PhD-FSTM-2023-083
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on October 13th 2023 in Esch-sur-Alzette
to obtain the degree of

DOCTEUR DE L'UNIVERSIT'E DU LUXEMBOURG EN SCIENCES DE L'INGÉNIEUR

by

VU MINH CHAU
born on August 18th 1989 in Long An, Vietnam

CONSTITUTIVE MODELLING OF NON-LINEAR ISOTROPIC ELASTICITY USING DEEP REGRESSION NEURAL NETWORKS

Dissertation defence committee:

Professor Andreas ZILIAN, Dissertation Supervisor
Professor, Université du Luxembourg

Dr Khiem NGUYEN
Lecturer, University of Glasgow

Professor Christoph SCHOMMER, Chairman
Professor, Université du Luxembourg

Dr Michal HABERA
Research scientist, Rafinex S.à r.l.

Dr Lars BEEX, Vice Chairman
Research scientist, Université du Luxembourg

To my ever-supportive parents, whose love and encouragement propelled me through this journey.

Acknowledgements

I would like to express my appreciation to my supervisor, Andreas Zilian, whose guidance and mentorship played a pivotal role in shaping this dissertation. Gratitude extends to the members of the Zilian group, including Lan Shang, Chrysovalantou Kalaitzidou, Michal Habera, and Hamidreza Dehghani, with whom I have engaged in discussions on both professional and personal fronts.

I also wish to acknowledge the supportive and friendly atmosphere provided by the members at the Department of Engineering at the University of Luxembourg. My heartfelt thanks go to the friends within the Computational Mechanics research group, whose unwavering support was pivotal in navigating the challenges of graduate life.

This work would have been significantly more challenging without the unwavering support of my parents and my two brothers. Their genuine concern for my well-being and unwavering encouragement sustained me through the highs and lows of this research journey.

To my girlfriend, Nguyen Thi Phuong Thao, your presence illuminated many dark days, and your support throughout this dissertation writing process has been truly invaluable.

Abstract

Deep neural networks (DNNs) have emerged as a promising approach for constitutive modelling of advanced materials in computational mechanics. However, achieving physically realistic and stable numerical simulations with ANNs can be challenging, especially when dealing with large deformations, that can lead to non-convergence effects in the presence of local stretch/stress peaks.

This PhD dissertation introduces a novel approach for data-driven modelling of non-linear compressible isotropic materials, focusing on predicting the large deformation response of 3D specimens. The proposed methodology formulates the underlying hyperelastic deformation problem in the principal space using principal stretches and principal stresses, in which the corresponding constitutive relation is captured by a deep neural network surrogate model. To ensure constitutive requirements of the model while preserving the robustness of underlying numerical solution schemes, several physics-motivated constraints are imposed on the architecture of the DNN, such as objectivity, growth condition, normalized condition, and Hill's inequalities.

Furthermore, the prediction phase utilizes a constitutive blending approach to overcome divergence in the Newton-Raphson process, which can occur when solving boundary value problems using the Finite Element Method. The work also presents a machine learning-finite element pipeline for modelling non-linear compressible isotropic materials, involving determining automatically the hyperparameters, training, and integrating the ANN operator into the finite element solver using symbolic representation.

The proposed formalism has been tested through numerical benchmarks, demonstrating its ability to describe non-trivial load-deformation trajectories of 3D test specimens accurately. Overall, the thesis presents a complete and general formalism for data-driven modelling of non-linear compressible isotropic materials that overcomes the limitations of existing approaches.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Introduction

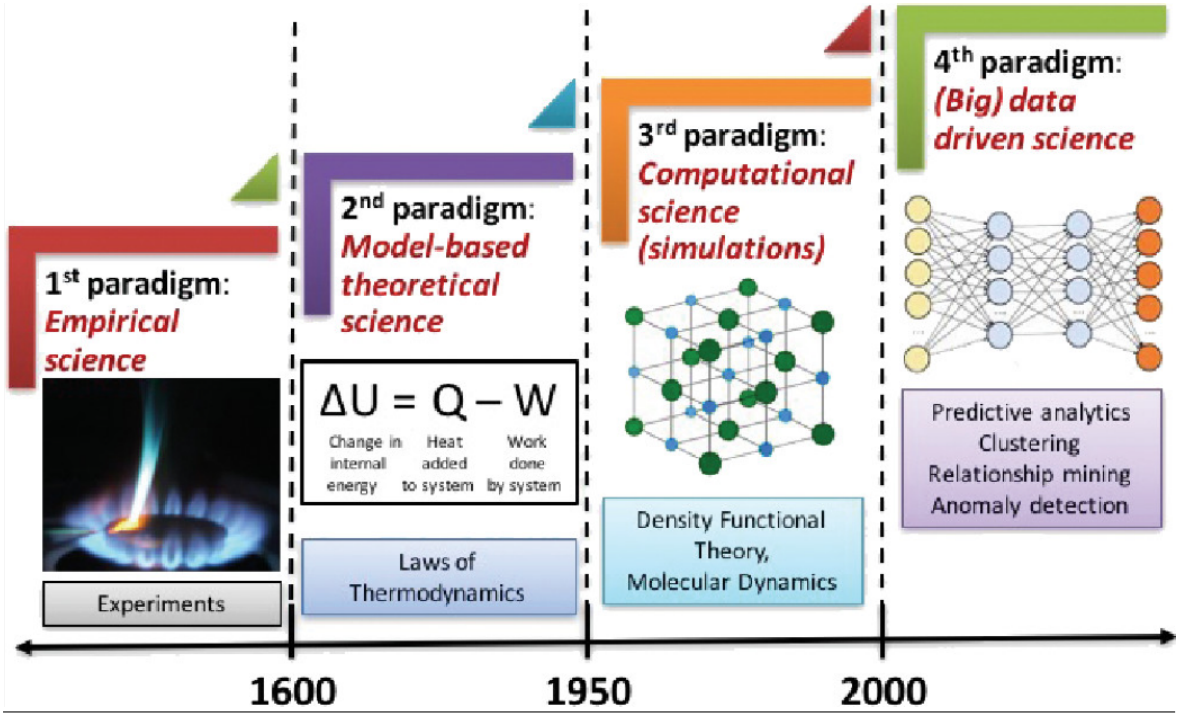
The data revolution has been rapidly emerging in a wide variety of disciplines including natural and social sciences, engineering, and information technology. As a result, a tremendous of raw data produced from instruments, sensors, or simulations need to be managed and analysed (Bell, Hey, and Szalay, 2009; Hey, Tansley, and Tolle, 2009a) to make it *understandable*.

Data-driven methods, also known as the "Fourth Paradigm" (Hey, Tansley, and Tolle, 2009b), have emerged to address these challenges by extracting knowledge directly from structured or unstructured data. These methods utilise machine learning, statistical learning, data analytics, and optimization to determine a mathematical model to develop effective solutions and aid in decision-making (Dhar, 2013).

Across scientific disciplines, transformative results have been accomplished such as astroinformatics (Borne, 2010), bioinformatics (Hogeweg, 31-Mar-2011), image recognition (Krizhevsky, Sutskever, and G. E. Hinton, 2012), cognitive science (Lake, Salakhutdinov, and Tenenbaum, 2015), just to name a few.

In the field of computational engineering science, not only the volume of the available data (e.g. NOMAD Repository (Draxl and Scheffler, 2019)) has increased dramatically, but the tools for producing and analysing it have also considerably improved over the past decades, including sensors, computational hardware, and software (W. K. Liu et al., 2019). Inheriting from these advancements and the rise of high-performance computing, data-driven methods for mechanics have emerged as a promising trend. These methods allow users to input physical processes into the data-driven solver, which can predict the correct output even without a well-defined model. However, the solver is demanded to show the capability of capturing the complex behaviour and constituting governing equations of the system (Montáns et al., 2019). Therefore, there are significant challenges in developing novel computational algorithms that

Figure 1.1 The four paradigms of science are empirical, theoretical, computational, and data-driven (Agrawal and Choudhary, 2016). It begins with empirical science, which involves pure experiments and observations. Then, there was the rise of model-based theoretical science characterized by mathematical equations and laws. Subsequently, the third paradigm introduced computational science, building upon the foundation of the second era. Finally, (big) data-driven science marked a significant milestone in scientific development, emphasizing the need to establish novel computational algorithms.



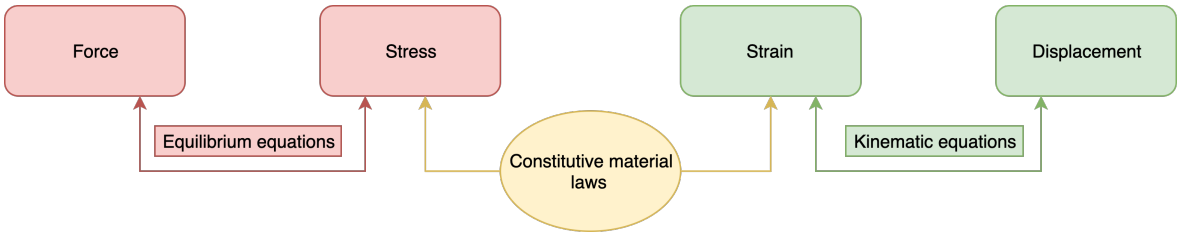
deploy data-driven paradigms without relying on analytical equations, but instead operate directly on experimental data.

The exploration of experimental data is not a new concept. Originally, only experimental science provided the empirical understanding of nature, representing the first of the four paradigms of scientific discovery. Figure 1.1 by Agrawal and Choudhary (2016) illustrates all four paradigms of science throughout the history of human research.

One of the very first laws that can be mentioned is the work of Archimedes, a member of the earliest empirical researchers, who sought to solve the practical problem known as the "Wreath Problem". He established a profound practice of buoyancy measurement and formulated it in an undoubtedly *empirical common language* that can be checked by co-practitioners (Nierlich, 2005). However, this explanation remained primarily based on attraction and had not been a systematic scientific activity.

Subsequently, the development of calculus contributed to the second paradigm i.e. *theoretical models* and generalizations. Various mathematical formulations of "laws" have been discovered, namely Kepler's Laws, Newton's Laws of Motion, Maxwell's equations, and the laws of

Figure 1.2 Constitutive equations show the relationship between applied load and stress can be described through equilibrium equations whereas that of strain-displacement is characterised using kinematic equations.



thermodynamics, among others. These seminal breakthroughs led to the ability to generalize the behaviour of natural phenomena in regimes where no data had been observed (Brunton, Proctor, and Kutz, 2016).

Soon, complex and nonlinear scientific problems eventually grew too complicated to solve analytically, which has given rise to the urge for an alternative approach. Numerical simulation played its role in solving these tasks based on the theoretical models of the second paradigm for the last several decades, marked as the third paradigm (Hey, Tansley, and Tolle, 2009b), *computational science*. Finite elements, finite differences, finite volumes, molecular dynamics simulations, and mesh-free methods are all excellent examples of this computational era.

In more detail, simulation models (computational era) in classical continuum mechanics are based on the universal principles of physical equations known as conservation laws (balance principles: mass, momentum, and energy), the second law of thermodynamics derived back from the second paradigm. Although these conservation laws hold for all continuum bodies, they cannot distinguish the material types in the case of deformable bodies. Therefore, supplementary laws are established according to the observed physical behaviour of real materials under specific conditions of interest in order to determine their responses regarding *constitutive laws* or material laws (Holzapfel, 2002). Figure 1.2 shows that constitutive equations bridge the gap between the equilibrium relations of applied load and stress and the kinematic behaviour of a continuum body (strain-displacement expressions). They are the missing block required to complete the prediction of the behaviour of materials.

Constitutive equations consist of physical-based, phenomenological-based, and empirical models. The first two types of material models require significant knowledge of the physical phenomena to construct (Marckmann and Verron, 2006). The difference between them is that in physical constitutive relations, the material parameters have physical meaning, whereas the phenomenological approach is mainly concerned with fitting mathematical equations to data obtained from experiments. The coefficients of the phenomenological approach do not necessarily explain physical behaviour. In contrast, empirical models directly derive the mathematical equations from the analysis of experimental data while minimizing the need for an explicit understanding of the physical problem. This approach is particularly helpful when

there is a lack of information about the underlying physics of the process (Solomatine, See, and Abrahart, 2008).

Nevertheless, all types of constitutive equations contain errors and uncertainties, often leading to failures in representing new materials (T. T. Kirchdoerfer, 2017). Hence, it becomes necessary to reconsider the objectives of computational algorithms and improve the linear and nonlinear continuum governing equations that are derived from limited and simplified experimental data.

An example approach is to acquire material data generated by experiments and simulations in order to reinvestigate and extract more accurate information, leading to the construction of new constitutive equations. This strategy is known as *data-driven methods*, representing the fourth paradigm, which has driven science to a point where the complexity of the target equations is no longer a major limitation (Montáns et al., 2019).

Developing alternative data-driven approaches to extract physical knowledge from systems has been pursued by numerous researchers, but these techniques still exhibit case sensitivity. Specifically, for the purpose of extracting constitutive equations, various types of data-driven measurements have been employed. One such approach is Data-Driven Computational Mechanics (DDCM), introduced by T. Kirchdoerfer and Ortiz (2016). DDCM utilizes data distance minimization techniques to generate classical solutions in elasticity. While it can circumvent the explicit use of constitutive relations, it requires a large number of data points as critical requirements.

Recently, Brunton, Proctor, and Kutz (2016) proposed a promising direction for identifying governing equations in nonlinear dynamical systems. This approach combines sparse regression (Hastie, Tibshirani, and Friedman, 2009; Tibshirani, 1996) with genetic algorithm (Koza and Koza, 1992) to select the most accurate representation of data from a pool of simple functions and partial derivatives.

In the same manner, Martius and Lampert (2016) modified the activation functions of a shallow Artificial Neural Network (ANN) called equation learner (EQL) with simple equations and operators frequently found in dynamics problems. This allows for the construction of functional relations from observed data, and the obtained equations enable the extrapolation of results to the unseen parts of the parameter space. However, it is worth noting that EQL may experience difficulties in learning some mathematical expressions even in the interpolation region.

The task of identifying material laws from experimental data has been expressed by those mentioned approaches, still, it remains a challenging question. Therefore, it is necessary to employ some form of *machine learning* to overcome their main limitations, particularly the ability to handle extensive big data.

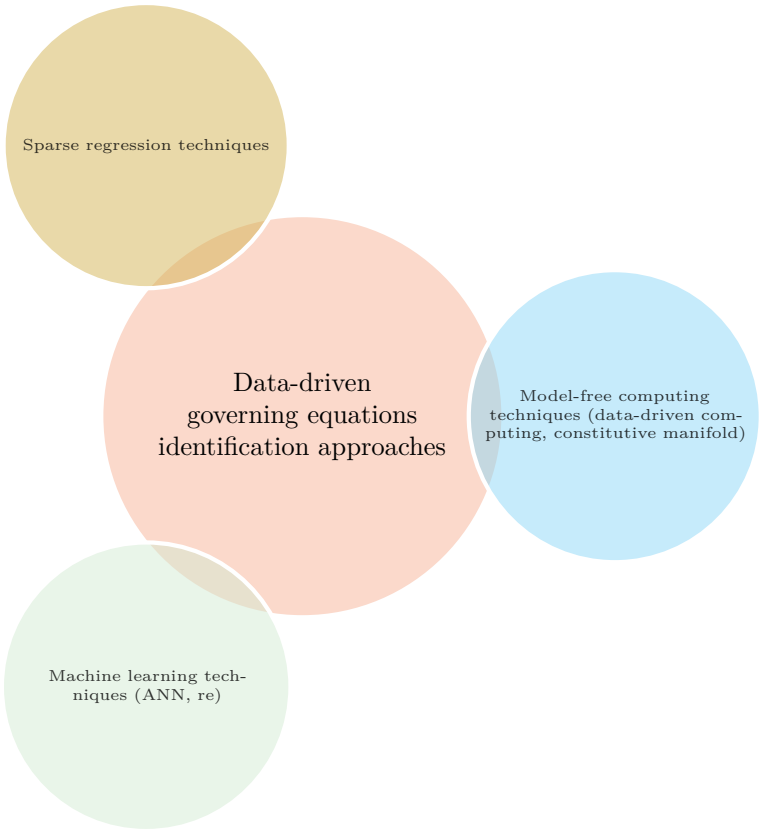
In this project, the author explores the feasibility of:

- Developing an explainable ANN that can extract the constitutive equations from observed datasets by imposing physical constraints. This modified version of ANN aims to provide an interpretable representation of the underlying physics.
- Carefully investigating the advantages and disadvantages of the proposed methods, as well as determining their effective domains of application.

1.2 Review of state-of-the-art

Data-driven applications in computational mechanics have been developed in several directions. Recently, Model-free computing acts as an empirical treatment which presents the ability to derive the relationships of interested variable fields directly from data. The other trend is sparsity-promoting techniques that can discover the governing equations of a system of dynamic data. As shown in Figure 1.3, one of the most important branches is the Machine Learning approach which is the earliest and broadest method that has been established and constantly advanced at present.

Figure 1.3 Data-driven methods



1.2.1 Model-free computing method

Among the first researchers who attempted to implement the fourth paradigm in computational mechanics (T. Kirchdoerfer and Ortiz, 2016), data-driven computing did not employ any material constitutive models directly but integrated the computed strain, stress pairs through a given experimental dataset in order to minimize their distance that best satisfies conservation laws and relevant boundary conditions corresponding to the specific problem. For instance, the three-dimensional classical elasticity solutions are recovered and conditions for convergence have been derived by Conti, S. Müller, and Ortiz (2018). This data-driven practice referred to as *data-driven computing* showed good convergence, especially in comparison to a classical finite element model analysis. Further extension to viscoelastoplasticity Chinesta et al., 2017, static nonlinear elasticity problem within the finite deformation theory by L. T. K. Nguyen and Keip (2018). Similarly, a modified framework for small-strain elasticity called *Data-Driven Identification* proposed by Leygue et al. (2018) has been combined *data-driven computing* solver and *Data-Driven Identification* algorithm which is applicable to non-linear elastic behaviours only. Moreover, modelling errors and uncertainty are discarded by avoiding material modelling empiricism, likewise for the loss of experimental information. However, it is sensitive to noise and/or outliers in the dataset. Thus, developments for solving these issues are explained in (T. Kirchdoerfer and Ortiz, 2017), solver improvements (Ayensa-Jiménez et al., 2018; Kanno, 2018; L. T. K. Nguyen, Aydin, and Cyron, 2022). In terms of the time-dependent problem, a new solver based on both distance-minimizing and entropy-maximizing schemes is expressed in (Eggersmann et al., 2019; T. Kirchdoerfer and Ortiz, 2018). Still, this method also contains its own errors, and in order to obtain precision of constraints that affected directly the convergence of the algorithm, a vast amount of data points is needed. It only can be solved partly by L. T. K. Nguyen, Aydin, and Cyron (2022).

Closely relevant, Peherstorfer and Willcox (2015, 2016) constructed reduced-order models from data also without having direct knowledge of the constitutive models by inferring the full-order operators. Likewise, Ibañez et al. (2017) and Ibañez et al. (2016) introduced a more general approach by applying manifold learning methodologies to define a so-called *constitutive manifold*, a low-dimensional embedding for stress-strain pairs, which is a data-based constitutive equation. This type of data-driven method is based on pure data regression, hence posing the risks that violate basic laws of thermodynamics due to noisy data, and the degree of accuracy of approximated results. To overcome this problem, González, Chinesta, and Cueto (2019) present a novel paradigm, also without employing constitutive equations, uses the GENERIC (General Equation for Non-Equilibrium Reversible-Irreversible Coupling) framework which guarantees the thermodynamical consistency. By not adopting the constraints from the material theory (i.e. not applying the material constitutive equations), these model-free approaches may demand a large number of datasets in order to generate the constraints. Therefore, it is necessary to determine the least amount of data required for the training process of a specific model K. Wang, Sun, and Du (2019). Therefore, Ibáñez et al. (2019)

presented a hybrid constitutive modeling methodology that combined constitutive equations of plasticity models and experimental data via sparse proper generalized decomposition (s-PGD) technique (Ibáñez et al., 2018). By exploiting the benefit of contained constitutive equations, this hybrid model produced the error convergence more accurately and efficiently than the ones without a model (Ibáñez Pinillo, 2019).

1.2.2 Machine learning method

Another typical type of data-driven method is Artificial Neural Networks (ANN), which has experienced tremendous success in wide-ranging fields of material modelling including the construction of neural network constitutive models (NNCMs) (Hoerig, Ghaboussi, and Insana, 2019; Hoerig, Ghaboussi, and Insana, 2017, 2018), reduced order model (Hesthaven and Ubbiali, 2018), solving and learning systems involving ODEs and PDEs (Long et al., 2017; Qin, K. Wu, and Xiu, 2019; Raissi, Perdikaris, and Karniadakis, 2019; Raissi and Karniadakis, 2018; Raissi, Perdikaris, and Karniadakis, 2017b,c), Gaussian process regression (Raissi, Perdikaris, and Karniadakis, 2017a).

Artificial neural systems are a computational methodology, which attempts to copy the modelling of a human mind with the end goal to figure out how to perform undertakings as humans do. In spite of the fact that neural systems endeavour to imitate the human brain, they are far less perplexing for the real brain. As definition stated by Haykin (2009): A neural network is a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects: (1) knowledge is acquired by the network from its environment through a learning process; (2) interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge. Neural networks perform a variety of tasks such as prediction, pattern classification, or fitting arbitrarily complex nonlinear models to multidimensional data to any desired accuracy. This powerful capability of function fitting can be extended to multivariate techniques including multiple linear regression and nonlinear regression (Samarasinghe, 2016).

This first application of this approach, proposed by Ghaboussi et al. (1998), Ghaboussi J., Garrett J. H., and Wu X. (1991), X. Wu and Ghaboussi (1995), and Yagawa and Okuda (1996) is to model the constitutive relationship of 2-D plane concrete extracted from the experimental data. Specifically, the measured strains serve as inputs, and the applied stresses act as outputs, the information flows through neurons without human action. Once the network is trained, the model can be implemented in a boundary value problem straightforwardly, in exactly the same way as a conventional phenomenological model. Further development expressed in (Ghaboussi et al., 1998; Hashash, Jung, and Ghaboussi, 2004a) described the numerical implementation in Finite Element Analysis, in these works, the material stiffness matrix is unique and represented by the NN model which is an attempt to explain the “black-box”. Other applications for composite (Haj-Ali Rami et al., 2001), strain-softening material models

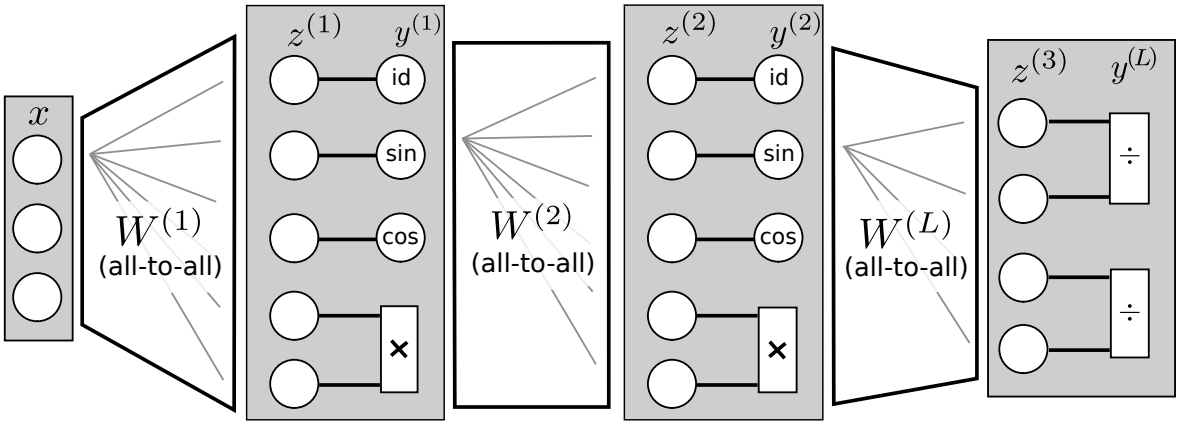
(Kaklauskas Gintaris and Ghaboussi Jamshid, 2001) hyperelastic materials and implemented in commercial software (Shen et al., 2004), as well as other applications (Lefik and Schrefler, 2003; Stoffel, Bamer, and Markert, 2019; K. Wang and Sun, 2018). However, generally, this approach consists of several problems namely a high demand in the training dataset, high dimensionality that leads to inconsistency in prediction performance, and easy getting under-fitting or over-fitting unless carefully calibrated the NN's hyper-parameters.

In order to explain the "black-box" of ANN and extract the knowledge to show an explicit relationship between input and output, many tools in the field of data mining have been developed to provide human-understandable syntax for a trained ANN model. Those knowledge extraction algorithms from the feed-forward ANN can be classified as three catalogues: decomposition, pedagogical, and eclectic (Young and Weckman, 2010). The decomposition approaches to analyse the entire internal structures of the ANN (activation functions, weights, neurons), then concludes the significance of the features or set rule-based form equations ("If-Then" or "M-of-N") (Huysmans, Baesens, and Vanthienen, 2006; Setiono and H. Liu, 1997; Towell and Shavlik, 1993). In contrast, the pedagogical measurements ignore all the internal components and only investigate the input-output relationship based on the information obtained from the original ANN model. The rules extracted have the form of decision trees or to quantify input significance (Breiman, 2017; Rangwala, 2006; Zhou, Jiang, and Chen, 2003). The eclectic one is a combination of the two mentioned approaches which are rare in practice and lie in the middle on the translucency spectrum, (Hruschka and Ebecken, 2006; Mohamed, 2011; Thrun, 1995). However, most of the algorithms above are the application of classification problems while engineering interests fall in regression sub-fields. Moreover, the rule-based form equations hardly show the ability to express explicitly the governing laws of materials laws. Although many efforts have been reported (Chan, 2017; Odajima et al., 2008; Saito and Nakano, 2002; Setiono and Thong, 2004) in which the authors used the piecewise linear artificial neural network (PWL-ANN), the resulted equations have a form of multiple linear regressions which create a complicated system of equations. Additionally, the core of PWL-ANN is to try to approximate the sigmoid activation function which is outdated.

Recently, new methods reported by Montavon, Samek, and K.-R. Müller (2018) applied for interpreting the Deep ANN using Layer-wise relevance propagation, and Taylor decomposition is only used to evaluate the influence of input features.

Some other works involved to extract the symbolic equations are (Martius and Lampert, 2016; Sahoo, Lampert, and Martius, 2018), in which the authors employ a general feed-forward network with modification in activation called EQL, it resembles *sum-product networks (SPNs)* (Poon and Domingos, 2011) and *Pi-Sigma networks (PSNs)* (Shin and Ghosh, 1991). EQL can learn a function from experimental data, and generalize different regions of data space even extrapolate to unseen parts. The extension ($EQL^{\nabla\cdot}$ shown in Figure 1.4) carried by Sahoo, Lampert, and Martius (2018) covered the limitations of EQL namely not able to present divisions, improve unreliable identification of the true functional relation. Notably,

Figure 1.4 Network architecture of Equation Learner $EQL^{\nabla\cdot}$ (Sahoo, Lampert, and Martius, 2018))



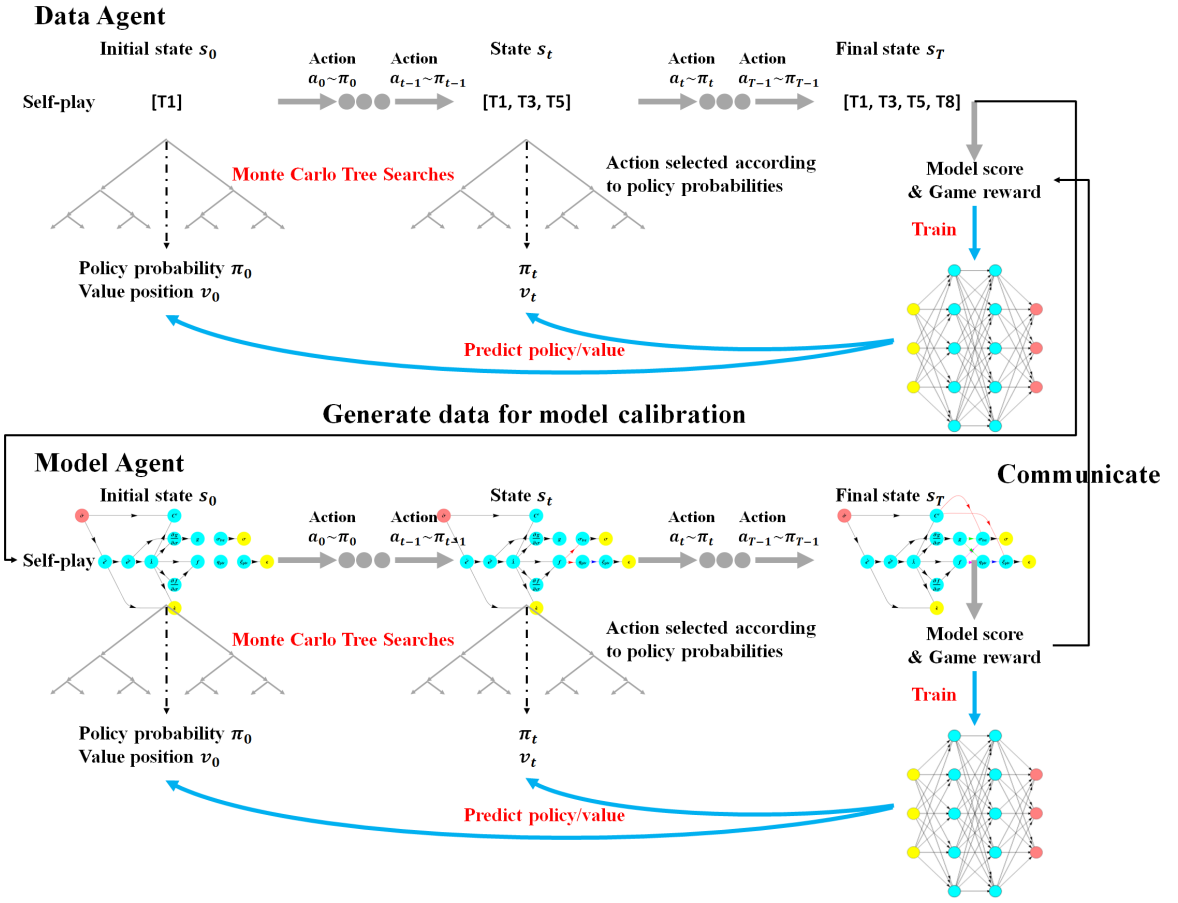
in the case of optimization problems that are not trivial and contain many local minima, the algorithm only produces approximation solutions and exhibits poor extrapolated ability eventually.

