^{\tau lp}, r_{iJ,t}^{\tau lp\to p+1}, r_{iJ,t}^{\mathrm{ref},p\to p+1}, \tau\right], \quad \text{where} \quad i \in C_J, \tag{5.47}$$

where C_J is the cluster represented by supernode J, $r_{iJ,t}^{\tau lp\to p+1}$ is the current distance between nodes i and J, and $r_{iJ,t}^{\mathrm{ref},p\to p+1}$ is the reference distance. In our notation we use capital indices for coarse nodes and lowercase indices for fine nodes.

We compute messages using a temporal U-Net:

$$\left[\mathbf{m}_{iJ,t}^{\tau lp\to p+1}\right]_{t=0}^{T-1} = \mathrm{UNet1D}\left(\left[\mathbf{e}_{iJ,t}^{\tau lp}\right]_{t=0}^{T-1}\right). \tag{5.48}$$

The coarse node embedding is obtained by aggregating these messages

$$\mathbf{h}_{J,t}^{\tau l+1p+1} = \frac{1}{\|C_J\|}\sum_{i\in C_J}\mathbf{m}_{iJ,t}^{\tau lp\to p+1}, \tag{5.49}$$

whereas the coarse position is updated using a weighted aggregation of fine microfluctuations

$$\vec{x}_{I,t}^{\tau l+1p+1} = \vec{x}_{I,t}^{\mathrm{aff},p+1} + \sum_{i\in C_J}\alpha_{iJ,t}\vec{w}_{i,t}^{p}, \tag{5.50}$$

where $\alpha_{iJ,t} = \mathrm{softmax}_{i\in C_J}\left(\mathbf{m}_{iJ,t}^{\tau lp\to p+1}[0]\right)$ is an attention score for the position update, obtained from one channel of the message. This allows the coarse node to focus on the most relevant fine nodes in its cluster when updating its position.

This is followed by enforcing the constraints as explained in Subsection 5.3.9. See Figure 5.9 for a schematic overview of the pooling step.

Stage 3 — All-pairs periodic attention at p = 1

At the coarse level, we allow all clusters to interact via self-attention. We use a modified attention mechanism that incorporates geometric pair features, and no absolute position encoding to ensure equivariance.

We first compute queries, keys and values for each coarse node at each time step:

$$\left[\mathbf{Q}_{I,t}\right]_{t=0}^{T-1} = \mathrm{Conv}_Q\left(\left[\mathbf{h}'^{\tau lp}_{I,t}\right]_{t=0}^{T-1}\right), \tag{5.51}$$

$$\left[\mathbf{K}_{I,t}\right]_{t=0}^{T-1} = \mathrm{Conv}_K\left(\left[\mathbf{h}'^{\tau lp}_{I,t}\right]_{t=0}^{T-1}\right), \tag{5.52}$$

$$\left[\mathbf{V}_{I,t}\right]_{t=0}^{T-1} = \mathrm{Conv}_V\left(\left[\mathbf{h}'^{\tau lp}_{I,t}\right]_{t=0}^{T-1}\right), \tag{5.53}$$

where $\mathbf{h}'^{\tau lp}_{I,t} = [\mathbf{h}^{\tau lp}_{I,t}, \tau, \|\delta\vec{w}^{\tau lp}_{I,t}\|]$. Here, $\mathbf{Q}_{I,t}$, $\mathbf{K}_{I,t}$ and $\mathbf{V}_{I,t}$ denote the query, key and value embeddings used in the attention mechanism: queries determine what information a node attends to, keys determine which other nodes and time steps are relevant, and values contain the information that is aggregated.

Geometric pair features are defined as:

$$\mathbf{f}_{IJ,t}^{\tau lp} = \left[r_{IJ,t}^{\tau lp}, r_{IJ,t}^{\mathrm{ref},p}, \varepsilon_{IJ,t}^{\tau lp}\right], \tag{5.54}$$

which are used to compute additional pairwise terms

$$[c_{IJ,t}]_{t=0}^{T-1} = \text{ConvNet}_c\left(\left[\mathbf{f}_{IJ,t}^{\tau lp}\right]_{t=0}^{T-1}\right) \in \mathbb{R}^{N_p \times N_p \times T}, \tag{5.55}$$

$$[b_{IJ,t}]_{t=0}^{T-1} = \text{ConvNet}_b\left(\left[\mathbf{f}_{IJ,t}^{\tau lp}\right]_{t=0}^{T-1}\right) \in \mathbb{R}^{N_p \times N_p \times T}, \tag{5.56}$$

$$[v_{IJ,t}]_{t=0}^{T-1} = \text{ConvNet}_v\left(\left[\mathbf{f}_{IJ,t}^{\tau lp}\right]_{t=0}^{T-1}\right) \in \mathbb{R}^{N_p \times N_p \times T}, \tag{5.57}$$

where N_p is the number of nodes at pooling level p.

The attention weights are then:

$$A_{IJ,t} = \text{softmax}_J\left(\frac{\mathbf{Q}_{I,t}\mathbf{K}_{J,t}}{\sqrt{D_{\text{attn}}}} + b_{IJ,t}\right), \tag{5.58}$$

where D_{attn} is the dimensionality of the keys and queries, used for numerical stability.

The node embeddings are then updated as

$$\left[\mathbf{h}_{I,t}^{\tau l+1p+1}\right]_{t=0}^{T-1} = \left[\mathbf{h}_{I,t}^{\tau lp+1}\right]_{t=0}^{T-1} + \sum_{J\in\mathcal{P}} \text{Conv}\left(\left[\mathbf{A}_{IJ}^{\tau lp}\left(V_{I,t}^{\tau lp} + v_{IJ,t}^{\tau lp}\right)\right]_{t=0}^{T-1}\right), \tag{5.59}$$

where $\mathcal{P}$ is the set of nodes at level p.

Positions are updated using both spatial and temporal directions, analogous to Equation (5.32):

$$\begin{aligned}\Delta\vec{x}_I^{\tau l+1p} = &\sum_{J\in\mathcal{P}} A_{IJ}^{\tau lp} \tanh\left(c_{IJ}^{\tau lp}\right)\hat{\vec{r}}_{IJ,t}^{\tau lp} \\ &+ (\mathbf{h}_{I,t}^{\tau l+1p}[0])\delta\hat{\vec{w}}_{IJ,t-1/2}^{\tau lp} \\ &+ (\mathbf{h}_{I,t}^{\tau l+1p}[1])\delta\hat{\vec{w}}_{IJ,t+1/2}^{\tau lp}\end{aligned} \tag{5.60}$$

The positions are then again adjusted to respect the physical constraints, as described in Subsection 5.3.9.

Figure 5.10 shows a schematic overview of the attention step.

Stage 4 — Local message passing at $\mathbf{p = 1}$
We perform another EGNN update on the coarse graph to refine global interactions.

Stage 5 — Unpooling: $\mathbf{p + 1 \to p}$
Coarse features are propagated back to fine nodes, with skip connections from Stage 1. Each fine node receives:

$$\mathbf{g}_{i,t}^{\tau lp} = \left[\mathbf{h}_{i,t}^{\tau lp}, \mathbf{h}_{J,t}^{\tau lp+1}, \tau, r_{iJ,t}^{\tau lp\to p+1}, r_{iJ,t}^{\text{ref},p\to p+1}\right], \qquad i \in C_J, \tag{5.61}$$

and updates its embedding:

$$\left[\mathbf{h}_{i,t}^{\tau l+1p}\right]_{t=0}^{T-1} = \text{ConvNet}\left(\left[\mathbf{g}_{i,t}^{\tau lp}\right]_{t=0}^{T-1}\right). \tag{5.62}$$

Positions are updated using coarse information:

$$\vec{x}_{i,t}^{\tau l+1 p} = \vec{x}_{i,t}^{\tau l p} + c_{i,t}^{\tau l p} \vec{w}_{J,t}^{\tau l p+1} \tag{5.63}$$

$$c_{i,t}^{\tau l p} = 1 + \tanh\left(\mathbf{h}_{i,t}^{\tau l p}[0]\right), \tag{5.64}$$

where $c_{i,t}^{\tau l p}$ functions as a gate, and (1+tanh) ensures the gate lies in $(0, 2)$, modulating the coarse information. Figure 5.11 shows a schematic overview of the unpooling step.

Stage 6 — Refinement at $\mathbf{p = 0}$
Finally, additional local message-passing steps refine the prediction at the fine scale.

5.4.6 Evaluation Metrics

In our evaluation, we wish to measure two aspects: 1) the accuracy of the predictions, and 2) the diversity of the predictions, i.e., whether the model is able to capture the full distribution of possible buckling modes. Fortunately, in our case the ground truth distribution of possible solutions is discrete and quite small (max. 4 solutions), which makes evaluation easier than in a general generative modeling setting.

For the accuracy, we look at the mean squared error (MSE) of the predicted node positions $\vec{x}_{i,t}^{\text{pred}}$ after integrating the predicted flow field compared to the closest ground-truth solution, the same alignment as used in symmetric coupling during training, i.e.,

$$\text{MSE-tC} = \frac{1}{N T_{\text{valid}}} \|\mathbf{X}^{\text{pred}} - \mathbf{X}^{\text{gt},*}\|^2, \tag{5.65}$$

$$\text{where } \mathbf{X}^{\text{gt},*} = \arg\min_{\mathbf{X}^{\text{gt},a}} \|\mathbf{X}^{\text{pred}} - \mathbf{X}^{\text{gt},a}\|^2. \tag{5.66}$$

Here, a runs over the possible valid solutions, and $\mathbf{X}^{\text{pred}} = \left[\vec{x}_{i,t}^{\text{pred}}\right]_{i\in[0,N-1]}^{t\in[0,T_{\text{valid}}-1]}$ is one entire predicted trajectory, where T_{valid} is the number of valid time steps in this sample. That is, for each sampled trajectory $\mathbf{X}^{\text{pred}} \in \mathbb{R}^{N\times T_{\text{valid}}\times 3}$, $\mathbf{X}^{\text{gt},*}$ is the ground-truth solution closest to the prediction $\mathbf{X}^{\text{pred}}$ among all symmetry-equivalent alternatives. We then take the mean of this MSE-tC over all samples in the test set to get the reported MSE-tC.

For the diversity, we look at the coverage of all possible solutions, which looks at the distribution $P(a)$ of generated samples over the different solutions. As our ground truth distribution $Q(a)$ we take the uniform distribution $Q(a) = q_a = 1/A$ with A the number of possible solutions (since all modes are equally likely). We then compute the Jensen-Shannon divergence (JSD) between the distribution over S samples from the model and the uniform distribution:

$$\text{JSD}(p, q) = \frac{1}{2}\sum_{a=0}^{A-1} p_a \log \frac{p_a}{\frac{p_a+\frac{1}{A}}{2}} + \frac{1}{2}\sum_{a=0}^{A-1} \frac{1}{A} \log \frac{\frac{1}{A}}{\frac{p_a+\frac{1}{A}}{2}}. \tag{5.67}$$

$$p_a = s_a/S, \tag{5.68}$$

where s_a is the number of generated samples that are closest to the ground-truth solution a. For this, we use the same mode assignment as for the MSE-tC. We calculate the JSD for each trajectory in the test set and take the mean over trajectories to obtain the mean JSD (MJSD). Since the ground truth probabilities q_i depend on the number of solutions, this metric is reported separately for the cases with 2 and 4 possible solutions. In the case of only 1 possible solution, we do not report this metric because it is not meaningful.

We also compare the obtained MJSD values to the expected value of the MJSD under a random sampling of S samples from the ground truth distribution, because even a perfectly uniform distribution over the modes may not show perfect uniformity in the generated samples due to the finite number of samples. This expected value is indicated in the results Tables as "Baseline: average perfect model".

A common type of failed prediction we observed is failure to 'split' a trajectory in different modes and instead always predicting a trajectory that is close to the mean of all possible solutions. For that reason, we also report the 'Unclassified Fraction' (UF), which is the fraction of solutions that is considered "unclassified" because it is closer to the mean of the possible solutions than to any of the valid solutions.

The ideal metric would naturally be the Wasserstein distance between the predicted distribution of trajectories and the ground truth distribution, but this is computationally too expensive because of the high dimensionality of the problem. However, we consider the combination of MSE-tC, MJSD and UF to be a good surrogate metric.

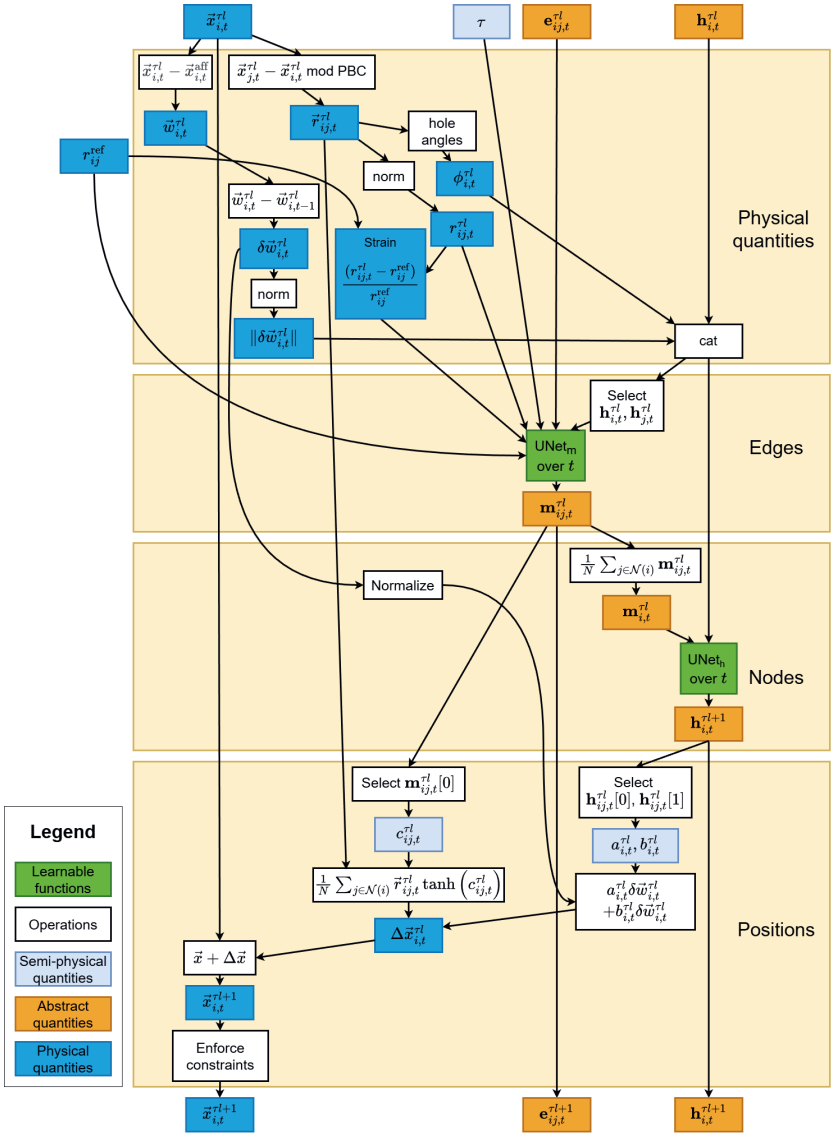


Figure 5.8: Schematic overview of one message-passing step.

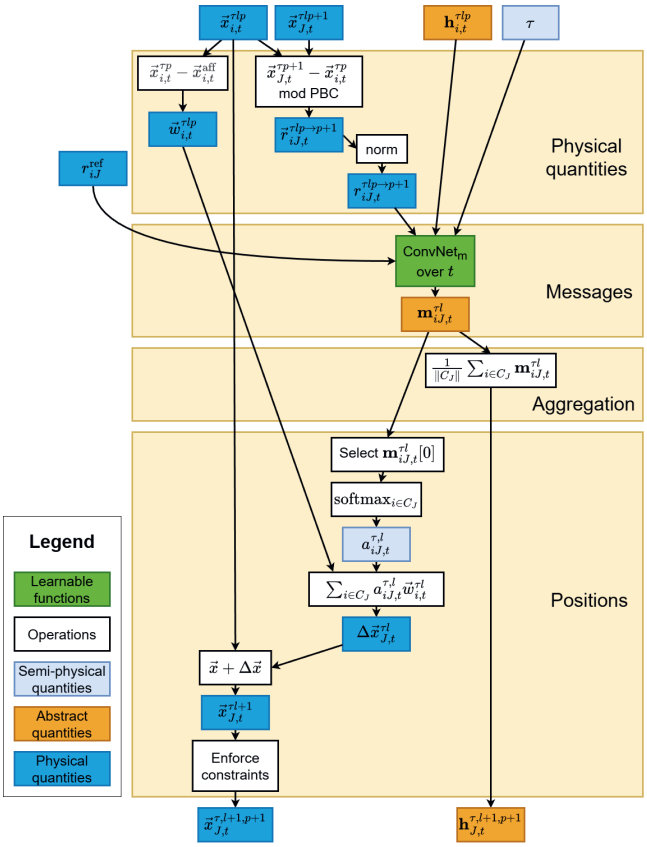


Figure 5.9: Schematic overview of a pooling step.

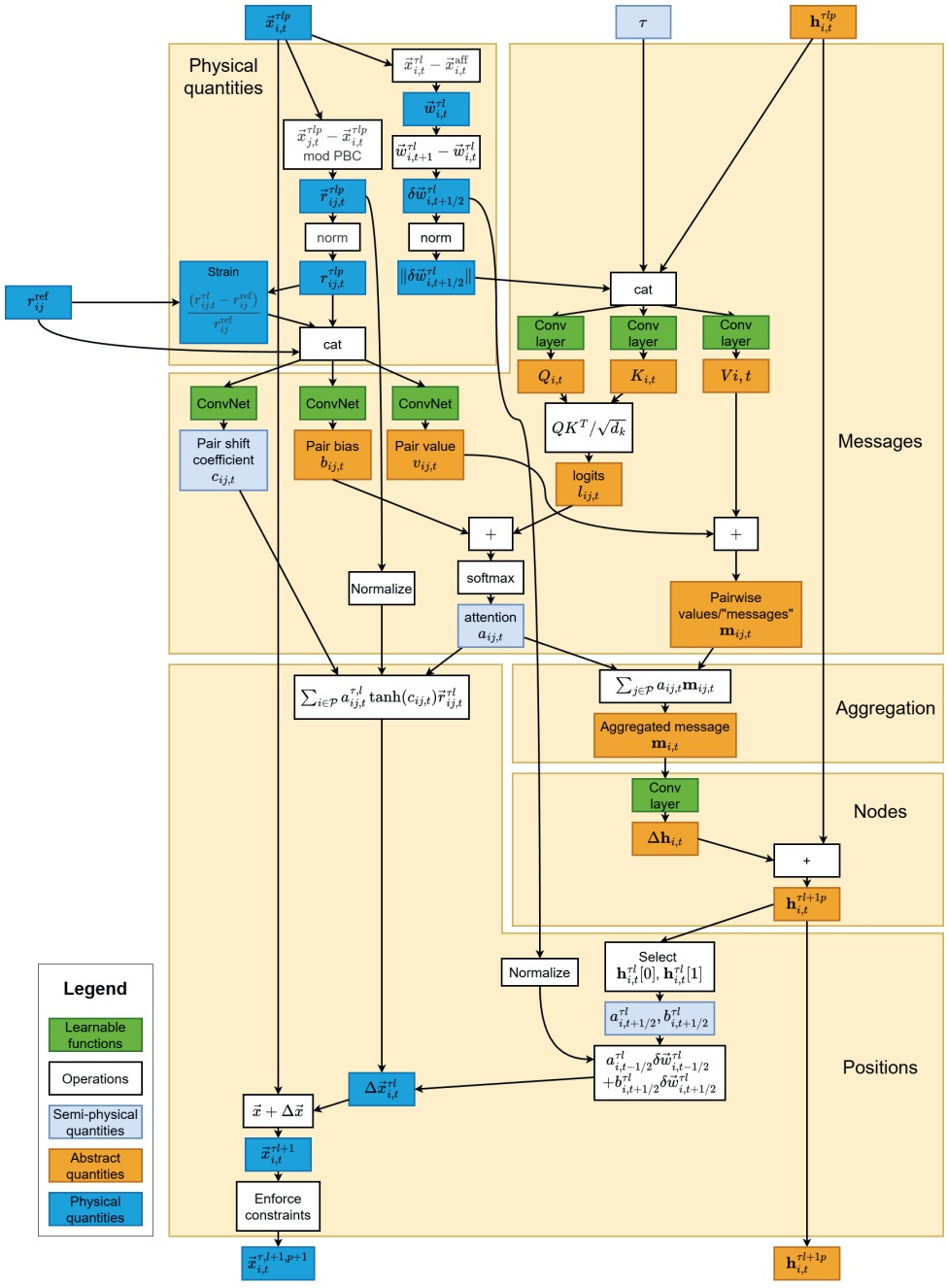


Figure 5.10: Schematic overview of an attention step.

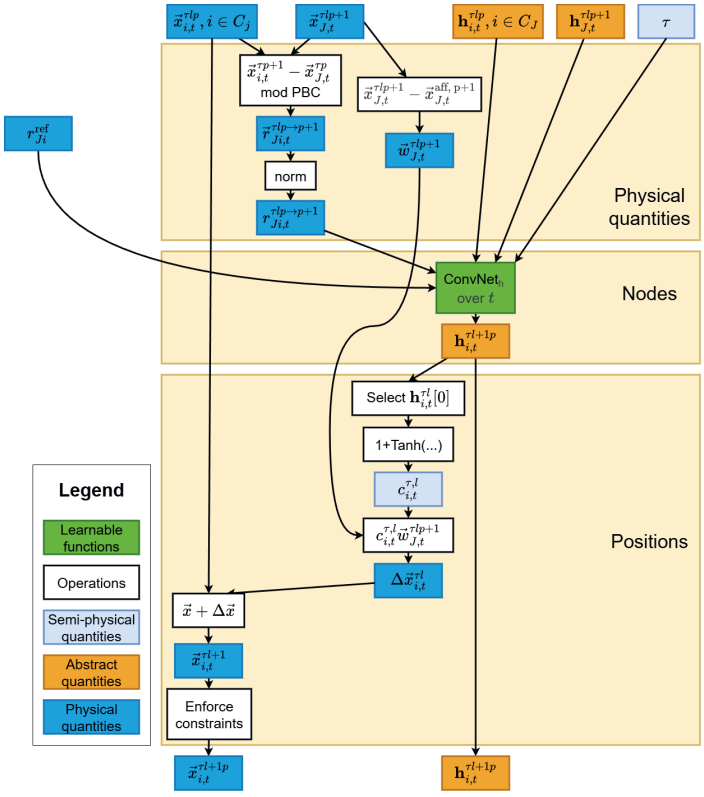


Figure 5.11: Schematic overview of an unpooling step.

5.5 Results

5.5.1 Static Model: Distribution over Buckling Modes

We evaluate the static flow matching GNN on the stacked-holes dataset. See Table 5.2 for results of the static GNN, EGNN and SimEGNN, trained with no coupling (always select the first ground truth solution to train on), random coupling (select a random solution to train on) or symmetric coupling (select the closest solution to train on). Clearly, the GNN is the best performing model[8]. The higher the number of possible solutions A, the more all the models struggle. A possible explanation for this is the uneven representation of the different solution multiplicities in the training data. Table 5.2 also shows that the more symmetries are present in the model, the less the model benefits from symmetric coupling; the GNN benefits the most, the EGNN benefits a little, while the SimEGNN does not seem to benefit at all. For both the GNN and the EGNN, the benefit is mostly visible for higher number of solutions A, which is expected since the different types of coupling are identical when there is only one possible solution. Noticeably, for all models there is no problem with mode collapse (i.e., the MJSD is always close to that of a theoretically perfect model), even for models trained on a single solution per data point. This confirms the built-in symmetry of the models.

Figure 5.12 shows some examples of predictions by the EGNN trained with symmetric coupling. The model is able to capture the deformations adequately, and is able to find all possible solutions. However, it increasingly struggles as the number of solutions increases.

Table 5.2: Results for the static flow matching GNN on the stacked-holes dataset, separated by the number of ground truth solutions A. 16 integration steps, sets of 8 predictions.

		$A=1$	$A=2$			$A=4$		
	Coupling	MSE-tC ($\times10^{-5}$)	MSE-tC ($\times10^{-3}$)	MJSD ($\times10^{-2}$)	UF ($\times10^{-1}$)	MSE-tC ($\times10^{-3}$)	MJSD ($\times10^{-2}$)	UF ($\times10^{-1}$)
GNN	None	**3.01**	5.78	3.44	2.89	6.61	7.90	2.15
	Random	3.90	2.19	1.87	2.10	3.50	7.59	1.68
	Symmetric	3.35	**0.556**	2.00	**0.744**	**1.82**	7.01	**0.489**
EGNN	None	7.68	2.46	3.30	3.15	5.00	6.96	2.85
	Random	5.57	2.91	2.03	3.73	5.19	6.36	3.23
	Symmetric	6.76	1.70	2.34	2.15	3.77	5.69	1.66
SimEGNN	None	13.3	3.31	2.83	2.90	6.55	5.68	3.40
	Random	12.9	3.62	1.81	3.50	7.46	5.32	3.61
	Symmetric	10.8	4.06	1.89	3.87	7.30	6.96	3.83
MJSD Worst case: total mode collapse				21.6			38.0	
MJSD Baseline: average perfect model				1.84			6.22	

[8]The static GNN outperformed the EGNN in the comparison above. However, this behavior did not carry over to initial experiments with time-dependent flow matching on the full dataset, where the GNN performs poorly. For this reason, we used the EGNN as the backbone for the hierarchical model developed here. More work is needed to understand why the relative performance of the GNN and EGNN differs between the static and time-dependent settings.

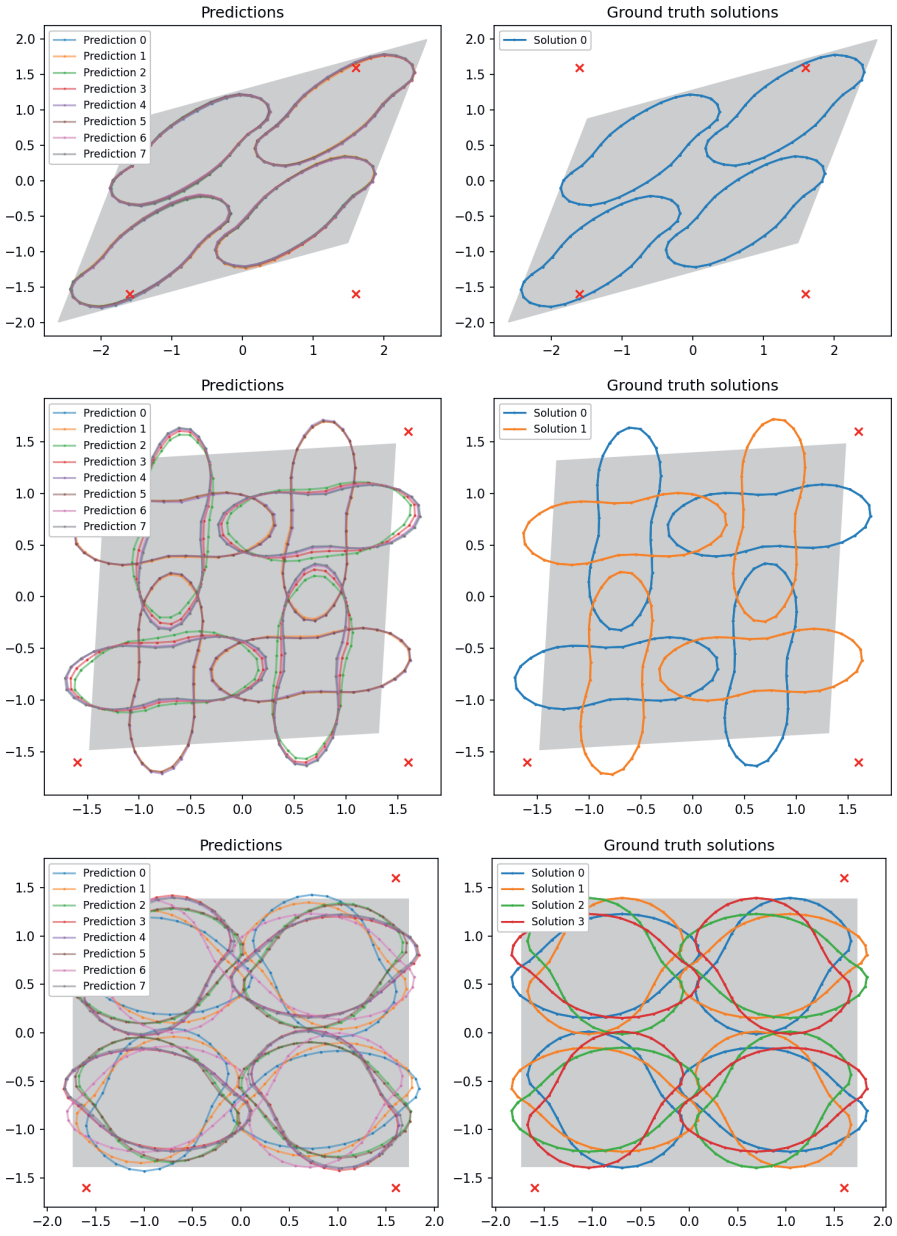


Figure 5.12: Deformations predicted by the EGNN trained with symmetric coupling. The left figures show 8 different predictions by the model (16 integration steps), the right figure shows the possible ground truths. Top row: 1 possible solution. Middle row: 2 possible solutions. Bottom row: 4 possible solutions. The gray parallelogram indicates affine RVE deformation due to $\mathbf{F}$. The red crosses indicate positions of the corners in the reference configuration.

Table 5.3: Results for the single-trajectory overfit experiment with a hierarchical model, comparing with and without symmetric coupling and 50/50 with/without. 16 flow matching integration steps, sets of 64 predictions.

Setting	MSE-tC	MJSD	UF
Reference model			
Full model	1.34×10^{-6}	1.01×10^{-2}	**0**
Architecture ablation			
Non-hierarchical	1.08×10^{-3}	1.96×10^{-3}	7.81×10^{-2}
Feature ablation			
No $\delta\vec{w}_{i,t}$	6.19×10^{-5}	1.01×10^{-2}	**0**
No angle $\phi_{i,t}$	$\mathbf{1.29 \times 10^{-6}}$	1.26×10^{-2}	**0**
Coupling ablation			
Random coupling	7.07×10^{-6}	1.10×10^{-3}	**0**
Hybrid coupling (50%/50%)	1.38×10^{-6}	1.22×10^{-4}	**0**
MJSD Worst case: total mode collapse	–	2.2×10^{-1}	–
MJSD Baseline: average perfect model	–	1.89×10^{-3}	–

5.5.2 Single-Trajectory Overfit: Time-Dependent Models

To verify that the time-dependent models are able to memorize a trajectory before evaluating generalization, we train both the non-hierarchical and hierarchical models on a single trajectory from the wallpaper dataset.

This works near perfectly. See Figure 5.13 for several predictions of the hierarchical model compared to both ground truths (from FEM), and Table 5.3 for quantitative results, comparing the influence of 1) removing the hierarchy as explained in Subsection 5.3.8, 2) training with symmetric coupling versus random coupling versus hybrid coupling (50% random, 50% symmetric coupling) as explained in Subsection 5.3.4 and 3) removing the microfluctuation increment $\delta\vec{w}_{i,t}$ or angle $\phi_{i,t}$ features, as explained in Subsection 5.3.5 and 5.3.10, respectively.