Regarding the hybrid method, for the first time, K. Wang and Sun (2019) and K. Wang, Sun, and Du (2019) introduced a multi-agent meta-modelling game to generate datasets, and scheme that makes predictions of the constitutive relationship of elastoplastic materials. Two discrete mathematics routines namely directed multigraph and decision tree learning are the cores of this practice. Artificial intelligence (AI) data agent generates data from real or simulation experiments and the AI modeller agent evaluates all objective functions of possible labelled directed multi-graphs (presented for the constitutive laws) that link the source (i.e. strain history) to the target (i.e. stress) in order to indicate the optimal path (Figure 1.5). This process resembles a more human-like continuous decision-making procedure in which constitutive laws is simplified in term of forming graph edges and the goal is to maximize the model score. Through a deep reinforcement learning framework (Sutton, 1992) which operated an automated trial-and-error process, the computer agent is able to self-improve constitutive laws effectively (similar to the algorithm of AlphaGo Zero Silver et al., 2017). During the process, the AI agents optimised their decisions automatically without human intervention. Hence, applications of graph theories such as directed graphs in meta-modelling provide us with hierarchical information on the connections among events and mechanisms.

1.2.3 Symbolic regression method

Most of the problems in the engineering fields are related to uncovering equations for the observations also known as *symbolic regression* tasks where the evolution computational perform on a curtain function space. Both the form of the equation and its corresponding parameters are searched simultaneously which differs from the traditional linear and nonlinear regression methods that need a given equation to fit parameters. Schmidt and Lipson (2009) have discovered the physical laws such as invariants and conserved quantities in the form

Figure 1.5 Multi-player interactive deep reinforcement learning for generating optimal strategy to automate the modelling, calibration, and validation of an elastoplastic model by K. Wang, Sun, and Du, 2019



of analytical expressions without any prior knowledge, although computational resources exploded for high-dimensional problems.

Recently, Brunton, Proctor, and Kutz (2016) proposed a promising direction, so-called sparse identification of nonlinear dynamics (SINDy), attempting to discover the governing equations from noised experimental data of nonlinear dynamics problems. Standing on the foundation of a seminal breakthrough in (Bongard and Lipson, 2007; Schmidt and Lipson, 2009), this treatment combines spare regression (Hastie, Tibshirani, and Friedman, 2009; Tibshirani, 1996) with genetic algorithm (Koza and Koza, 1992) to select the most accurate representation of data from a pool of simple functions and partial derivatives. The sparsity technique illustrated the abilities of highly efficient computation and avoiding over-fitting by selecting parsimonious models that balance model accuracy with complexity via Pareto analysis also in biological networks(Mangan et al., 2016). Further development was proposed by Rudy et al. (2017) to deal with spatiotemporal, high-dimensional data using innovative sampling strategies. Other improvements of selecting candidates models comparing the statistical criterion AIC/BIC

Mangan et al., 2017. However, there is an outstanding issue in the appropriate selections of the sparsifying function basis and measurement coordinates. In practice, those selections are usually based on the knowledge of users regarding the physical characteristics of the systems. Therefore, to tackle those issues, Champion et al. (2019) designed an auto-encoder, which is based on deep neural networks, to discover a coordinate transformation thus enabling the SINDy method to learn the governing equations and their corresponding coordinate systems successfully. Numerous applications have been reported in (Bhat, 2019; Kaiser, Kutz, and Brunton, 2018; S. Li et al., 2019; X. Li et al., 2019; Quade et al., 2018)

1.3 Thesis scope and overview

Among the three main methods, the model-free approach does not present explicit models or hypotheses to interpret outcomes. Plus, it comes at the cost of demanding an extensive volume of data. Symbolic regression, on the other hand, may generate lengthy expressions that pose challenges, particularly in high-dimensional problems. Additionally, the integration of established *prior* knowledge remains unaddressed in existing works. In contrast, the machine learning approach offers the potential for indirect interpretation and the incorporation of physical prerequisites. Weber, Geiger, and Wagner (2021) enforced constraints related to energy conservation, normalization, and material symmetries.

The core focus of this thesis revolves around the advancement of a modified Deep Neural Network (DNN) capable of extracting insights from experimental data. The overarching goal is to formulate empirical material laws rooted in centuries of scientific understanding, which trace back to the second and third paradigms. The objective is to reduce the risks of violating basic principles of physical laws and alleviate the computational burden associated with searching through enormous data ranges.

The approach taken involves several key steps:

- A tailored version of DNN for identifying governing equations. This version is designed to satisfy the essential physical requirements of commonly used material laws.
- A sufficient number of Finite Element Analysis (FEA) simulations, ideally conducted using the `FEniCSx` framework (Logg and Wells, 2010b), generate the required input data.
- The central modifications to the DNN are twofold: (a) the activation functions are adjusted to ensure the final functional form satisfies the mathematical implications of physically motivated requirements, and (b) the loss function is constructed based upon well-known generalized material theories such as Baker-Ericksen inequalities and Hill's inequalities (Baker and Ericksen, 1954; Hill, 1970).

The inputs and outputs of the DNN correspond to squared principal stretches and principal strains, respectively. Once the DNN model is appropriately trained, it is integrated into

the `FEniCSx` framework, effectively transforming the Finite Element Analysis tool into an intelligent module. The anticipated outcome of this endeavour is a `FEniCSx` module capable of predicting strain-stress fields in hyperelastic simulations. Notably, the process involves the systematic recording of strengths and limitations, with the overarching aim of analyzing the applicability range of interpretable DNN models within the realm of computational mechanics.

Chapter 2

Nonlinear isotropic elastic materials

Contents

2.1 Introduction

Building on the historical perspective and profound understanding of finite elasticity established in the mid-20$^{\text{th}}$ century, this chapter addresses nonlinear isotropic elastic materials and their behaviour under large deformations and rotations. Traditional linear elasticity theory often is unable to treat large deformations and rotations, necessitating the development and refinement of more sophisticated models and methodologies. The scope of this study is confined to purely mechanical theories, thereby excluding thermodynamic variables such as temperature and entropy.

Over the years, profound theoretical findings in this field, many validated by experimental data, have led to a significant increase in the application of various rubber-like materials such as synthetic elastomers, polymers, biological tissues, and natural rubber in diverse fields. The foundations set for finite elasticity now support important new fields of study. These include the elasticity of biological materials, the mechanics of constrained continua, theories of rods and shells, microstructural mechanics, and the theory of homogenization, data-driven constitutive modelling, among others.

This chapter prepares the theory of the constitutive equations in elastic materials, outlining the fundamental physical requirements and their corresponding mathematical consequences. By systematically considering these requirements, the scope of material under investigation is narrowed down to compressible non-linear isotropic materials. To begin, the chapter discusses the kinematics of finite deformation (section 2.2), detailing the concepts of motions, and deformations. Subsequently, an analysis of strain tensors and spectral representation of strain states are performed, providing an understanding of geometric changes that occur during deformation.

Next, section 2.3 reviews stress analysis, balance laws, and stress tensors. The spectral representation of the stress state is presented, providing a foundation for understanding the stress developments induced under strain formulated in principal space.

Following that, the stress-strain relation of nonlinear isotropic elastic materials is studied, with a focus on the physically driven requirements (section 2.4). Herein, the principles of material frame indifference and material symmetry transformations are discussed. Moreover, the concept of isotropic elastic material and undistorted configurations are introduced, alongside an exploration of the growth conditions in large strains. The two most common physically driven requirements, namely Baker–Ericksen and Hill's constitutive inequalities, are analysed, shedding light on their connections with the mathematically driven constitutive requirements.

Overall, the aim of this chapter is to outline requirements to facilitate the process of identifying material laws that satisfy physical plausibility. To conclude this chapter, an example of an ideal material candidate that satisfies all the requirements, namely hyperelastic materials, is

presented (section 2.4.9). Examples of compressible stretch-based hyperelastic models are also provided.

2.2 Kinematical description of finite deformation

2.2.1 Motions

A body $\mathcal{B}$ is defined as a three-dimensional (3D) differentiable manifold, comprised of a collection of material points, referred to as *continuum particles*[1]. The location of each particle can be defined within a coordinate system, denoted by triples of real numbers $(x_1, x_2, x_3) \in \mathbb{R}^3$. The 3D Euclidean space, denoted as $\mathbb{R}^3$, composed of a fixed origin O and three orthogonal basis vectors $\boldsymbol{e}_a$, with $a = 1, 2, 3$. The compact domain $\Omega_0 \in \mathbb{R}^3$, with boundary $\partial\Omega_0$, represents the reference configuration of the body $\mathcal{B}_0$ at a specific time instant $t = t_0$. The current configuration of the body, $\mathcal{B}_t$, is occupied by the domain Ω_t, with its associated boundary denoted as $\partial\Omega_t$. A material particle Q in the reference configuration is identified by its position vector $\boldsymbol{X}(Q)$. In the presence of finite deformation, as illustrated in Fig. 2.1, a motion $\boldsymbol{\chi}$ transports the material point Q from its location in the reference configuration $\mathcal{B}_0$ to an updated position in the current configuration $\mathcal{B}_t$, forming a trajectory; Q is considered to have undergone a *displacement*, or a shift in position. Then, relative to the coordinate system, the position of Q in the current configuration is described by the vector function $\boldsymbol{x}(Q)$. This can be represented as:

$$\boldsymbol{x} = \boldsymbol{\chi}(\boldsymbol{X}, t), \quad \boldsymbol{X} = \boldsymbol{\chi}^{-1}(\boldsymbol{x}, t), \tag{2.1}$$

where $\boldsymbol{\chi}^{-1}$ denotes the inverse of the deformation mapping $\boldsymbol{\chi}$ at any fixed time t. It is important to note that $\boldsymbol{\chi}$ is orientation preserving, implying that the physical material is incapable of self-penetration or inverting the orientation of its material coordinates. This property is described per definition by the condition $\det \nabla \boldsymbol{\chi}(\boldsymbol{X}, t) > 0$ (Oden, 2011, p. 5).

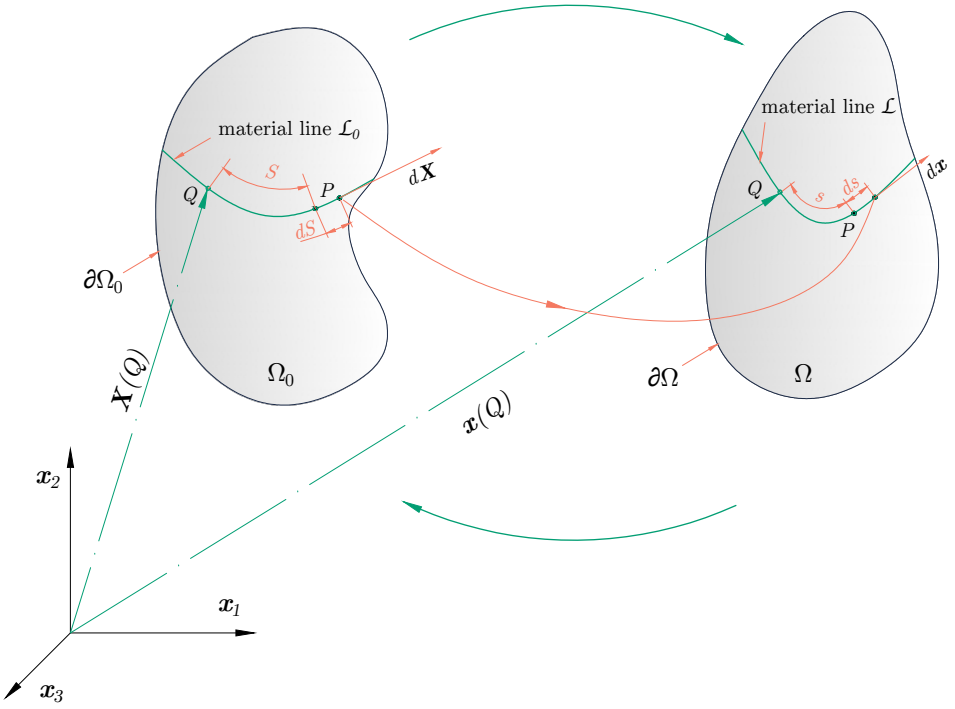
The *velocity* and *acceleration* of the particle Q are defined respectively by

$$\dot{\boldsymbol{x}}(\boldsymbol{X}, t) = \frac{\partial \boldsymbol{x}}{\partial t}(\boldsymbol{X}, t)\,, \qquad \ddot{\boldsymbol{x}} = \frac{\partial \dot{\boldsymbol{x}}}{\partial t}(\boldsymbol{X}, t)\,. \tag{2.2}$$

Field quantities expressed in terms of the reference configuration $(\boldsymbol{X}, t)$ are said to be in the *material* (or referential) description. In contrast, field quantities expressed in terms of the current configuration $(\boldsymbol{x}, t)$ are said to be in the *spatial* (or current) description. Since we analyse the deformation of the body at some fixed time t, we can omit t in (2.1) and

[1]The term "continuum particle" relates to a group of numerous molecules that, despite their large quantity, are sufficiently small to be regarded as a singular entity. This contrasts with a mass point in Newtonian mechanics or an individual atom/molecule. (Holzapfel, 2000).

Figure 2.1 The material line $\mathcal{L}_0$ is transformed from reference configuration Ω_0 to $\mathcal{L}$ in Ω_t.



write:

$$\boldsymbol{x} = \boldsymbol{\chi}(\boldsymbol{X}), \quad \boldsymbol{X} = \boldsymbol{\chi}^{-1}(\boldsymbol{x})\,. \tag{2.3}$$

Therefore, the *displacement* field of the continuum body reads:

$$\boldsymbol{u}(\boldsymbol{X}) = \boldsymbol{\chi}(\boldsymbol{X}) - \boldsymbol{X}\,. \tag{2.4}$$

The deformation of the continuum body in the vicinity of a point $\boldsymbol{X}$ is characterized by the *deformation gradient* $\boldsymbol{F}$, defined as:

$$\boldsymbol{F} = \frac{\partial \boldsymbol{\chi}(\boldsymbol{X})}{\partial \boldsymbol{X}} = \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} = \boldsymbol{I} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}}\,, \tag{2.5}$$

where $\boldsymbol{I}$ is the identity tensor. Since the motion is orientation preserving, $\boldsymbol{F}$ is a non-singular tensor, which implies that the local ratio of the current to reference volume of a differential material volume element bounding the $\boldsymbol{X}$ is:

$$J = \det \boldsymbol{F} > 0\,. \tag{2.6}$$

2.2.2 Deformations

As depicted in Fig. 2.1, the tangent element $d\boldsymbol{X}$ of a material line $\mathcal{L}_0$ in Ω_0 is transformed into the tangent element $d\boldsymbol{x}$ of its deformed image line $\mathcal{L}$ in Ω_t by deformation gradient tensor $\boldsymbol{F}$. This transformation is described by:

$$d\boldsymbol{x} = \boldsymbol{F} d\boldsymbol{X}. \tag{2.7}$$

Consider $|d\boldsymbol{X}| = dS$ and $|d\boldsymbol{x}| = ds$ as the undeformed and deformed lengths, where S and s represent the arc length parameters of $\mathcal{L}_0$ and $\mathcal{L}$, respectively. The *stretch* λ, which is defined as the ratio of deformed to undeformed lengths, can be expressed as:

$$\lambda = \frac{ds}{dS}. \tag{2.8}$$

Consequently, (2.7) can be reformulated as:

$$\lambda \boldsymbol{e} = \boldsymbol{F} \boldsymbol{e}_0 \,. \tag{2.9}$$

Here, $\boldsymbol{e} = d\boldsymbol{x}/ds$ and $\boldsymbol{e}_0 = d\boldsymbol{X}/dS$ denote the unit vectors tangent to $\mathcal{L}$ and $\mathcal{L}_0$ at $\boldsymbol{x}$ and $\boldsymbol{X}$, respectively. The squared length of the tangent element in the reference configuration can be expressed as follows:

$$dS^2 = |d\boldsymbol{X}|^2 = d\boldsymbol{X}^\top d\boldsymbol{X} = (\boldsymbol{F}^{-1} d\boldsymbol{x})^\top (\boldsymbol{F}^{-1} d\boldsymbol{x}) = d\boldsymbol{x}^\top (\boldsymbol{F}\boldsymbol{F}^\top)^{-1} d\boldsymbol{x}, \tag{2.10}$$

whereas, in the current configuration:

$$ds^2 = |d\boldsymbol{x}|^2 = d\boldsymbol{x}^\top d\boldsymbol{x} = d\boldsymbol{X}^\top \boldsymbol{F}^\top \boldsymbol{F} d\boldsymbol{X}. \tag{2.11}$$

The right and left Cauchy-Green deformation tensors are introduced as:

$$\boldsymbol{C} = \boldsymbol{F}^\top \boldsymbol{F}\,; \quad \boldsymbol{b} = \boldsymbol{F}\boldsymbol{F}^\top \,. \tag{2.12}$$

Both $\boldsymbol{C}$ and $\boldsymbol{b}$ are *symmetric* and *positive definite* at each $\boldsymbol{X} \in \Omega_0$ and each $\boldsymbol{x} \in \Omega_t$, respectively. These tensors embody the more general and physically useful *polar decomposition theorem* (Truesdell and Noll, 2004, p. 52) when applied to the deformation gradient tensor $\boldsymbol{F}$:

$$\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U} = \mathbf{v}\boldsymbol{R}\,. \tag{2.13}$$

The rigid body rotation of a material point is characterized by the proper orthogonal tensor $\boldsymbol{R}$ ($\boldsymbol{R} \in SO(3)$[1]). The deformations are described by the unique, positive, symmetric tensors $\boldsymbol{U}$ and $\mathbf{v}$, which are known as the *right* and *left stretch tensors*, respectively.

It is usually difficult [2] to compute $\boldsymbol{U}$ and $\mathbf{v}$ through the polar decomposition, so they are often obtained from their squares instead. By utilising the *Square-Root theorem* (see Gurtin (1981, p. 13); Oden (2011, p. 19)), the tensors $\boldsymbol{U}$ and $\mathbf{v}$ can also be presented as:

$$\boldsymbol{U}^2 = \boldsymbol{U}\boldsymbol{U} = \boldsymbol{C}\,, \tag{2.14}$$

$$\mathbf{v}^2 = \mathbf{v}\mathbf{v} = \boldsymbol{b}\,. \tag{2.15}$$

Using the polar decomposition (2.13), with $\boldsymbol{R}^{-1} = \boldsymbol{R}^\top$ or $\boldsymbol{R}^\top\boldsymbol{R} = \boldsymbol{I}$, the relation between $\boldsymbol{U}$ and $\mathbf{v}$ can be found

$$\mathbf{v} = \boldsymbol{F}\boldsymbol{R}^{-1} = \boldsymbol{F}\boldsymbol{R}^\top = \boldsymbol{R}\boldsymbol{U}\boldsymbol{R}^\top\,, \tag{2.16}$$

By (2.13), (2.15), and (2.16), the important relations of $\boldsymbol{C}$ and $\boldsymbol{b}$ are revealed, namely

$$\boldsymbol{b} = \mathbf{v}^2 = \boldsymbol{R}\boldsymbol{U}\boldsymbol{R}^\top\boldsymbol{R}\boldsymbol{U}\boldsymbol{R}^\top = \boldsymbol{R}\boldsymbol{U}^2\boldsymbol{R}^\top = \boldsymbol{R}\boldsymbol{C}\boldsymbol{R}^\top\,. \tag{2.17}$$

2.2.3 Strain tensors

Generalised strain measures, initially introduced by Seth (1961, 1966) and further investigated by Hill (1968), have played a vital role in the development of material constitutive equations for various types of elastic materials, such as hyperelasticity, nonlinear viscoelasticity, and rubber-like elasticity (cf. (R. W. Ogden, 1997, p. 89–91, 118, 156, 159); (Holzapfel, 2000, p. 88); (Beex, 2019; Mihai et al., 2017)).

The expressions for these strain measures in material description are respectively given by:

$$\boldsymbol{E}^{(m)} = \begin{cases} \frac{1}{m}(\boldsymbol{C}^{m/2} - \boldsymbol{I})\,; & \text{if} \quad m > 0\,, \\ \frac{1}{2}\ln\boldsymbol{C}\,; & \text{if} \quad m = 0\,. \end{cases} \tag{2.18}$$

[1] $SO(3)$ is the special orthogonal group comprising orthogonal tensors $\boldsymbol{Q}$ satisfying $\boldsymbol{Q}^\top\boldsymbol{Q} = \boldsymbol{I}$ and $\det\boldsymbol{Q} = 1$. It represents proper rotation transformations in three-dimensional space, preserving both distances and orientations. Specifically, when $\mathbf{Q}$ corresponds to a rotation, it is referred to as a proper orthogonal tensor, whereas if it represents a reflection, it is considered an improper orthogonal tensor (Holzapfel, 2000, p. 16).

[2] See Higham and Noferini (2016) for numerical computation of polar decomposition (for 3×3 matrix) without using the singular value decomposition.

For the spatial description, the expressions of the generalised strain follows:

$$\boldsymbol{e}^{(m)} = \begin{cases} \frac{1}{m}(\boldsymbol{I} - \boldsymbol{b}^{-m/2})\,; & \text{if} \quad m < 0\,, \\ \frac{1}{2}\ln \boldsymbol{b}\,; & \text{if} \quad m = 0\,. \end{cases} \tag{2.19}$$

Here, $m \in \mathbb{Z}$ (Morman, 1986), where distinct values of m correspond to different strain tensors:

- $m = 2, \quad \boldsymbol{E}^{(2)} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I})$, Green-Lagrangian strain tensor (R. W. Ogden, 1997, p. 89-90)
- $m = 1, \quad \boldsymbol{E}^{(1)} = \boldsymbol{C}^{1/2} - \boldsymbol{I} = \boldsymbol{U} - \boldsymbol{I}$, Biot strain tensor (Biot, 1965)
- $m = 0, \quad \boldsymbol{E}^{(0)} = \ln \boldsymbol{C}^{1/2} = \ln \boldsymbol{U}$, Hencky[1] (logarithmic, true) strain tensor (Hencky, 1928)[2].
- $m = -2, \quad \boldsymbol{e}^{(-2)} = \frac{1}{2}(\boldsymbol{I} - \boldsymbol{b}^{-1})$, Euler-Almansi strain tensor (Holzapfel, 2000, p. 81).

The material is *unstrained* at $\boldsymbol{X}$, i.e. $\boldsymbol{E}^{(m)} = \boldsymbol{0}$, when $\boldsymbol{U} = \mathbf{v} = \boldsymbol{I}$. Moreover, $\boldsymbol{E}^{(m)}$ is an isotropic tensor expressed in term of $\boldsymbol{C}$ for $m > 0$, and of $\boldsymbol{b}$ for $m < 0$ respectively. Proof of this property is presented in Section 2.2.4.

The logarithmic strain tensors, $m = 0$, hold particular interest in non-linear constitutive theories because of their asymptotic and monotonic properties, it is also found useful in the theory of plasticity (Xiao, Bruhns, and Meyers, 1997). They have been extensively studied in literature such as Bruhns, Xiao, and Meyers (2001), Hill (1968), Hoger (1987), Neff, Eidel, and R. J. Martin (2016), Neff, Ghiba, and Lankeit (2015), and Seth (1961, 1966).

2.2.4 Spectral representation of strain state

Constitutive relations in finite elasticity, especially in cases of soft tissue, hyperfoam, and rubber-like materials are usually formulated in terms of principal stretches. Furthermore, they may be more intuitive to formulate the boundary value problems of the stretch-based elastic model in principal space. This subsection provides the foundation of the strain tensors in spectral representation.

Let $\boldsymbol{N}_a$ denote a set of mutually orthogonal and normalized eigenvectors of the material stretch tensor $\boldsymbol{U}$, with corresponding eigenvalues $\{\lambda_a\}$, for $a = 1, 2, 3$:

$$\boldsymbol{U}\boldsymbol{N}_a = \lambda_a \boldsymbol{N}_a \qquad \text{with} \qquad |\boldsymbol{N}_a| = 1\,, \tag{2.20}$$

where $|\cdot|$ denotes the norm of a vector.

[1] According to Neff, Münch, and R. Martin (2016), the logarithmic strain tensor was actually first introduced by Becker (1893), which was published 35 years before Hencky's works. It will be referred to as the logarithmic strain tensor in the following.

[2] A newly typeset version can be found in Neff, Münch, and R. Martin (2016), see also Neff, Eidel, and R. Martin (2014) for a recent English translation.

By combining (2.14) with (2.20), the eigenvalue problem for $\boldsymbol{C}$ is obtained

$$\boldsymbol{C}\boldsymbol{N}_a = \boldsymbol{U}^2\boldsymbol{N}_a = \lambda_a^2\boldsymbol{N}_a\,, \qquad a = 1,2,3\,. \tag{2.21}$$

The eigenvalues $\{\lambda_1, \lambda_2, \lambda_3\}$ of the right stretch tensor $\mathbf{U}$ are referred to as the *principal stretches* (Holzapfel, 2000, p. 89), corresponding to the three principal material lines. Conversely, the eigenvalues of the symmetric tensor $\boldsymbol{C}$ correspond to the squares of the principal stretches, denoted as λ_a^2. Note that both $\boldsymbol{U}$ and $\boldsymbol{C}$ share the same orthogonal eigenbasis, namely the set $\{\boldsymbol{N}_a\}$, which represents the *principal referential directions*.

The eigenvalue problem for $\mathbf{v}$ can be expressed using (2.16) as follows (Holzapfel, 2000, p. 90):

$$\mathbf{v}(\boldsymbol{R}\boldsymbol{N}_a) = \boldsymbol{R}\boldsymbol{U}\boldsymbol{R}^\top(\boldsymbol{R}\boldsymbol{N}_a) = \boldsymbol{R}\boldsymbol{U}\boldsymbol{N}_a = \lambda_a(\boldsymbol{R}\boldsymbol{N}_a)\,. \tag{2.22}$$

Therefore, the eigenvectors of $\mathbf{v}$, given by $\boldsymbol{R}\boldsymbol{N}_a$, correspond to the rotated versions of the eigenvectors of $\boldsymbol{U}$ and $\boldsymbol{C}$ under $\boldsymbol{R}$. These eigenvectors are known as the *principal spatial directions*, denoted as $\boldsymbol{n}_a$, and satisfy:

$$\boldsymbol{n}_a = \boldsymbol{R}\boldsymbol{N}_a \qquad \text{with} \qquad |\boldsymbol{n}_a| = 1\,, \qquad a = 1,2,3\,. \tag{2.23}$$

Hence, the two-point tensor $\boldsymbol{R}$ rotates the unit orthogonal eigenvectors $\boldsymbol{N}_a$, embedded into the reference configuration, into the corresponding unit orthogonal eigenvectors $\boldsymbol{n}_a$, embedded into the deformed configuration (Curnier and Rakotomanana, 1991).

Similarly, combining (2.15), the eigenvalue problem for $\boldsymbol{b}$ can be formulated as:

$$\boldsymbol{b}(\boldsymbol{R}\boldsymbol{N}_a) = \mathbf{v}^2(\boldsymbol{R}\boldsymbol{N}_a) = \boldsymbol{R}\boldsymbol{U}^2\boldsymbol{R}^\top(\boldsymbol{R}\boldsymbol{N}_a) = \lambda_a^2(\boldsymbol{R}\boldsymbol{N}_a) = \lambda_a^2\boldsymbol{n}_a\,. \tag{2.24}$$

This equation demonstrates that $\boldsymbol{b}$ also shares the same eigenvectors as $\mathbf{v}$. In addition, they can be obtained by rotating the corresponding principal spatial directions of $\boldsymbol{U}$ and $\boldsymbol{C}$ using $\boldsymbol{R}$ as indicated in (2.23).

The principal stretches are typically obtained through their squares, as the squares of these principal stretches correspond to the eigenvalues of $\boldsymbol{C}$ and $\boldsymbol{b}$. This can be done by applying the *spectral decomposition* (see Gurtin (1981, p. 11)) theorem[1] to positive definite tensors, $\boldsymbol{C}$ and $\boldsymbol{b}$ become (for distinct eigenvalues case)

$$\boldsymbol{C} = \boldsymbol{U}^2 = \sum_{a=1}^{3} c_a\,\boldsymbol{N}_a \otimes \boldsymbol{N}_a = \sum_{a=1}^{3} \lambda_a^2\,\boldsymbol{N}_a \otimes \boldsymbol{N}_a\,, \tag{2.25}$$

$$\boldsymbol{b} = \mathbf{v}^2 = \sum_{a=1}^{3} b_a\,\boldsymbol{n}_a \otimes \boldsymbol{n}_a = \sum_{a=1}^{3} \lambda_a^2\,\boldsymbol{n}_a \otimes \boldsymbol{n}_a\,, \tag{2.26}$$

[1] An algorithm for symbolically computing the spectral decomposition of 3x3 matrices. Habera and Zilian (2021)

where $\otimes$ denotes the tensor product, and $c_a = b_a = \lambda_a^2 > 0$ represent the eigenvalues of $\boldsymbol{C}$, $\boldsymbol{b}$, respectively.

The principal invariants of $\boldsymbol{C}$, denoted I_1, I_2, I_3, are as follows (Holzapfel, 2000, p. 25):

$$I_1(\boldsymbol{C}) = \operatorname{tr}(\boldsymbol{C}) = c_1 + c_2 + c_3 = \lambda_1^2 + \lambda_2^2 + \lambda_3^2, \tag{2.27}$$

$$I_2(\boldsymbol{C}) = \frac{1}{2}\left[(\operatorname{tr}(\boldsymbol{C}))^2 - \operatorname{tr}(\boldsymbol{C}^2)\right] = c_1c_2 + c_1c_3 + c_2c_3 = \lambda_1^2\lambda_2^2 + \lambda_2^2\lambda_3^2 + \lambda_3^2\lambda_1^2, \tag{2.28}$$

$$I_3(\boldsymbol{C}) = \det \boldsymbol{C} = J^2 = c_1c_2c_3 = \lambda_1^2\lambda_2^2\lambda_3^2. \tag{2.29}$$

The two-point tensors $\boldsymbol{F}$ and $\boldsymbol{R}$ are generally non-symmetric and may not have a spectral presentation. However, one can employ (2.23), (2.13), and (2.25) to express them in terms of principal stretches and principal directions as follows:

$$\boldsymbol{R} = \boldsymbol{R}\boldsymbol{I} = \sum_{a=1}^{3} (\boldsymbol{R}\boldsymbol{N}_a) \otimes \boldsymbol{N}_a = \sum_{a=1}^{3} \boldsymbol{n}_a \otimes \boldsymbol{N}_a \,, \tag{2.30}$$

where $\boldsymbol{N}_a \otimes \boldsymbol{N}_a = \boldsymbol{I}$ is used. Then

$$\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U} = \boldsymbol{R}\sum_{a=1}^{3} \lambda_a (\boldsymbol{N}_a \otimes \boldsymbol{N}_a) = \sum_{a=1}^{3} \lambda_a (\boldsymbol{R}\boldsymbol{N}_a) \otimes \boldsymbol{N}_a = \sum_{a=1}^{3} \lambda_a \left(\boldsymbol{n}_a \otimes \boldsymbol{N}_a\right). \tag{2.31}$$

Furthermore, $\{\lambda_a\}$ are the roots of the characteristic polynomial

$$p(\lambda_a^2) = \lambda_a^6 - \lambda_a^4 I_1(\boldsymbol{C}) + \lambda_a^2 I_2(\boldsymbol{C}) - I_3(\boldsymbol{C}) = 0 \qquad a = 1, 2, 3\,. \tag{2.32}$$

Morman (1986) showed the solutions of this equation as (see Habera and Zilian (2021) for general case of 3×3 matrices):

$$\lambda_a = \frac{1}{\sqrt{3}}\left[I_1 + 2\left(I_1^2 - 3I_2\right)^{1/2} \cos\frac{1}{3}(\phi + 2\pi a)\right]^{1/2}, \tag{2.33}$$

or

$$c_a = b_a = \lambda_a^2 = \frac{1}{3}\left[I_1 + 2\left(I_1^2 - 3I_2\right)^{1/2} \cos\frac{1}{3}(\phi + 2\pi a)\right], \tag{2.34}$$

where

$$\phi = \arccos\left[\frac{2I_1^3 - 9I_1I_2 + 27I_3}{2(I_1^2 - 3I_2)^{3/2}}\right]. \tag{2.35}$$

Let $\boldsymbol{f}(\boldsymbol{C})$ be a tensor function, then the spectral representation is as follows (Morman, 1986; Simo and Taylor, 1991):

$$\boldsymbol{f}(\boldsymbol{C}) = \boldsymbol{C}^{m/2} = \sum_{a=1}^{3} c_a^{(m/2)} \boldsymbol{N}_a \otimes \boldsymbol{N}_a \,. \tag{2.36}$$

Application of the spectral theorem on the $\boldsymbol{C}^{m/2}$ shows:

$$\boldsymbol{C}^{m/2} = \begin{cases} \sum_{a=1}^{3} \lambda_a^m \boldsymbol{N}_a \otimes \boldsymbol{N}_a & \lambda_1 \neq \lambda_2 \neq \lambda_3 \,, \\ \lambda_1^m \boldsymbol{I} + (\lambda_3^m - \lambda_1^m) \boldsymbol{N}_3 \otimes \boldsymbol{N}_3 & \lambda_1 = \lambda_2 \neq \lambda_3 \,, \\ \lambda_1^m \boldsymbol{I} & \lambda_1 = \lambda_2 = \lambda_3 \,, \end{cases} \tag{2.37}$$

in which $\boldsymbol{C}^{m/2}$ is an isotropic tensor function, i.e

$$\boldsymbol{Q}\boldsymbol{C}^{m/2}\boldsymbol{Q}^\top = (\boldsymbol{Q}\boldsymbol{C}\boldsymbol{Q}^\top)^{m/2} \,, \quad \boldsymbol{Q} \in SO(3) \,. \tag{2.38}$$

Proof. Rising the power of $2/m$ to the left hand side of (2.38), and with (2.37), we can obtain

$$\begin{aligned} (\boldsymbol{Q}\boldsymbol{C}^{m/2}\boldsymbol{Q}^\top)^{2/m} &= \sum_{a=1}^{3} (\lambda_a^m)^{2/m} \boldsymbol{Q}\boldsymbol{N}_a \otimes \boldsymbol{N}_a \boldsymbol{Q}^\top \\ &= \boldsymbol{Q} \left[\sum_{a=1}^{3} \lambda_a^2 \boldsymbol{N}_a \otimes \boldsymbol{N}_a \right] \boldsymbol{Q}^\top = \boldsymbol{Q}\boldsymbol{C}\boldsymbol{Q}^\top \,. \end{aligned}$$

□

The isotropic property of $\boldsymbol{C}^{m/2}$ implies the isotropic properties of the generalised strain $\boldsymbol{E}^{(m)}$ (2.18) for the cases of $m \neq 0$.

Proof. For the case of $m > 0$:

$$\boldsymbol{E}^{(m)}(\boldsymbol{C}) = \frac{1}{m}(\boldsymbol{C}^{m/2} - \boldsymbol{I}) \,.$$

$$\begin{aligned} \boldsymbol{Q}\boldsymbol{E}^{(m)}\boldsymbol{Q}^\top &= \boldsymbol{Q} \left[\frac{1}{m}(\boldsymbol{C}^{m/2} - \boldsymbol{I}) \right] \boldsymbol{Q}^\top \\ &= \frac{1}{m}(\boldsymbol{Q}\boldsymbol{C}^{m/2}\boldsymbol{Q}^\top - \boldsymbol{Q}\boldsymbol{I}\boldsymbol{Q}^\top) \\ &= \frac{1}{m}(\boldsymbol{Q}\boldsymbol{C}^{m/2}\boldsymbol{Q}^\top - \boldsymbol{I}) \,, \\ &= \boldsymbol{E}^{(m)}(\boldsymbol{Q}\boldsymbol{C}\boldsymbol{Q}^\top) \qquad \text{(apply (2.37))} \end{aligned}$$

A similar procedure can be used for the case $m < 0$. □

Based on (2.37), the spectral presentation of the generalised strain tensors in a material description can be written as (for distinct eigenvalues):

$$\boldsymbol{E}^{(m)}(\boldsymbol{C}) = \sum_{a=1}^{3} e_a \boldsymbol{N}_a \otimes \boldsymbol{N}_a = \begin{cases} \frac{1}{m} \sum_{a=1}^{3} (\lambda_a^m - 1)\, \boldsymbol{N}_a \otimes \boldsymbol{N}_a\,; & \text{if} \quad m \neq 0\,, \\ \sum_{a=1}^{3} \ln(\lambda_a)\, \boldsymbol{N}_a \otimes \boldsymbol{N}_a\,; & \text{if} \quad m = 0. \end{cases} \tag{2.39}$$

Hence, the principal values of these strain measures are given by:

$$e_a(\lambda_a) = \begin{cases} \frac{1}{m}(\lambda_a^m - 1) & m \neq 0\,, \\ \ln \lambda_a\,, & m = 0\,. \end{cases} \tag{2.40}$$

Note that while (2.40) represents the most commonly used family of generalised strain measures, there are numerous valid options available for selecting a strain measure's function, $f(\lambda_a)$. However, the exact conditions for $f(\lambda_a)$ have not reached a consensus.

Hill (1968) requires $f(\lambda_a)$ to be any sufficiently smooth monotone function, satisfying $f(1) = 0$, $\frac{\partial f(\lambda_a)}{\partial \lambda_a}(1) = 1$. R. W. Ogden (1997, p. 118) imposes additional requirements that $f(\lambda_a)$ to be infinitely differentiable and $\frac{\partial f}{\lambda_a} > 0, \forall \lambda_a \in (0, \infty)$ to hold (Neff, Eidel, and R. J. Martin, 2016). Therefore, appropriate strain measurements should be selected for each material in order to obtain a simplified form of the underlying stress-strain relation (Truesdell and Noll, 2004, p. 348). This property allows for the condensation of the nonlinear effects of deformation into the definition of strain and reduces the reliance on representing the nonlinear characteristics of the constitutive equations (Morman, 1986). Later in this work, this property will be leveraged to construct a novel activation (subsection 4.3.1) that provides a physically meaningful interpretation for the surrogate model using artificial neural networks.

2.3 Stress analysis

2.3.1 Balance laws

In the previous section, the kinematical aspects of the motion and deformation of a continuum body were discussed. Motion and deformation are the result of interactions between the material particles within the body. One consequence of these interactions is *stress*. The motion and deformation may be the result of the forces that act on a part $\mathcal{P}$ of a body $\mathcal{B}$ with mass M which can be divided into two categories: a distribution of contact force $\boldsymbol{t}_n$ per unit *deformed area*, denoted as a, of the boundary $\partial\mathcal{P}$ of $\mathcal{P}$, and a distribution of body force $\boldsymbol{f}_b$ per unit volume V of $\mathcal{P}$ in current configuration Ω.

The total forces and torques acting on a part $\mathcal{P}$ of a body $\mathcal{B}$ at time t, denoted as $\boldsymbol{\mathcal{F}}(\mathcal{P}, t)$ and $\boldsymbol{\mathcal{T}}(\mathcal{P}, t)$ respectively, are related to the momentum and the moment of momentum of the material points of $\mathcal{B}$ about a fixed point in a reference frame, following Euler's laws of motion

(Beatty, 1987):

$$\boldsymbol{\mathcal{F}}(\mathcal{P},t) = \int_{\partial\mathcal{P}} \boldsymbol{t}_n da + \int_{\mathcal{P}} \boldsymbol{f}_b dV = \frac{d}{dt}\int_{\mathcal{P}} \dot{\boldsymbol{x}} dM\,, \tag{2.41}$$

$$\boldsymbol{\mathcal{T}}(\mathcal{P},t) = \int_{\partial\mathcal{P}} \boldsymbol{x}\times\boldsymbol{t}_n da + \int_{\mathcal{P}} \boldsymbol{x}\times\boldsymbol{f}_b dV = \frac{d}{dt}\int_{\mathcal{P}} \boldsymbol{x}\times\dot{\boldsymbol{x}} dM\,. \tag{2.42}$$

Here, recall (2.2), and $dM = \rho dV$ is the material element of mass with density ρ per unit volume in Ω. The principle of balance of mass also states that $dM = \rho_0 dV_0$, with ρ_0 being the density of mass per unit volume V_0 in reference configuration Ω_0. Thus, the local equation of continuity for the two densities is $\rho = J\rho_0$.

Application of the first law (2.41) to an arbitrary tetrahedral element leads to the *Cauchy's stress theorem: the stress vector* $\boldsymbol{t}_n$ *is a linear transformation of the unit normal vector* $\boldsymbol{n}$ *and the Cauchy's (or true) stress tensor* $\boldsymbol{\sigma}$,

$$\boldsymbol{t}_n = \boldsymbol{\sigma}\boldsymbol{n}\,. \tag{2.43}$$

Combining (2.2), (2.41), (2.42), and (2.43) yields

$$\operatorname{div}\boldsymbol{\sigma} + \boldsymbol{f}_b = \rho\ddot{\boldsymbol{x}}\,, \tag{2.44}$$

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^\top\,, \tag{2.45}$$

They are called *Cauchy's first law and second law of motion* (Beatty, 1987), respectively (see also R. W. Ogden, 1997, p. 148 - 150).

2.3.2 Stress tensors

In solid mechanics, the Cauchy stress indicates the contact force distribution in the current configuration Ω. However, this can be inconvenient since the deformed configuration is often not known *a priori*. Therefore, the *engineering stress tensor* $\boldsymbol{P}$, also known as the *first Piola-Kirchhoff stress tensor*, is introduced to define the contact force distribution $\boldsymbol{t}_n = \boldsymbol{P}\boldsymbol{N}$ (Holzapfel, 2000) per unit *undeformed area* A_0 in the reference configuration Ω_0, where $\boldsymbol{N}$ is the exterior unit normal vector to $\partial\mathcal{P}$ in Ω_0. The relation with the true stress tensor is derived using Nanson's formula as follows:

$$\boldsymbol{P} = J\boldsymbol{\sigma}\boldsymbol{F}^{-\top}\,, \tag{2.46}$$

similar to $\boldsymbol{F}$, $\boldsymbol{P}$ is a two-point tensor and non-symmetric.

For computational approaches, it may be more useful to employ the symmetric positive definite $\boldsymbol{S}$, known as the *second Piola-Kirchhoff stress tensor*. It does not have a direct physical

interpretation in terms of surface tractions, defined as

$$\boldsymbol{S} = J\boldsymbol{F}^{-1}\boldsymbol{\sigma}\boldsymbol{F}^{-\top} \quad \text{and} \quad \boldsymbol{P} = \boldsymbol{F}\boldsymbol{S}\,. \tag{2.47}$$

Note that the work conjugate pairs of the deformation tensor $\boldsymbol{F}$ and Green's strain tensor $\boldsymbol{E}^{(2)} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I})$ are represented by $\boldsymbol{P}$ and $\boldsymbol{S}$, respectively (see (R. W. Ogden, 1997, sect. 3.5) for more details).

Let $\boldsymbol{T}^{(m)}$ denote the generalized stress tensor, which represents the corresponding conjugate stresses (R. W. Ogden, 1997, p. 159) of the generalized strain $\boldsymbol{E}^{(m)}$ (2.18). By collecting the conjugate pairs of the generalized strain tensors, $\boldsymbol{E}^{(m)}$, at each m, the corresponding work conjugate paired stress tensors in **material description** are as follows:

- $m = 2$, $\boldsymbol{T}^{(2)} = \boldsymbol{S}$: 2nd Piola-Kirchhoff stress tensor.
- $m = -2$, $\boldsymbol{T}^{(-2)} = J\boldsymbol{F}^{\top}\boldsymbol{\sigma}\boldsymbol{F}$: weighted convected stress tensor (Farahani and Naghdabadi, 2000).
- $m = 1$, in general, $\boldsymbol{T}^{(1)} = \frac{1}{2}(\boldsymbol{S}\boldsymbol{U} + \boldsymbol{U}\boldsymbol{S})$ (Farahani and Naghdabadi, 2000). However, for special case where $\boldsymbol{T}^{(m)}$ is coaxial with $\boldsymbol{E}^{(m)}$, then $\boldsymbol{T}^{(1)} = \boldsymbol{T}^{B} = \boldsymbol{R}^{\top}\boldsymbol{P}$: symmetric Biot (Jaumanann) stress tensor (R. W. Ogden, 1997, p. 1960).
- $m = 0$, in general, the formulation of the conjugate stress tensor $\boldsymbol{T}^{(0)}$ corresponding to $\boldsymbol{E}^{(0)} = \ln \boldsymbol{U}$ is given by equation (49) as proposed by (Hoger, 1987) for the general case. In case of $\boldsymbol{T}^{(m)}$ is coaxial with $\boldsymbol{E}^{(m)}$, $\boldsymbol{T}^{(0)} = \boldsymbol{R}^{\top}\boldsymbol{\tau}\boldsymbol{R}$: the rotated Kirchhoff stress tensor (where $\boldsymbol{\tau}$: symmetric Kirchhoff stress tensor).

In addition to the stress tensors mentioned, there exist numerous other stress tensors with distinct characteristics. The selection of a particular stress tensor depends on the specific application and its requirements. Notably, if the deformation gradient is known, it is possible to uniquely determine all other stress tensors from one of them (Bertram, 2021).

2.3.3 Spectral representation of stress state

Since the second Piola-Kirchhoff stress is a symmetric positive definite second-order tensor, it may be presented in the spectral form:

$$\boldsymbol{S} = \sum_{a=1}^{3} s_a \boldsymbol{G}_a \otimes \boldsymbol{G}_a \qquad \text{with} \quad |\boldsymbol{G}_a| = 1\,, \tag{2.48}$$

where $s_a, \boldsymbol{G}_a, a = 1, 2, 3$ are the principal values and principal direction of the symmetric positive definite 2nd Piola-Kirchhoff stress tensor $\boldsymbol{S}$, respectively.

Let consider the case of *isotropic materials* (subsection 2.4.5), the principal directions of the stretch tensor $\boldsymbol{C}$ coincide with the principal directions of the stress $\boldsymbol{S}$, i.e. $\boldsymbol{N}_a = \boldsymbol{G}_a$. Using

(2.31), the Cauchy stress tensor $\boldsymbol{\sigma}$ becomes:

$$\begin{aligned}\boldsymbol{\sigma} = J^{-1}\boldsymbol{F}\boldsymbol{S}\boldsymbol{F}^{\top} &= J^{-1}\left(\sum_{a=1}^{3}\lambda_a \boldsymbol{n}_a\otimes\boldsymbol{N}_a\right)\left(\sum_{a=1}^{3}s_a\boldsymbol{G}_a\otimes\boldsymbol{G}_a\right)\left(\sum_{a=1}^{3}\lambda_a \boldsymbol{n}_a\otimes\boldsymbol{N}_a\right)^{\top}\\ &= J^{-1}\sum_{a=1}^{3}\lambda_a^2 s_a(\boldsymbol{n}_a\otimes\boldsymbol{N}_a)(\boldsymbol{G}_a\otimes\boldsymbol{G}_a)(\boldsymbol{N}_a\otimes\boldsymbol{n}_a)\\ &= J^{-1}\sum_{a=1}^{3}\lambda_a^2 s_a(\boldsymbol{n}_a\otimes\boldsymbol{N}_a)[\boldsymbol{G}_a\otimes\boldsymbol{n}_a(\boldsymbol{G}_a\cdot\boldsymbol{N}_a)]\\ &= \sum_{a=1}^{3}\underbrace{J^{-1}\lambda_a^2 s_a}_{\sigma_a}\underbrace{(\boldsymbol{G}_a\cdot\boldsymbol{N}_a)\boldsymbol{n}_a}_{\boldsymbol{g}_a}\otimes\underbrace{\boldsymbol{n}_a(\boldsymbol{G}_a\cdot\boldsymbol{N}_a)}_{\boldsymbol{g}_a}\,.\end{aligned}$$

where σ_a and $\boldsymbol{g}_a$, $a=1,2,3$ denote the principal values and the principal directions of the Cauchy stress, respectively. Using identity (2.30), obtaining:

$$\boldsymbol{g}_a = (\boldsymbol{G}_a\cdot\boldsymbol{N}_a)\boldsymbol{n}_a = \boldsymbol{G}_a(\boldsymbol{N}_a\boldsymbol{R}\boldsymbol{N}_a) = \boldsymbol{G}_a\boldsymbol{R}^{\top}\boldsymbol{N}_a\boldsymbol{N}_a = \boldsymbol{R}\boldsymbol{G}_a\,. \tag{2.49}$$

Eq. (2.49) indicates that the orthonormal vectors $\boldsymbol{g}_a$, with $|\boldsymbol{g}_a|=1$, live in the current configuration whereas $\boldsymbol{G}_a$ are in the reference configuration.

Finally, the spectral form of $\boldsymbol{\sigma}$ and the symmetric Kirchhoff stress tensor $\boldsymbol{\tau}$:

$$\boldsymbol{\sigma} = \sum_{a=1}^{3}\sigma_a\,\boldsymbol{g}_a\otimes\boldsymbol{g}_a\,;\quad \sigma_a = J^{-1}\lambda_a^2 s_a\,, \tag{2.50}$$

$$\boldsymbol{\tau} = J\boldsymbol{\sigma} = \sum_{a=1}^{3}J\sigma_a\,\boldsymbol{g}_a\otimes\boldsymbol{g}_a = \sum_{a=1}^{3}\tau_a\,\boldsymbol{g}_a\otimes\boldsymbol{g}_a\,. \tag{2.51}$$

where $\tau_a = J\sigma_a, a=1,2,3$ are the principal values of $\boldsymbol{\tau}$.

The two-point 1$^{\text{st}}$ Piola-Kirchhoff stress tensor $\boldsymbol{P}$, in general, is non-symmetric and may not have a spectral representation. Nonetheless, one can present it in the principal terms by employing definition (2.47), (2.31), and (2.49)

$$\begin{aligned}\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S} &= \left(\sum_{a=1}^{3}\lambda_a\boldsymbol{n}_a\otimes\boldsymbol{N}_a\right)\left(\sum_{a=1}^{3}s_a\boldsymbol{G}_a\otimes\boldsymbol{G}_a\right)\\ &= \sum_{a=1}^{3}\lambda_a s_a(\boldsymbol{n}_a\otimes\boldsymbol{N}_a)(\boldsymbol{G}_a\otimes\boldsymbol{G}_a)\\ &= \sum_{a=1}^{3}\lambda_a s_a(\boldsymbol{N}_a\cdot\boldsymbol{G}_a)\boldsymbol{n}_a\otimes\boldsymbol{G}_a\\ &= \sum_{a=1}^{3}\lambda_a s_a\,\boldsymbol{g}_a\otimes\boldsymbol{G}_a\,.\end{aligned}$$

Denoting $p_a = \lambda_a s_a$, then:

$$\boldsymbol{P} = \sum_{a=1}^{3} p_a \, \boldsymbol{g}_a \otimes \boldsymbol{G}_a \, . \tag{2.52}$$

Note that only in the case of *isotropic materials*, the principal directions of the stretch tensor $\boldsymbol{C}$ coincide with the principal directions of the stress $\boldsymbol{S}$, i.e. $\boldsymbol{N}_a = \boldsymbol{G}_a$, and $\boldsymbol{n}_a = \boldsymbol{g}_a$.

The generalised stress tensor in a material description, $\boldsymbol{T}^{(m)}$, can be presented using the spectral presentation (for $m = 0, 1, 2$ cases):

$$\boldsymbol{T}^{(m)} = \begin{cases} \sum_{a=1}^{3} t_a \, \boldsymbol{G}_a \otimes \boldsymbol{G}_a \, ; & \text{if} \quad m > 0 \, , \\ \sum_{a=1}^{3} t_a \, \boldsymbol{g}_a \otimes \boldsymbol{g}_a \, ; & \text{if} \quad m = 0 \, . \end{cases} \tag{2.53}$$

where, t_a: the principal values of the generalised stress tensors, and

- $m = 2$, $t_a = s_a$.
- $m = 1$, $t_a = p_a$, for isotropic materials.
- $m = 0$, $t_a = \tau_a$, for isotropic materials.

2.4 Physically driven constitutive requirements

Thus far, the deformation in a continuous body has been analyzed without reference to any material properties the body may possess. However, the deformation of a body results from the inherent constitutive nature of the materials in response to forces and torques acting upon it.

Constitutive equations are the closure of the governing equations and provide the links between strain and stress (R. W. Ogden, 1997, p. 169). These are utilized to generally express the stress state of any point in the body in response to an arbitrary deformation at any given time t. Constitutive equations (or response functions) can be either mathematically generalised (axiomatic) or based upon experimental data (empirical). In practice, a combination of both methods is used to establish a comprehensive model. For material laws to be considered physically plausible, they must satisfy certain empirical conditions that are widely accepted as prerequisites for an accurate representation of material behaviour.

In nonlinear elasticity, the conditions/assumptions on the constitutive relation are known as *constitutive requirements*. These requirements have been approached from two distinct perspectives. The first perspective, referred to as the classical approach, focuses on the mechanical response of material models and establishes well-known restrictions such as the Baker-Ericksen inequalities and Hill's inequalities, etc. (Baker and Ericksen, 1954; Hill, 1968, 1979; Truesdell and Noll, 2004). The second perspective, which is motivated by mathematical

considerations, emphasizes the restrictions that must be imposed on constitutive equations to obtain solutions for boundary-value problems. This approach is concerned with concepts such as quasi-convexity and ellipticity, etc. as discussed in (Ball, 1976; Truesdell and Noll, 2004).

The following assumptions form the basis for a theory of purely mechanical phenomena in continuous media (Bertram, 2021, p. 156); (Truesdell and Noll, 2004, p. 60):

- *Principle of determinism:* The stresses at a material point at an instant time t are determined by the history (but not the future) motion of the body.
- *Principle of local action:* The stresses at a material point depend on the motion of only a finite neighbourhood of that point.
- *Principle of local action for simple materials:* The stresses at a material point depends on the motion of only its infinitesimal neighbourhood (meaning that only the deformations of the tangent space of the material point have influence, not the whole body).

Further careful consideration must be taken in the selection of these constraints to prevent undue restrictions. They should not only provide physical realism but also possess the versatility to encompass a wide spectrum of material behaviours.

2.4.1 Principle of material frame indifference

Considering a specific class of elastic materials such as Cauchy-elastic materials (path-independent simple elastic material), the constitutive equation may be expressed in the general form (Holzapfel, 2000, p. 197):

$$\boldsymbol{\sigma}(\boldsymbol{x}, t) = \mathfrak{g}(\boldsymbol{F}(\boldsymbol{X}, t), \boldsymbol{X}) \,, \tag{2.54}$$

where $\mathfrak{g}$ is the response function associated with the Cauchy stress tensor $\boldsymbol{\sigma}$. Further assuming that the elastic material is *isothermal* and *homogeneous* leads to the stress relations (Holzapfel, 2000):

$$\boldsymbol{\sigma} = \mathfrak{g}(\boldsymbol{F}), \tag{2.55}$$

$$\boldsymbol{P} = J\boldsymbol{\sigma}\boldsymbol{F}^{-\top} = \mathfrak{G}(\boldsymbol{F}). \tag{2.56}$$

Principle of material frame indifference:[1] *Constitutive equations involving thermodynamic and mechanical variables must be invariant under change of frame* (Truesdell and Noll, 2004, sec. 19).

[1] See (Truesdell and Noll, 2004, p. 45) for the history of the principle of material frame-indifference.

It indicates that the constitutive equations (2.55), (2.56) are not affected by (proper orthogonal) rigid-body motions $\boldsymbol{Q}$, hence (Truesdell and Noll, 2004, sec. 43):

$$\mathfrak{g}(\boldsymbol{QF}) = \boldsymbol{Q}\mathfrak{g}(\boldsymbol{F})\boldsymbol{Q}^{\top}\,, \tag{2.57}$$

$$\mathfrak{G}(\boldsymbol{QF}) = \boldsymbol{Q}\mathfrak{G}(\boldsymbol{F})\,. \tag{2.58}$$

Conditions (2.57) and (2.58) must hold for every non-singular $\boldsymbol{F}$, and all orthogonal $\boldsymbol{Q}$.

By applying polar decomposition $\boldsymbol{F} = \boldsymbol{RU}$ on the right hand side of (2.57), obtaining the alternative form:

$$\boldsymbol{Q}\mathfrak{g}\boldsymbol{Q}^{\top} = \mathfrak{g}(\boldsymbol{QRU}). \tag{2.59}$$

Choosing $\boldsymbol{Q} = \boldsymbol{R}^{\top}$, leads to the reduced form of the stress response:

$$\boldsymbol{\sigma} = \mathfrak{g}(\boldsymbol{U}) = \boldsymbol{R}\mathfrak{g}(\boldsymbol{U})\boldsymbol{R}^{\top}\,. \tag{2.60}$$

Thus, it shows that the response functions of elastic materials are invariant with respect to the rotation $\boldsymbol{R}$. The same procedure applied on (2.58) leads to

$$\boldsymbol{P} = \mathfrak{G}(\boldsymbol{F}) = \boldsymbol{R}\mathfrak{G}(\boldsymbol{U})\,. \tag{2.61}$$

Utilising $J = \det \boldsymbol{U}$, and (2.60), yields the general relation for the second Piola-Kirchhoff stress tensor:

$$\boldsymbol{S} = J\boldsymbol{F}^{-1}\boldsymbol{\sigma}\boldsymbol{F}^{-\top} = \det \boldsymbol{U}\boldsymbol{U}^{-1}\mathfrak{g}(\boldsymbol{U})\boldsymbol{U}^{-1}\,. \tag{2.62}$$

Recalling that $\boldsymbol{U} = \boldsymbol{C}^{1/2}$, the (2.62) can be presented as a tensor-valued function of $\boldsymbol{C}$:

$$\boldsymbol{S} = \mathfrak{H}(\boldsymbol{C})\,. \tag{2.63}$$

Then $\boldsymbol{S}$ is also invariant of observers. The Cauchy stress tensor can be obtained from $\boldsymbol{S}$ by:

$$\boldsymbol{\sigma} = \frac{1}{J}\boldsymbol{F}\mathfrak{H}(\boldsymbol{C})\boldsymbol{F}^{\top} = \mathfrak{h}(\boldsymbol{C})\,. \tag{2.64}$$

2.4.2 Material symmetry transformation

A further restriction is the material symmetry properties of Cauchy-elastic materials. Consider a thought experiment where a body is subjected to two tests (Beatty, 1987):

- In the first test, the body is deformed at $\boldsymbol{X}$ in reference configuration Ω_0 by deformation $\boldsymbol{F}$ relative to $\mathbb{R}^3$ frame resulting in the response function:

$$\boldsymbol{\sigma}^{(1)} = \mathfrak{g}(\boldsymbol{F}) = \mathfrak{h}(\boldsymbol{C})\,.$$

- In the second test, in the same configuration Ω_0 as before, the body is first subjected to a specified rigid body rotation $\boldsymbol{Q}$ ($\boldsymbol{Q} \in SO(3)$) in the frame $\mathbb{R}^3$, and then the same deformation $\boldsymbol{F}$ is applied resulting in the body ending up at a different orientation in frame $\mathbb{R}^3$ with $\widehat{\boldsymbol{F}} = \boldsymbol{F}\boldsymbol{Q}$. In this case, $\widehat{\boldsymbol{C}} = \widehat{\boldsymbol{F}}^{\top}\widehat{\boldsymbol{F}} = \boldsymbol{Q}^{\top}\boldsymbol{C}\boldsymbol{Q}$, and the response function is:

$$\boldsymbol{\sigma}^{(2)} = \mathfrak{g}(\widehat{\boldsymbol{F}}) = \mathfrak{h}(\boldsymbol{Q}^{\top}\boldsymbol{C}\boldsymbol{Q}).$$

In general, the response functions of both tests are not expected to be the same. However, if the material response at $\boldsymbol{X}$, before and after the given rotation $\boldsymbol{Q}$, for both experiments is the same ($\boldsymbol{\sigma}^{(1)} = \boldsymbol{\sigma}^{(2)}$), then:

$$\mathfrak{h}(\boldsymbol{C}) = \mathfrak{h}(\boldsymbol{Q}^{\top}\boldsymbol{C}\boldsymbol{Q})\,, \tag{2.65}$$

where $\mathfrak{h}$ is a tensor-valued function of the symmetric second-order tensor $\boldsymbol{C}$. Then (R. W. Ogden, 1997, p. 183),

$$\mathfrak{g}(\boldsymbol{F}\boldsymbol{Q}) = \mathfrak{g}(\boldsymbol{F})\,. \tag{2.66}$$

Here, special rotations $\boldsymbol{Q}$ that satisfy equations (2.65) and (2.66) are known as material symmetry transformation (Beatty, 1987). These transformations form a subgroup of a more general *symmetry group of the material* $\mathscr{G}$ at $\boldsymbol{X}$ with respect to the reference configuration Ω_0. A material symmetry group is defined as a set of linear transformations which maps the reference configuration onto mechanically indistinguishable configurations.