It shows that none of the models suffer from mode collapse, i.e., all models are able to capture both modes. Furthermore, it shows that the hierarchical aspect, the symmetric coupling and the use of $\delta\vec{w}_{i,t}$ improve the performance, whereas the angle has no significant effect.

The non-hierarchical model still manages to make decent predictions most of the time, despite the fact that it uses 9 message-passing steps. However, it takes longer to get there, and during training shows big spikes in the validation MSE-tC. See Figure 5.14.

On further investigation, these spikes are caused by the fact that the model swings wildly between giving highly accurate predictions and highly inaccurate predictions, where the latter is caused by the nodes seemingly failing to coordinate; in other words, they do not agree on which of the two valid solutions they should predict, so a patchwork of both is predicted instead. See Figure 5.15 for an example. This behavior

empirically confirms the presence of a global coordination problem: the model must ensure consistent mode selection across all nodes. Non-hierarchical models, even when theoretically deep enough, struggle with global communication due to oversmoothing and/or oversquashing, leading to spatially inconsistent predictions. This confirms the need for a hierarchical model to improve node coordination.

Additionally, we show the influence of the number of flow matching integration steps with or without symmetric coupling in Figure 5.16. Clearly, all models need only few integration steps, and even with just 1 integration step, the models successfully bifurcate (UF = 0). Interestingly, the symmetric coupling seemingly converges to a better accuracy. This is somewhat counterintuitive, because the main goal of symmetric coupling is to straighten the flow paths, which would reduce the number of integration steps needed, but not necessarily improve the final prediction in the limit of infinite integration steps. The figure also shows that the MSJD is not sensitive to the number of integration steps, because symmetry preservation is enforced by construction.

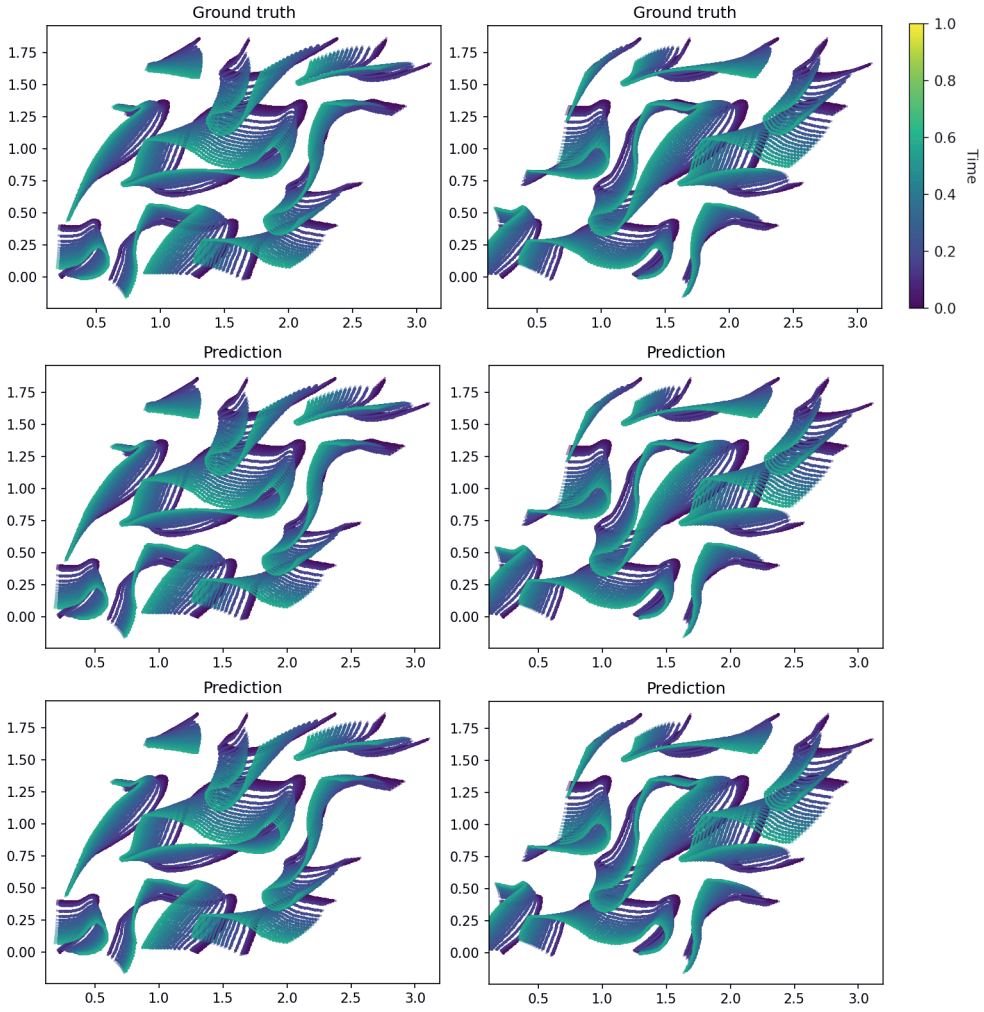


Figure 5.13: Top row: the two possible solutions for the selected trajectory. Middle and bottom rows: four different predictions from the hierarchical model overfitted on this trajectory (16 integration steps), showing that it can capture both modes with near-perfect accuracy. The figures show all the loading time steps in the trajectory, with color indicating the loading time t. The trajectories end before $t = 1$ because the ground truth trajectory is cut at contact.

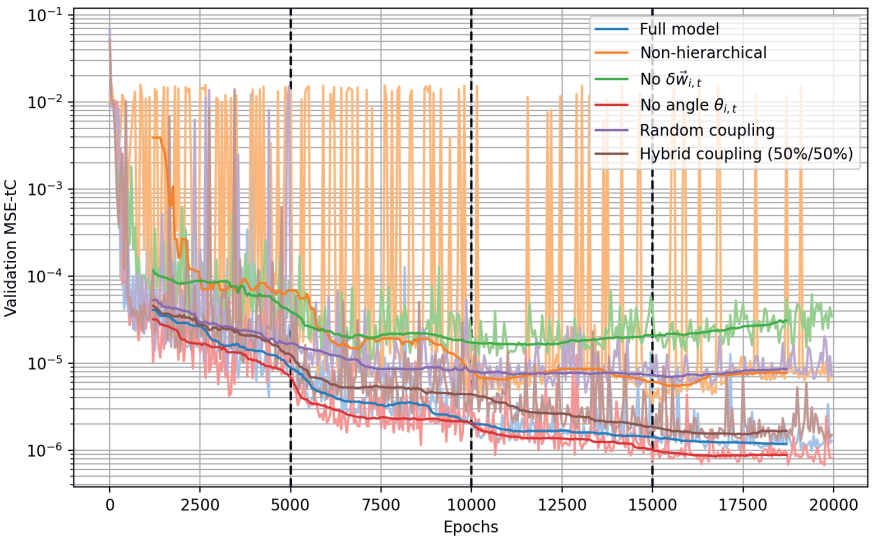


Figure 5.14: MSE-tC during training for the single-trajectory overfit experiment, computed every 50 epochs from 1 sample. 8 flow matching integration steps. Pastel colors show the entire curve, saturated colors show the running median over 2500 epochs (i.e., 50 samples). Black dashed vertical lines indicate changes in the learning rate. The tracking of validation MSE-tC was inconsistent across runs with respect to the invalid time steps, hence the figure cannot be used to compare among models quantitatively.

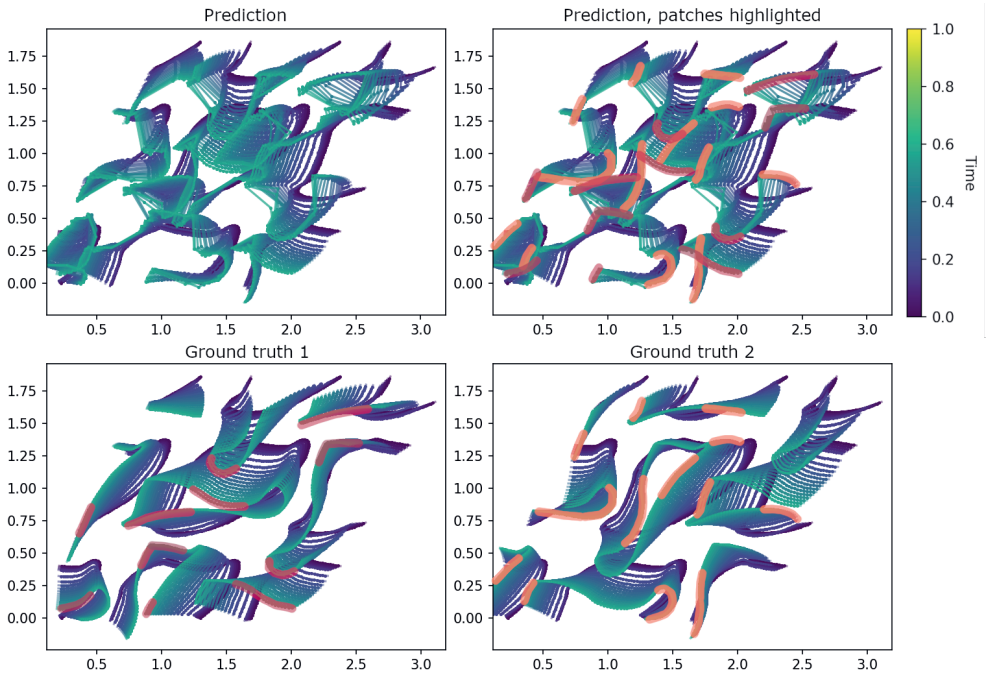


Figure 5.15: Top left: an inaccurate prediction from the non-hierarchical model (16 flow matching integration steps), showing that the nodes fail to agree on which of the two valid solutions to predict. The rest of the figures show which patches of the prediction correspond to which ground truth trajectory. Each figure shows all the loading time steps in the trajectory, with color indicating the loading time t. The trajectory is masked for some timesteps before $t = 1$ because the ground truth trajectory is cut at contact.

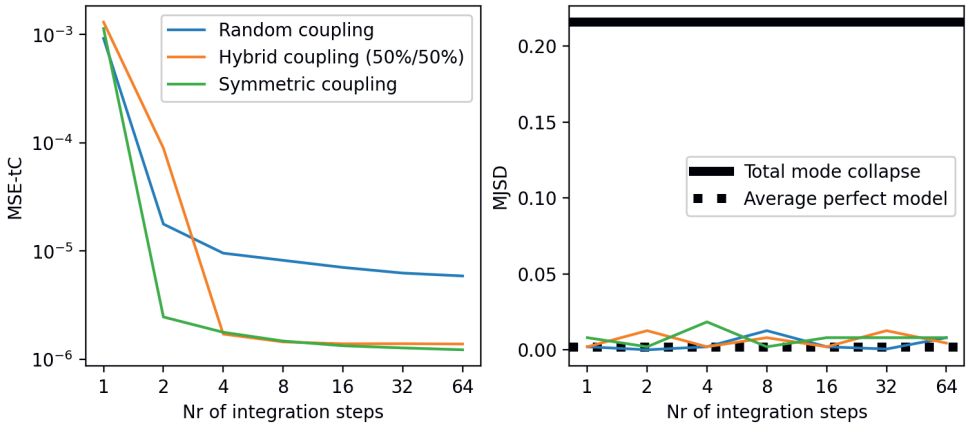


Figure 5.16: Results for the single-trajectory overfit experiment with a hierarchical model, showing the influence of the number of flow matching integration steps on the performance, comparing with and without and 50/50 symmetric coupling (sets of 32 predictions). All models for all numbers of integration steps have a UF = 0.

5.5.3 Full Wallpaper Dataset: Comparison of Time-Dependent Models

We train and evaluate both time-dependent models on the test data of the full wallpaper-group dataset.

The results are shown in Table 5.4. Additionally, we show some predictions by the hierarchical model on the test data in Figures 5.17 to 5.20, where the first subfigure ('Reference configuration + $\mathbf{F}_{\text{end}}$') shows the reference configuration (solid black outline) and how it is deformed by $\mathbf{F}_t$ at the last valid time step (orange arrows deforming it to the gray shape with the dashed black outline). The second figure ('Prediction') shows the predicted trajectory, the third figure ('Closest ground truth') shows the ground truth solution that is closest to the predicted. The title indicates $\mathbf{F}_t$ at the last valid time step. The 'Prediction' and 'Ground truth' figures show all the loading time steps in the trajectory, where the color indicates the loading time t. Some trajectories end before $t = 1$ because the ground truth trajectory is cut at contact. They were created using 32 flow matching integration steps.

Additionally, for some trajectories we create 16 predictions to investigate the spread over possible solutions. The last valid time step of each of these trajectories is shown in Figures 5.21 to 5.25, where the predictions are colored according to which ground truth solution is closest to it, with black indicating the mean of all solutions is closest.

The following observations can be made:

- The performance of the non-hierarchical model is actually slightly better for $A = 1$, where global coordination is not as important, and on par with the hierarchical model for $A = 2$ and $A = 4$ in terms of MSE-tC, but it is significantly worse in terms of UF.
- The hierarchical model is already converged with only 8 flow matching integration steps, as increasing the number of steps to 16 does not improve overall performance. This indicates the symmetric coupling is already effective at straightening the flow paths.
- Both models show good performance on trajectories that do not buckle, as evidenced by the low MSE-tC for $A = 1$ in Table 5.4. See also Figure 5.17 for an example from the hierarchical model.
- The performance on trajectories that do buckle is less good, as evidenced by the higher MSE-tC for $A = 2$ and $A = 4$ in Table 5.4, and Figures 5.18 to 5.25. This is probably partly because predicting a multimodal trajectory is simply more difficult than a unimodal trajectory due to the more complicated flow field needed, but additionally the fact that these trajectories (especially $A = 4$) are underrepresented in the dataset probably plays a role. Further oversampling of these trajectories might help.
- The most common failure mode is failure to 'split' a trajectory in different modes and instead always predicting something close to the mean of all possible solutions, as evidenced by the high value of the UF in Table 5.4. Especially Figure 5.24 reveals a lot of such examples.
- Figures 5.18 to 5.20 show that the hierarchical model *is* capable of predicting buckling; in each example there is a very clear, abrupt symmetry breaking

happening.

- If there are a lot of buckling ligaments, the hierarchical model has trouble coordinating all of them, as evidenced by Figure 5.22. Since these ligaments would most likely not have a strong interaction between them, it is likely there are multiple possible solutions that are energetically close to each other, which might be making this a particularly hard case.
- Another possible model failure is predicting the wrong buckling mode, as shown in Figure 5.23, where the model sometimes predicts a different shear mode (e.g., holes in the same column will deform the same), highlighted in yellow, even though in the ground truth the shear mode is along the upper-left to lower-right diagonal. We double-checked, and it turns out these buckling modes are also possible, although for slightly different $\mathbf{F}$. These predictions are therefore close to an admissible response rather than entirely incorrect.
- There are no problems with mode collapse, as evidenced by the low value of the MJSD in Table 5.4 and shown in Figures 5.21 to 5.25, where the predictions are not necessarily all *good*, but they do not show a bias towards one of the modes. This is expected because the equivariance of the model enforces symmetry of the probability distribution.
- The hierarchical model appears to have learned that ligaments changing width is undesirable, as shown in Figure 5.22, where the predictions may be inaccurate, but the ligament widths look very accurate. This is mostly likely aided by the graph representation, whose rays connect opposite hole boundaries across the ligaments, which we hypothesize biases the model to prioritize keeping the ligament width reasonable, which encourages opposite sides of ligaments to move together.

Table 5.4: Results for the hierarchical time-dependent flow matching GNN on the full wallpaper test dataset, separated by number of ground truth solutions A. 8 or 16 flow matching integration steps, sets of 8 predictions.