It is important to note that the material symmetry group $\mathscr{G}$ depends on the chosen reference configuration. Altering the reference configuration results in different material symmetry group presenting the same material symmetry (cf. (Negahban and Wineman, 1989) for further details).

In addition, the material itself can be characterized by material symmetry groups, $\mathscr{G}$ (Bertram, 2021, p. 189). The identity transformation $\{\boldsymbol{I}\}$ is contained in every group, making the set $\{\boldsymbol{I}\}$ the minimal symmetry group. A material possessing this minimal symmetry group is known as a *triclinic material*, where $\mathscr{G} = \{\boldsymbol{I}\}$. In contrast, *(elastic) fluids* exhibit the maximum symmetry group, which consists of the unimodular (volume preserving) transformations,

$\{\mathscr{U}^+ = \boldsymbol{H} | \det \boldsymbol{H} = 1\}$[1]. All other materials have symmetry groups that lie between these two extremes $\{\boldsymbol{I}\} \subseteq \mathscr{G} \subseteq \mathscr{U}^+$ (Bertram, 2021, p. 190), such as monoclinic, orthotropic, transversely isotropic, and isotropic materials (Negahban and Wineman, 1989).

2.4.3 Undistorted and stress-free configurations

Since the material symmetry groups $\mathscr{G}$ of an elastic material depend on the reference configurations[2], the response of the material is also influenced by these configurations. Therefore, it is important to fix reference configurations relative to which the material symmetry group holds.

Consider the vanishing deformation state, $\boldsymbol{F} = \boldsymbol{I}$, where the Cauchy stress tensor becomes $\boldsymbol{\sigma}(\boldsymbol{F}) = \mathfrak{g}(\boldsymbol{I})$. Subsequently, due to the combined effects of material frame indifference (2.57) and material symmetry (2.58) (R. W. Ogden, 1997, p. 184), the following relation is obtained:

$$\mathfrak{g}(\boldsymbol{I}) = \boldsymbol{Q}\mathfrak{g}(\boldsymbol{I})\boldsymbol{Q}^\top . \tag{2.67}$$

This implies that $\mathscr{G}$ associated with this configuration contains the proper orthogonal group $SO(3)$, and consequently, equation (2.67) holds for all proper orthogonal matrices $\boldsymbol{Q}$. Furthermore,

$$\mathfrak{g}(\boldsymbol{I}) = \alpha \boldsymbol{I} , \tag{2.68}$$

where α is a scalar. Therefore, in the considered reference configuration, the stress is hydrostatic, and the deformation associated with such reference configurations is a pure dilatation (potentially coupled with a rotation). This reference configuration is referred to as an *undistorted configuration* of the material.

It is convenient to choose an undistorted configuration that is *stress-free* (*natural configuration*) as the reference configuration[3]. In this stress-free configuration, the response functions

[1] $\boldsymbol{H} \in SO(3)$ is restricted in this work, however, it implied non-orthogonal elements also can be admitted. Furthermore, $\{\mathscr{U}^- = \boldsymbol{H} | \det \boldsymbol{H} = -1\}$ is valid in general, however, it is not considered since the negative values of the determinant of the deformations correspond to *non-physically* realistic deformations.

[2] Noll (2009) coined the terms "configuration" and "deformation" in what is now referred to as "placement" and "transplacement" (e.g. (Bertram, 2021)). He expressed regret for introducing those terms in the 1960s and offered an apology for any confusion they may have caused.

In this work, the author continues to use the 'old terms' for their current work but plans to adopt the new terms in future works.

[3] The reference configuration does not have to be an undistorted or actual configuration that the body occupies during motion. It can be any configuration that the body is capable of occupying. For example, the natural, undeformed state of an automobile tire at rest on a store rack may serve as a reference configuration. The inflated, toroidal state of the tire when mounted on a car wheel can also be used as a reference configuration. The body must be capable of occupying the chosen reference configuration, but it is not necessary that it actually does so (Beatty, 1987).

satisfy:

$$\boldsymbol{\sigma} = \mathfrak{g}(\boldsymbol{F} = \boldsymbol{I}) = \boldsymbol{0}\,, \tag{2.69}$$

$$\boldsymbol{P} = \mathfrak{G}(\boldsymbol{F} = \boldsymbol{I}) = \boldsymbol{0}\,, \tag{2.70}$$

$$\boldsymbol{S} = \mathfrak{H}(\boldsymbol{C} = \boldsymbol{I}) = \boldsymbol{0}\,. \tag{2.71}$$

2.4.4 Growth conditions in large strains

Another restriction upon the constitutive relations describes the intuitive idea that "infinite stress must accompany extreme strains" (Antman, 1983; Ciarlet, 1988) and are supported by direct physical observation. This is called *growth conditions*, and the fundamental concept behind it is that infinite stress should be associated with extreme strain (Antman, 1983), i.e.:

$$\|\boldsymbol{P}\| \longrightarrow +\infty \text{ as any eigenvalue } (\lambda_a(\boldsymbol{C})) \text{ of } \boldsymbol{C} \text{ approach } +\infty\,, \tag{2.72}$$

$$\|\boldsymbol{P}\| \longrightarrow -\infty \text{ as any eigenvalue } (\lambda_a(\boldsymbol{C})) \text{ of } \boldsymbol{C} \text{ approach } 0^+\,, \tag{2.73}$$

where the $\|\boldsymbol{P}\| = (\boldsymbol{P} : \boldsymbol{P})^{1/2}$ is the tensor norm of $\boldsymbol{P}$. It is observed that if an eigenvalue of $\boldsymbol{C}$ were to approach ∞ or 0, we have the following equivalences (Ciarlet, 1988, p. 158):

$$\lambda_a(\boldsymbol{C}) \to 0^+ \Longleftrightarrow \det \boldsymbol{F} \to 0^+\,, \tag{2.74}$$

$$\lambda_a(\boldsymbol{C}) \to +\infty \Longleftrightarrow \det \boldsymbol{F} \to +\infty\,, \tag{2.75}$$

$$\lambda_a(\boldsymbol{C}) \to +\infty \Longleftrightarrow \|\boldsymbol{F}\| \to +\infty\,, \tag{2.76}$$

$$\lambda_a(\boldsymbol{C}) \to +\infty \Longleftrightarrow \|\boldsymbol{F}\| \to +\infty\,. \tag{2.77}$$

2.4.5 Isotropic elastic materials

If a material possesses the material symmetry group such that $\mathscr{G} = SO(3)$ in at least one reference configuration, then its mechanical behaviour exhibits no preferred direction. Such material is called an *isotropic elastic material* (R. W. Ogden, 1997, p. 184). This implies that for any proper orthogonal tensor $\boldsymbol{Q}$, the relationship given by (2.65) holds. By choosing $\boldsymbol{Q} = \boldsymbol{R}^\top$ and using the relation (2.17), it can be shown that isotropic materials must satisfy the necessary condition:

$$\mathfrak{h}(\boldsymbol{C}) = \mathfrak{h}(\boldsymbol{Q}^\top \boldsymbol{C} \boldsymbol{Q}) = \mathfrak{h}(\boldsymbol{R}^\top \boldsymbol{C} \boldsymbol{R}) = \mathfrak{h}(\boldsymbol{b})\,. \tag{2.78}$$

This condition means that, for a given deformation, the response function has the same values whether $\boldsymbol{C}$ or $\boldsymbol{b}$ is used as the independent variable. Isotropic material holds the fundamental invariance relation (Holzapfel, 2000, p. 200):

$$\boldsymbol{Q}\mathfrak{h}(\boldsymbol{b})\boldsymbol{Q}^\top = \mathfrak{h}(\boldsymbol{Q}\boldsymbol{b}\boldsymbol{Q}^\top)\,. \tag{2.79}$$

The isotropic tensor function $\mathfrak{h}(\boldsymbol{b})$, which satisfies (2.79), may be presented in the form (Gurtin, 1981, p. 233):

$$\boldsymbol{\sigma} = \mathfrak{h}(\boldsymbol{b}) = \alpha_0 \boldsymbol{I} + \alpha_1 \boldsymbol{b} + \alpha_2 \boldsymbol{b}^2 \,; \qquad \alpha_j = \alpha_j[I_1(\boldsymbol{b}), I_2(\boldsymbol{b}), I_3(\boldsymbol{b})] \,, \tag{2.80}$$

where $\alpha_j, j = 1, 2, 3$ are three scalar functions, and $\{\alpha_j\}$ depend on three invariants of tensor $\boldsymbol{b}$. Representation (2.80) is known as the *first representation theorem for isotropic tensor functions* (Gurtin, 1981, p. 233), and is the most general form of a constitutive equation for isotropic materials.

Now, since $\boldsymbol{b}^{-1}$ exists, an alternative form of the constitutive equation for isotropic materials can be shown (Gurtin, 1981, p. 235):

$$\boldsymbol{\sigma} = \mathfrak{h}(\boldsymbol{b}) = \beta_0 \boldsymbol{I} + \beta_1 \boldsymbol{b} + \beta_{-1} \boldsymbol{b}^{-1} \,; \qquad \beta_k = \beta_k[I_1(\boldsymbol{b}), I_2(\boldsymbol{b}), I_3(\boldsymbol{b})] \,, \tag{2.81}$$

where, $\beta_k, k = 0, 1, -1$, are also three scalar functions of the three invariants of $\boldsymbol{b}$, expressed as:

$$\beta_0 = \alpha_0 - I_2 \alpha_2 \,; \quad \beta_1 = \alpha_1 + I_1 \alpha_2 \,; \quad \beta_{-1} = I_3 \alpha_2 \,. \tag{2.82}$$

Representation (2.81) is known as the *second representation theorem for isotropic tensor functions*.

2.4.6 Baker–Ericksen inequalities

The Baker–Ericksen (BE) inequalities (Baker and Ericksen, 1954; Fosdick and Šilhavý, 2006; Marsden and Hughes, 1994; Marzano, 1983; Mihai et al., 2017; Neff, Ghiba, and Lankeit, 2015; Truesdell and Noll, 2004) are critical necessary requirements[1] for reasonable material behaviours, dealing directly with the mechanical response of the material models. The applicability of the BE-inequalities has been extended to nonlinear transversely isotropic materials (Humphrey, Strumpf, and Yin, 1990; May-Newman and Yin, 1998) and further generalised for anisotropic elastic solids by Fosdick and Šilhavý (2006).

The original work by (Baker and Ericksen, 1954) demonstrated that the BE-inequalities are necessary and sufficient for *the greater principal stress to occur in the direction of the greater principal stretch*, as expressed by:

$$(h_a - h_b)(\lambda_a - \lambda_b) \geq 0 \,; \quad \lambda_a \neq \lambda_b \,; \quad a, b = 1, 2, 3 \,. \tag{BE-inequalities}$$

[1] Among other possible restrictions on the response function, $\mathfrak{g}(\boldsymbol{F})$, such as (1) physically motivated requirements: *P-C inequality*, *P-C inequality*, *T-E inequalities*, *IFS* conditions, *E-T inequalities*, *O-F inequality*, etc. and (2) mathematically motivated requirements: *GCN condition*, *rank-one convexity*, *quasi-convexity*, and *poly-convexity* (Noll, 2006).

Here, h_a represents the principal stresses (eigenvalues of the response function $\mathfrak{h}(\boldsymbol{C})$) and λ_a represents the principal stretches (eigenvalues of $\boldsymbol{C}$), reveal the ordering of principal stresses and stretches in the same sequence.

2.4.7 Hill's constitutive inequalities

Given the various assumptions that have been made to arrive at a simplified version of a material model, such as isotropic elasticity. It is important to ensure that the model satisfies certain fundamental physical requirements. These requirements are often based on observations from experimental tests and are referred to as physically driven requirements (R. W. Ogden, 1997; Voss, 2021). Additionally, when applying the material law to solve boundary-value problems (see Section 3.1), mathematically driven restrictions emerge. These restrictions are associated with the existence and uniqueness of solutions to the governing differential equations and boundary conditions (R. W. Ogden, 1997, p. 170). Naturally, these two viewpoints must be consistent with each other.[1]

2.4.7.1 Preliminarily

Return to the deformation and velocity expressions represented in (2.1) and (2.2), with the time variable included. The time derivative of $\boldsymbol{F}$ at fixed $\boldsymbol{X}$ (relative to $\boldsymbol{x}$) be:

$$\dot{\boldsymbol{F}} = \frac{\partial}{\partial t}\left(\frac{\partial \boldsymbol{\chi}}{\partial \boldsymbol{X}}\right) = \frac{\partial}{\partial \boldsymbol{X}}\left(\frac{\partial \boldsymbol{\chi}}{\partial t}\right) = \frac{\partial \dot{\boldsymbol{x}}}{\partial \boldsymbol{X}} = \frac{\partial \dot{\boldsymbol{x}}}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}} = \underbrace{\frac{\partial \dot{\boldsymbol{x}}}{\partial \boldsymbol{x}}}_{\boldsymbol{l}}\boldsymbol{F}\,. \tag{2.83}$$

where $\boldsymbol{l}$ is the *spacial velocity gradient* tensor, then (Holzapfel, 2000, pp. 97):

$$\boldsymbol{l} = \dot{\boldsymbol{F}}\boldsymbol{F}^{-1}\,. \tag{2.84}$$

The velocity gradient tensor can be decomposed into the sum of a symmetric part $\boldsymbol{d}$ and a skew part $\boldsymbol{w}$ as follows:

$$\boldsymbol{d} = \frac{1}{2}(\boldsymbol{l} + \boldsymbol{l}^{\top})\,; \quad \boldsymbol{w} = \frac{1}{2}(\boldsymbol{l} - \boldsymbol{l}^{\top})\,. \tag{2.85}$$

The symmetric part $\boldsymbol{d}$ is referred to as *rate of deformation tensor* (Hill (1968, 1970) and R. W. Ogden (1997) preferred it as Eulerian strain-rate), whereas $\boldsymbol{w}$ is called the *spin tensor* or *rate of rotation* (c.f (Chakrabarty, 2012, pp. 44)). Both $\boldsymbol{d}$ and $\boldsymbol{w}$ are quantities spacial configuration (hence, the low case bold symbols).

[1] One must also exercise caution when using purely mathematical requirements without any connection to the physical situation, for example, polyconvexity, as it may exclude buckling modes (Hill, 1970; Neff, Ghiba, and Lankeit, 2015; Voss, 2021). While polyconvexity is useful for verifying the existence of a solution corresponding to the minimization problem, its physical integration of material behaviour is not yet fully understood (Carroll, 2009; Voss, 2021).

The normal component of the Eulerian strain-rate are:

$$d_{11} = \frac{\dot{\lambda}_1}{\lambda_1}\,, \quad d_{22} = \frac{\dot{\lambda}_2}{\lambda_2}\,, \quad d_{33} = \frac{\dot{\lambda}_3}{\lambda_3}\,. \tag{2.86}$$

where $\dot{\lambda} = \frac{\partial \lambda}{\partial t}$ is the rate of change of λ.

Assuming no two stretches are equal throughout this section express the spin components of the material principal axes as follows:

$$w_1 = \frac{2\lambda_2\lambda_3}{\lambda_2^2 - \lambda_3^2} d_{23}\,; \quad w_2 = \frac{2\lambda_3\lambda_1}{\lambda_3^2 - \lambda_1^2} d_{31}\,; \quad w_3 = \frac{2\lambda_1\lambda_2}{\lambda_1^2 - \lambda_2^2} d_{12}\,. \tag{2.87}$$

The rate of change of the principal Kirchhoff stresses (2.51) are:

$$\dot{\tau}_1 = \lambda_1 \frac{\partial \tau_1}{\partial \lambda_1} d_{11}\,, \quad \dot{\tau}_2 = \lambda_1 \frac{\partial \tau_2}{\partial \lambda_2} d_{22}\,, \quad \dot{\tau}_3 = \lambda_1 \frac{\partial \tau_3}{\partial \lambda_3} d_{33} \tag{2.88}$$

For the case of isotropic Cauchy elasticity, the objective stress rate as follows (R. Ogden, 1970):

$$\mathring{\boldsymbol{s}}^{(m)} = \mathring{\boldsymbol{\tau}} - \frac{m}{2}(\boldsymbol{\sigma d} + \boldsymbol{d\sigma})\,; \quad \mathring{s}_{ab} = \mathring{\tau}_{ab} - \frac{m}{2}(\sigma_{ac} d_{cb} + \sigma_{bc} d_{ca})\,, \tag{2.89}$$

where $\mathring{(\bullet)}$ symbol denotes the Jaumann (or rigid-body) derivative.

The components of the Jaumann derivative of Kirchhoff stress is as follows:

$$\begin{cases} J\mathring{\tau}_{aa} = \sum_{a=1}^{3} \lambda_a \frac{\partial \tau_a}{\partial \lambda_a} d_{aa} & \text{for normal components,} \\ J\mathring{\tau}_{ab} = \frac{\lambda_a^2 + \lambda_b^2}{\lambda_a^2 - \lambda_b^2}(\tau_a - \tau_b) d_{ab} & \text{for shear components}\,. \end{cases} \tag{2.90}$$

Therefore, (2.89), in terms of normal and shear components, becomes:

$$\begin{cases} J\mathring{s}_{11}^{(m)} = \left(\lambda_1 \frac{\partial \tau_1}{\partial \lambda_1} - m\tau_1\right) d_{11} + \lambda_2 \frac{\partial \tau_1}{\partial \lambda_2} d_{22} + \lambda_3 \frac{\partial \tau_1}{\partial \lambda_3} d_{33}\,, & \text{etc.,} \\ J\mathring{s}_{12}^{(m)} = \left[\frac{\lambda_1^2 + \lambda_2^2}{\lambda_1^2 - \lambda_2^2}(\tau_1 - \tau_2) - \frac{m}{2}(\tau_1 + \tau_2)\right] d_{12} & \text{etc.}\,. \end{cases} \tag{2.91}$$

The material rates of change of the principal generalised strains (2.40) and princial generalised stresses (2.53) on the material principal axes is as follows (Hill, 1970):

$$\dot{e}_a^{(m)} = \dot{\lambda}_a \lambda_a^{m-1}\,; \quad \dot{t}_a^{(m)} = \sum_{a=1}^{3} \frac{\partial t_a^{(m)}}{\partial e_a^{(m)}} \dot{e}_a^{(m)}\,. \tag{2.92}$$

The material rates of change $\dot{\boldsymbol{E}}$ and $\dot{\boldsymbol{T}}$ are expressed as:

$$\begin{cases} \dot{E}^{(m)}_{aa} = \dot{e}^{(m)}_a\,; \quad \dot{T}^{(m)}_{aa} = \dot{t}^{(m)}_a & \text{for normal components,} \\ \dot{E}^{(m)}_{ab} = (e^{(m)}_a - e^{(m)}_b)w_k\,; \quad \dot{T}^{(m)}_{ab} = (t^{(m)}_a - t^{(m)}_b)w_k\,; \quad k \neq a \neq b & \text{for shear components.} \end{cases} \tag{2.93}$$

2.4.7.2 Constitutive inequalities

Hill (1970) introduced the first inequality requiring a positive scalar product between each stress-rate and strain-rate pair:

$$\mathring{\boldsymbol{s}}^{(m)} : \boldsymbol{d} > 0 \qquad \text{for all strain}\,. \tag{2.94}$$

Here m denotes the type of conjugate strain-stress pair as introduced in (2.18). The second Hill's inequality is established from (2.94) for isotropic Cauchy elastic materials. The key finding that (2.94) leads to the generally weaker inequality based on the general strains measures (2.18) and their corresponding conjugate stresses:

$$\dot{\boldsymbol{T}}^{(m)} : \dot{\boldsymbol{E}}^{(m)} > 0 \qquad \text{for all strain-rates}\,. \tag{2.95}$$

As noted in Hill (1970), generally, inequality (2.94) is distinct from (2.95).

Using (2.91) and (2.92), inequality (2.94) may be expressed in an alternative form (R. W. Ogden, 1970)

$$J\mathring{\boldsymbol{s}}^{(m)} : \boldsymbol{d} = \sum_{a=1}^{3}\sum_{b=1}^{3} \frac{\partial t^{(m)}_a}{\partial e^{(m)}_b} \dot{e}^{(m)}_a \dot{e}^{(m)}_b + \sum_{a\neq b} \psi_{ab} d^2_{ab} \quad > 0\,, \tag{2.96}$$

where

$$\psi_{ab} = \psi_{ba} = \frac{\tau_a - \tau_b}{\lambda_a^2 - \lambda_b^2}(\lambda_a^2 + \lambda_b^2) - \frac{m}{2}(\tau_a + \tau_b) \tag{2.97}$$

By (2.92) and (2.93), the inequality yields (Hill, 1970):

$$\dot{\boldsymbol{T}}^{(m)} : \dot{\boldsymbol{E}}^{(m)} = \sum_{a=1}^{3}\sum_{b=1}^{3} \frac{\partial t^{(m)}_a}{\partial e^{(m)}_b} \dot{e}^{(m)}_a \dot{e}^{(m)}_b + \sum_{k\neq a\neq b} (t^{(m)}_a - t^{(m)}_b)(e^{(m)}_a - e^{(m)}_b) w_k^2 \quad > 0\,. \tag{2.98}$$

Expressions (2.96) and (2.98) are positive if and only if its independent components are positive, namely

$$\sum_{a=1}^{3}\sum_{b=1}^{3}\frac{\partial t_a^{(m)}}{\partial e_b^{(m)}}\dot{e}_a^{(m)}\dot{e}_b^{(m)} > 0\,, \tag{2.99}$$

$$\psi_{12} > 0\,, \qquad \psi_{13} > 0\,, \qquad \psi_{31} > 0\,, \tag{2.100}$$

$$(t_a^{(m)} - t_b^{(m)})(e_a^{(m)} - e_b^{(m)}) > 0\,; \qquad e_a^{(m)} \neq e_b^{(m)}\,. \tag{2.101}$$

Equation (2.99) implies (2.101) as it holds in every configuration. This states that the ordering of principal stresses $t_a^{(m)}$ in the same algebraic sequence as $e_a^{(m)}$, which, clearly, implies (BE-inequalities). Therefore, by satisfying Hill's inequality, $\mathring{\boldsymbol{s}}^{(m)} : \boldsymbol{d} > 0$, the constitutive model also satisfies the BE-inequalities in the case of compressible isotropic solid.

Hill (1970) concluded the implications of inequalities (2.94) and (2.95) for isotropic compressible solid:

- The same algebraic sequence or the principal stresses and principal strains,

$$(t_a^{(m)} - t_b^{(m)})(e_a^{(m)} - e_b^{(m)}) > 0\,; \qquad e_a^{(m)} \neq e_b^{(m)}\,. \tag{2.102}$$

- The Jacobian matrix:

$$\boldsymbol{H}^{(m)} = \begin{bmatrix} \frac{\partial t_1^{(m)}}{\partial e_1^{(m)}} & \frac{\partial t_1^{(m)}}{\partial e_2^{(m)}} & \frac{\partial t_1^{(m)}}{\partial e_3^{(m)}} \\ \frac{\partial t_2^{(m)}}{\partial e_1^{(m)}} & \frac{\partial t_2^{(m)}}{\partial e_2^{(m)}} & \frac{\partial t_2^{(m)}}{\partial e_3^{(m)}} \\ \frac{\partial t_3^{(m)}}{\partial e_1^{(m)}} & \frac{\partial t_3^{(m)}}{\partial e_2^{(m)}} & \frac{\partial t_3^{(m)}}{\partial e_3^{(m)}} \end{bmatrix} \text{ is positive definite for all strains.} \tag{2.103}$$

  Or equivalently, the symmetric part, $\frac{1}{2}(\boldsymbol{H}^{(m)} + [\boldsymbol{H}^{(m)}]^\top)$, is also symmetric positive definite

2.4.8 Constitutive model for non-linear isotropic elastic materials

So far, the constitutive relations that approximate the actual macroscopic behaviour of materials are discussed. Therefore, a data-driven constitutive material law, denotes as:

$$\boldsymbol{T}^{(m)} = \mathfrak{f}(\boldsymbol{E}^{(m)})\,, \tag{2.104}$$

is required to satisfy the principle of material frame indifference (sect. 2.4.1) and material symmetry (sect. 2.4.2), which imposes certain limitations on its form.

By leveraging the spectral representation of the generalised strain tensor and its conjugate stress tensor, these assumptions are automatically fulfilled:

$$\sum_{a=1}^{3} t_a^{(m)} \boldsymbol{g}_a \otimes \boldsymbol{g}_a = \sum_{a=1}^{3} \mathfrak{f}(e_a^{(m)}) \boldsymbol{N}_a \otimes \boldsymbol{N}_a \,. \tag{2.105}$$

Therefore, aiming for a symmetric tonsorial form of the strains-stress pairs is preferable due to its compatibility with the spectral decomposition. Among the generalised strains/stresses options in the material description, where $m \in \{2, 1, 0\}$, the preference leans towards $m = 2$. This choice allow both $\boldsymbol{E}^{(2)} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I})$ and its stress conjugate pair, $\boldsymbol{S}$, to be symmetric tensors. The only other option with both quantities are symmetric tensors is $m = 0$, however, $\boldsymbol{E}^{(0)} = \ln \boldsymbol{U}$ whose eigenvalues are computed from tensor $\boldsymbol{C}$.

Henceforth, this work primarily focuses on the case of $m = 2$ for the strain-stress relation:

$$\sum_{a=1}^{3} s_a \, \boldsymbol{g}_a \otimes \boldsymbol{g}_a = \sum_{a=1}^{3} \mathfrak{f}(c_a) \, \boldsymbol{N}_a \otimes \boldsymbol{N}_a \,. \tag{2.106}$$

Additionally, the consideration narrows down to constitutive laws containing undistorted reference configurations (sect. 2.4.3), i.e.,

$$\mathfrak{f}(\boldsymbol{I}) = \boldsymbol{0} \,. \tag{2.107}$$

This further restricts materials to solid elastic materials. Hereafter, the isotropic constraint (sect. 2.4.5) are in place, further simplifying the focus of this study to **non-linear isotropic compressible elastic solids**. Consequently, $\boldsymbol{N}_a = \boldsymbol{g}_a$,

$$\sum_{a=1}^{3} s_a \, \boldsymbol{N}_a \otimes \boldsymbol{N}_a = \sum_{a=1}^{3} \mathfrak{f}(c_a) \, \boldsymbol{N}_a \otimes \boldsymbol{N}_a \,. \tag{2.108}$$

To this end, the inputs and outputs for the data-driven surrogate models are clearly defined. The principal values of the right Cauchy–Green tensor, $\boldsymbol{c} = [c_1, c_2, c_3]^\top$, serve as inputs, and the principal values of the 2$^{\text{nd}}$ Piola-Kirchhoff stress tensor, $\boldsymbol{s} = [s_1, s_2, s_3]^\top$, act as outputs for the model.

Furthermore, the strain-stress relation $\mathfrak{f}(\boldsymbol{c})$ should fulfill the growth condition (set. 2.4.4), Hill's inequality (which implies the Baker-Ericksen inequality as discussed in sect. 2.4.7). Specific details for adapting the requirements will be presented in Chapter 4.

2.4.9 Ideal reference candidate: Hyperelastic materials

Hyperelastic material, also known as *Green elastic material*, represents a special case of Cauchy elastic material. It assumes the existence of a *strain-energy function*, denoted as $\Psi = \Psi(\boldsymbol{F})$. which measures the energy stored in an elastic body due to deformation. Although primarily

driven by mathematical considerations, hyperelasticity utilises common calculus methods to establish the existence of a minimizer. It fulfils all the physically driven requirements and their corresponding mathematical consequences outlined in previous subsections. Additionally, it also ensures the conservation of elastic energy, which is a physically desirable property (Carroll, 2009). Therefore, hyperelasticity serves as an ideal reference candidate for the verification and validation of new data-driven techniques.

Henceforth in this work, synthetic datasets generated from hyperelastic materials are employed to train artificial neural networks. This approach aims to use a well-established model like hyperelasticity to assess the performance of the novel DNN-based surrogate models in capturing unknown constitutive laws.