Nr of solutions	$A=1$	$A=2$			$A=4$		
Nr of trajectories	1409	356			71		
	MSE-tC ($\times 10^{-4}$)	MSE-tC ($\times 10^{-3}$)	MJSD ($\times 10^{-2}$)	UF	MSE-tC ($\times 10^{-3}$)	MJSD ($\times 10^{-2}$)	UF
Non-hierarchical model, 8 steps	**3.06**	**1.62**	2.41	0.924	2.38	7.98	0.951
Full model, 8 steps	5.36	1.67	2.56	0.849	**2.35**	6.38	0.896
Full model, 16 steps	6.53	1.80	2.55	**0.812**	2.47	6.65	**0.894**
MJSD Worst case: total mode collapse			21.6			38.0	
MJSD Baseline: average perfect model			1.84			6.22	

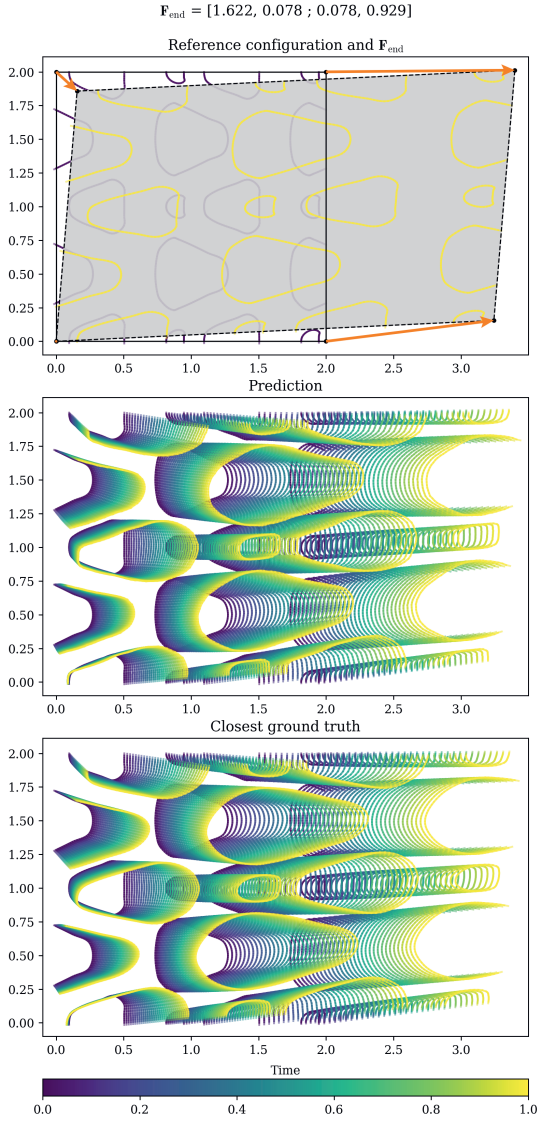


Figure 5.17: Non-buckling trajectory under predominantly horizontal stretching, with only one ground-truth solution ($A = 1$). Top: reference configuration (solid black outline) and affine deformation by $\mathbf{F}_t$ at the last valid time step (dashed black outline). Middle: predicted trajectory using 32 flow matching integration steps. Bottom: closest ground-truth trajectory. The title shows $\mathbf{F}_t$ at the last valid time step. Colors in the middle and bottom panels indicate loading time t. Trajectories may terminate before $t = 1$ because ground truth data are cut at contact. The prediction closely matches the ground truth.

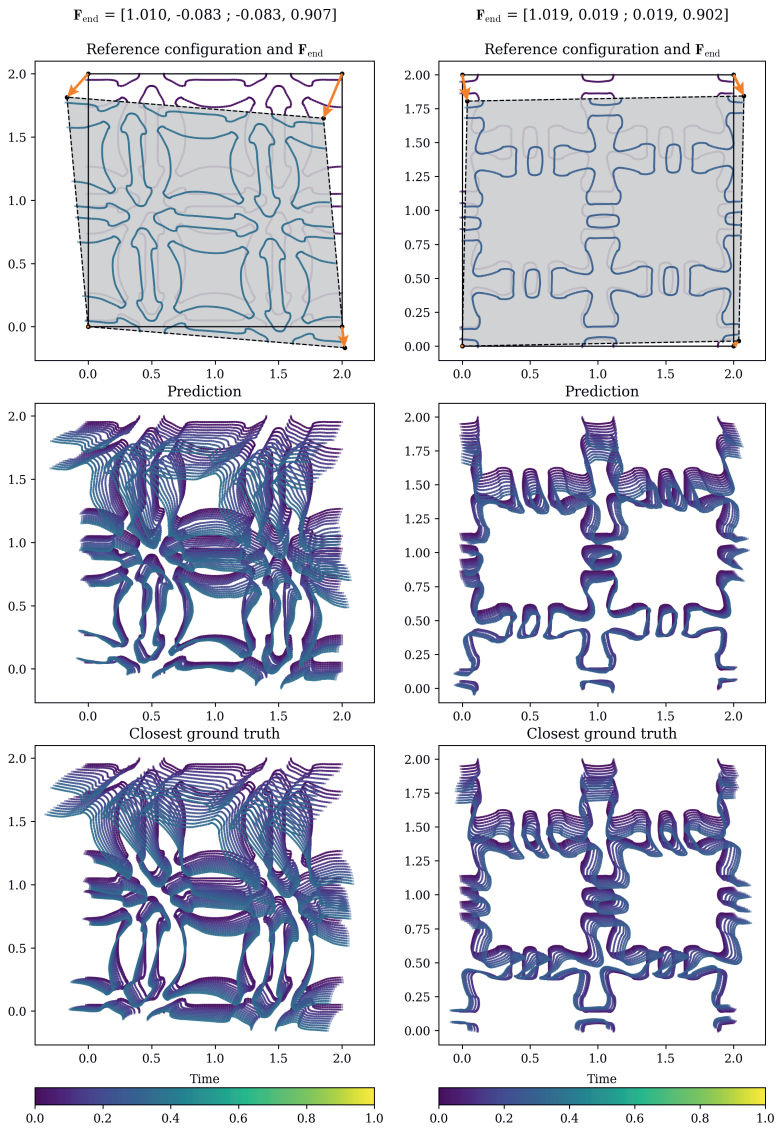


Figure 5.18: Buckling test trajectories with two admissible solutions ($A = 2$) under mainly uniaxial compression, with and without additional shear, for two microstructures with multiple reflection symmetries in the reference configuration. Top: reference configuration (solid black outline) and affine deformation by $\mathbf{F}_t$ at the last valid time step (dashed black outline). Middle: predicted trajectory using 32 flow matching integration steps. Bottom: closest ground-truth trajectory. The titles show $\mathbf{F}_t$ at the last valid time step. Colors in the middle and bottom panels indicate loading time t. Trajectories may terminate before $t = 1$ because ground truth data are cut at contact. The generated predictions buckle in a pattern similar to one of the ground truth solutions.

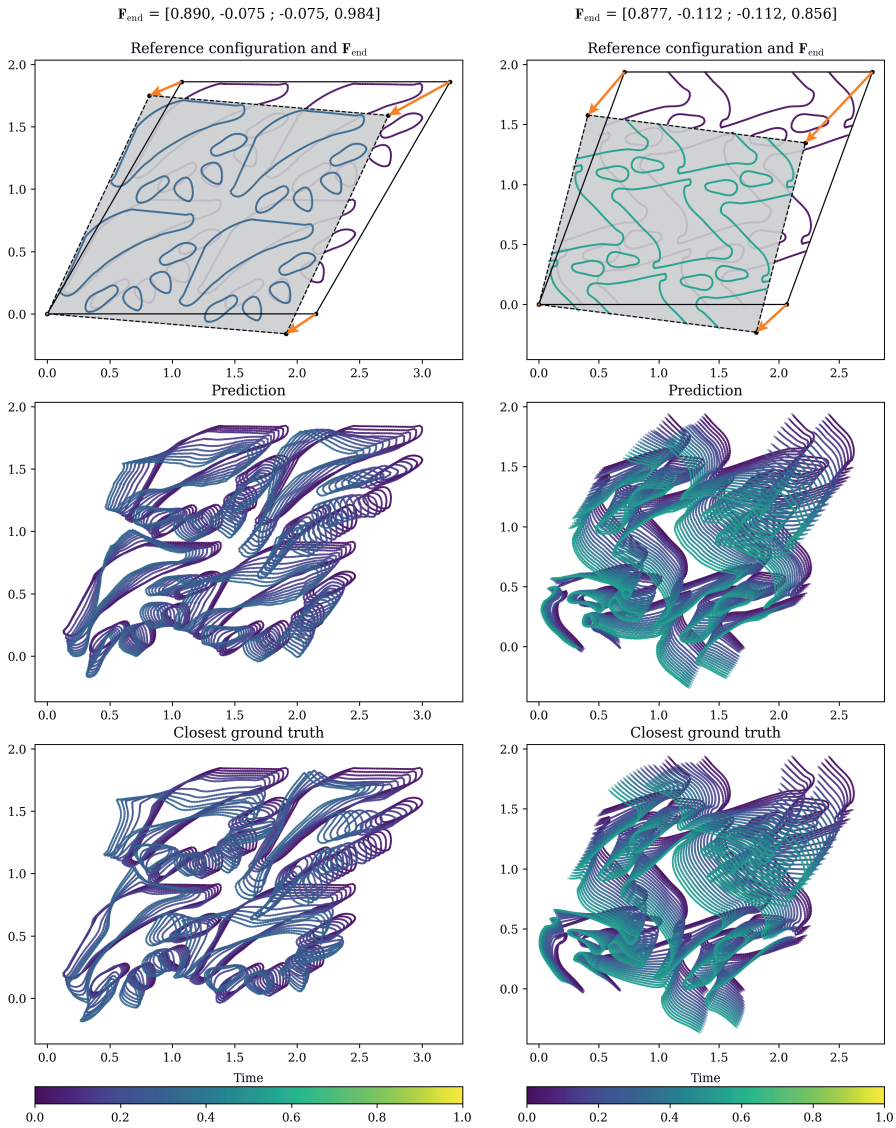


Figure 5.19: Buckling test trajectories with two admissible solutions ($A = 2$) under compressive loading with negative shear, for two microstructures without any reflection symmetry. Top: reference configuration (solid black outline) and affine deformation by $\mathbf{F}_t$ at the last valid time step (dashed black outline). Middle: predicted trajectory using 32 flow matching integration steps. Bottom: closest ground-truth trajectory. The titles show $\mathbf{F}_t$ at the last valid time step. Colors in the middle and bottom panels indicate loading time t. Trajectories may terminate before $t = 1$ because ground truth data are cut at contact. The generated predictions buckle in a pattern similar to one of the ground truth solutions.

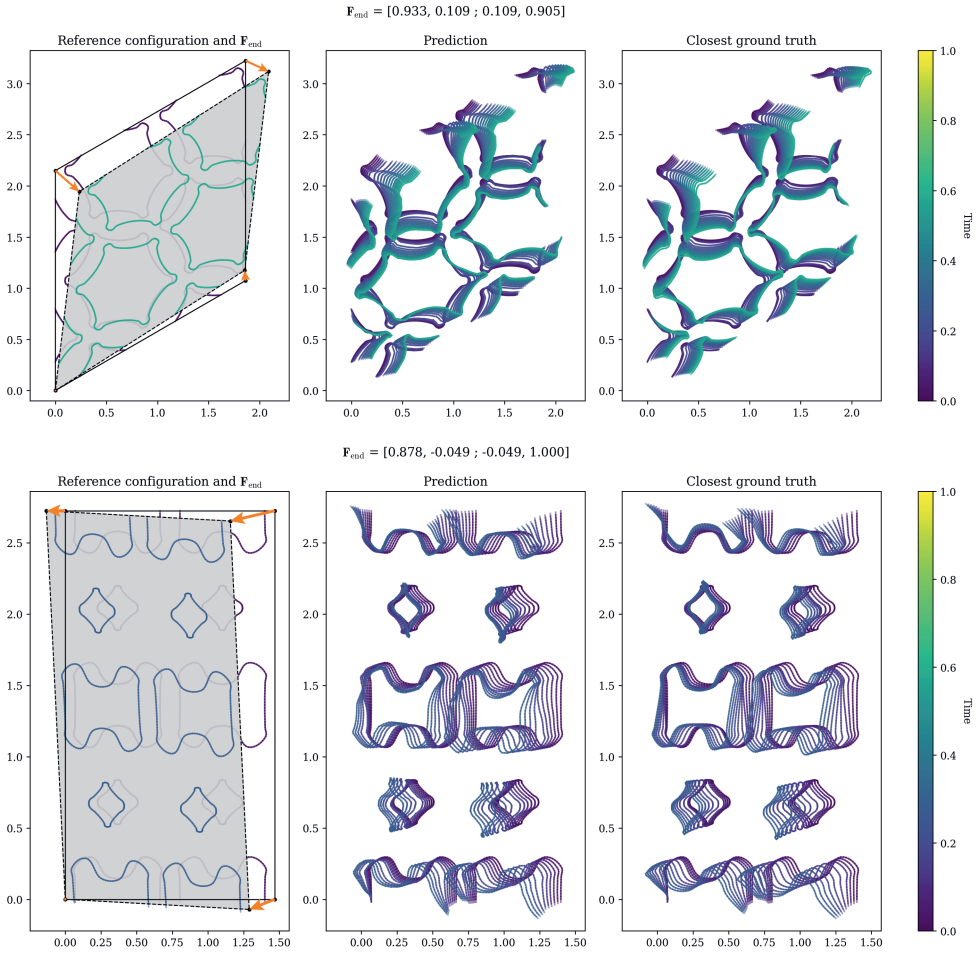


Figure 5.20: Buckling test trajectories with two admissible solutions ($A = 2$) under shear-compression loading. Left: reference configuration (solid black outline) and affine deformation by $\mathbf{F}_t$ at the last valid time step (dashed black outline). Middle: predicted trajectory using 32 flow matching integration steps. Right: closest ground-truth trajectory. The titles show $\mathbf{F}_t$ at the last valid time step. Colors in the middle and bottom panels indicate loading time t. Trajectories may terminate before $t = 1$ because ground truth data are cut at contact. The generated predictions buckle in a pattern similar to one of the ground truth solutions.

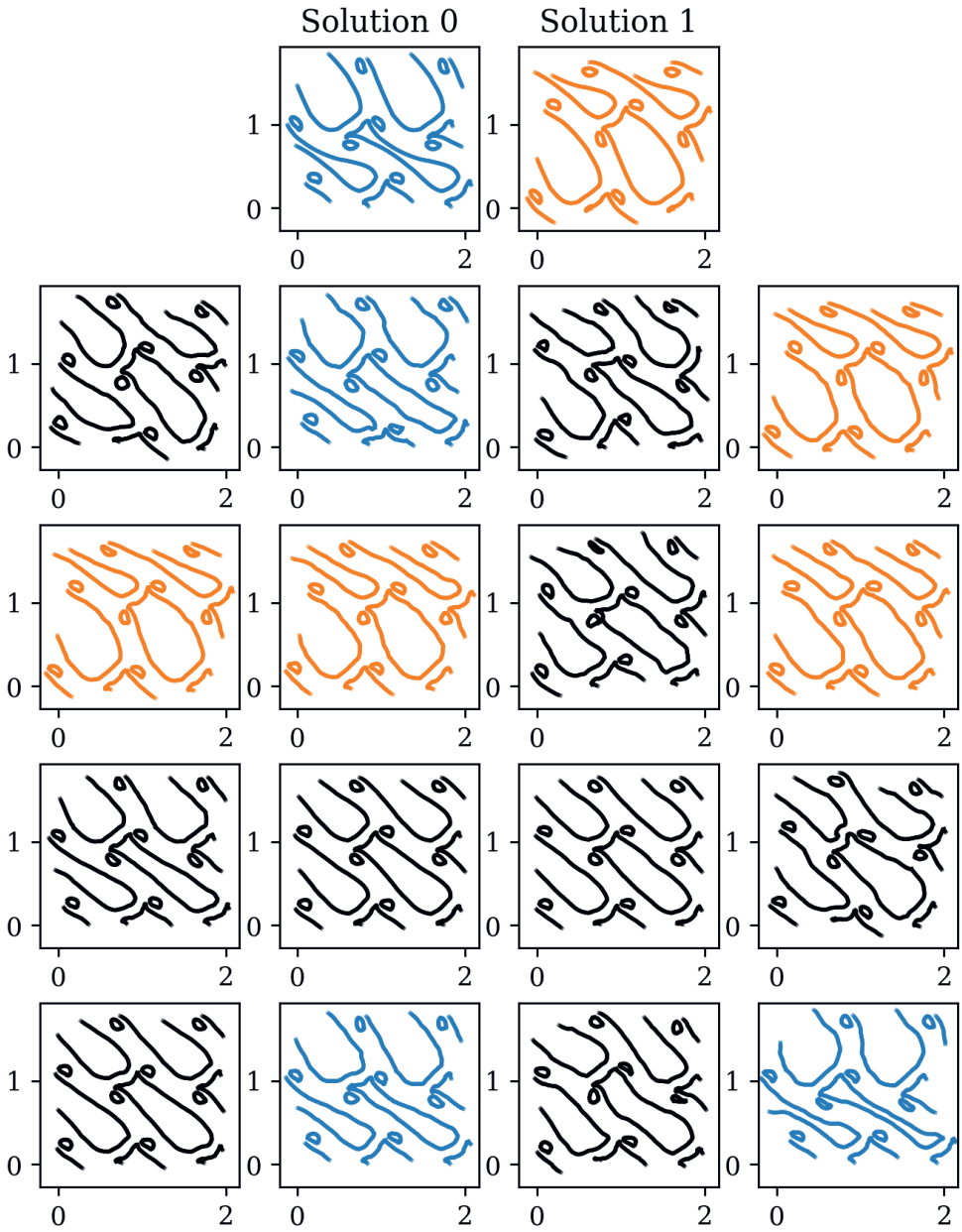


Figure 5.21: Buckling case with two ground truth solutions ($A = 2$), with long diagonal ligaments. Top row: the two ground truths. Other rows: 16 predictions for the same trajectory, showing the spread over possible solutions. Only the last valid time step is shown. Each prediction is colored according to the closest ground-truth solution; black indicates that the mean of all solutions is closest. The figure illustrates that the model can predict both modes, but does not always consistently do so, and many times predicts a solution closer to the average, unbuckled configuration.

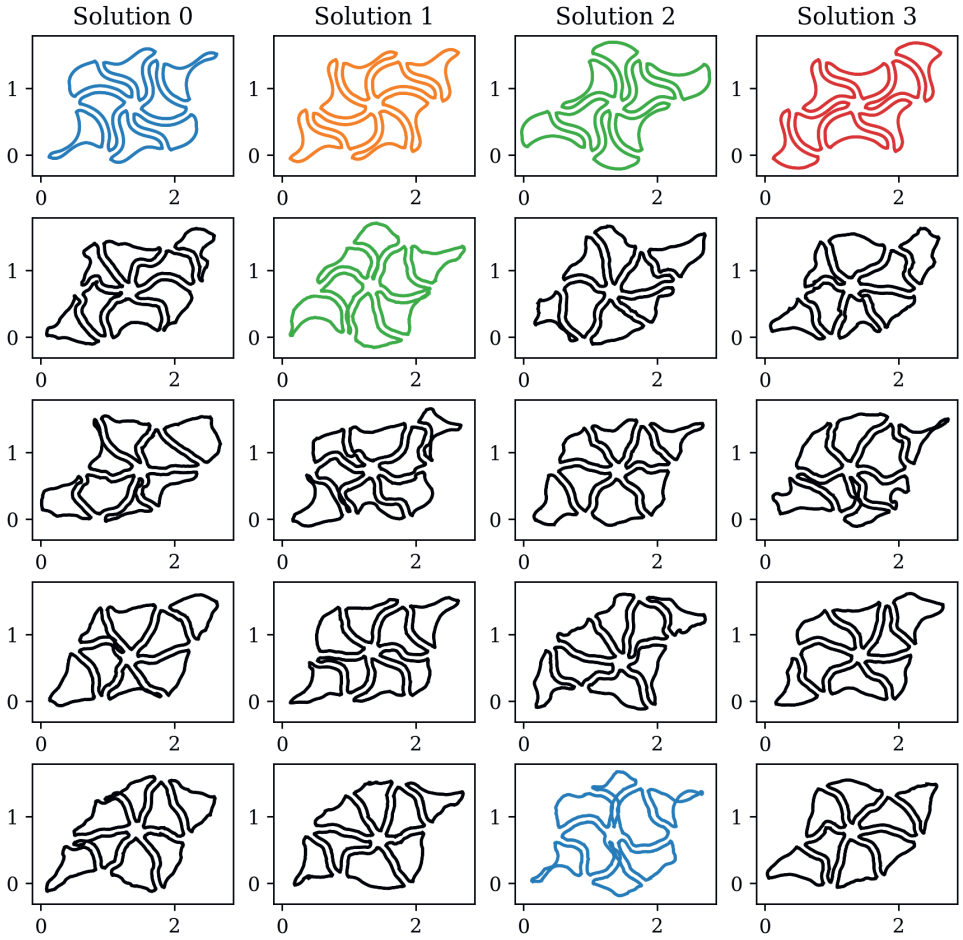


Figure 5.22: Buckling case with four ground truth solutions ($A = 4$), for a highly symmetric microstructure with many thin ligaments. Top row: the four ground truths. Other rows: 16 predictions for the same trajectory, showing the spread over possible solutions. Only the last valid time step is shown. Each prediction is colored according to the closest ground-truth solution; black indicates that the mean of all solutions is closest. The figure illustrates that the model struggles with coordinating many thin ligaments.

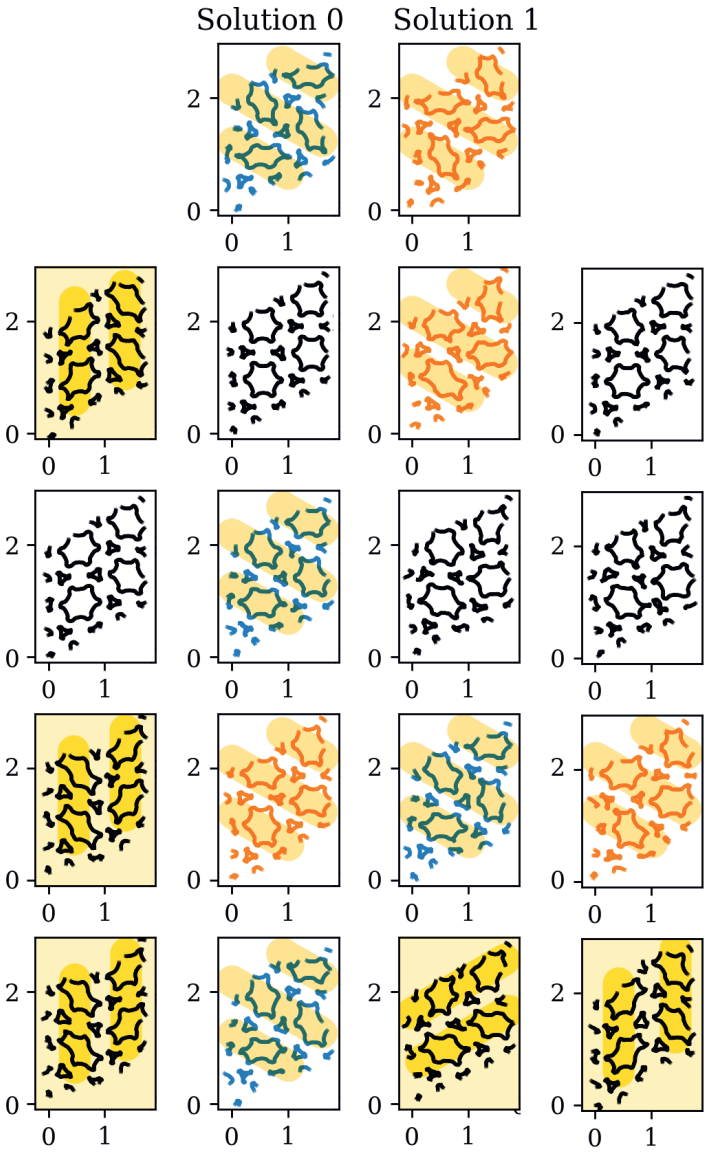


Figure 5.23: Buckling case with two ground truth solutions ($A = 2$). Top row: the two ground truths. Other rows: 16 predictions for the same trajectory, showing the spread over possible solutions. Only the last valid time step is shown. Each prediction is colored according to the closest ground-truth solution; black indicates that the mean of all solutions is closest. Highlighted in yellow are solutions that are technically incorrect but still show a mode that is close to being valid. The yellow stripes show how holes in the same column/row deform similarly.

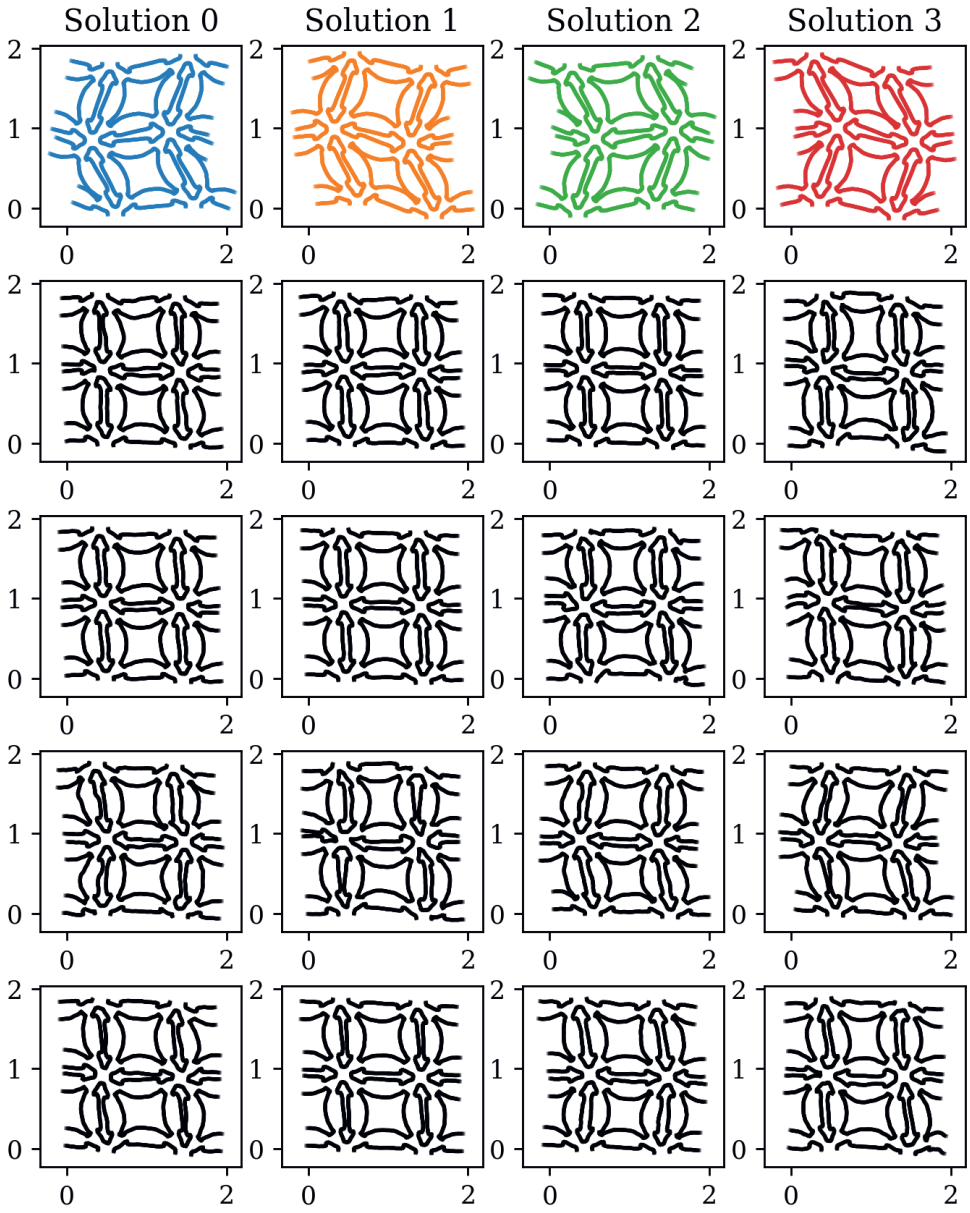


Figure 5.24: Buckling case with four ground truth solutions ($A = 4$), resulting in both horizontal and vertical zigzag patterns. Top row: the four ground truths. Other rows: 16 predictions for the same trajectory, showing the spread over possible solutions. Only the last valid time step is shown. Each prediction is colored according to the closest ground-truth solution; black indicates that the mean of all solutions is closest. The figure illustrates that the model predicts a solution closer to the average, unbuckled configuration.

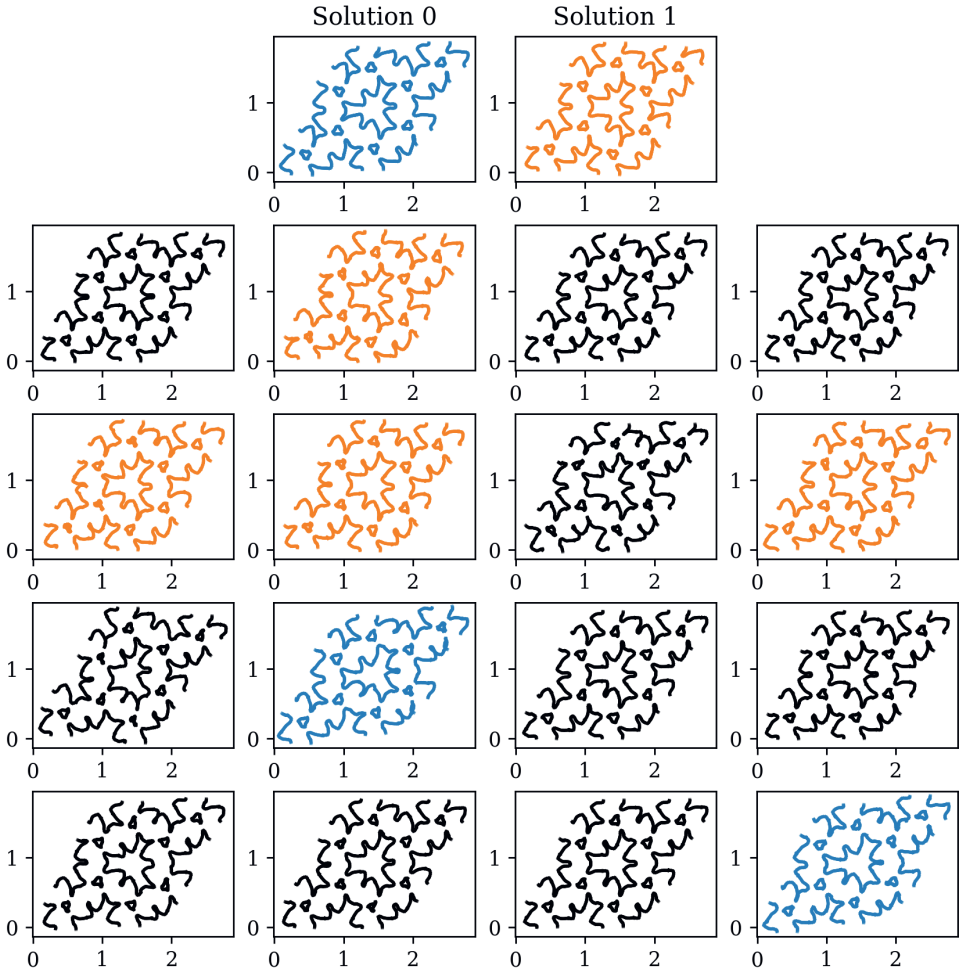


Figure 5.25: Buckling case with two ground truth solutions ($A = 2$) for a complicated, irregular microstructure. Top row: the two ground truths. Other rows: 16 predictions for the same trajectory, showing the spread over possible solutions. Only the last valid time step is shown. Each prediction is colored according to the closest ground-truth solution; black indicates that the mean of all solutions is closest. The figure illustrates that the model can predict both modes, but does not always consistently do so, and many times predicts a solution closer to the average, unbuckled configuration.