The following assumptions are applied:

- Only considering the time-independent resulting deformations.
- The material is homogeneous, and any effect of temperature is ignored, i.e. $\Psi(\boldsymbol{X}, t) = \Psi(\boldsymbol{X})$.
- The entire mechanical energy utilized to deform the material is stored as elastic potential energy. When the deformed body recovers its initial configuration, an equivalent amount of energy is released.
- The elastic behaviour can be described by Ψ.

The mechanical energy principle leads to the following differential equation of mechanical energy balance (Beatty, 1987; Holzapfel, 2000):

$$\frac{D}{Dt}\left[\Psi(\boldsymbol{X})\right] = \operatorname{tr}(\boldsymbol{P} : \dot{\boldsymbol{F}}^{\top}) = \operatorname{tr}(\mathfrak{G}(\boldsymbol{F}) : \dot{\boldsymbol{F}}^{\top}) = \mathfrak{G}(\boldsymbol{F}) : \dot{\boldsymbol{F}} \tag{2.109}$$

$$\mathfrak{G}(\boldsymbol{F}) : \dot{\boldsymbol{F}} - \frac{D}{Dt}\left[\Psi(\boldsymbol{X})\right] = 0 \tag{2.110}$$

$$\left(\mathfrak{G}(\boldsymbol{F}) - \frac{\partial \Psi(\boldsymbol{F})}{\partial \boldsymbol{F}}\right) : \dot{\boldsymbol{F}} = 0\,. \tag{2.111}$$

As $\boldsymbol{F}$ and $\dot{\boldsymbol{F}}$ can be chosen arbitrarily, Eq. (2.111) yields the following general constitutive equation for hyperelastic materials:

$$\boldsymbol{P} = \mathfrak{G}(\boldsymbol{F}) = \frac{\partial \Psi(\boldsymbol{F})}{\partial \boldsymbol{F}}\,. \tag{2.112}$$

The Cauchy stress is determined by (Holzapfel, 2000, p. 210):

$$\boldsymbol{\sigma} = J^{-1}\boldsymbol{F}\left(\frac{\partial \Psi(\boldsymbol{F})}{\partial \boldsymbol{F}}\right)^{\top} = 2J^{-1}\boldsymbol{F}\frac{\partial \Psi(\boldsymbol{C})}{\partial \boldsymbol{C}}\boldsymbol{F}^{\top}\,. \tag{2.113}$$

Alternative expressions of first and second Piola-Kirchhoff stress follow:

$$\boldsymbol{P} = 2\boldsymbol{F}\frac{\partial \Psi(\boldsymbol{C})}{\partial \boldsymbol{C}}\,; \quad \boldsymbol{S} = 2\frac{\partial \Psi(\boldsymbol{C})}{\partial \boldsymbol{C}}\,. \tag{2.114}$$

Since Green elasticity is a special case of Cauchy elasticity, all its properties attributed to $\mathfrak{G}(\boldsymbol{F})$ carry over to $\frac{\partial \Psi(\boldsymbol{F})}{\partial \boldsymbol{F}}$, then, all the inequalities mentioned in Section 2.4 hold for hyperelastic materials.

Hyperelastic materials satisfy the principle of material indifference (subsection 2.4.1),

$$\Psi(\boldsymbol{Q}\boldsymbol{F}) = \Psi(\boldsymbol{F})\,; \quad \forall \boldsymbol{Q} \in SO(3)\,. \tag{2.115}$$

By choosing $\boldsymbol{Q} = \boldsymbol{R}^{\top}$, and with $\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U}$, it follows that,

$$\Psi(\boldsymbol{F}) = \Psi(\boldsymbol{R}^{\top}\boldsymbol{R}\boldsymbol{U}) = \Psi(\boldsymbol{U}) = \widetilde{\Psi}(\boldsymbol{C})\,. \tag{2.116}$$

The principle of material transformations (section 2.4.2) is also satisfied by hyperelasticity:

$$\widetilde{\Psi}(\boldsymbol{C}) = \widetilde{\Psi}(\boldsymbol{Q}^{\top}\boldsymbol{C}\boldsymbol{Q}) \tag{2.117}$$

For convenience, choosing the natural configuration (section 2.4.3 as the reference configuration, where the minimum of the strain-energy is assumed to be zero. This is also regarded as the *normalisation condition*, expressed as:

$$\Psi(\boldsymbol{I}) = \widetilde{\Psi}(\boldsymbol{I}) = 0\,. \tag{2.118}$$

At finite strains, the response function $\mathfrak{G}$ is required to satisfy the assumption of *growth conditions* (subsection 2.4.4). Then, the corresponding conditions for hyperelasticity are that an infinite amount of strain-energy is necessary for expanding the continuum body to an infinite size or compressing it to a vanished volume point (Holzapfel, 2000, p. 208). They are expressed as:

$$\lim_{\det \boldsymbol{F} \to +\infty} \widetilde{\Psi}(\boldsymbol{C}) = +\infty, \qquad \lim_{\det \boldsymbol{F} \to 0^{+}} \widetilde{\Psi}(\boldsymbol{C}) = +\infty\,, \tag{2.119}$$

Furthermore, the strain-energy function must also satisfy certain mathematical conditions to ensure material stability and the existence of solutions to boundary-value problems[1].

[1]Specifically, it must satisfy quasi-convexity, continuity, and coercivity conditions. Detailed discussions can be found in Ball (1976), Marsden and Hughes (1994), and Neff, Ghiba, and Lankeit (2015).

Considering the case of isotropy (subsection 2.4.5), the strain-energy of *isotropic hyperelastic materials* can be expressed as:

$$\widetilde{\Psi}(\boldsymbol{C}) = \widetilde{\Psi}(\boldsymbol{b}), \tag{2.120}$$

indicates that the strain energy of isotropic hyperelastic material has the same value for $\boldsymbol{C}$ or $\boldsymbol{b}$. Furthermore, it can be presented in terms of the principal values of $\boldsymbol{C}$:

$$\widetilde{\Psi}(\boldsymbol{C}) = \widetilde{\Psi}(\lambda_1(\boldsymbol{C}), \lambda_2(\boldsymbol{C}), \lambda_3(\boldsymbol{C}))\,, \tag{2.121}$$

Follow that the stress tensors may be presented in terms of principal stretches as:

$$\boldsymbol{\sigma} = \sum_{a=1}^{3} J^{-1}\lambda_a \frac{\partial \widetilde{\Psi}}{\partial \lambda_a} \boldsymbol{n}_a \otimes \boldsymbol{n}_a\,, \tag{2.122}$$

$$\boldsymbol{\tau} = \sum_{a=1}^{3} \lambda_a \frac{\partial \widetilde{\Psi}}{\partial \lambda_a} \boldsymbol{n}_a \otimes \boldsymbol{n}_a\,, \tag{2.123}$$

$$\boldsymbol{P} = \sum_{a=1}^{3} \frac{\partial \widetilde{\Psi}}{\partial \lambda_a} \boldsymbol{n}_a \otimes \boldsymbol{N}_a\,, \tag{2.124}$$

$$\boldsymbol{S} = \sum_{a=1}^{3} \frac{1}{\lambda_a} \frac{\partial \widetilde{\Psi}}{\partial \lambda_a} \boldsymbol{N}_a \otimes \boldsymbol{N}_a\,. \tag{2.125}$$

Here, $\boldsymbol{N}_a = \boldsymbol{G}_a$ and $\boldsymbol{n}_a = \boldsymbol{g}_a$ for isotropic elastic materials as noted in Subsection 2.3.3.

2.4.10 Examples of compressible stretch-based strain-energy functions

In this work, selected strain-energy functions for compressible hyperelastic media are employed to demonstrate numerical simulations of the boundary value problem in principal space as well as to generate synthetic datasets for the training of artificial neural networks.

The first model is *Ogden-Storåkers* model (R. W. Ogden, Saccomandi, and Sgura, 2004; Storåkers, 1986), defined by

$$\Psi_{Ogden} = \sum_{i=1}^{M} \frac{\mu_i}{\alpha_i}\left[\lambda_1^{\alpha_i} + \lambda_2^{\alpha_i} + \lambda_3^{\alpha_i} - 3 + \frac{1}{\beta}\Big(J^{-\alpha_i\beta} - 1\Big)\right]. \tag{2.126}$$

where M is a positive integer, determining the number of terms in the strain energy function, μ_i are shear moduli, and α_i, β are the material constants. β relates to Possion's ratio ν by:

$$\beta = \frac{\nu}{1-2\nu}\,.$$

The ground-state shear modulus and the corresponding bulk modulus are

$$\mu_0 = \frac{1}{2}\sum_{i=1}^{M} \mu_i\alpha_i\,, \quad K_0 = \left(\beta + \frac{1}{3}\right)\sum_{i=1}^{M} \mu_i\alpha_i\,. \tag{2.127}$$

To ensure material stability in stress-free ground-states, i.e. $\mu_0 > 0$ and $K_0 > 0$, requiring the inequalities:

$$\mu_i \alpha_i > 0\,; \quad \beta > -\frac{1}{3}\,. \tag{2.128}$$

for each $i = 1, \ldots, M$. However, in general, if $M > 3$, these inequalities do not need to hold for every i (R. W. Ogden, Saccomandi, and Sgura, 2004).

The second model is a compressible version of the *Mooney-Rivlin* model, proposed by (Blatz and Ko, 1962). The strain-energy functions of the compressible Mooney-Rivlin model can be written using Eq. (2.126) by setting $M = \alpha_1 = -\alpha_2 = 2$ as follows:

$$\Psi_{MR} = \frac{\mu_1}{2}\left[\lambda_1^2 + \lambda_2^2 + \lambda_3^2 - 3 + \frac{1}{\beta}\left(\frac{1}{J^{2\beta}} - 1\right)\right] - \frac{\mu_2}{2}\left[\frac{1}{\lambda_1^2} + \frac{1}{\lambda_2^2} + \frac{1}{\lambda_3^2} - 3 + \frac{1}{\beta}\left(J^{2\beta} - 1\right)\right] \tag{2.129}$$

Chapter 3

Compressible non-linear elastic materials boundary value problems: principal space formulation

Contents

This chapter presents two novel contributions aimed at improving the understanding and efficient formulation of the isotropic compressible materials in principle space in the context of boundary value problems (BVPs). The first contribution is a novel concise formulation of the principle of virtual work in principal space for isotropic materials. This formulation simplifies the virtual work expression, requiring only the principal strains and stresses rather than the full tensorial forms typically used in the literature. Isotropic compressible hyperelastic materials are chosen and implemented to solve BVP benchmark examples. The benchmark scenarios are kept as close as possible to real-work working conditions, enabling to calibrate the mechanical parameters with experimental data.

This leads to the second contribution which involves the application of a new method for identifying the material parameters of hyperelastic materials using constrained optimization. By incorporating constraints into the optimization problem, highly accurate results that also satisfy the prior constraints are achieved. This new approach is particularly valuable for acquiring optimal material parameters that ensure the material to satisfy *prior* conditions without the need for post-processing techniques.

These novel contributions are implemented in FEniCSx to showcase the effectiveness of these new techniques in solving BVP benchmark examples. Notably, this marks the first time that stretch-based hyperelastic materials are implemented in FEniCSx. Subsequently, these benchmark examples are used to generate a dataset for training deep neural network (DNN)-based surrogate models, which will be explored in later chapters.

The current chapter starts with an overview of boundary-value problems and the principle of virtual work in Section 3.1. The linearization of the principle of virtual work in principal space will be discussed in Section 3.2, followed by the novel formulation of its vectorization in Section 3.3. Section 3.4 will present the constrained optimization method for hyperelastic material parameter identification. In Section 3.4, the novel techniques developed in the previous sections will be validated through comparison with experimental data. Section 3.5 validates the principal space formulation and the results of Section 3.4 through comparison with experimental data

3.1 Boundary value problems

Returning to subsection 2.2.1, recall that the domain $\Omega_0 \in \mathbb{R}^3$, with its boundary $\partial\Omega_0$ in the reference configuration. Now, let $\partial\Omega_D$ and $\partial\Omega_F$ define complementary partitions of $\partial\Omega_0$. On $\partial\Omega_D$, the displacement $\overline{\boldsymbol{u}}(\boldsymbol{X})$ is prescribed (Dirichlet boundary conditions), while on $\partial\Omega_F$, the first Piola-Kirchhoff traction vector $\bar{\boldsymbol{t}}_F(\boldsymbol{X})$ is prescribed (Neumann boundary conditions), such that $\partial\Omega = \partial\Omega_D \cup \partial\Omega_F, \partial\Omega_D \cap \partial\Omega_F = \varnothing$.

Assuming *static* deformations of the body, the governing equations, or the *strong form* of the nonlinear *boundary-value problem* in terms of the unknown displacement field $\boldsymbol{u}(\boldsymbol{X})$ in the

reference configuration are:

$$\operatorname{Div} \boldsymbol{P} + \boldsymbol{f}_B = \boldsymbol{0} \qquad \text{in} \quad \Omega_0\,, \tag{3.1}$$

$$\boldsymbol{u} = \overline{\boldsymbol{u}} \qquad \text{on} \quad \partial\Omega_D\,, \tag{3.2}$$

$$\boldsymbol{t}_F = \boldsymbol{P}\boldsymbol{n}_F = \overline{\boldsymbol{t}}_F \qquad \text{on} \quad \partial\Omega_F\,, \tag{3.3}$$

where $\operatorname{Div} \boldsymbol{P} = \frac{\partial P_{ij}}{\partial x_j}\boldsymbol{e}_i$. $\boldsymbol{P}$ and $\boldsymbol{f}_B$ represent the first Piola-Kirchhoff stress and the body forces term in reference configuration, respectively. $\boldsymbol{n}_F$ is the unit normal vector of the boundary surface $\partial\Omega_F$. Analytical solutions for the set of equations (3.1) - (3.3) are only possible for some special cases. Therefore, numerical methods such as the finite element method are often employed to approximate solutions based on the basis of variational principles.

To obtain the variational form, the strong form is multiplied by an arbitrary test function $\delta\boldsymbol{u}$ (in the sense of the scalar product) integrated over Ω_0. Then, applying the divergence theorem yields:

$$\mathcal{F}(\boldsymbol{u}, \delta\boldsymbol{u}) = \int_{\Omega_0} \left[\boldsymbol{P} : \delta\boldsymbol{F} - \boldsymbol{f}_B \cdot \delta\boldsymbol{u}\right] dV_0 - \int_{\partial\Omega_F} \overline{\boldsymbol{t}}_F \cdot \delta\boldsymbol{u}\, dS = 0 \quad \forall \delta\boldsymbol{u} \text{ s.t. } \delta\boldsymbol{u} = \boldsymbol{0} \text{ on } \partial\Omega_D\,. \tag{3.4}$$

Here, $\delta\boldsymbol{u}$ is also referred to as the virtual displacement field, and $\delta\boldsymbol{F}$ is the first variation of the deformation gradient $\boldsymbol{F}$, is given by

$$\delta\boldsymbol{F} = \frac{\partial \delta\boldsymbol{u}}{\partial \boldsymbol{X}}\,. \tag{3.5}$$

The principle of virtual work states that the work done by internal stresses within a deformed elastic body must be equal to the work done by external forces applied to the body. The functions:

$$\delta W_{int}(\boldsymbol{u}, \delta\boldsymbol{u}) = \int_{\Omega_0} \left[\boldsymbol{P} : \delta\boldsymbol{F}\right] dV_0\,, \tag{3.6}$$

$$\delta W_{ext}(\boldsymbol{u}, \delta\boldsymbol{u}) = \int_{\Omega_0} \boldsymbol{f}_B \cdot \delta\boldsymbol{u}\, dV_0 + \int_{\partial\Omega_F} \overline{\boldsymbol{t}}_F \cdot \delta\boldsymbol{u}\, dS\,, \tag{3.7}$$

are known as the *internal virtual work* and the *external virtual work*, respectively. Note that, here, this general equation has not incorporated any specific form of the constitutive closure $\boldsymbol{P}$ yet.

3.2 Principle of virtual work in principal space: Linearization

The finite element method is commonly employed to numerically solve the nonlinear boundary-value problem (3.4) and determine the unknown field $\boldsymbol{u}$. Newton-type methods are frequently used due to their efficiency in solving this type of problem. They require a *consistent*

linearization of all relevant quantities to generate *recurrence update formulas*. This approach replaces the nonlinear problem with a series of linear problems that are solved iteratively, resulting in a quadratic convergence rate near the solution.

To simplify the numerical demonstrations later on, the body force $\boldsymbol{f}_b$ and tractions $\bar{\boldsymbol{t}}_F$ are assumed to be independent of the deformation of the body. This implies that the corresponding linearization of the external virtual work (3.7) vanishes, i.e. $D_{\Delta \boldsymbol{u}} \delta \widehat{W}_{ext} = 0$. As a result, the linearization of the variational equation (3.4) only affects the internal virtual work δW_{int}.

Generally, the general work conjugate pairs $(\boldsymbol{T}^{(m)}, \boldsymbol{E}^{(m)})$, where m specifies the type of strain and stress, can be employed to present the generalised form of the internal virtual work δW_{int}:

$$\delta W_{int}(\boldsymbol{u}, \delta \boldsymbol{u}) = \int_{\Omega_0} \boldsymbol{T}^{(m)} : \delta \boldsymbol{E}^{(m)} \, dV_0 \,. \tag{3.8}$$

Considering the case of *isotropic elastic solid* only, the symmetric stress tensor $\boldsymbol{T}^{(m)}$ is coaxial with $\boldsymbol{E}^{(m)}$, equivalently, $\boldsymbol{T}^{(m)}$ is coaxial with $\boldsymbol{U}$ (R. W. Ogden, 1997, p. 160, 336), then, their spectral forms (subsections 2.2.4 and 2.3.3):

$$\boldsymbol{E}^{(m)} = \sum_{a=1}^{3} e_a(\boldsymbol{u}) \boldsymbol{N}_a \otimes \boldsymbol{N}_a \,; \quad \boldsymbol{T}^{(m)} = \sum_{a=1}^{3} t_a(e_a(\boldsymbol{u})) \boldsymbol{N}_a \otimes \boldsymbol{N}_a \,, \quad a = 1, 2, 3 \,. \tag{3.9}$$

Here, t_a and e_a are the principal stresses and strains, respectively. Equipping with these properties, the spectral representation of (3.8) is:

$$\delta W_{int}(\boldsymbol{u}, \delta \boldsymbol{u}) = \int_{\Omega_0} \left[\sum_{a=1}^{3} t_a \boldsymbol{N}_a \otimes \boldsymbol{N}_a : \delta \left(\sum_{a=1}^{3} e_a \boldsymbol{N}_a \otimes \boldsymbol{N}_a \right) \right] dV_0 \,. \tag{3.10}$$

This equation can be simplified as follows:

$$\delta W_{int}(\boldsymbol{u}, \delta \boldsymbol{u}) = \sum_{a=1}^{3} \int_{\Omega_0} t_a(e_a(\boldsymbol{u})) \delta e_a(\boldsymbol{u}) \, dV_0 \,. \tag{3.11}$$

Remark. Recall that $\{\boldsymbol{N}_a\}$ form an orthogonal basis with $|\boldsymbol{N}_a| = 1$, i.e. $\boldsymbol{N}_a \cdot \boldsymbol{N}_b = \delta_{ab}$ (subsection 2.2.4). Therefore, the change of a vector with constant length is always orthogonal to the vector itself (Holzapfel, 2000, p. 91), expressed as:

$$\boldsymbol{N}_a \cdot \delta \boldsymbol{N}_a = 0 \,. \tag{3.12}$$

Denote $\boldsymbol{M}_a = \boldsymbol{N}_a \otimes \boldsymbol{N}_a$ as the eigenprojectors of $\boldsymbol{U}$ with identities $\boldsymbol{M}_a : \boldsymbol{M}_b = \delta_{ab}$. Furthermore:

$$\begin{aligned}
\boldsymbol{M}_a : \delta\boldsymbol{M}_a &= (\boldsymbol{N}_a \otimes \boldsymbol{N}_a) : \delta(\boldsymbol{N}_a \otimes \boldsymbol{N}_a) \\
&= (\boldsymbol{N}_a \otimes \boldsymbol{N}_a) : (\delta\boldsymbol{N}_a \otimes \boldsymbol{N}_a + \boldsymbol{N}_a \otimes \delta\boldsymbol{N}_a) \\
&= (\boldsymbol{N}_a \otimes \boldsymbol{N}_a) : (\delta\boldsymbol{N}_a \otimes \boldsymbol{N}_a) + (\boldsymbol{N}_a \otimes \boldsymbol{N}_a) : (\boldsymbol{N}_a \otimes \delta\boldsymbol{N}_a) \\
&= \underbrace{(\boldsymbol{N}_a \cdot \delta\boldsymbol{N}_a)}_{0}(\boldsymbol{N}_a \cdot \boldsymbol{N}_a) + (\boldsymbol{N}_a \cdot \boldsymbol{N}_a)\underbrace{(\boldsymbol{N}_a \cdot \delta\boldsymbol{N}_a)}_{0} = 0 \qquad \text{(apply (3.12))}
\end{aligned}$$

Consequently, the internal virtual work in the spectral form can be simplified as follows:

$$\begin{aligned}
\delta W_{int}(\boldsymbol{u}, \delta\boldsymbol{u}) &= \int_{\Omega_0} \left[\sum_{a=1}^{3} t_a \boldsymbol{N}_a \otimes \boldsymbol{N}_a : \delta \left(\sum_{a=1}^{3} e_a \boldsymbol{N}_a \otimes \boldsymbol{N}_a \right) \right] dV_0 \\
&= \int_{\Omega_0} \left[\sum_{a=1}^{3} t_a \boldsymbol{M}_a : \delta \left(e_a \boldsymbol{M}_a \right) \right] dV_0 \\
&= \int_{\Omega_0} \left[\sum_{a=1}^{3} t_a \boldsymbol{M}_a : (\delta e_a \boldsymbol{M}_a + e_a \delta \boldsymbol{M}_a) \right] dV_0 \\
&= \int_{\Omega_0} \sum_{a=1}^{3} \left[t_a \delta e_a \underbrace{(\boldsymbol{M}_a : \boldsymbol{M}_a)}_{1} + t_a e_a \underbrace{(\boldsymbol{M}_a : \delta \boldsymbol{M}_a)}_{0} \right] dV_0 \\
&= \int_{\Omega_0} \sum_{a=1}^{3} t_a \delta e_a \, dV_0.
\end{aligned}$$

□

3.3 Principle of virtual work in principal space: Vectorization

Let $\boldsymbol{e}^{(m)} = [e_1, e_2, e_3]^\top$ and $\boldsymbol{t}^{(m)} = [t_1, t_2, t_3]$ be the vectors of the principal generalised strains and stresses, respectively, and $\delta\boldsymbol{e}^{(m)} = [\delta e_1, \delta e_2, \delta e_3]^\top$ be the variation of $\boldsymbol{e}^{(m)}$. Then, the principle of virtual work in terms of principal values (Eq. (3.11)) can be expressed in vectorized form as:

$$\delta W_{int}(\boldsymbol{u}, \delta\boldsymbol{u}) = \int_{\Omega_0} \boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right) \cdot \delta\boldsymbol{e}^{(m)}(\boldsymbol{u}) dV_0 \tag{3.13}$$

Linearisation of (3.13) can be expressed as:

$$D_{\Delta \boldsymbol{u}} \delta W_{int} = \int_{\Omega_0} \left[\boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right) \cdot D_{\Delta \boldsymbol{u}} \delta \boldsymbol{e}^{(m)}(\boldsymbol{u}) + \delta \boldsymbol{e}^{(m)}(\boldsymbol{u}) \cdot D_{\Delta \boldsymbol{u}} \boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right) \right] dV_0 \tag{3.14}$$

$$= \int_{\Omega_0} \left[\delta \boldsymbol{e}^{(m)}(\boldsymbol{u}) \cdot D_{\Delta \boldsymbol{u}} \boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right) \right] dV_0 \,, \tag{3.15}$$

where $D_{\Delta \boldsymbol{u}} \delta \boldsymbol{e}^{(m)}(\boldsymbol{u})$ characterizes the directional derivative of $\delta \boldsymbol{e}^{(m)}(\boldsymbol{u})$ at $\boldsymbol{u}$ in the direction of $\Delta \boldsymbol{u}$. Similarly, $D_{\Delta \boldsymbol{u}} \boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right)$ is the linearization of $\boldsymbol{t}^{(m)}$, which is specified by:

$$D_{\Delta \boldsymbol{u}} \boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right) = \frac{\partial \boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right)}{\partial \boldsymbol{e}^{(m)}(\boldsymbol{u})} \cdot D_{\Delta \boldsymbol{u}} \boldsymbol{e}^{(m)}(\boldsymbol{u}) = \boldsymbol{H}^{(m)}(\boldsymbol{u}) \cdot D_{\Delta \boldsymbol{u}} \boldsymbol{e}^{(m)}(\boldsymbol{u}) \,, \tag{3.16}$$

where $\boldsymbol{H}^{(m)}(\boldsymbol{u}) = \frac{\partial \boldsymbol{t}^{(m)} \left(\boldsymbol{e}^{(m)}(\boldsymbol{u}) \right)}{\partial \boldsymbol{e}^{(m)}(\boldsymbol{u})}$.

The final linearization form of Eq. (3.13) reads:

$$D_{\Delta \boldsymbol{u}} \delta W_{int} = \int_{\Omega_0} \left[\delta \boldsymbol{e}^{(m)}(\boldsymbol{u}) \cdot \boldsymbol{H}^{(m)}(\boldsymbol{u}) \cdot D_{\Delta \boldsymbol{u}} \delta \boldsymbol{e}^{(m)}(\boldsymbol{u}) \right] dV_0 \,. \tag{3.17}$$

The novel formulation in Eq. (3.17) only involves vectors and second-order tensors, which, to the best of the author's knowledge, has not been introduced in the literature. It rewrites the principal virtual work in the spectral representation of isotropic elastic materials to be represented in a highly concise. It greatly simplifies the numerical implementation of the boundary-value problems in the case of stretch-based isotropic elastic materials (such as Ogden model, Shariff model (Shariff, 2000), etc.).

Considering the case of $m = 2$, the generalized strain (2.40) becomes the Green-Lagrange strain $\boldsymbol{E}$ and has the conjugate pair with the 2^{nd} Piola-Kirchhoff stress $\boldsymbol{S}$. Let $\boldsymbol{c} = [c_1, c_2, c_3]^\top$ and $\boldsymbol{s} = [s_1, s_2, s_3]$ be the vectors of the principal values of $\boldsymbol{C}$ and $\boldsymbol{S}$, respectively. Then, the vector of principal values of $\boldsymbol{E}$ is $\boldsymbol{e}^{(2)} = \frac{1}{2}(\boldsymbol{c} - \boldsymbol{1})$, where $\boldsymbol{1} = [1, 1, 1]^\top$. The derivative of $\boldsymbol{s}$ with respect to $\boldsymbol{c}$ reads:

$$H^{(2)}_{ab}(\boldsymbol{u}) = \frac{\partial t^{(2)}_a}{\partial e^{(2)}_b} = 2 \frac{\partial s_a}{\partial (c_b - 1)} = 2 \frac{\partial s_a}{\partial c_b} \frac{\partial c_b}{\partial (c_b - 1)} = 2 \frac{\partial s_a}{\partial c_b} \frac{1}{\frac{\partial c_b}{\partial (c_b - 1)}} = 2 \frac{\partial s_a}{\partial c_b} \,. \tag{3.18}$$

It follows that,

$$D_{\Delta \boldsymbol{u}} \delta W_{int} = \frac{1}{2} \int_{\Omega_0} \left[\delta \boldsymbol{c}(\boldsymbol{u}) \cdot \boldsymbol{H}^{(2)}(\boldsymbol{u}) \cdot D_{\Delta \boldsymbol{u}} \boldsymbol{c}(\boldsymbol{u}) \right] dV_0 \,, \tag{3.19}$$

where, $\delta \boldsymbol{c}$ is the variation of $\boldsymbol{c}$.

3.4 Fitting hyperelastic model's parameters using constrained optimization method

Elastic materials such as rubber, polymer foams, and soft tissues, require mechanical testing to determine their properties. Different tests are necessary depending on the property of interest. For example, the isotropy property can be determined by uniaxial tension/compression tests, where the material can be considered isotropic if the stretches in the lateral directions of the loading direction are not varied too much. Shear tests help determine the shear modulus of the material, which measures how much a material resists deformation when subjected to shear stress. Volumetric compression tests or imaging modalities such as ultrasound can be used to identify the compressibility property.

Common materials testing usually consists of several modes of mechanical tests, such as uniaxial, volumetric, biaxial, pure shear, simple shear, or combinations of those. In this study, compressible hyperelastic materials namely polyethylene foam is of interest. However, experimental data for certain types of materials are often scarce. Therefore, to generate a sufficiently rich dataset for data-driven methods, data from both, experiments and numerical simulations, are often combined.

For hyperelastic materials, a process called material parameter identification is employed to fit the model's strain energy function to experimental test cases. Through this process, the optimal model is then employed in numerical simulations to generate an accurate and reliable synthetic dataset for the training process of ANNs.

To model the finite deformation of compressible hyperelastic materials, more detailed experimental data are necessary than for incompressible materials since the lateral stretches can be calculated from the incompressibility condition. Information about the lateral stretches λ_T is important for determining the compressibility-related parameters of the strain energy Ψ, where $J \neq 1$ (Anssari-Benam and Horgan, 2022; Kossa and Berezvai, 2016; El-Ratal and Mallick, 1996). While many studies have reported on simple tension/compression of polymeric foams, there are relatively few that include information on the lateral stretches λ_T. Therefore, this work utilises experimental datasets of polymeric foams that include λ_T, namely uniaxial and equibiaxial compression (Kossa and Berezvai, 2016).