5.6 Conclusion

We presented a probabilistic framework for predicting buckling in mechanical metamaterials. This approach enables the capture of buckling through a probability distribution over possible trajectories. This is essential for symmetry-breaking systems, where a fixed microstructure and loading condition can admit multiple physically valid buckling modes. In contrast to deterministic models, the proposed model can generate all symmetry-related buckling configurations with equal probability, in line with the underlying physics.

Our approach combines equivariant graph neural networks with flow matching and time bundling that ensures coherence over time. In addition, we adapted symmetric coupling to the metamaterial buckling context and introduced a hierarchical architecture with all-pairs periodic attention at the coarse level, enabling long-range spatial dependencies beyond local message passing. We showed that this global communication is essential in coordinating all nodes into a coherent buckling mode. Together, these components provide a structured way to model both the multiplicity and the global coherence of buckling behavior.

The three model variants (static, non-hierarchical time-dependent, and hierarchical time-dependent) address increasing levels of complexity: from generating distributions over static buckling modes, to predicting time-dependent loading trajectories, to modeling geometrically diverse metamaterials with hierarchical long-range interactions. The results show that the proposed framework is able to capture symmetry-related buckling modes, generate multiple plausible outcomes for a single input, and preserve temporal consistency along deformation trajectories.

5.6.1 Main achievements and contributions

The main achievements and contributions of this work can be summarized as follows:

- We treated metamaterial buckling as a probabilistic learning problem rather than a deterministic regression problem. This makes it possible to represent multiple admissible post-buckling responses for the same geometry and loading condition.
- We combined equivariant graph neural networks with flow matching for symmetry-breaking buckling problems. In this formulation, the learned probability distribution respects the symmetries of the underlying physical system, while individual samples may still break those symmetries.
- We adapted symmetric coupling to the metamaterial buckling setting. This allows physically equivalent symmetry-related buckling modes to be handled consistently during training, instead of penalizing predictions solely because they select a different but equivalent mode. We show that the resulting model needs only a small number of 8 flow matching integration steps during inference.
- We incorporated time bundling into the time-dependent flow-matching model, so that complete deformation trajectories are generated coherently rather than as independent predictions at separate loading steps.
- We designed an equivariant, time-dependent hierarchical architecture for graph-structured metamaterials by combining local message passing with coarse-level

all-pairs periodic attention. While hierarchical graph architectures are not new in general, this combination is important here because buckling requires global coordination across the full unit cell.
- We demonstrated the approach on both a controlled stacked-holes benchmark and the more diverse wallpaper-group dataset. The stacked-holes results show that the method can recover competing buckling modes and generate symmetry-related solutions with the correct qualitative structure.
- We showed that global coordination is a central architectural requirement for probabilistic buckling prediction. The comparison between non-hierarchical and hierarchical variants demonstrates that local message passing alone is insufficient for reliably organizing a full structure into a coherent post-buckling mode.
- We identified both the promise and the current limitations of equivariant generative modeling for buckling metamaterials. The results show clear successes on controlled examples, while the wallpaper dataset highlights remaining challenges in accuracy, data coverage, and computational efficiency.

These achievements show that equivariant generative modeling is a promising direction for learning the response of buckling metamaterials. At the same time, the results also clarify where the current framework may be further approved, particularly when applied to the full diversity of the wallpaper dataset.

5.6.2 Limitations

The goal of this current work was to investigate whether probabilistic, equivariant flow matching can be used for high-dimensional buckling trajectories in graph-structured metamaterials. To the best of our knowledge, there is currently no established state-of-the-art model that can simultaneously handle arbitrary graph-based metamaterial geometries, symmetry-related multimodal buckling responses, full loading trajectories, and global coordination of post-buckling modes. From this perspective, the present model should be viewed as a proof of concept that demonstrates the feasibility of this approach, while also making clear which parts still require improvement.

The main limitations of the current model are its accuracy on the most challenging wallpaper dataset and its computational cost. On the wallpaper dataset, the model does not always predict the correct buckling response and can still produce outputs that remain too close to the average of possible solutions. Given that the performance on the stacked-holes benchmark *is* quite good and the probabilistic and hierarchical components substantially improve the modeling of symmetry-breaking responses, this indicates that most likely the problem is not with the probabilistic approach using flow matching, but that the model still struggles with the geometric diversity of this dataset. It is possible that this is due to limitations of the dataset, or the geometric expressiveness of the network.

Inference is considerably faster than the original finite-element simulations, but the comparison should be interpreted carefully. The original FEM model took on average 112 seconds per trajectory; our model takes on average 0.912 seconds per trajectory when using 8 flow matching integration steps. This is a substantial speed-up per trajectory. However, the original FEM model is not yet GPU-enabled and was run on a laptop, whereas our model runs on an H100 GPU, processes trajectories in

batches and removes the bulk nodes. In addition, training the model takes about 4.5 days, which means that the practical benefit depends on the number of trajectories generated after training. For small numbers of queries, the training cost may outweigh the inference-time savings; for large-scale sampling, uncertainty quantification, or inverse design loops, the cost becomes easier to justify.

Finally, the current work is limited to two-dimensional geometries and a single hyperelastic material model. The EGNN framework itself is not restricted to two dimensions, but in three dimensions, creating a training dataset will be more difficult (e.g., 219 space groups instead of 17 wallpaper groups) and computationally expensive. Additionally, the admissible buckling patterns would become more complicated, twisting, bending, and localized deformation modes that are absent from the present planar setting. It would also need to handle larger graph structures. Contact is not included in the simulations used for training, even though many buckling trajectories in the wallpaper dataset approach contact shortly after buckling. This limits both the available training data and the range of post-buckling behavior that the model can learn.

5.6.3 Future Work

Future work can go in three directions: 1) improving the model in terms of accuracy and computational efficiency, 2) extending the model and its techniques to more complex settings, and 3) applying the model to real-world design problems.

Improving the model We see many possible ways of still improving the model, including but not limited to the following.

- **Addressing the Picasso problem:** A possible explanation for the limited accuracy on the wallpaper group dataset is related to the *Picasso problem* in equivariant architectures: it is not enough to recognize and predict the right local features; they must also be arranged correctly. In facial recognition, identifying individual facial features (eyes, nose, mouth) is not enough; the facial features need to be arranged in the right way too to actually look like a face. Similarly, in our setting, the orientation of holes relative to each other is crucial for the buckling behavior. Although our model is $E(n)$-equivariant and can in principle represent such orientations, the equivariant information it uses is restricted to edge vectors defined by the graph connectivity. These vectors are not freely constructed or recombined by the network, and can only be 'propagated' to other nodes through changes in node position. They therefore provide limited flexibility for encoding and communicating relational geometric information across the structure. As a result, the model may struggle to capture the correct global arrangement of features, potentially limiting accuracy [97]. More expressive equivariant architectures that learn and propagate richer equivariant features could mitigate this issue [58–61]. Another potential solution is to abandon (some) equivariance (but in various smart ways), which has also seen recent successes in complex high-dimensional physical systems [77, 90, 93, 101].
- **Dataset:** Another contributing factor may be data limitations. Even though the current wallpaper dataset is extensive with more than 12,000 trajectories,

the number of geometries (1020) is not actually that large, and moreover, a lot of trajectories do not actually buckle, and of the ones that do buckle, a lot run into contact soon after buckling and are therefore prematurely cut short. In the future, a dataset that oversamples loadcases with a lot of compression and includes contact might improve the performance of the model.

- **Graph representation:** The graph representation could also be optimized more, e.g., by investigating the value of the different edge types (some might be redundant), and improving the clustering or the number of levels in the hierarchy.
- **Similarity equivariance:** We also still believe that similarity equivariance [1] deserves a closer look, if only to see what exactly is limiting its performance in this setting.
- **Smarter pooling:** Another option is to pool loading time and nodes at the same time, i.e., the bottleneck of the graph U-Net is also the bottleneck of the temporal U-Net, and the representation is therefore reduced in both time *and* space.
- **Optimizing efficiency:** Regarding efficiency, there is room for improvement by using more GPU-friendly architectures (e.g., using FlashAttention [102], but with pairwise distances under periodic boundary conditions). Another direction is reducing the number of flow matching integration steps needed, e.g., by implementing Rectified Flow [103].

Extending the framework A natural next step is to extend the approach to more realistic settings, including three-dimensional geometries, more complex material models (e.g., viscoelasticity or plasticity), and multi-material or multi-physics systems (e.g., magnetic actuation).

More generally, the framework of equivariant flow matching with symmetric coupling could be applied to other types of symmetry-breaking bifurcations beyond mechanical buckling. Extending symmetric coupling to continuous symmetry groups is a particularly interesting direction, e.g., periodic time translations in oscillatory fluid flows or buckling in 3D.

Applying the model Finally, integrating the model into inverse design and topology optimization pipelines could enable the exploration of design spaces under uncertainty. The probabilistic nature of the predictions is particularly well-suited for robust design, where multiple possible outcomes must be considered.

Bibliography

[1] F. Hendriks, V. Menkovski, M. Doškář, M. G. Geers, O. Rokoš, Similarity equivariant graph neural networks for homogenization of metamaterials, Computer Methods in Applied Mechanics and Engineering 439 (2025) 117867.

[2] K. Bertoldi, V. Vitelli, J. Christensen, M. Van Hecke, Flexible mechanical metamaterials (oct 2017). doi:10.1038/natrevmats.2017.66.

[3] X. Ren, R. Das, P. Tran, T. D. Ngo, Y. M. Xie, Auxetic metamaterials and structures: a review, Smart materials and structures 27 (2) (2018) 023001.

[4] K. Bertoldi, P. M. Reis, S. Willshaw, T. Mullin, Negative poisson's ratio behavior induced by an elastic instability, Advanced Materials 22 (3) (2010) 361–366. doi:10.1002/adma.200901956.

[5] D. Goswami, S. Liu, A. Pal, L. G. Silva, R. V. Martinez, 3d-architected soft machines with topologically encoded motion, Advanced functional materials 29 (24) (2019) 1808713.

[6] A. Bekele, M. A. Wadee, A. Phillips, Enhancing energy absorption through sequential instabilities in mechanical metamaterials, Royal Society Open Science 10 (8) (2023).

[7] P. Wang, F. Casadei, S. Shan, J. C. Weaver, K. Bertoldi, Harnessing buckling to design tunable locally resonant acoustic metamaterials, Physical review letters 113 (1) (2014) 014301.

[8] J. Li, Y. Wang, W. Chen, Y.-S. Wang, R. Bao, Harnessing inclusions to tune post-buckling deformation and bandgaps of soft porous periodic structures, Journal of Sound and Vibration 459 (2019) 114848.

[9] S. Ning, D. Chu, F. Yang, H. Jiang, Z. Liu, Z. Zhuang, Characteristics of band gap and low-frequency wave propagation of mechanically tunable phononic crystals with scatterers in periodic porous elastomeric matrices, Journal of Applied Mechanics 88 (5) (2021) 051001.

[10] J. Shim, C. Perdigou, E. R. Chen, K. Bertoldi, P. M. Reis, Buckling-induced encapsulation of structured elastic shells under pressure, Proceedings of the National Academy of Sciences 109 (16) (2012) 5978–5983.

[11] D. Yang, B. Mosadegh, A. Ainla, B. Lee, F. Khashai, Z. Suo, K. Bertoldi, G. Whitesides, Buckling of elastomeric beams enables actuation of soft machines (2015).

[12] Y. Kim, G. A. Parada, S. Liu, X. Zhao, Ferromagnetic soft continuum robots, Science Robotics 4 (33) (2019) 7329. doi:10.1126/SCIROBOTICS.AAX7329.

[13] S. Janbaz, F. Bobbert, M. J. Mirzaali, A. A. Zadpoor, Ultra-programmable buckling-driven soft cellular mechanisms, Materials Horizons 6 (6) (2019) 1138–1147.

[14] J. Su, K. He, Y. Li, J. Tu, X. Chen, Soft materials and devices enabling sensorimotor functions in soft robots, Chemical Reviews 125 (12) (2025) 5848–5977.

[15] C. El Helou, P. R. Buskohl, C. E. Tabor, R. L. Harne, Digital logic gates in soft, conductive mechanical metamaterials, Nature communications 12 (1) (2021) 1633.

[16] J. Byun, A. Pal, J. Ko, M. Sitti, Integrated mechanical computing for autonomous soft machines, Nature Communications 15 (1) (2024) 2933.

[17] N. Ohno, D. Okumura, H. Noguchi, Microscopic symmetric bifurcation condition of cellular solids based on a homogenization theory of finite deformation, Journal of the Mechanics and Physics of Solids 50 (5) (2002) 1125–1153.

[18] J. Shim, S. Shan, A. Košmrlj, S. H. Kang, E. R. Chen, J. C. Weaver, K. Bertoldi, Harnessing instabilities for design of soft reconfigurable auxetic/chiral materials, Soft Matter 9 (34) (2013) 8198–8202.

[19] D. M. Kochmann, K. Bertoldi, Exploiting microstructural instabilities in solids and structures: from metamaterials to structural transitions, Applied mechanics reviews 69 (5) (2017) 050801.

[20] S. Janbaz, M. McGuinness, A. A. Zadpoor, Multimaterial control of instability in soft mechanical metamaterials, Physical Review Applied 9 (6) (2018) 064013.

[21] O. Faltus, M. Jirásek, M. Horák, M. Doškář, R. Peerlings, J. Zeman, O. Rokoš, Towards active stiffness control in pattern-forming pneumatic metamaterials, European Journal of Mechanics-A/Solids 112 (2025) 105632.

[22] R. Azulay, C. Combescure, J. Dirrenberger, Instability-induced pattern generation in architectured materials—a review of methods, International Journal of Solids and Structures 274 (2023) 112240.

[23] R. Azulay, C. Combescure, Predicting the post-bifurcated patterns of architectured materials using group-theoretic tools, Journal of the Mechanics and Physics of Solids 187 (2024) 105631.

[24] W. Wagner, P. Wriggers, A simple method for the calculation of postcritical branches, Engineering computations 5 (2) (1988) 103–109.

[25] F. Fujii, E. Ramm, Computational bifurcation theory: path-tracing, pinpointing and path-switching, Engineering Structures 19 (5) (1997) 385–392.

[26] F. Hendriks, V. Menkovski, M. Doškář, M. G. Geers, K. Verbeek, O. Rokoš, Wallpaper group-based mechanical metamaterials: Dataset including mechanical responses, Scientific Data (2025).

[27] R. Christensen, Sufficient symmetry conditions for isotropy of the elastic moduli tensor (1987).

[28] S. H. Kang, S. Shan, W. L. Noorduin, M. Khan, J. Aizenberg, K. Bertoldi, Buckling-induced reversible symmetry breaking and amplification of chirality using supported cellular structures (2013).

[29] J. M. Podestá, C. Méndez, S. Toro, A. E. Huespe, Symmetry considerations for topology design in the elastic inverse homogenization problem, Journal of the Mechanics and Physics of Solids 128 (2019) 54–78.

[30] R. P. Khawale, G. Vinal, R. Rai, W. W. Menasco, G. F. Dargush, Tiling-based lattice generation for structural property exploration, Materials & Design 247 (2024) 113391.

[31] O. Faltus, M. Doškář, J. Havelka, P. Rychnovskỳ, O. Rokoš, Non-uniform pneumatic actuation switches macroscopic properties of elastomeric honeycombs, Journal of the Mechanics and Physics of Solids (2026) 106594.