The Ogden-Storåkers strain energy function (eq. (2.126)) is re-calibrated to achieve better performance compared to that of (Kossa and Berezvai, 2016), with additional regularization terms $\mu_0 > 0$ and $K_0 > 0$. The principal values of the first Piola-Kirchhoff stress tensor generated by the model (2.126) are:

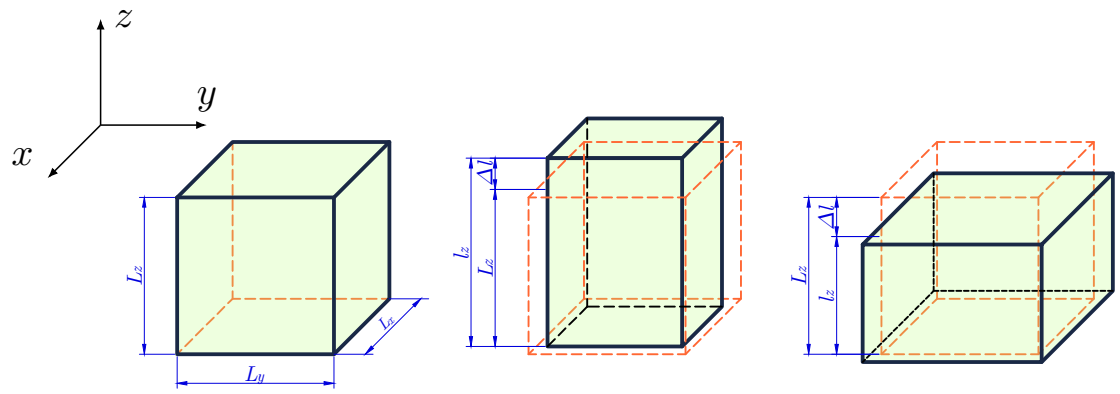
$$p_a = \frac{\partial \Psi_{Ogden}}{\partial \lambda_a}\,, \tag{3.20}$$

which can be easily computed using automatic differentiation[1], hence, explicit forms are not needed.

3.4.1 Uniaxial extension/compression (UA)

Consider a cube of isotropic material with edges parallel to the coordinate axes ($(x, y, z) \in \mathbb{R}^3$), having side lengths of $L_x = L_y = L_z = 1\,\text{mm}$ in the reference configuration. To induce axial tension/compression, a displacement $\overline{u}_z = \Delta l$ is prescribed along the third direction z (Fig. 3.1). The resulting dimensions of the deformed cube in the current configuration are l_x, l_y, l_z. The

Figure 3.1 Uniaxial tests on a unit cube under the prescribed displacement $\overline{u}_z = \Delta l$.



(a) Unit cube (b) Uniaxial tension (c) Uniaxial compression

stretch along the stretching direction is denoted by $\lambda_3 = \frac{l_z}{L_z} = \lambda$, where $\lambda > 1$ indicate tension, and $0 < \lambda < 1$ indicates compression. The lateral (transverse) stretches are defined as $\lambda_1 = \frac{l_x}{L_x}$ and $\lambda_2 = \frac{l_y}{L_y}$. In the case of isotropic material, $\lambda_1 = \lambda_2 = \lambda_T$, subscript T refers to the transverse direction.

The strain energy now is a function of λ and λ_T, i.e. $\Psi^{UA}(\lambda_T, \lambda_T, \lambda)$. In the uniaxial tension/compression test, the minor principal nominal stresses satisfy the zero transverse stress constraints:

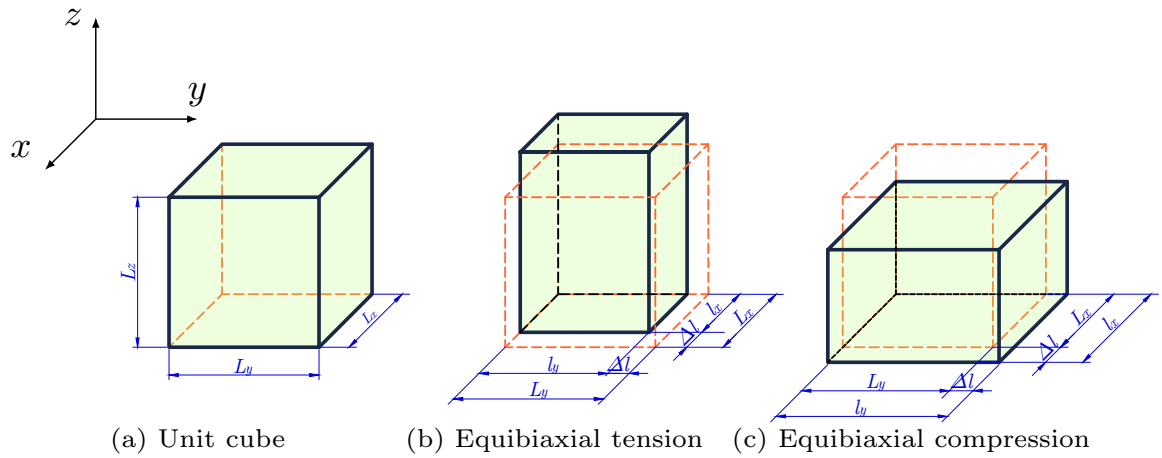
$$p_1^{UA} = p_2^{UA} = \frac{\partial \Psi^{UA}(\lambda_T, \lambda_T, \lambda)}{\partial \lambda_T} = 0\,, \tag{3.21}$$

[1]Automatic differentiation (`autodiff`) is a technique that automatically constructs a computational procedure for efficiently computing the gradients of mathematical functions. Therefore, it can simplify and improve the process of parameter identification in hyperelastic materials. For instance, by employing `autodiff`, the gradient of the strain energy with respect to the material parameters can be automatically computed, which is subsequently used in optimization algorithms to find the best-fit parameters. `JAXopt` (Blondel et al., 2022), a library built on top of `JAX` (Bradbury et al., 2018) framework are such tools that have been specifically designed for this purpose.

3.4.2 Equibiaxial extension/compression (BA)

For equibiaxial extension/compression, the specimen is compressed/stretched equally in two orthogonal directions. Displacements $\overline{u}_x = \overline{u}_y = \Delta l$ are prescribed on the facets along x-axis and y-axis of the unit cube (Fig. 3.2). The stretches along the compressing/stretching direction

Figure 3.2 Equibiaxial tests on a unit cube under the prescribed displacement $\overline{u}_x = \overline{u}_y = \Delta l$.



(a) Unit cube (b) Equibiaxial tension (c) Equibiaxial compression

are denoted by $\lambda_1 = \lambda_2 = \frac{l_x}{L_x} = \frac{l_y}{L_y} = \lambda$ and the lateral stretch is $\lambda_3 = \frac{l_z}{L_z} = \lambda_T$. The strain energy becomes $\Psi^{BA}(\lambda, \lambda, \lambda_T)$. Hence, the nominal stresses in the loading directions are equal, that is

$$p_1^{BA} = p_2^{BA} = \frac{\partial \Psi^{BA}(\lambda, \lambda, \lambda_T)}{\partial \lambda} \,. \tag{3.22}$$

The minor principal nominal stress satisfies the zero transverse stress constraints:

$$p_3^{BA} = \frac{\partial \Psi^{BA}(\lambda, \lambda, \lambda_T)}{\partial \lambda_T} = 0 \,, \tag{3.23}$$

3.4.3 Constrained optimization for parameter identification

Table 3.1[1] presents the experimental data used in this work, which consists of uniaxial compression and equibiaxial compression tests. Let $\mathbf{\Lambda}^{(t)}$ ((t): the type of the test case) be the matrix containing column vectors of a measure of stretch appropriate to the considered

[1]Note that a typographical error exists in the original publication (Kossa and Berezvai, 2016), where the unit of the uniaxial test is stated incorrectly. The correct unit for the uniaxial test is KPa, which has been confirmed by Professor Kossa through communication with the author.

Uniaxial compression			Biaxial compression		
$\lambda_i^{UA}[-]$	$\lambda_{Ti}^{UA}[-]$	p_i^{UA}[kPa]	$\lambda_i^{BA}[-]$	$\lambda_{Ti}^{BA}[-]$	p_i^{BA}[kPa]
0.95	1.00570	-9.10	0.95	1.01140	-13.67
0.90	1.01212	-17.30	0.90	1.02424	-25.99
0.85	1.01786	-26.00	0.85	1.03572	-36.04
0.80	1.02311	-33.80	0.80	1.04622	-47.33
0.75	1.02915	-41.41	0.78	1.05830	-60.36
0.70	1.03551	-50.36	0.70	1.07102	-75.82
0.65	1.04217	-61.15	0.65	1.08434	-96.05
0.60	1.04962	-74.04	0.60	1.09924	-123.13
0.55	1.05751	-89.57	0.55	1.11502	-166.03
0.50	1.06596	-108.67			
0.45	1.07543	-132.54			
0.40	1.08615	-163.49			
0.35	1.09849	-204.91			
0.30	1.11285	-263.43			
0.25	1.12816	-353.57			

Table 3.1 Experimental data for a polyethylene foam sample under uniaxial and equibiaxial compression, acquired from Kossa and Berezvai (2016).

deformation, given by

$$\boldsymbol{\Lambda}^{(UA)} = \begin{bmatrix} \lambda_1^{(UA)} & \lambda_{T1}^{(UA)} & \lambda_{T1}^{(UA)} \\ \lambda_2^{(UA)} & \lambda_{T2}^{(UA)} & \lambda_{T2}^{(UA)} \\ \vdots & \vdots & \vdots \\ \lambda_n^{(UA)} & \lambda_{Tn}^{(UA)} & \lambda_{Tn}^{(UA)} \end{bmatrix}; \qquad \boldsymbol{\Lambda}^{(BA)} = \begin{bmatrix} \lambda_1^{(BA)} & \lambda_1^{(BA)} & \lambda_{T1}^{(BA)} \\ \lambda_2^{(BA)} & \lambda_2^{(BA)} & \lambda_{T2}^{(BA)} \\ \vdots & \vdots & \vdots \\ \lambda_n^{(BA)} & \lambda_n^{(BA)} & \lambda_{Tn}^{(BA)} \end{bmatrix}. \tag{3.24}$$

Here, n is the number of experimental data points.

Let $\boldsymbol{Y}^{(t)}$ be the matrix containing the column vectors of experimental nominal stress corresponding to the stretch matrix $\boldsymbol{\Lambda}^{(t)}$, given by

$$\boldsymbol{Y}^{(UA)} = \begin{bmatrix} p_1^{(UA)} & p_{T1}^{(UA)} & p_{T1}^{(UA)} \\ p_2^{(UA)} & p_{T2}^{(UA)} & p_{T2}^{(UA)} \\ \vdots & \vdots & \vdots \\ p_n^{(UA)} & p_{Tn}^{(UA)} & p_{Tn}^{(UA)} \end{bmatrix} = \begin{bmatrix} p_1^{(UA)} & 0 & 0 \\ p_2^{(UA)} & 0 & 0 \\ \vdots & \vdots & \vdots \\ p_n^{(UA)} & 0 & 0 \end{bmatrix}; \tag{3.25}$$

$$\boldsymbol{Y}^{(BA)} = \begin{bmatrix} p_1^{(BA)} & p_1^{(BA)} & p_{T1}^{(BA)} \\ p_2^{(BA)} & p_2^{(BA)} & p_{T2}^{(BA)} \\ \vdots & \vdots & \vdots \\ p_n^{(BA)} & p_n^{(BA)} & p_{Tn}^{(BA)} \end{bmatrix} = \begin{bmatrix} p_1^{(BA)} & p_1^{(BA)} & 0 \\ p_2^{(BA)} & p_2^{(BA)} & 0 \\ \vdots & \vdots & \vdots \\ p_n^{(BA)} & p_n^{(BA)} & 0 \end{bmatrix}, \tag{3.26}$$

where the zero-columns are the transverse stress conditions (3.21) and (3.23) as proposed in (Kossa and Berezvai, 2016)[1]. The strain-energy function Ψ represents the material model, from which the predicted stress can be calculated and assembled as $\widehat{\boldsymbol{Y}}^{(t)}(\boldsymbol{\Lambda}^{(t)}, \boldsymbol{\gamma})$, where $\widehat{(\cdot)}$ denotes the predicted stress matrix and $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_m]$ represents a set of m material parameters to be identified.

Let $\mathcal{L}^{(t)}$ be the objective function defined as the sum of the mean squared values corresponding to each test (t):

$$\mathcal{L}^{(t)} = \frac{1}{n} \sum_{i}^{n} \left[\widehat{\boldsymbol{Y}}_i^{(t)}(\Lambda_i^{(t)}, \boldsymbol{\gamma}) - \boldsymbol{Y}_i^{(t)} \right]^2 , \tag{3.27}$$

where $\widehat{\boldsymbol{Y}}_i^{(t)}$ and $\Lambda_i^{(t)}$ are the i^{th} row of the predicted stress matrix $\widehat{\boldsymbol{Y}}$ and $\boldsymbol{\Lambda}^{(t)}$, respectively. Here, the zero transverse stress conditions, Eq (3.21) and Eq (3.23), are automatically fulfilled. Furthermore, the conditions (2.128) are imposed to achieve admissible solutions and material stability (R. W. Ogden, Saccomandi, and Sgura, 2004). The objective functions associated with these datasets and constraint conditions are as follows:

$$\mathcal{L}^{(UA)}(\boldsymbol{\gamma}) = \frac{1}{n} \sum_{i}^{n} \left[\widehat{\boldsymbol{Y}}_i^{(UA)}(\Lambda_i^{(UA)}, \boldsymbol{\gamma}) - \boldsymbol{Y}_i^{(UA)} \right]^2 , \tag{3.28}$$

$$\mathcal{L}^{(BA)}(\boldsymbol{\gamma}) = \frac{1}{n} \sum_{i}^{n} \left[\widehat{\boldsymbol{Y}}_i^{(BA)}(\Lambda_i^{(BA)}, \boldsymbol{\gamma}) - \boldsymbol{Y}_i^{(BA)} \right]^2 \tag{3.29}$$

$$\mathcal{L}_{total}(\boldsymbol{\gamma}) = \mathcal{L}^{(UA)}(\boldsymbol{\gamma}) + \mathcal{L}^{(BA)}(\boldsymbol{\gamma}) \tag{3.30}$$

$$\boldsymbol{\gamma}^* = \arg\min_{\boldsymbol{\gamma}} \mathcal{L}_{total} \tag{3.31}$$

$$\text{subject to } \sum_{i}^{M} \mu_i \alpha_i > 0 \quad \text{and} \quad \beta > -\frac{1}{3} . \tag{3.32}$$

$\widehat{\boldsymbol{Y}}^{(t)}(\boldsymbol{\Lambda}^{(t)}, \boldsymbol{\gamma})$ is nonlinear with respect to $\boldsymbol{\gamma}$, a non-linear least squares (NLS) problem arises, which can be numerically solved using the `scipy.optimize` package in `scipy` (Virtanen et al., 2020), offering a range of optimization algorithms for constrained and unconstrained problems. Conveniently, `jaxopt` has wrappers for `scipy.optimize` package which provides automatic differentiation and GPU acceleration, making it a suitable tool for large-scale optimization problems, as demonstrated in this work. Since (3.31) is a constrained non-linear optimization problem, the Sequential Least Squares Programming (SLSQP) method (Kraft, 1988) is adopted. As the NLS algorithm iterates towards a minimum for the problem (3.31), either a precision goal of `ftol` $= 1 \times 10^{-15}$ or a maximum of 1000 iterations will stop the iterative technique to obtain an optimal solution $\boldsymbol{\gamma}^*$. The overall performance of the calibrated model is measured through the *coefficient of determination* R^2, expressing the ratio of the variance of the predicted values to the variance of the observation values. The better the

[1] Assembling the pairs of data values $(\boldsymbol{\Lambda}^{(t)}, \boldsymbol{Y}^{(t)})$ in this form allows for the utilization of batchable and differentiable algorithms provided by `JAXopt`, resulting in a significant speedup of the optimization process.

model fits the data, the closer value of R^2 to 1, hence, $R^2 = 1$ indicates a perfect fit, the coefficient is expressed by:

$$R^2 = 1 - \frac{\sum_i^n (\widehat{y}_i - y_i)^2}{\sum_i^n (y_i - \overline{y})^2}\,, \tag{3.33}$$

where y_i and $\widehat{y}_i$ are the observation and predicted outputs values, respectively; $\overline{y}$ is the mean of all the observation data, and again, n is the number of data points. To evaluate the performance at a specific stretch value, *relative error* is put in use:

$$err = 100 \times \frac{|\widehat{y}_i - y_i|}{\max(0.5, |\widehat{y}_i|)}\,; \quad \text{for } i = 1, \ldots, n\,. \tag{3.34}$$

Here, a slight modification of the usual definition is that a factor 0.5 in the denominator to avoid division by small values of $|\widehat{y}_i|$ when the stretches are close to 1 which implies zero stress ($\widehat{y}_i = \widehat{p}_i = 0$).

Separately fitting the model on each UA or BA dataset yields poor performance for the other; i.e. the best-fitted optimal parameters for UA case showed worse agreement for BA case and vice versa, and therefore, these results are not presented. However, a better fit can be obtained by fitting both sets of Kossa's data simultaneously. The fitting process yielded the following observations:

- Possibility of finding a large number of optimal sets of parameters[1];
- The number and values of the optimal parameter sets are dependent on the parameter M, and are sensitive to the numerical accuracy requirements of the NLS algorithm used, such as the maximum number of iterations allowed, tolerance errors, and other relevant factors.

Table 3.2 shows three optimal sets of parameter values for hyperelastic models, namely the Mooney-Rivlin (2.129), the Ogden-Storåkers model (2.126) with $M = 2$ and $M = 3$. All three models were simultaneously fitted to simple compression and biaxial compression data. The optimal parameter sets have been fitted to the data with high accuracy as well as satisfied all the constraints (2.128).

The performance of both the 2-parameter and 3-parameter Ogden-Storåkers models is close to each other, whereas Mooney-Rivlin lags behind. This outcome is expected since the MR model has two fixed parameters: $\alpha_1 = -\alpha_2 = 2$, whereas the Ogden-Storåkers ($M = 2$) has all 5 optimized parameters. Ogden-Storåkers ($M = 3$) exhibits the best performance, reflected in the highest values of $R^2_{UA} = 0.999\,72$ and $R^2_{BA} = 0.999\,45$ for the UA and BA cases, respectively. Since the performance both Ogden-Storåkers based models are nearly

[1] Multiple sets of parameters can be found such that the model fits well with experimental data. Hence, a set of optimal parameters is not unique.

similar, the 3-parameter Ogden-Storåkers model is picked for further investigation and referred as to Ogden-Storåkers model only, for simplicity.

i	μ_i	α_i	β	R^2
		Mooney-Rivlin (2.129)		
1	0.058 05	2	0.25	0.994 02 (UA)
2	0.000 85	−2		0.971 63 (BA)
		Ogden-Storåkers ($M = 2$)		
1	1.993×10^{-9}	−10.998 05	0.105 41	0.996 91 (UA)
2	0.026 11	10.846 40		0.999 20 (BA)
Constraints	$\sum_i^3 \mu_i\alpha_i = 0.28319 > 0$		$\beta > -\frac{1}{3}$	
		Ogden-Storåkers ($M = 3$)		
1	0.009 86	−1.322 87	0.105 85	0.999 72 (UA)
2	5.107 38	0.008		0.999 45 (BA)
3	0.018 99	11.9025		
Constraints	$\sum_i^3 \mu_i\alpha_i = 0.25376 > 0$		$\beta > -\frac{1}{3}$	

Table 3.2 Parameter values for the Mooney-Rivlin, the Ogden-Storåkers model with $M = 2$ and $M = 3$, simultaneously fitted to simple compression and biaxial compression data in Table 3.1. All conditions are passed with high accuracy of R^2 values.

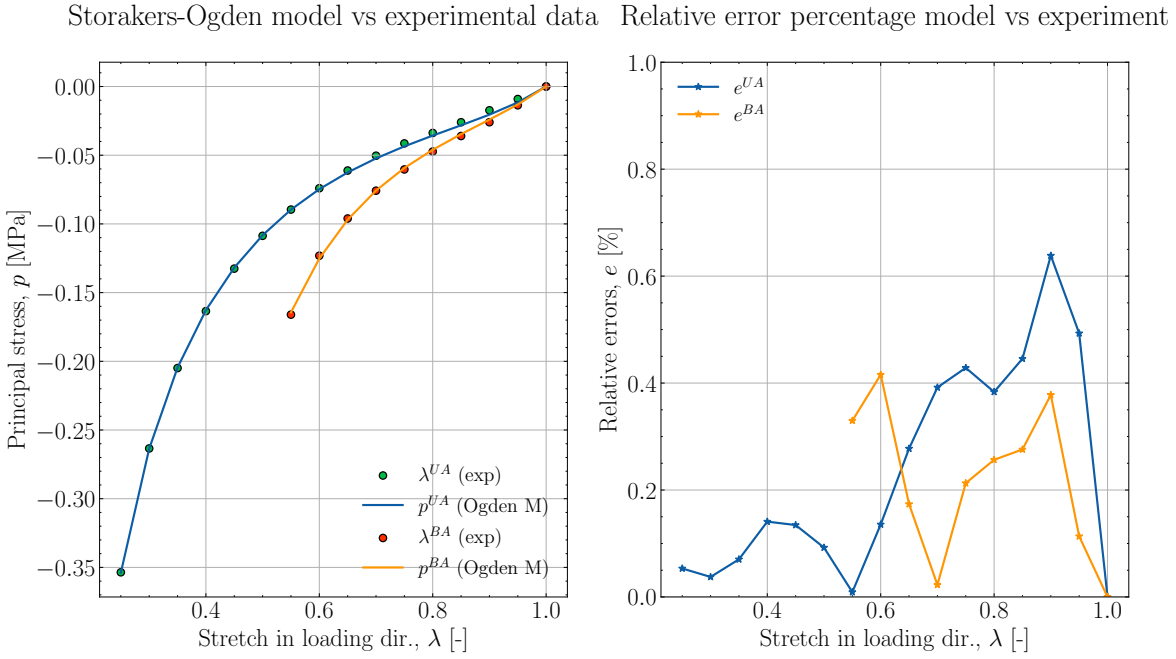
Fig. 3.3 shows the comparison between the experimental data and the fitted Ogden-Storåkers model for both uniaxial (UA) and biaxial (BA) cases. In the figure, the unit of stress is converted from kPa to MPa. The left plot compares the experimental principal values of the 1st Piola-Kirchhoff stresses to the stretches in the loading direction for both cases. The predicted principal stresses for UA and BA (p^{UA} and p^{BA}) are presented by the blue and orange solid lines, respectively, whereas the experimental data points (λ^{UA} and λ^{BA}) are represented by the filled markers, green dots for UA case and red dots for the BA case. The graph indicates a good fit between the model and the experimental data for both cases. The model predictions start at zero stress for $\lambda = 1$ and follow a similar trend for both cases, with decreasing stresses as λ decreases. The asymptotically decreasing trend of stresses with stretches indicates that the material is stiffer when compressed more in the loading direction. Additionally, the stresses are zero $p = 0$ at $\lambda = 1$ and decrease exponentially as the stretches λ approach zero, confirming the growth and normal requirements of hyperelasticity as discussed in Sect. 2.4.9.

The right plot shows the relative error percentage (Eq. (3.34)) between the model predictions and the experimental data for both cases. The errors of the UA case, err^{UA}, are shown in the blue asterisk line, and the ones for BA, err^{BA}, are shown in orange. The relative error percentages are all less than 1%, which demonstrates the high accuracy of the model. However, the graph also shows that there are some deviations from the model prediction, particularly

for UA at higher values of λ in the region $0.8 \leq \lambda \leq 1.0$, which could be due to the sensitive nature of the Ogden-Storåkers model's parameters in this region, as previously reported in Sect. 4.2 by R. W. Ogden, Saccomandi, and Sgura (2004).

Overall, the graph suggests that the model prediction is a good approximation of the experimental data, and satisfies all the constitutive requirements proposed in Sect. 2.4. This indicates that the Ogden-Storåkers model with the identified parameters in Table 3.2 is a good candidate for presenting this experimental dataset.

Figure 3.3 Comparison of the parameter-identified Ogden-Storåkers model ($M = 3$) and experimental data. The left graph shows the model is simultaneously fitted to uniaxial compression data (green dots) and biaxial compression data (red dots). The stress values of the fitted model are represented by the blue and orange lines for the uniaxial and biaxial cases, respectively. On the right graph, the relative errors [%] of the uniaxial (blue) and biaxial (orange) compression are plotted against the stretches.



3.5 Numerical model validation with experimental data

In this subsection, the principal space formulation and set of optimal parameters for the stretch-based Ogden-Storåkers model, developed in the previous sections, will be validated through comparison with experimental data. Finite element (FE) simulations of a compressible polyethylene foam unit cube specimen, subjected to UA and BA compression tests, are conducted to assess the accuracy and reliability of the proposed methods. In this work, `FEniCSx`[1] is employed to seek solutions to the boundary-value problem. By comparing the

[1] `FEniCSx` (Alnaes et al., 2014; Kirby and Logg, 2006; Logg and Wells, 2010a; Logg, Wells, and Hake, 2012; Scroggs, Baratta, et al., 2022; Scroggs, Dokken, et al., 2022) is a free and open-source finite element analysis

simulation results to published experimental data (Table 3.1), the effectiveness of the principal space formulation and constrained optimization techniques can be thoroughly assessed. This validation process is essential for ensuring that the novel approaches presented in this thesis are suitable for practical applications in the study of hyperelastic materials and related boundary value problems.

3.5.1 Weak form of uniaxial/equibiaxial tests

For UA and BA tests described in subsections 3.4.1 and 3.4.2, the prescribed displacement boundary conditions are $\overline{u}_z^{(UA)} = \Delta l$ (Fig. 3.1) for UA case, and $\overline{u}_x^{(BA)} = \overline{u}_y^{(BA)} = \Delta l$ (Fig. 3.2) for BA. The deformation gradients $\boldsymbol{F}^{(UA)}$, $\boldsymbol{F}^{(BA)}$ of the UA and BA tests, respectively, are as follows:

$$\boldsymbol{F}^{(UA)} = \begin{bmatrix} \frac{u_x^{(UA)}}{L_x} + 1 & 0 & 0 \\ 0 & \frac{u_y^{(UA)}}{L_y} + 1 & 0 \\ 0 & 0 & \frac{u_z^{(UA)}}{L_z} + 1 \end{bmatrix} = \begin{bmatrix} \lambda_T & 0 & 0 \\ 0 & \lambda_T & 0 \\ 0 & 0 & \lambda \end{bmatrix}; \tag{3.35}$$

$$\boldsymbol{F}^{(BA)} = \begin{bmatrix} \frac{u_x^{(BA)}}{L_x} + 1 & 0 & 0 \\ 0 & \frac{u_y^{(BA)}}{L_y} + 1 & 0 \\ 0 & 0 & \frac{u_z^{(BA)}}{L_z} + 1 \end{bmatrix} = \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda_T \end{bmatrix}. \tag{3.36}$$

For the isotropic case, the corresponding left and right Cauchy-Green tensors are equal and can be expressed as:

$$\boldsymbol{C}^{(UA)} = \boldsymbol{b}^{(UA)} = \begin{bmatrix} \lambda_T^2 & 0 & 0 \\ 0 & \lambda_T^2 & 0 \\ 0 & 0 & \lambda^2 \end{bmatrix}; \tag{3.37}$$

$$\boldsymbol{C}^{(BA)} = \boldsymbol{b}^{(BA)} = \begin{bmatrix} \lambda^2 & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & \lambda_T^2 \end{bmatrix}. \tag{3.38}$$

The vector $\boldsymbol{c}^{(t)}$ of three principal values of $\boldsymbol{C}^{(t)}$ can be expressed as (recall subsection 3.3):

$$\boldsymbol{c}^{(UA)} = [c_1^{(UA)}, c_2^{(UA)}, c_3^{(UA)}] = [\lambda_T^2, \lambda_T^2, \lambda^2]; \tag{3.39}$$

$$\boldsymbol{c}^{(BA)} = [c_1^{(BA)}, c_2^{(BA)}, c_3^{(BA)}] = [\lambda^2, \lambda^2, \lambda_T^2] \tag{3.40}$$

3.5.2 Numerical results

The benchmark simulations consist of UA and BA tests performed on a compressible 3D closed-cell polyethylene foam unit cube with the dimension of 1 mm × 1 mm × 1 mm. The

framework that provides a flexible and efficient platform for solving partial differential equations numerically. It allows users to quickly develop and solve complex mathematical models using the finite element method.

numerical model was utilized on a mesh composed of $20 \times 20 \times 20$ grid, comprising 9261 nodes and 52808 tetrahedron elements.

In the case of UA compression test, the boundary conditions are defined as follows: at the top side (the $x - y$ facet at $z = 1.0$), the z-component of $\boldsymbol{u}^{(UA)}$ is subjected to stepped displacements, which are prescribed to match the stretch values of UA case listed in Table 3.1. For instance, a displacement at $u_z^{(UA)} = -0.3\,\text{mm}$ is prescribed, resulting in stretch values of $\lambda^{(UA)} = 0.7$ in the negative direction along the z-axis. At the bottom face, the boundary conditions are composed as follows:

- The z-component of all nodes on the bottom side (the $x - y$ facet at $z = 0$) is fixed.
- The x-component of all nodes along the line $A_1(0.5, 0.0, 0.0) - B_1(0.5, 1.0, 0.0)$ is fixed.
- The y-component of all nodes along the line $A_2(0.0, 0.5, 0.0) - B_2(1.0, 0.5, 0.0)$ is fixed.
- The z-component of all nodes on the top side (the $x - y$ facet at $z = 1.0$) is subjected to stepped displacements which match the stretch values of UA case listed in Table 3.1.

In the case of BA compression test, the following boundary conditions are defined:

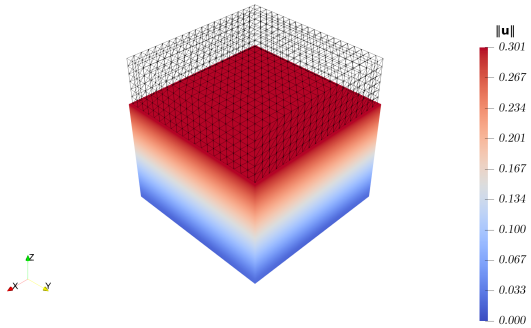
- The z-component of all nodes on the bottom face is fixed.
- The x-component of all nodes on the back side, the $y - z$ facet at $x = 0$) is fixed.
- The y-component of all nodes on the left side, the $x - z$ facet at $y = 0$) is fixed.
- The x-component of all nodes on the front side, the $y - z$ facet at $x = 1.0$) is subjected to stepped displacements which match the stretch values of BA case listed in Table 3.1.
- The y-component of all nodes on the left side, the $x - z$ facet at $y = 0$) is subjected to stepped displacements which match the stretch values of BA case listed in Table 3.1.

These boundary condition settings allow for transverse deformations to occur freely during the simulations.

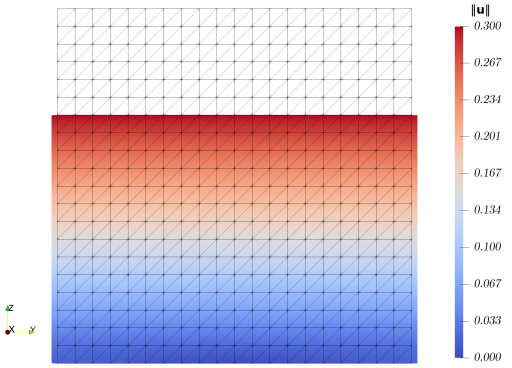
One example of the results of the UA and BA compression simulations of the unit cube at the stretch value at $\lambda^{(t)} = 0.7$ is chosen to depict in Fig. 3.4. The colour map presents the magnitude of the displacement field, denoted as $\|\boldsymbol{u}\|$. Fig. 3.4a and Fig. 3.4c show the deformed unit cube plotted in colour and the undeformed shape plotted in black wireframe for reference. Overall, the prescribed compression of the two cases is clearly observable, with a smooth transition of colour from the bottom face to the top face at the value $u_z^{(UA)} = -0.3\,\text{mm}$ for UA test and $u_x^{(BA)} = u_y^{(BA)} = -0.3\,\text{mm}$.

Fig. 3.4b and Fig. 3.4d present planar views of the $Y - Z$ plane at $x = 0.5$ of $\|\boldsymbol{u}\| \in [0, 0.3]$ for both tests. It is evident that the transverse sides of the loading axis are deformed similarly indicating the homogeneity and isotropy of the material.

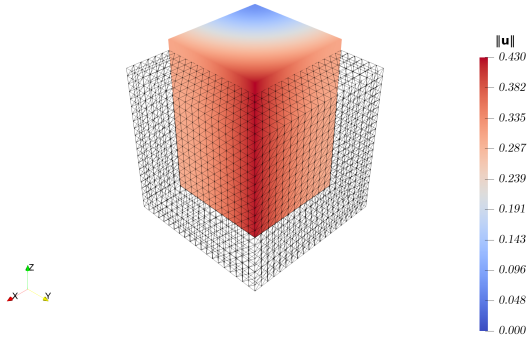
Figure 3.4 Simulation of UA and BA compression tests of compressible closed-cell polyethylene foam unit cube. (a) and (c) 3D view of $\|\boldsymbol{u}\|$ of both tests. (b) and (d) $y-z$ plane at $x=0.5$ of $\|\boldsymbol{u}\|$ of both tests.



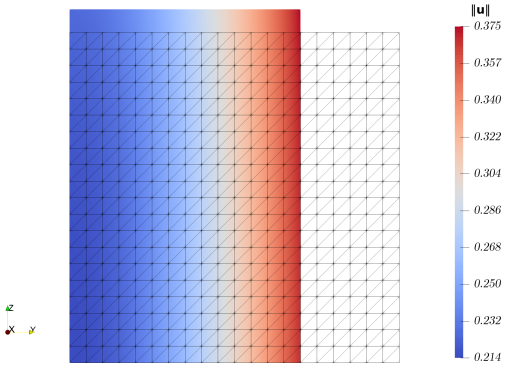
(a) $\lambda^{(UA)}=0.7$, 3D view of the unit cube under uniaxial compression: magnitude of displacement field $\|\boldsymbol{u}\|$.



(b) Plane $Y-Z$ view at $x=0.5$ of the unit cube under uniaxial extension $\lambda^{(UA)}=0.7$: magnitude of displacement field $\|\boldsymbol{u}\|$.



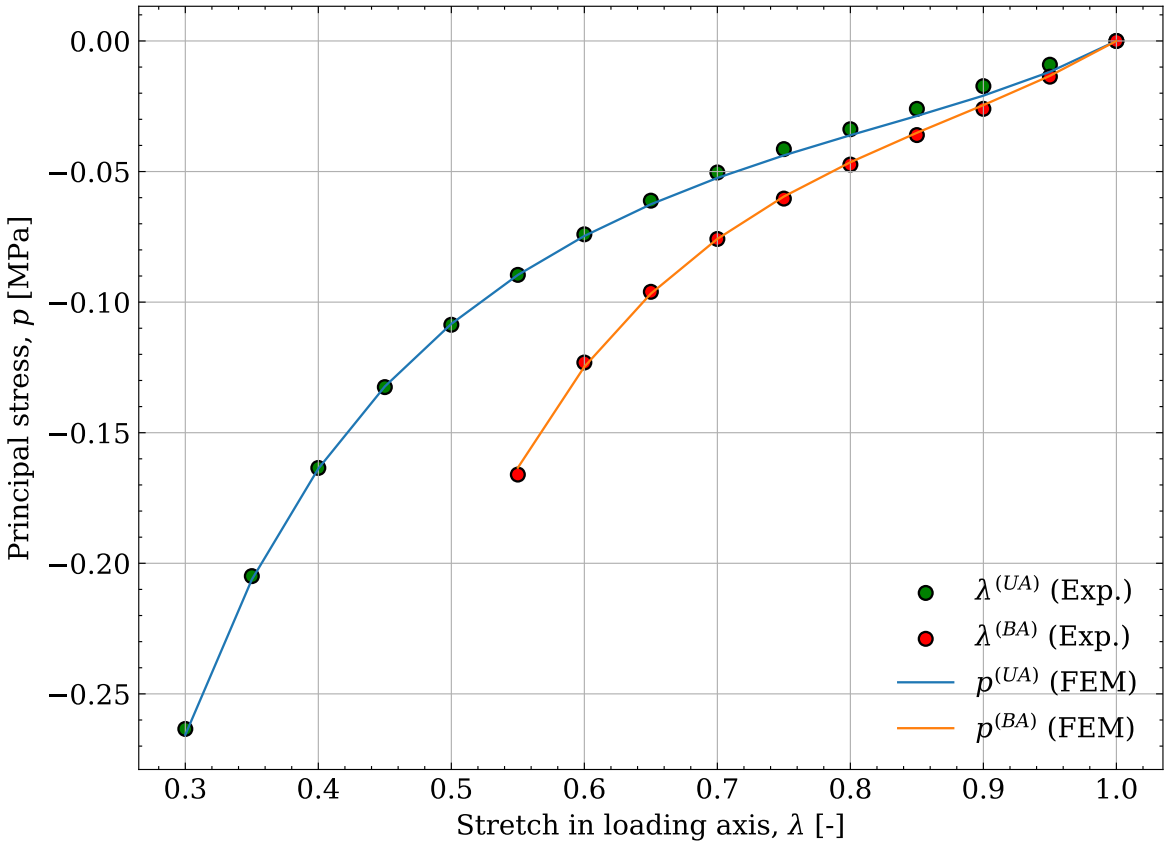
(c) $\lambda^{(BA)}=0.7$, 3D view of the unit cube under equibiaxial compression: magnitude of displacement field $\|\boldsymbol{u}\|$.



(d) Plane $Y-Z$ view at $x=0.5$ of the unit cube under equibiaxial extension $\lambda^{(BA)}=0.7$: magnitude of displacement field $\|\boldsymbol{u}\|$.

Since the stress field remains in a constant state during these compression tests, it is convenient to use the stresses of these tests for validating the agreement between FE results and the experimental data. Figure 3.5 presents a comparison between the FEM results of the Ogden-Storåkers hyperelastic model, formulated in principal space, with the identified parameters form section 3.4, and the Kossa's experimental data (Table 3.1). The figure demonstrates a good agreement between the FE predictions and the experimental data for both tests. The average relative errors, $err^{UA} = 0.298\%$ and $err^{BA} = 0.185\%$, for uniaxial compression test and equibiaxial compression respectively.

Figure 3.5 Comparison between FEM (Ogden-Storåkers) vs experimental data (UA and BA compression tests). The average relative errors are $err^{UA} = 0.298\%$ and $err^{BA} = 0.185\%$ for UA and BA tests, respectively.

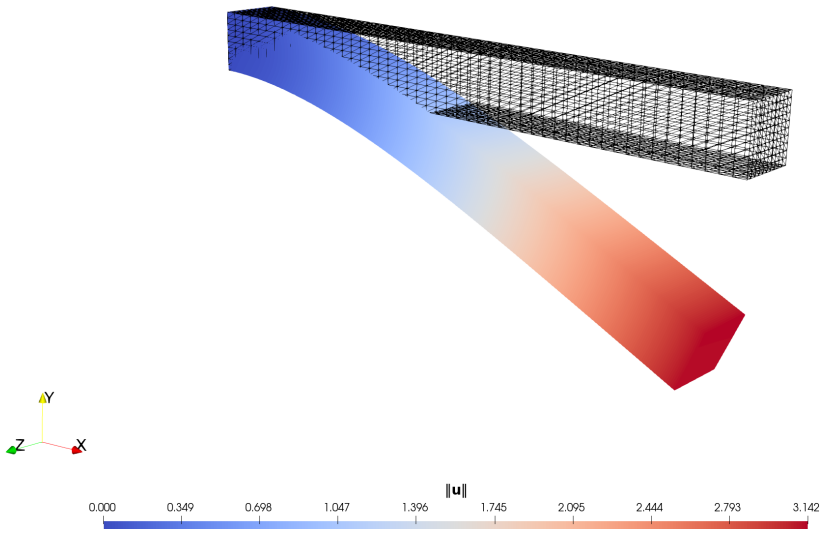


Next, a closed-cell polyethylene foam cantilever beam subjected to uniformly distributed load is employed for further numerical illustrations. The Mooney-Rivlin model with its optimal parameters (Table 3.2) is employed for this test case. The cantilever beam has the dimension of 10 mm × 1 mm × 1 mm, and the boundary conditions for the beam are as follows:

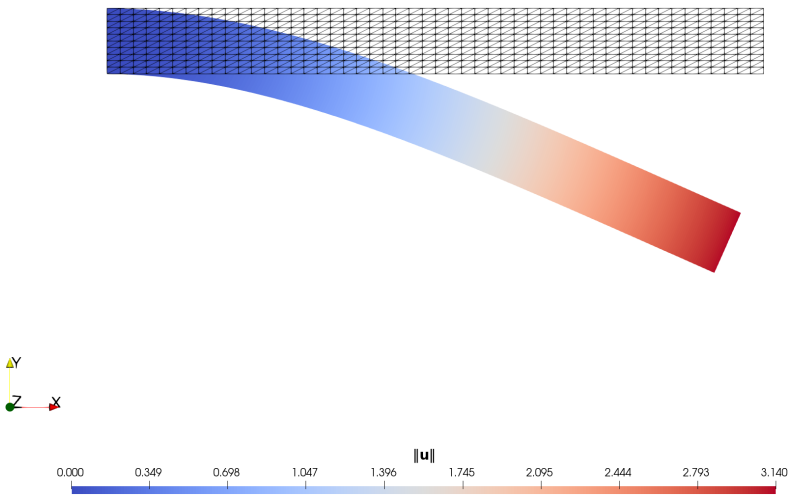
- All nodes on the left plane $y - z$ at $x = 0$ is fixed .
- All nodes on the top plane $x - z$ at $y = 1$ mm are subject to a distributed load, $q = 30$ Pa.

A grid of $20 \times 5 \times 5$ is used for modelling the beam. The mesh composed of 6171 nodes and 34 408 tetrahedron elements. Figure 3.6 presents the results of the cantilever beam test case. The magnitude of the displacement field, $\|\boldsymbol{u}\|$, is shown in colour, whereas the black wire-frame is the undeformed beam for reference. The 3D view of the test case is depicted in 3.6a, and the cross sections $X - Y$ at $z = 0.5$ of the magnitude of displacement field and the principal stress filed are illustrated in 3.6b and 3.6c, respectively. Overall, the expected behaviours are demonstrated nicely in this test case for the Mooney-Rivlin model. the largest value of $\|\boldsymbol{u}\|$ is at the tips of the beam and the stress, $\|\boldsymbol{p}\|$, concentrates at the fixed support.

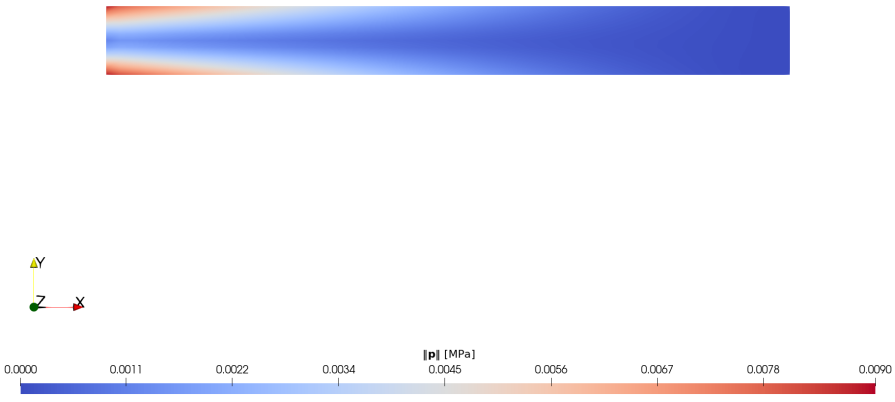
Figure 3.6 Simulation of the cantilever beam under uniformly distributed load, $q = 30\,\text{Pa}$, using the Mooney-Rivlin model. (a) 3D view of $\|\boldsymbol{u}\|$. (b) $X - Y$ plane at $z = 0.5$ of $\|\boldsymbol{u}\|$.



(a) Magnitude of displacement field: 3D view.



(b) Magnitude of displacement field: plane $X - Y$ view at $z = 0.5$.



(c) Magnitude of principal 1^{st} Piola-Kirchhoff stress: plane $X - Y$ view at $z = 0.5$.

Chapter 4

Physical motivated deep artificial neural networks

Contents

4.1 Introduction

In previous subsections, the constitutive requirements for modelling non-linear elastic deformation have been discussed. It is important to note that while these requirements seem intuitive and physically reasonable, they only serve as an approximation of real-world behaviour. In practice, there is no single mathematical model capable of perfectly describing the behaviour of any elastic material, particularly for large strains. Further restrictions can be achieved on the basis of what is observed in experimental tests on particular materials. Comprehensive experimental data can enable the development of more specialized forms of constitutive equations, and data-driven techniques can be applied to extract information from the data.

Sect. 2.4 provides *indications* on how to construct constitutive relations that satisfy underlying physical requirements and are mathematically well-posed. While adding constraints can ensure the admissibility of the physical predictions, it can also lead to restricted space, which can make it difficult to train data-driven models like deep neural networks (DNNs) to discover the underlying relations. Therefore, one must be cautious to add requirements that encourage correct physical behaviours while also maintaining flexibility for the training process of DNNs-based surrogate models.

This work takes into consideration physical requirements such as the Hill's inequalities, growth and normalisation conditions. Incorporating these requirements will encourage physical consistency while maintaining flexibility for the training process of DNNs-based surrogate models. In machine learning terminology, this approach is known as incorporating domain-specific expert knowledge. It can be a highly effective way to enhance the modelling quality of neural networks in various applications. By utilizing domain-specific knowledge, the learning process of neural networks can be directed towards physically accurate solutions, thus preventing them from predicting unphysical responses. Therefore, the requirements outlined in Section 2.4 are incorporated as prior knowledge to enhance the quality of neural networks for constitutive modelling within this study. As detailed in Subsection 2.4.8, the input/output pairs for the DNNs are identified. Furthermore, the DNNs-based response function should obey the physical constraints, namely stress-free condition (2.4.3), and growth conditions (2.4.4).

The fulfilment of Hill's inequalities translates to the requirement that the Jacobian matrix of the outputs with respect to inputs should be positive definiteness (2.103). Additionally, BE-inequalities indicates that partially monotonic of outputs with respect to inputs.

Clearly, for the case of non-linear isotropic elasticity, satisfying the Hill's inequalities ensures the satisfaction of the BE-inequalities. However, in the extension to non-isotropic elasticity, Hill's inequalities may not be valid. Instead, BE-inequalities may hold, as discussed in (Fosdick and Šilhavý, 2006; Humphrey, Strumpf, and Yin, 1990; May-Newman and Yin,

1998). Therefore, this work also introduces methodologies aimed at incorporating separately monotone behaviour of the outputs with respect to the inputs of a neural network.

Enforcing monotonicity with respect to a subset of the inputs is proposed in many real-world applications such as economic modelling (Feelders, 2000), fairness and security concerns (Akhtar et al., 2021; Cole and Williamson, 2019). It may help to improve the generalization capabilities (Milani Fard et al., 2016; You et al., 2017) and assist in the interpretability of a model (A.-P. Nguyen et al., 2023). However, to the best of the author's knowledge, there are no works that incorporate the monotonic constraints to DNNs for constitutive modelling. Plus, incorporating growth and normalisation conditions directly into the architecture of DNNs has also not been reported. The Jacobian matrix of the outputs of DNNs-based model with respect to its inputs has been discussed in Leng, Calve, and Tepole (2021) but only for 2D configurations.

This chapter presents a novel approach such that all these conditions are considered. It is a hybrid method that combines altering the network architecture as well as modifying the training process of the modified network.

In the proposed approach, a novel activation function, the Logarithmic Linear Unit (LOGLU), is introduced to handle feature extraction in the initial layers of the DNN. The LOGLU activation is designed to meet the requirements of a strain measure's function, ensuring monotonicity and respecting the inputs. Additionally, point-wise losses are utilized as heuristic regularization to enforce separate monotonicity and positive definiteness constraints of the Jacobian matrix of the neural network. Next, the modified DNNs-based model, obtained after training, is implemented in `FEniCSx` to model compressible non-linear isotropic materials.

This Chapter is structured as follows. Section 4.2 provides a brief overview of deep feedforward neural networks, activation functions, hyperparameters optimization, gradient-based learning, etc. Section 4.3 introduces novel hybrid approaches for incorporating physical constraints in constitutive modelling. Finally, Section 4.4 describes the implementation of the trained surrogate model in `FEniCSx` framework.

4.2 Deep artificial neural networks

Deep feedforward networks, also known as *feedforward neural networks* or *multilayer perceptrons* (MLPs), constitute one of the fundamental models of deep learning. The primary objective of a feedforward network is to approximate some response functions $f^*(\boldsymbol{x})$, for example, in the case of a regressor, where $\widehat{\boldsymbol{y}} = f(\boldsymbol{x})$ maps an input $\boldsymbol{x}$ to an output $\widehat{\boldsymbol{y}}$[1]. The mapping is defined as $\widehat{\boldsymbol{y}} = f(\boldsymbol{x}; \boldsymbol{w}, \boldsymbol{b})$, and the network learns the optimal values of the parameters, $\boldsymbol{w}$ and $\boldsymbol{b}$, to achieve the best possible function approximation.

[1]The abused notations $\boldsymbol{x}, \boldsymbol{y}, \widehat{\boldsymbol{y}}, \widehat{y}, f$ are distinct in this section and unrelated to those used in previous sections.

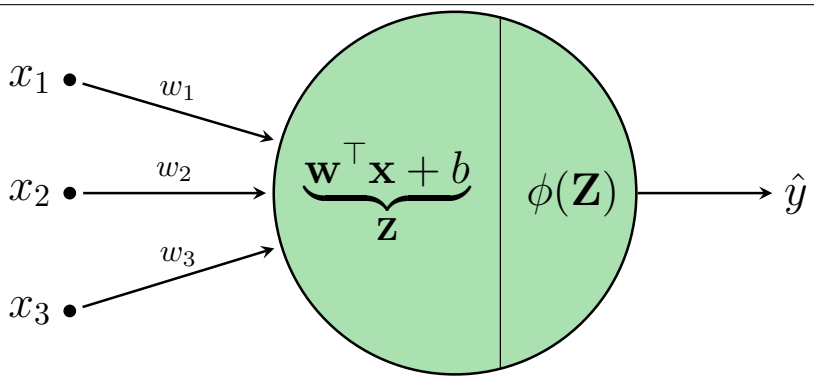
These models are called *feedforward* networks because the data flow through the function starts from the input $\boldsymbol{x}$, passing through intermediate computations, and finally to output $\widehat{\boldsymbol{y}}$. The information flows only in one direction and there are no feedback connections in which the output of the model is fed back. If these models are enhanced by adding feedback connections, they become known as recurrent neural networks.

Feedforward networks are a fundamental tool for machine learning practitioners and have a significant impact on many commercial applications. For instance, convolutional networks, a specific type of feedforward network, are commonly used for object recognition in photographs (Farabet et al., 2013; Krizhevsky, Sutskever, and G. E. Hinton, 2017; Szegedy et al., 2014). Feedforward networks represent a crucial step in the development of recurrent networks, which have proven to be highly successful in natural language processing applications (G. Hinton et al., 2012) and Large Language Models (e.g., ChatGPT, LLaMA, GPT-4) (Brown et al., 2020; Touvron et al., 2023). Moreover, they are also useful in the constitutive modelling of new advanced materials by discovering their underlying physical patterns (Fuhg, Marino, and Bouklas, 2021; Hashash, Jung, and Ghaboussi, 2004b; Klein et al., 2021; Linka et al., 2021; L. M. Wang, Linka, and Kuhl, 2023; Weber, Geiger, and Wagner, 2021).

4.2.1 Single-layer perceptron

The single-layer perceptron is a type of feedforward neural network that comprises a single layer of nodes, also known as artificial *neurons*. The perceptron was first introduced by Rosenblatt (1957) as a computational model for learning patterns in data. In its simplest form,

Figure 4.1 The Structure of a Single Neuron in an Artificial Neural Network: An input data vector $\boldsymbol{x} = [x_1, x_2, x_3]$, it is passed to the neuron through the black line with corresponding weights $\boldsymbol{w} = [w_1, w_2, w_3]$. The weighted inputs are then summed with a bias term b and passed through an activation function $\phi(\cdot)$, producing the output of the neuron, denoted by z. The activation function introduces non-linearity into the model, allowing for more complex patterns to be learned.



the single layer perceptron (SLP) (Fig. 4.1) takes a set of N input values $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$ and produces a single output value $\widehat{y}$, i.e. $\widehat{y} = f(\boldsymbol{x}; \boldsymbol{w}, b)$. Where $\boldsymbol{\theta}$ is a set of weights $\boldsymbol{w} = [w_1, w_2, \ldots, w_n]$ and a bias term b. Each input value is connected to the output node via a weighted connection w_i and bias b, then passed through the summation function. Finally, the

resulting summation is fed into a nonlinear activation function $\phi(\cdot)$ to produce the final output of the perceptron. Mathematically, SLP can be expressed as a vector-to-scalar function:

$$\widehat{y}(\boldsymbol{x};\boldsymbol{w},b) = f(\boldsymbol{x};\boldsymbol{w},b) = \phi\left(\boldsymbol{w}^\top\boldsymbol{x}+b\right) = \phi\left(\sum_{j=1}^{N} w_j x_j + b\right). \tag{4.1}$$

The SLP served as an important starting point for the development of more complex neural network architectures such as the multi-layer perceptron.

4.2.2 Deep multi-layer perceptron

When the width of the SLP is expanded to include multiple neurons in a layer, it can represent a vector-to-vector function, where each neuron acts in parallel. Multiple SLPs can be arranged in a chain to create a Multi-Layer Perceptron (MLP), which is the most commonly used structure of neural networks. For example, connect three functions $f^{(1)}, f^{(2)}, f^{(3)}$ in series to form $f(\boldsymbol{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\boldsymbol{x})))$. This is often denoted by $f(\boldsymbol{x}) = f_3 \circ f_2 \circ f_1(\boldsymbol{x})$, where $\circ$ is the function composition operator. In this case, $f^{(1)}$ is called the *input layer* of the network, $f^{(3)}$ is called the *output layer*, and any intermediate layers, e.g. $f^{(2)}$, are named the *hidden layers*. When a neural network has more than two hidden layers, it is commonly called *deep* forward neural networks (DNNs).

In the data flow, a series of functional transformations occur within the network. First, linear combinations of the inputs and the corresponding parameter matrix $\boldsymbol{W}^{(1)}$ and bias $\boldsymbol{b}^{(1)}$ of the first layer are constructed, followed by the application of the activation function $\phi^{(1)}(\cdot)$ to obtain the output of the first layer, $\boldsymbol{h}^{(1)}$. The second layer then takes $\boldsymbol{h}^{(1)}$ as input, and so on until the output $\widehat{\boldsymbol{y}}$ of the whole network is computed:

$$\boldsymbol{h}^{(1)} = f^{(1)}(\boldsymbol{x};\boldsymbol{W}^{(1)},\boldsymbol{b}^{(1)}) = \phi^{(1)}\left(\boldsymbol{W}^{(1)\top}\boldsymbol{x}+\boldsymbol{b}^{(1)}\right) = \phi^{(1)}\left(\sum_{i=1}^{N}\sum_{j=1}^{M} w_{ji}^{(1)} x_i + b_j^{(1)}\right), \tag{4.2}$$

$$\widehat{\boldsymbol{y}} = f^{(2)}(\boldsymbol{h}^{(1)};\boldsymbol{W}^{(2)},\boldsymbol{b}^{(2)}) = \phi^{(2)}\left(\boldsymbol{W}^{(2)\top}\boldsymbol{h}^{(1)}+\boldsymbol{b}^{(2)}\right) = \phi^{(2)}\left(\sum_{j=1}^{M}\sum_{k=1}^{K} w_{kj}^{(2)} h_j^{(1)} + b_k^{(2)}\right). \tag{4.3}$$

Here $j = 1,\ldots,M$ and K are the total numbers of outputs. The activation functions $\phi^{(1)}, \phi^{(2)}$ introduce non-linearity into the model, allowing for the learning of complex patterns in the data. These various states can be combined to give the overall network function that takes the vectorized form:

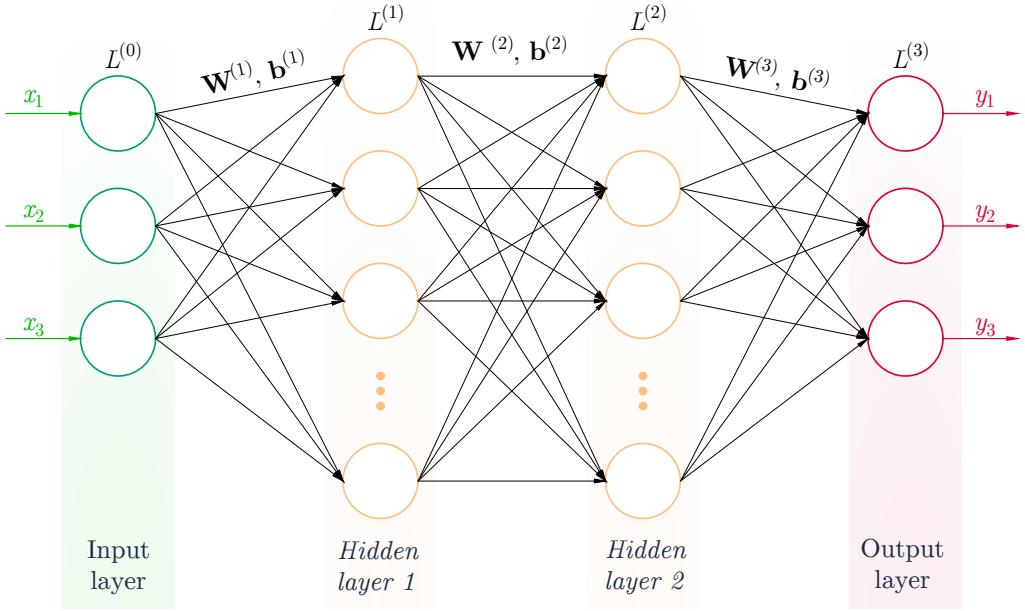
$$\widehat{\boldsymbol{y}} = \boldsymbol{h}^{(2)} \circ \boldsymbol{h}^{(1)}(\boldsymbol{x}) \tag{4.4}$$

$$= \phi^{(2)}\left(\boldsymbol{W}^{(2)\top}\phi^{(1)}\left(\boldsymbol{W}^{(1)\top}\boldsymbol{x}+\boldsymbol{b}^{(1)}\right)+\boldsymbol{b}^{(2)}\right). \tag{4.5}$$

Thus, this network model is simply a nonlinear function mapping of a set of input variables $\boldsymbol{x}$ to a set of output variables $\widehat{\boldsymbol{y}}$ controlled by adjustable parameters $\boldsymbol{W}, \boldsymbol{b}$.

This function can be visually represented as a network diagram, as shown in Figure 4.2. The process of evaluating the network (4.5) can then be interpreted as a forward propagation of information through the network, which involves two stages of processing, each resembling the perceptron model discussed in Subsection 4.2.1. The network architecture depicted in Figure 4.2 is the most commonly used architecture in practice. It is important to note that

Figure 4.2 A Deep Neural Network Architecture. The network consists of multiple hidden layers, the input layer takes in the raw data, which is then processed through a series of hidden layers before producing the final output in the output layer. The connections between the layers are weighted, and the weights and bias $\boldsymbol{W}^{(l)}, \boldsymbol{b}^{(l)}$ are learned during the training process.



the terminology for counting the number of layers in such networks can be ambiguous in the literature. The network in Figure 4.2 may be referred to as a 2-hidden layer network, which counts the number of 2 hidden layers of units, an input layer and an output layer.