[32] F. Maurin, C. Claeys, E. Deckers, W. Desmet, Probability that a band-gap extremum is located on the irreducible brillouin-zone contour for the 17 different plane crystallographic lattices, International Journal of Solids and Structures 135 (2018) 26–36.

[33] F. Hendriks, O. Rokoš, M. Doškář, M. G. Geers, V. Menkovski, Equivariant flow matching for symmetry-breaking bifurcation problems, arXiv preprint arXiv:2509.03340 (2025).

[34] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, M. Le, Flow matching for generative modeling, arXiv preprint arXiv:2210.02747 (2022).

[35] M. S. Albergo, E. Vanden-Eijnden, Building normalizing flows with stochastic interpolants, arXiv preprint arXiv:2209.15571 (2022).

[36] X. Zhu, F. Wang, R. Walters, J. Shi, Se (3)-equivariant diffusion policy in spherical fourier space, arXiv preprint arXiv:2507.01723 (2025).

[37] K. B. t. Brinke, K. Minartz, V. Menkovski, Flow matching for geometric trajectory simulation, arXiv preprint arXiv:2505.18647 (2025).

[38] V. Kouznetsova, W. Brekelmans, F. Baaijens, An approach to micro-macro modeling of heterogeneous materials, Computational mechanics 27 (1) (2001) 37–48.

[39] R. Hill, Elastic properties of reinforced solids: some theoretical principles, Journal of the Mechanics and Physics of Solids 11 (5) (1963) 357–372.

[40] M. G. Geers, V. G. Kouznetsova, K. Matouš, J. Yvonnet, Homogenization methods and multiscale modeling: nonlinear problems, Encyclopedia of computational mechanics second edition (2017) 1–34.

[41] J. Yvonnet, Computational homogenization of heterogeneous materials with finite elements, Vol. 258, Springer, 2019.

[42] K. Bertoldi, M. C. Boyce, S. Deschanel, S. M. Prange, T. Mullin, Mechanics of deformation-triggered pattern transformations and superelastic behavior in periodic elastomeric structures, Journal of the Mechanics and Physics of Solids 56 (8) (2008) 2642–2668.

[43] J. T. Overvelde, K. Bertoldi, Relating pore shape to the non-linear response of periodic elastomeric structures, Journal of the Mechanics and Physics of Solids 64 (1) (2014) 351–366. doi:10.1016/j.jmps.2013.11.014.

[44] E. Polukhov, D. Vallicotti, M. A. Keip, Computational stability analysis of periodic electroactive polymer composites across scales, Computer Methods in Applied Mechanics and Engineering 337 (2018) 165–197. doi:10.1016/j.cma.2018.01.020.

[45] G. Geymonat, S. Müller, N. Triantafyllidis, Homogenization of nonlinearly elastic materials, microscopic bifurcation and macroscopic loss of rank-one convexity, Archive for Rational Mechanics and Analysis 122 (3) (1993) 231–290. doi:10.1007/BF00380256.

[46] N. Triantafyllidis, M. D. Nestorović, M. W. Schraad, Failure surfaces for finitely strained two-phase periodic solids under general in-plane loading, Journal of Applied Mechanics, Transactions ASME 73 (3) (2006) 505–515. doi:10.1115/1.2126695.

[47] K. Bertoldi, M. C. Boyce, S. Deschanel, S. M. Prange, T. Mullin, Mechanics of deformation-triggered pattern transformations and superelastic behavior in periodic elastomeric structures, Journal of the Mechanics and Physics of Solids 56 (8) (2008) 2642–2668. doi:10.1016/j.jmps.2008.03.006.

[48] G. Zhang, N. Feng, K. Khandelwal, A computational framework for homogenization and multiscale stability analyses of nonlinear periodic materials, International Journal for Numerical Methods in Engineering 122 (22) (2021) 6527–6575. doi:10.1002/nme.6802.

[49] F. Hendriks, V. Menkovski, M. Doškář, M. Geers, K. Verbeek, O. Rokoš, Wallpaper group-based mechanical metamaterials and their mechanical responses (Jul. 2025). doi:10.5281/zenodo.18172947.

[50] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, P. W. Battaglia, Learning Mesh-Based Simulation with Graph Networks (oct 2020). arXiv:2010.03409.

[51] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, P. W. Battaglia, Learning to Simulate Complex Physics with Graph Networks, arXiv (feb 2020). arXiv:2002.09405.

[52] L. Wang, Y.-C. Chan, F. Ahmed, Z. Liu, P. Zhu, W. Chen, Deep generative modeling for mechanistic-based learning and design of metamaterial systems, Computer Methods in Applied Mechanics and Engineering 372 (2020) 113377. doi:10.1016/j.cma.2020.113377.

[53] H. T. Kollmann, D. W. Abueidda, S. Koric, E. Guleryuz, N. A. Sobh, Deep learning for topology optimization of 2D metamaterials, Materials and Design 196 (2020) 109098. doi:10.1016/j.matdes.2020.109098.

[54] J. Storm, I. B. Rocha, F. van der Meer, A microstructure-based graph neural network for accelerating multiscale simulations, Computer Methods in Applied Mechanics and Engineering 427 (2024) 117001.

[55] R. B. Jakobsen, T. S. Kristensen, J. Storm, I. B. Rocha, T. Andriollo, Predicting plastic strain localization in porous solids using graph neural networks, Available at SSRN 5828366.

[56] L. Patrignani, S. T. Pinho, Graph neural networks with hybrid local-global attention for effective prediction of mechanical response in structures, Computer Methods in Applied Mechanics and Engineering 452 (2026) 118753.

[57] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, G. E. Dahl, Neural message passing for quantum chemistry, in: International conference on machine learning, Pmlr, 2017, pp. 1263–1272.

[58] Y.-L. Liao, A. J. Hoffman, S. C. Shen, A. Duval, S. W. Norwood, T. Smidt, Equiformerv3: Scaling efficient, expressive, and general se (3)-equivariant graph attention transformers, arXiv preprint arXiv:2604.09130 (2026).

[59] I. Batatia, D. P. Kovacs, G. Simm, C. Ortner, G. Csányi, Mace: Higher order equivariant message passing neural networks for fast and accurate force fields, Advances in neural information processing systems 35 (2022) 11423–11436.

[60] K. Schütt, O. Unke, M. Gastegger, Equivariant message passing for the prediction of tensorial properties and molecular spectra, in: International conference on machine learning, PMLR, 2021, pp. 9377–9388.

[61] J. Brandstetter, R. Hesselink, E. van der Pol, E. J. Bekkers, M. Welling, Geometric and physical quantities improve e (3) equivariant message passing, arXiv preprint arXiv:2110.02905 (2021).

[62] V. G. Satorras, E. Hoogeboom, M. Welling, E (n) equivariant graph neural networks, in: International conference on machine learning, PMLR, 2021, pp. 9323–9332.

[63] L. F. Guilhoto, A. Kaushal, P. Perdikaris, Multimodal scientific learning beyond diffusions and flows, arXiv preprint arXiv:2602.00960 (2026).

[64] T. E. Smidt, M. Geiger, B. K. Miller, Finding symmetry breaking order parameters with euclidean neural networks, Physical Review Research 3 (1) (2021) L012002.

[65] G. Fabiani, H. Vandecasteele, S. Goswami, C. Siettos, I. G. Kevrekidis, Enabling local neural operators to perform equation-free system-level analysis, arXiv preprint arXiv:2505.02308 (2025).

[66] A. Della Pia, D. G. Patsatzis, G. Rozza, L. Russo, C. Siettos, Surrogate normal-forms for the numerical bifurcation and stability analysis of navier-stokes flows via machine learning, Communications in Nonlinear Science and Numerical Simulation (2026) 110137.

[67] Z. Zou, Z. Wang, G. Em Karniadakis, Learning and discovering multiple solutions using physics-informed neural networks with random initialization and deep ensemble, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 481 (2325) (2025).

[68] H. Lawrence, V. Portilheiro, Y. Zhang, S.-O. Kaba, Improving equivariant networks with probabilistic symmetry breaking, arXiv preprint arXiv:2503.21985 (2025).

[69] K. Minartz, Y. Poels, S. Koop, V. Menkovski, Equivariant neural simulators for stochastic spatiotemporal dynamics, Advances in Neural Information Processing Systems 36 (2023) 38930–38957.

[70] A. Tong, K. Fatras, N. Malkin, G. Huguet, Y. Zhang, J. Rector-Brooks, G. Wolf, Y. Bengio, Improving and generalizing flow-based generative models with minibatch optimal transport, arXiv preprint arXiv:2302.00482 (2023).

[71] L. Klein, A. Krämer, F. Noé, Equivariant flow matching, Advances in Neural Information Processing Systems 36 (2023) 59886–59910.

[72] Y. Song, J. Gong, M. Xu, Z. Cao, Y. Lan, S. Ermon, H. Zhou, W.-Y. Ma, Equivariant flow matching with hybrid probability transport for 3d molecule generation, Advances in Neural Information Processing Systems 36 (2023) 549–568.

[73] J. Brandstetter, D. Worrall, M. Welling, Message passing neural pde solvers, arXiv preprint arXiv:2202.03376 (2022).

[74] K. Chen, X. Chen, Z. Yu, M. Zhu, H. Yang, Equidiff: A conditional equivariant diffusion model for trajectory prediction, in: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2023, pp. 746–751.

[75] J. Han, M. Xu, A. Lou, H. Ye, S. Ermon, Geometric trajectory diffusion models, Advances in Neural Information Processing Systems 37 (2024) 25628–25662.

[76] C. Tie, Y. Chen, R. Wu, B. Dong, Z. Li, C. Gao, H. Dong, Et-seed: Efficient trajectory-level se (3) equivariant diffusion policy, in: International Conference on Learning Representations, Vol. 2025, 2025, pp. 60114–60132.

[77] F. Sestak, A. Toshev, A. Fürst, G. Klambauer, A. Mayr, J. Brandstetter, Lam-slide: Latent space modeling of spatial dynamical systems via linked entities, Advances in Neural Information Processing Systems 38 (2026) 140166–140218.

[78] E. Perez, F. Strub, H. De Vries, V. Dumoulin, A. Courville, Film: Visual reasoning with a general conditioning layer, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 32, 2018.

[79] S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, B. Kozinsky, E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials, Nature communications 13 (1) (2022) 2453.

[80] Y.-L. Liao, B. Wood, A. Das, T. Smidt, Equiformerv2: Improved equivariant transformer for scaling to higher-degree representations, in: International Conference on Learning Representations, Vol. 2024, 2024, pp. 39282–39309.

[81] Q. Li, Z. Han, X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 32, 2018.

[82] K. Oono, T. Suzuki, Graph neural networks exponentially lose expressive power for node classification, arXiv preprint arXiv:1905.10947 (2019).

[83] U. Alon, E. Yahav, On the bottleneck of graph neural networks and its practical implications, arXiv preprint arXiv:2006.05205 (2020).

[84] F. Di Giovanni, T. K. Rusch, M. M. Bronstein, A. Deac, M. Lackenby, S. Mishra, P. Veličković, How does over-squashing affect the power of gnns?, arXiv preprint arXiv:2306.03589 (2023).

[85] A. Brandt, Multiscale scientific computation: Review 2001, Multiscale and Multiresolution Methods: Theory and Applications (2002) 3–95.

[86] W. Hackbusch, Parabolic multi-grid methods, in: Proc. of the sixth int'l. symposium on Computing methods in applied sciences and engineering, VI, 1985, pp. 189–197.

[87] J. Xu, Iterative methods by space decomposition and subspace correction, SIAM review 34 (4) (1992) 581–613.

[88] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, J. Leskovec, Hierarchical graph representation learning with differentiable pooling, Advances in neural information processing systems 31 (2018).

[89] J. Han, W. Huang, T. Xu, Y. Rong, Equivariant graph hierarchy-based neural networks, Advances in Neural Information Processing Systems 35 (2022) 9176–9187.

[90] M. Zhdanov, M. Welling, J.-W. Van De Meent, Erwin: A tree-based hierarchical transformer for large-scale physical systems, arXiv preprint arXiv:2502.17019 (2025).

[91] M. Fortunato, T. Pfaff, P. Wirnsberger, A. Pritzel, P. Battaglia, Multiscale meshgraphnets, arXiv preprint arXiv:2210.00612 (2022).

[92] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, Advances in neural information processing systems 30 (2017).

[93] B. Alkin, A. Fürst, S. Schmid, L. Gruber, M. Holzleitner, J. Brandstetter, Universal physics transformers: A framework for efficiently scaling neural operators, Advances in Neural Information Processing Systems 37 (2024) 25152–25194.

[94] J. Lee, I. Lee, J. Kang, Self-attention graph pooling, in: International conference on machine learning, pmlr, 2019, pp. 3734–3743.

[95] H. Gao, S. Ji, Graph u-nets, in: international conference on machine learning, PMLR, 2019, pp. 2083–2092.

[96] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, Journal of statistical mechanics: theory and experiment 2008 (10) (2008) P10008.

[97] C. K. Joshi, C. Bodnar, S. V. Mathis, T. Cohen, P. Lio, On the expressive power of geometric graph neural networks, in: International conference on machine learning, PMLR, 2023, pp. 15330–15355.

[98] J. Gasteiger, J. Groß, S. Günnemann, Directional message passing for molecular graphs, arXiv preprint arXiv:2003.03123 (2020).

[99] J. Gasteiger, S. Giri, J. T. Margraf, S. Günnemann, Fast and uncertainty-aware directional message passing for non-equilibrium molecules, arXiv preprint arXiv:2011.14115 (2020).

[100] J. Gasteiger, F. Becker, S. Günnemann, Gemnet: Universal directional graph neural networks for molecules, Advances in neural information processing systems 34 (2021) 6790–6802.

[101] J. Abramson, J. Adler, J. Dunger, R. Evans, T. Green, A. Pritzel, O. Ronneberger, L. Willmore, A. J. Ballard, J. Bambrick, et al., Accurate structure prediction of biomolecular interactions with alphafold 3, Nature 630 (8016) (2024) 493–500.

[102] J. Shah, G. Bikshandi, Y. Zhang, V. Thakkar, P. Ramani, T. Dao, Flashattention-3: Fast and accurate attention with asynchrony and low-precision, Advances in Neural Information Processing Systems 37 (2024) 68658–68685.

[103] X. Liu, C. Gong, Q. Liu, Flow straight and fast: Learning to generate and transfer data with rectified flow, arXiv preprint arXiv:2209.03003 (2022).

CHAPTER 6

Discussion and Outlook

This thesis has investigated how machine learning can be applied to the simulation of buckling mechanical metamaterials. The central difficulty is that these systems combine several challenging features: complex geometry, nonlinear deformation, periodicity, symmetry, and multiple admissible post-buckling responses. Standard surrogate models are often not well suited to this setting, because they may ignore physical symmetries, require narrow geometry classes (e.g., only square RVEs), or collapse multiple valid buckling modes into a single averaged prediction.

The broader motivation is the design of better mechanical metamaterials. Efficient design requires predictions that are not just fast, but that are also able to process diverse geometries, respect the structure of the underlying mechanics and represent mode multiplicity. Such predictions can eventually be used inside inverse-design and optimization loops. The contributions of this thesis should be seen as steps towards that goal.

6.1 Revisiting the research questions

The work was organized around four research questions.

RQ1. How can neural-network surrogate models be designed so that they respect the symmetries and physical constraints of mechanical metamaterials?
Chapter 2 addressed this question by developing SimEGNN, a similarity-equivariant graph neural network for homogenization of periodic porous metamaterials. The model represents the microstructure as a graph and predicts both local deformation fields and homogenized quantities such as strain energy density, stress, and stiffness. By incorporating Euclidean equivariance, periodicity, scale equivariance, and a tensorial output structure, the model builds important physical constraints directly into the architecture. The results show that for the chosen test system, these inductive biases significantly improve accuracy compared to models that rely mainly on data augmentation.

RQ2. How can a dataset be constructed to study and learn mechanical responses across a broad range of loading conditions and metamaterial designs, including all periodic planar symmetry groups?
Chapter 3 addressed this question by introducing a dataset of two-dimensional wallpaper-group-based mechanical metamaterials, which relieves a key bottleneck in the development of data-driven approaches: the lack of comprehensive datasets. The dataset contains 1,020 microstructures spanning all 17 wallpaper groups, together with simulated mechanical responses under multiple loading

paths in the finite-strain regime, including cases with buckling. This provides a broader basis for learning and benchmarking than datasets restricted to a small set of canonical unit cells or loading cases. It also supports the study of how symmetry influences mechanical response, including anisotropy, auxeticity, buckling tendency, and the multiplicity of possible deformation modes.

RQ3. How can machine-learning models predict symmetry breaking, where one symmetric input may lead to multiple valid outputs?
Chapter 4 addressed this question by formulating symmetry-breaking bifurcation as a probabilistic learning problem. Instead of predicting a single deterministic output, the proposed equivariant flow-matching framework learns a distribution over admissible outcomes. This distinction is essential: individual samples may break symmetry, while the full distribution should still respect the symmetry of the governing system. Symmetric coupling was introduced as a practical training strategy for such symmetric multimodal distributions. The examples showed that equivariant generative models can capture multiple coexisting solutions in controlled bifurcation problems without collapsing them into nonphysical averages.

RQ4. How can equivariant generative models be scaled to realistic buckling trajectories in mechanical metamaterials?
Chapter 5 addressed this question by combining equivariant graph neural networks, flow matching, temporal modeling, and hierarchical architectures into one probabilistic framework for buckling trajectories. This moves beyond low-dimensional proof-of-concept bifurcation problems towards graph-structured metamaterial systems with many degrees of freedom. The model can generate multiple possible deformation trajectories for a given microstructure and loading path, and the hierarchical architecture helps coordinate local and global deformation patterns. The results on both the stacked-holes benchmark and the wallpaper-group dataset demonstrate that this direction is feasible, while also showing that accuracy, computational cost, and scalability remain open challenges.

6.2 Main achievements and novelties

The main achievements of this thesis can be summarized as follows.

- A graph-based surrogate modeling framework was developed for nonlinear periodic metamaterials, in which geometry is used directly as input rather than being reduced to user-defined geometrical parameters.
- Physical symmetries were incorporated explicitly into graph neural networks for metamaterial homogenization, including translation, rotation, reflection, periodicity, scale equivariance, and the transformation behavior of scalar, vector, and tensor quantities.
- A systematic dataset of wallpaper-group-based mechanical metamaterials was created, covering all 17 planar periodic symmetry groups and multiple finite-strain loading paths. This dataset provides both training data for machine-learning models and a resource for studying symmetry-property relationships.

- Symmetry-breaking buckling was reformulated as a probabilistic prediction problem. This addresses the fundamental mismatch between deterministic regression and the one-to-many nature of post-buckling response.
- Equivariant flow matching was adapted to bifurcation problems, with symmetric coupling used to exploit the equivalence between symmetry-related outcomes during training.
- The probabilistic approach was scaled from controlled low-dimensional systems to high-dimensional graph-structured metamaterial trajectories by combining equivariant GNNs, temporal modeling, and hierarchical long-range communication.

Together, these contributions establish a coherent approach for learning the response of nonlinear, symmetry-breaking mechanical metamaterials. The work does not yet provide a complete automated design tool, but it identifies and demonstrates several of the components such a tool would require: fast surrogate prediction, geometry-aware data representation, systematic datasets, symmetry-respecting architectures, and probabilistic treatment of multiple possible post-buckling modes.

6.3 Implications for metamaterial design

The original motivation for this thesis was not only to predict metamaterial behavior faster, but ultimately to make the design of metamaterials more efficient. In design, one is rarely interested in a single forward simulation. Instead, many candidate geometries must be evaluated, compared, optimized, and made robust against uncertainty. Buckling makes this especially difficult because the same design may admit several possible responses.

The framework developed in this thesis contributes to this design problem in three ways. First, graph-based surrogate models can accelerate the evaluation of candidate microstructures while preserving geometric flexibility. Second, the wallpaper-group dataset provides a structured way to explore a much broader symmetry and geometry space than a single canonical unit cell. Third, probabilistic generative models allow the design process to consider not only an average response, but the distribution of possible responses.

In this sense, the thesis highlights the potential of machine learning in metamaterial design, and reframes its role. Rather than treating machine learning as a black-box replacement for finite-element simulations, it argues for models whose data representation, architecture, and output distribution reflect the structure of the mechanics. This is particularly important for systems governed by symmetry and instability, where the physically meaningful prediction is often not a single deformation field but a structured set of possible outcomes.

6.4 Limitations and outlook

The results also point to several directions that remain open. These open directions should be interpreted in the context of the difficulty of the problem: current state-of-the-art models do not yet provide a general solution for probabilistic prediction of

buckling trajectories in arbitrary mechanical metamaterials. The final framework in this thesis is therefore best viewed as a proof of concept and an exploration of what such models require.

- **Data representation.** Graphs are a natural representation for finite-element meshes and porous microstructures, because they can handle irregular geometry and topology. However, the best representation for generative metamaterial design is still an open question. Pixel-based, mesh-based, graph-based, point-cloud, implicit, and hybrid representations each have advantages. Future work should investigate which representations best support both accurate forward prediction and efficient inverse design.
- **Model architecture and global coordination.** Buckling is not only a local phenomenon. Different parts of the structure must coordinate to realize a physically consistent mode. The hierarchical architecture used in this thesis is one way to introduce long-range communication, but more expressive equivariant architectures, attention mechanisms, multiscale graph representations, or physics-informed constraints may be needed to improve performance on highly diverse datasets.
- **Generative modeling efficiency.** Flow matching provides a principled way to model multimodal output distributions, but the current models remain computationally expensive to train. For use in design loops, future work should focus on faster sampling, more efficient training and better uncertainty calibration.
- **Physical scope.** The present work is limited to two-dimensional porous geometries and a single class of hyperelastic material models. Extending the framework to three-dimensional metamaterials, different material laws, contact, dynamics, multi-material systems, and experimental data would be necessary for broader engineering use.
- **Datasets and validation.** The wallpaper-group dataset broadens the available design space, but it is still a finite and idealized collection of simulated examples. Future datasets should include more diverse geometry-generation procedures, manufacturing constraints, imperfections, contact, and, where possible, experimental validation. This would make learned models more reliable for practical design.
- **Inverse design and optimization.** Another important next step is to integrate probabilistic surrogate models into inverse-design pipelines. Such pipelines should optimize not only a target mechanical response, but also robustness with respect to mode selection, imperfections, and loading variability. Generative models could be used both to predict possible responses of a candidate design and to propose new candidate geometries.
- **Broader applicability.** Future research could extend the proposed framework to other problems in structural and solid mechanics with non-unique, symmetry-breaking, or path-dependent responses, such as shell buckling, wrinkling, plastic localization, fracture, delamination, and phase transformations. In such systems, deterministic surrogates may be insufficient because the same nominal geometry and loading can lead to several admissible outcomes. The combination of equivariant graph-based representations, probabilistic generative modeling, and hierarchical long-range communication could therefore provide a

general modeling strategy for mechanical systems in which geometry, symmetry, instability, and solution multiplicity play a role.

Overall, this thesis shows that equivariant and probabilistic machine learning provides a promising route towards faster and more reliable modeling of buckling mechanical metamaterials. Its main message is that the complexity of these systems should not be ignored or averaged out. Geometry, symmetry, instability, and multimodality are central features of the physics and should therefore be built into the learning problem. By combining structured datasets, graph-based equivariant architectures, and generative modeling, this thesis provides a foundation for future tools that can simulate, analyze, and ultimately design mechanical metamaterials with nonlinear and multimodal responses.

Curriculum Vitae



Fleur Hendriks was born on April 11^{th}, 1995, in Weert, the Netherlands. She completed her secondary education (gymnasium) at Het College Weert. During her time there, she developed several programming projects. Her final school project focused on the mathematical analysis and computational modeling of electoral systems, foreshadowing her interest in mathematics, computation, and modeling.

In 2013, Fleur started her bachelor's degree in Applied Physics at Eindhoven University of Technology. She developed a strong interest in computational physics, programming, and machine learning, which she continued to pursue during her master's degree in Applied Physics, specializing in Plasma Physics and Radiation Technology. During her master's, she completed an internship at ASML on simulations of ion transport in narrow channels. For her graduation project, she developed a permutation-invariant H_2–H_2 potential energy surface for studying vibrational energy transfer, combining quantum chemistry with molecular dynamics. She completed her master's degree cum laude in 2021, with a GPA of 8.7/10 and a Theory for Technology acknowledgement.

In 2021, Fleur started her PhD at Eindhoven University of Technology, working at the intersection of machine learning, applied physics, and mechanical engineering. Her doctoral research focused on machine-learning methods for the simulation of mechanical metamaterials, with particular attention to graph neural networks, geometric deep learning, generative AI, and physical symmetries. She also attended the Generative AI Summer School at the Technical University of Denmark. During her PhD, she developed the Similarity-Equivariant Graph Neural Network for metamaterial homogenization, constructed a dataset of wallpaper-group-based mechanical metamaterials, and developed probabilistic generative models for buckling and symmetry-breaking bifurcation problems. Her work resulted in several first-author publications, including papers in Computer Methods in Applied Mechanics and Engineering, Scientific Data, and the NeurIPS workshop Machine Learning and the Physical Sciences.

Alongside her research, Fleur served as a teaching assistant for the Solid Mechanics course, supervised bachelor's and master's students, and organized the Bifurcation Hackathon. At the Engineering Mechanics Symposium, her work was recognized with the Best Poster Award in 2023 and the Best Presentation Award in 2024.

Beyond academia, Fleur has pursued creative and technical hobbies and side projects, including graph visualizations and an Android app. Since 2024, she has volunteered at MakersPlaats at De Bibliotheek Eindhoven.

After completing her PhD thesis, Fleur joined DIFFER as a postdoctoral researcher, where she works on artificial intelligence for nuclear fusion.

List of publications

First-author publications

- **F. Hendriks**, V. Menkovski, M. Doškář, M. G. D. Geers, and O. Rokoš. "Similarity equivariant graph neural networks for homogenization of metamaterials." *Computer Methods in Applied Mechanics and Engineering*, 439, 117867, 2025.
- **F. Hendriks**, V. Menkovski, M. Doškář, M. G. D. Geers, K. Verbeek, and O. Rokoš. "Wallpaper group-based mechanical metamaterials: Dataset including mechanical responses." *Scientific Data*, 2025.
- **F. Hendriks**, O. Rokoš, M. Doškář, M. G. D. Geers, and V. Menkovski. "Equivariant flow matching for symmetry-breaking bifurcation problems." In NeurIPS Workshop *Machine Learning and the Physical Sciences*, 2025.
- **F. Hendriks**, V. Menkovski, M. Doškář, M. G. D. Geers, and O. Rokoš. "Probabilistic Modeling of Buckling in Mechanical Metamaterials." In preparation, 2026.

Other publications

- K. Minartz, F. Hendriks, S. M. Koop, A. Corbetta, and V. Menkovski. "Discovering interaction mechanisms in crowds via deep generative surrogate experiments." *Scientific Reports*, 15.1, 10385, 2025.
- L. Gaynutdinova, P. Havlásek, O. Rokoš, F. Hendriks, and M. Doškář. "Autoregressive U-Net for full-field prediction of shrinkage-induced damage in concrete." Submitted for journal publication, 2026.

Awards

- **Best Poster Award**, 2023, Engineering Mechanics symposium. *Graph Neural Networks with Embedded Symmetries for Robust Computational Homogenization*
- **Best Presentation Award**, 2024, Engineering Mechanics symposium. *Embedding Symmetries into Graph Neural Networks for Efficient Computational Homogenization*

Acknowledgements

This PhD would not have been completed without the support of many people, and I would like to thank everyone who has contributed, directly or indirectly, whether formally in the form of supervision or a collaboration, or informally through discussion, advice, a shoulder to cry on or a pleasant distraction.

First of all I would like to thank my supervisors, Ondrej, Vlado, Martin and Marc, for their guidance, support and encouragement. I appreciate how everyone was willing to stretch beyond their own field of expertise and learn something new, and how through communication and collaboration we were able to make progress and learn from each other.

Ondrej, I feel grateful to have been able to learn so much from you, both about metamaterials and how to be a good researcher, as well as about how to bring out the best in people while always staying kind. Your advice has been invaluable. I feel very lucky to have had a supervisor who gave me freedom to explore my own ideas and interests, while also steering me in the right direction and providing guidance and support.

Vlado, I appreciate your insight and deep knowledge of machine learning. Conversations with you have always been inspiring with their various tangents, and I have learned a lot from you about how to approach research problems and how to communicate them. Thank you also for organizing the group lunches and trying to impart your bouldering skills on me.

Martin, I appreciate your enthousiasm and your patience in helping me work with your code. Even from the Czech Republic, you were just as much a part of our little team as anyone else. Thank you for your help and attention to detail.

Marc, thank you for sharing your wisdom as a senior researcher, and for your support in arranging an extension near the end of my PhD, which allowed me to finish my project without too much added stress.

I would also like to thank my other collaborators, Kevin, Liya, Koen, Simon, Alessandro, Anna, Johannes, Martin W and all the other people I have collaborated with during my PhD. I have learned a lot from you, and I am grateful for your help and enthousiasm.

My time at TU/e would not have been the same without my office mates in various offices, the first and foremost of which was Gemini 4.14, where I spent many hours working, discussing, and laughing with Adrià, Alicia, Bart, Britt, Daniek, Elmer, Esther, Eva, Evianne, Federica, Justina, Jules, Kaja, Lena, Luca, Meike, Moeen, Niels, Parisa, Rick, Rienk, Sil, Tara, Tijmen. Thanks for the many nice office outings and the many good conversations.

I additionally spent some time in offices with various MoM group members, including Benjamin, Cassio, Etienne, Fatih, Kylian, Kaan, Marwan, Phani Ram, Robin, Ruben, Sebastián, Tigernach, Umar. Thanks for the fun distractions, including but not limited to good conversations, the dart board and the pepernoten.

To the Metaforum office, Kiet, Koen M, Mahdi, Marko, Rachna, Simon, Yoeri, thank you for the insightful discussions, rubber-ducking and the many good laughs.

Other of the MoM group, Anne, Avi, Ayush, Bram, Britt, Carlos, Casper, Clément, Cong, Georgia, Job, Lei, Lloyd, Noud, Nanditha, Rani, Renan, Sadık, Sven, Tom, Vahid, Zhi-Hua, and all the others,thank you all for the nice chats.

I am grateful to all the BEP and MSc students I worked with, Joël, Alexandra, Dennis, Mart, Linsy, Manon and Koen G.

I would like to thank Alice and Rachel for the administrative support and efficient help, and for providing a friendly face to turn to with questions.

Thank you also to the people from the Makersplaats, where I have enjoyed volunteering and learning new ways to be creative.

I would like to express my gratitude to my family, who kept me sane with the many trips and creative outings and discussions.

Ome Dick, thanks for the nice vacations we had together and the creative brainstorming.

Meike, your creativity and humor are worth a lot to me and I always have fun doing creative things with you.

Dorus, my data science buddy, I enjoyed your humor and your opinions on AI and building our app together.

A special thank you to my parents, who have always supported me. Thank you for the very obvious support such as taking care of me when healing from a broken neck, and the less obvious support of encouraging me to relax and have a healthy life-work balance, and all the little things that show how much you care.

Lastly, I would like to thank my paranymphs, Liya and Robin. Your friendship and support has been invaluable. Liya, thank you for the fun trip to Prague and Switzerland, the regular video calls, and for commiserating with me about the ups and downs of PhD life. Robin, thank you for your support and advice and for the many adventures you drag me into. I look forward to the rest of our lives together.



9 789465 346816 >