For a deep feedforward neural network, it can be generalized by adding L hidden layers $\{\boldsymbol{h}^l\}_{l=1,\cdots,L}$, where l denotes the l^{th} hidden layer. The network is fully connected, so each neuron in each layer receives connections from all the neurons in the previous layer. This means each neuron has its own bias, and weight for every pair of units in two consecutive

layers. Therefore, the computations of the network are written in vectorized as:

$$\boldsymbol{h}^{(1)} = \phi^{(1)}\left(\boldsymbol{W}^{(1)\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right), \tag{4.6}$$

$$\boldsymbol{h}^{(l)} = \phi^{(l)}\left(\boldsymbol{W}^{(l)\top}\boldsymbol{h}^{(l-1)} + \boldsymbol{b}^{(l)}\right); \quad l = 2, \ldots, L, \tag{4.7}$$

$$\widehat{\boldsymbol{y}} = \phi^{(L+1)}\left(\boldsymbol{W}^{(L+1)\top}\boldsymbol{a}^{(l)} + \boldsymbol{b}^{(L+1)}\right). \tag{4.8}$$

Note that the activation, $\phi^{(l)}$, can be different for different layers. Then, the generalized equation of a deep feedforward neural network with L hidden layers is

$$\widehat{\boldsymbol{y}} = \boldsymbol{h}^{(L+1)} \circ \boldsymbol{h}^{(L)} \circ \cdots \circ \boldsymbol{h}^{(1)}(\boldsymbol{x}) \tag{4.9}$$

$$= \phi^{(L+1)}\left(\boldsymbol{W}^{(L+1)\top}\phi^{(l)}\left(\boldsymbol{W}^{(l)\top}\phi^{(L-1)}\left(\ldots\phi^{(1)}\left(\boldsymbol{W}^{(1)\top}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right)\ldots\right) + \boldsymbol{b}^{(l)}\right) + \boldsymbol{b}^{(L+1)}\right). \tag{4.10}$$

The constitutive equation from Section 2.4 for non-linear elastic materials can be formulated using a deep feedforward neural network to create a DNN-based surrogate material model. This model aims to approximate the response functions $\mathfrak{g}(\boldsymbol{F})$ by taking the flattened form of the deformation gradient tensors $\boldsymbol{F}_{vec} = [F_{11}, F_{12}, \ldots, F_{33}]$ as inputs, and predicting the flattened form of the Cauchy stress tensor $\boldsymbol{\sigma}_{vec} = [\sigma_{11}, \sigma_{12}, \ldots, \sigma_{33}]$ as outputs. Hence, the DNN-based surrogate material model becomes:

$$\boldsymbol{\sigma}_{vec} = \phi^{(2)}\left(\boldsymbol{W}^{(2)\top}\phi^{(1)}\left(\boldsymbol{W}^{(1)\top}\boldsymbol{F}_{vec} + \boldsymbol{b}^{(1)}\right) + \boldsymbol{b}^{(2)}\right). \tag{4.11}$$

Equation (4.11) is the simplest form of a DNN-based material model, which employs a 2-hidden layer DNN (eq. (4.5)). In this case, the optimal weights and biases $\boldsymbol{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$ need to be determined, making this a *multivariate regression problem* with continuous values. Clearly, the inputs and outputs choices will change when considering further physical requirements that a DNN-based surrogate material model must satisfy.

4.2.3 Activation functions

Activation functions are an essential component of neural networks. They are non-linear transformations applied element-wise to the output of a neuron to introduce non-linearity into the model, allowing for more complex patterns and relationships in data to be learned. They are a crucial component of deep neural networks, where, without them, the model would simply be a linear function. There are several types of activation functions commonly used in neural networks for numerical predictive models.

Tanh (Hyperbolic Tangent)

$$\phi(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}; \qquad \phi(x) \in [-1, 1]. \tag{4.12}$$

The Tanh function maps input values to a range between -1 and 1. Its output is zero-centered, making it useful for learning symmetric patterns. However, this function suffers from the saturation problem, where some neurons may become inactive and stop contributing to the learning process.

ReLU (Rectified Linear Unit)

$$\phi(x) = \max(0, x); \qquad \phi(x) \in [0, +\infty)\,. \tag{4.13}$$

The ReLU is currently the most widely used activation function in neural networks. The pioneering works by Fukushima (1969, 1975) on multi-layer neural networks initially introduced the concept of the "analog threshold element". However, it was not until Nair and G. E. Hinton (2010) that the interest in ReLU was popularized. ReLU is preferred due to its computational simplicity and ability to avoid the saturation problem encountered with the Tanh function. Nonetheless, ReLU also suffers from "dead neurons" when the input is negative. More detailed analysis of ReLU and Tanh in (Goodfellow, Bengio, and Courville, 2016, Chapter 6).

ELU (Exponential Linear Unit) (Clevert, Unterthiner, and Hochreiter, 2016)

$$\phi(\alpha, x) = \begin{cases} x & x > 0 \\ \alpha(e^x - 1) & x \leq 0 \end{cases}; \qquad \phi(x) \in [-\alpha, +\infty)\,, \tag{4.14}$$

where α is a trainable constant. In practice, $\alpha = 1.0$ is commonly chosen. The ELU activation function is a monotonic function created to address the problem of "dead neurons" encountered with ReLU. ELUs exhibit better-learning properties than ReLUs. Unlike ReLUs, ELUs can take negative values, allowing them to push the mean unit activations closer to zero, similar to the effect of batch normalization but without high computational complexity. Mean shifts towards zero accelerate learning by bringing the normal gradient closer to the neuron's natural gradient, resulting from a reduced bias shift effect. ELUs saturate to a negative value with smaller inputs, thereby decreasing the forward propagated variation and information.

Swish (Ramachandran, Zoph, and Le, 2017)

$$\phi(x) = \frac{x}{1 + e^{-\beta x}}\,, \tag{4.15}$$

where β is a trainable constant. In practice, $\beta = 1.0$ is commonly chosen. The Swish activation function is a smooth, non-monotonic function that has been shown to outperform ReLU and its variants in a range of deep learning applications. It can be viewed as a nonlinear interpolation between the linear function and the ReLU function.

Softplus (Dugas et al., 2000)

$$\phi(x) = \ln(1 + e^x); \qquad \phi(x) \in [0, +\infty)\,. \tag{4.16}$$

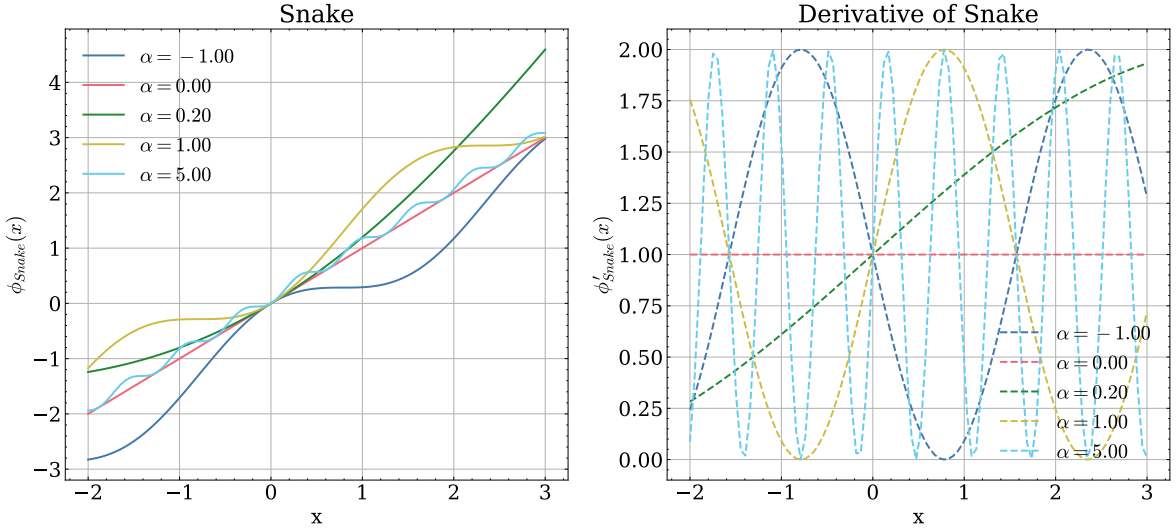
The Softplus function is a smooth, strictly positive, and monotonic function with properties similar to Swish. It can be considered a smooth approximation of ReLU that avoids non-differentiability at zero. Unlike ReLU, Softplus is differentiable everywhere, which can facilitate better training of deep neural networks. However, the Softplus function is computationally more expensive than the ReLU function.

Snake (Ziyin, Hartwig, and Ueda, 2020)

$$\texttt{Snake} = x + \frac{1}{\alpha}\sin^2(\alpha x)\,; \quad \texttt{Snake}' = 2\sin(\alpha x)\cos(\alpha x) + 1\,. \tag{4.17}$$

Here, α is a learnable parameter that controls the frequency of the periodic part. The behaviour of the `Snake` activation function at various values of the frequency controller α is illustrated in Figure 4.3. The negative and positive values of α result in a flipped version of the function. When $\alpha = 0$, the function becomes linear. For α values within the range $[0, 1]$, `Snake` appears as a smooth function. The `Snake` activation function exhibits several desirable

Figure 4.3 `Snake` and its first derivative at different $\alpha = [-1.0, 0.0, 0.2, 1.0, 5.0]$. The smoothness and periodicity of the function vary based on the value of α, indicating its adaptability for capturing complex patterns and oscillatory behaviours.



properties that make it a suitable candidate for constitutive learning models. The smoothness and monotonicity of the function enable efficient gradient-based optimization. Additionally, the learnable parameter α allows for the adjustment of the frequency of the periodic part of the function during training. This adaptability enables the model to better capture various data patterns and characteristics, ultimately resulting in improved performance. It provides an interesting alternative to traditional activation functions, such as the rectified linear unit

(ReLU) and Swish, due to its smoothness, monotonicity and periodicity (Ziyin, Hartwig, and Ueda, 2020).

Figure 4.4 Activation functions and their derivatives. The six activation functions (Tanh, ReLU, ELU, Swish, Softplus, and Shifted Softplus) are shown in separate subplots, with their corresponding first and second derivatives plotted in pink dashed and green dash-dotted lines, respectively. The x-axis represents the input values while the y-axis represents the output values and their derivatives.

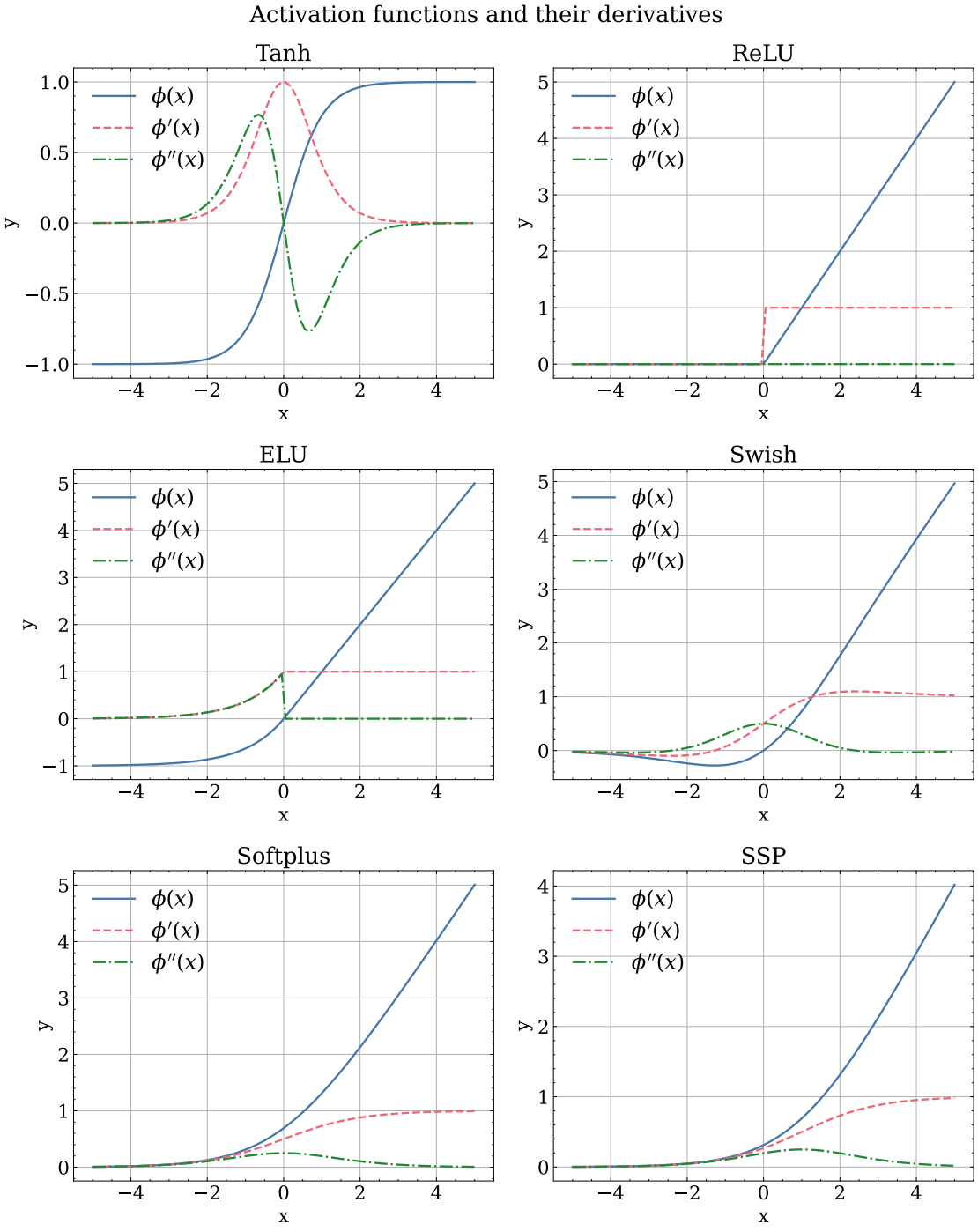


Figure 4.4 illustrates the common activation functions used in neural networks: Tanh, ReLU, ELU, Swish, Softplus, and Shifted Softplus (SSP). The figure presents the activation functions $\phi(x)$ and their first $\phi'(x)$ and second $\phi''(x)$ derivatives for each function. The x-axis represents the input value to the activation functions, while the y-axis corresponds to the output of the functions and their derivatives. This comparison provides insights into the behaviour of the different activation functions and their sensitivity to input changes. For instance, the ReLU function is non-differentiable at zero, while Softplus and Shifted Softplus offer smooth approximations to ReLU, ensuring differentiability at all input values. The ELU function can take on negative values for negative inputs, unlike ReLU, which is restricted to non-negative outputs. The Swish function offers a balance between linear and non-linear behaviour, depending on the input values, which has been shown to be advantageous in various deep learning applications.

In addition to selecting an appropriate activation function, it is also important to take into account the influence of the activation functions on the prediction properties of the neural network. For instance, when utilising specific architectures like monotonic neural networks (Sill, 1997), activation functions such as Tanh, ReLU, ELU, etc., prove to be advantageous due to their monotonic nature. This characteristic becomes beneficial in scenarios where the output needs to consistently increase or decrease over all or part of the inputs. The trade-off between computational efficiency and the impact on the generalization ability of the model is another aspect to consider when selecting an activation function. Table 4.1 summarises the properties of some common activation functions. Different activation functions may have varying degrees of computational intensity, which can affect the overall training time and efficiency of a neural network. For instance, smooth activation functions like Softplus or ELU, which address the "dead neuron" issue, may contribute to better generalization but come at the cost of increased computational cost due to more expensive mathematical operations like exponentiation and logarithms.

Table 4.1 Properties of common activation functions

Name	Monotonicity	Order of continuity	Range
Tanh	Yes	C^∞	$(-1, 1)$
ReLU	Yes	C^0	$[0, \infty)$
ELU	Yes	$\begin{cases} C^1 & \alpha = 1\,, \\ C^0 & \text{otherwise} \end{cases}$	$(-\alpha, \infty)$
Swish	No	C^∞	$[-0.278, \infty)$
Softplus	Yes	C^∞	$(0, \infty)$

4.2.4 Loss functions and training process

In the process of training a deep neural network, the primary goal is to find the optimal values for the parameters, denotes as $\boldsymbol{\theta}$ consisting of weights, $\boldsymbol{W}^{(l)}$, and biases, $\boldsymbol{b}^{(l)}$ with $l = 1, \cdots, L$,

that minimize the difference between the predicted outputs $\widehat{\boldsymbol{y}}$ and the true outputs $\boldsymbol{y}$ for a given set of inputs $\boldsymbol{x}$.

$$\mathcal{L} = f(y, \hat{y}) \tag{4.18}$$

To quantify this difference, loss function $\mathcal{L}$ (or objective function) is used to evaluate the performance of the model. The loss function plays a vital role in constructing the architecture of the machine learning model and enhancing its performance. In this work, the multivariate regression problem with continuous values is considered. Hence, six common loss functions for regression problems, which are introduced in Table 4.2 are the appropriate choices.

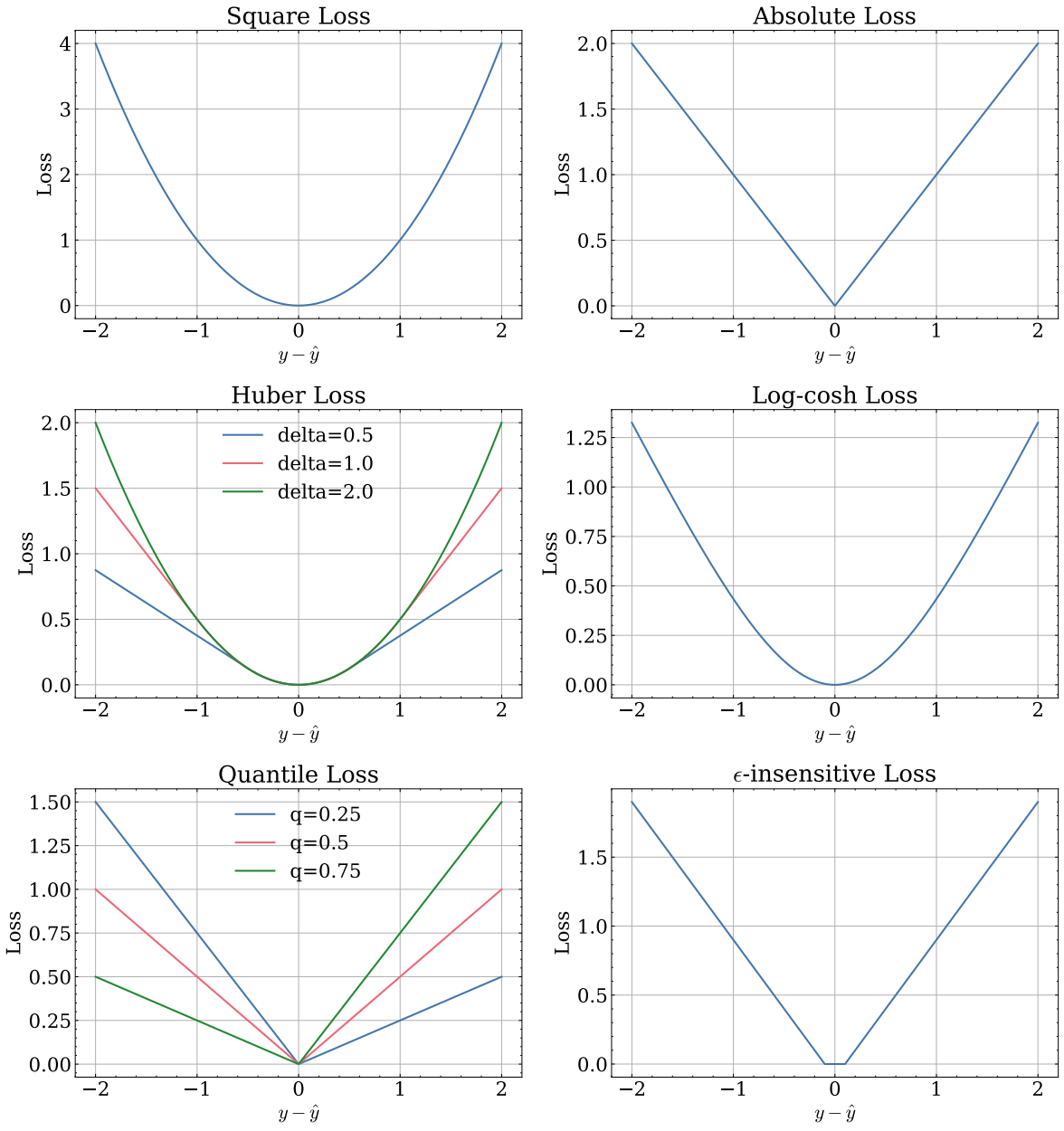
Figure 4.5 presents a visual comparison of these loss functions. The x-axis represents the error, $\mathcal{L} = (y - \hat{y})$, while the y-axis denotes the corresponding loss value. As seen in the figure, the Mean Squared Error (MSE) exhibits a quadratic relationship with the error, causing high sensitivity to outliers. In contrast, the Mean Absolute Error (MAE) is more robust to outliers but lacks smoothness at $\mathcal{L} = 0$, which may impede optimization.

Huber loss combines the properties of MSE and MAE, with the parameter δ controlling the transition between the two behaviours, making it more robust against outliers than mean square loss. For small errors, it behaves like an MSE, while for large errors, it behaves like an MAE.

Log-cosh loss offers a smooth and twice-differentiable alternative to Huber loss, as its first derivative is $\tanh(\mathcal{L})$. Quantile loss provides a more flexible approach with the parameter q. It estimates the conditional median or quantiles of the dependent outputs $\hat{y}$ across the values of the independent inputs x. This function can be viewed as an extension of the MAE, except for the 50^{th} percentile, where it is equivalent to the MAE.

Lastly, the ϵ-insensitive loss is designed to be robust to small errors, making it well-suited for support vector regression. Table 4.2 summarised the loss functions commonly used for regression tasks and indicated their advantages and disadvantages.

Figure 4.5 Comparison of six loss functions used in regression tasks. Each subplot shows the behaviour of the loss function $\mathcal{L}$ as the error of the true and predicted values $(y - \hat{y})$ varies. The Huber Loss and Quantile Loss plots include different parameter settings, illustrating the effect of varying δ and q, respectively. This visualization aids the understanding of the properties of each loss function and selecting the most suitable one for a given regression problem.



4.2.5 Training process

So far, the network is viewed as a general class of parameterized nonlinear functions mapping a vector input $\boldsymbol{x}$ to a vector output $\boldsymbol{y}$. The parameters of the network are needed to determine through the minimization of the loss function $\mathcal{L}(\boldsymbol{\theta})$.

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \,. \tag{4.24}$$

The process of finding the values of the network's parameters is called *training* or *fitting* (quite similar to the discussion in Section 3.4).

Consider a loss function $\mathcal{L}(\boldsymbol{\theta})$, which is a continuous function of the parameters $\boldsymbol{\theta}$. The gradient of $\mathcal{L}(\boldsymbol{\theta})$ with respect to the parameters is given by:

$$\boldsymbol{\nabla}_{\boldsymbol{\theta}}\mathcal{L} = \left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}^{(1)}}, \cdots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}^{(l)}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^{(1)}}, \cdots, \frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^{(l)}}\right]^\top \tag{4.25}$$

The gradient is computed backward, starting from the last layer's parameters toward the first ones and applying the chain rule. This process is called *backpropagation* or backward pass, and the backpropagation algorithm (Rumelhart, G. E. Hinton, and R. J. Williams, 1986) is usually used to numerically evaluate the gradient of the network. It consists of two passes through the network: a forward pass (as described in Subsection 4.2.2) and a backward pass. The most widely used optimization algorithm in deep learning is *gradient descent* (GD) and its variants.

Gradient Descent GD is an iterative optimization algorithm that updates the parameters of the model by moving in the direction of the negative gradient of the loss function with respect to the parameters. The update rule for the parameters is given by:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \boldsymbol{\nabla}_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}^{(t)}) \,, \tag{4.26}$$

where $\boldsymbol{\theta}^{(t)}$ represents the parameters at iteration t, η is the learning rate, and $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^{(t)})$ is the gradient of the loss function with respect to the parameters at iteration t.

Stochastic Gradient Descent (SGD) Standard GD can be computationally expensive, as it requires computing the gradient over the entire dataset. Stochastic Gradient Descent (Kiefer and Wolfowitz, 1952; Robbins and Monro, 1951) is an alternative that estimates the gradient using a random sample (or a mini-batch) of the dataset at each iteration. This reduces the computational cost and introduces some noise into the gradient, which can help escape local minima. Variants of SGD, such as Adagrad (Duchi, Hazan, and Singer, 2011), RMSProp, and Adam (Kingma and Ba, 2014), have been developed to improve convergence and adapt the learning rate during training. See (Bottou, Curtis, and Nocedal, 2018) for further discussions.

Table 4.2 Loss functions for regression tasks with their advantages and disadvantages

Loss Function		Advantages	Disadvantages
Mean Square Loss			
$\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=1}^{N}(\widehat{y}_i - y_i)^2$	(4.19)	Smooth, differentiable, easy to optimize Bishop, 2006	Sensitive to outliers, may overfit to extreme values Bishop, 2006
Mean Absolute Loss			
$\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=1}^{N}\lvert\widehat{y}_i - y_i\rvert$	(4.20)	Robust to outliers, more stable estimates Bishop, 2006	Not smooth at 0, harder to optimize.
Huber Loss			
$\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \begin{cases} \frac{1}{2}(\widehat{\boldsymbol{y}} - \boldsymbol{y})^2, & \text{if} \quad \lvert\widehat{\boldsymbol{y}} - \boldsymbol{y}\rvert < \delta \\ \delta\lvert\widehat{\boldsymbol{y}} - \boldsymbol{y}\rvert - \frac{1}{2}\delta^2\,, & \text{otherwise} \end{cases}$	(4.21)	Combines advantages of square and absolute loss, robust to outliers and smooth Huber, 1964	Requires tuning of δ, less interpretable Huber, 1964
Log-cosh Loss			
$\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \log(\cosh(\widehat{\boldsymbol{y}} - \boldsymbol{y}))\,.$	(4.22)	Have the advantages of Huber loss, smooth, quadratic differentiable (Jadon, 2020)	Less robust to outliers, harder to optimize than square loss (Q. Wang et al., 2022)
ϵ-insensitive Loss			
$\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \begin{cases} 0, & \text{if} \quad \lvert\widehat{\boldsymbol{y}} - \boldsymbol{y}\rvert < \epsilon \\ \lvert\widehat{\boldsymbol{y}} - \boldsymbol{y}\rvert - \epsilon\,, & \text{otherwise.} \end{cases}$	(4.23)	Robust to small errors, used in support vector regression Vapnik, Golowich, and Smola, 1996	Requires tuning of ϵ, less interpretable, less smooth Bishop, 2006

4.2.6 Hyperparameter optimization

Hyperparameter optimization (HPO) also referred to as *hyperparameter tuning*, is a crucial and intricate step in designing deep neural networks. This process aims to achieve optimal performance for a machine learning model by tuning hyperparameters (HPs), which govern various aspects of the network's architecture and training process. These HPs encompass parameters such as the number of layers, the number of hidden units per layer, the activation functions, the learning rate, etc. Effective calibration of these HPs can lead to improved performance and better generalization for a given task (Bischl et al., 2023; L. Yang and Shami, 2020).

The evaluation of model performance is quantified through an objective function specific to the chosen model, such as mean least squared error, accuracy, F1-score etc. For each tested set of HPs, the model is re-trained, and its performance score is computed based on the model's validation set. Commonly, there are two methods for selecting these hyperparameters: manual selection and automatic selection. To limit the laborious and non-reproducible nature of the manual trial-and-error approach in searching for effective hyperparameter settings, automatic HPO algorithms are usually preferred.

Sampling strategy entails a range of sampling algorithms available for automatic HPO, spanning from straightforward techniques such as grid search and random search (Bergstra and Bengio, 2012) to more sophisticated methods like Tree-structured Parzen Estimator (TPE) (Bergstra et al., 2011), Spearmint (Snoek, Larochelle, and Adams, 2012), Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen and Ostermeier, 2001), among others.

Nevertheless, one disadvantage of automatic HPO algorithms lies in their dependence on the completion of an entire training experiment, or *trial*, before valuable insights can be extracted. This is much less efficient in extracting valuable information early on compared to the manual HPO undertaken by a human expert. Human practitioners often identify problematic HPs at an early stage and can terminate the unpromising trials, this process is called *pruning*.

Pruning strategy is the approach utilised for terminating the unpromising training experiments, also known as automated stopping (Golovin et al., 2017). Pruning algorithms typically operate across two stages: (1) continuous monitoring of intermediate objective values, and (2) discontinuation of trials that fail to satisfy the predefined criteria. Some notable pruning algorithms include Bayesian parametric regression (Domhan, Springenberg, and Hutter, 2015; Swersky, Snoek, and Adams, 2014), Median Stopping Rule (Golovin et al., 2017), Hyperband (L. Li et al., 2018), etc.

The sampling strategy and pruning strategy can be combined flexibly to explore various settings, each combination is called *study*. In this work, `Optuna` (Akiba et al., 2019) framework is leveraged for implementing HPO of DNN-based constitutive model. The main procedure of HPO in this work is as follows:

1. Select the objective function e.g. loss function on the validation set.
2. Select HPs and their search space appropriately for each model by manual testing and/or domain knowledge.
3. Create studies by applying appropriate combinations of sampling strategy and pruner.
4. Run each study (optimization process) with a large number of trials to explore HPs.
5. Refine the search space based on the currently well-performing HP values if necessary.
6. Return the final optimal results as the best-performed HP configurations.

4.2.7 Generalization experiments of activation functions

During research and development of new techniques for a specific problem domain, it is common to test various activation functions. The choice of an activation function for a neural network should take into account not only the mathematical properties and performance of the function but also the trade-offs between computational efficiency and the impact on the generalization ability of the model.

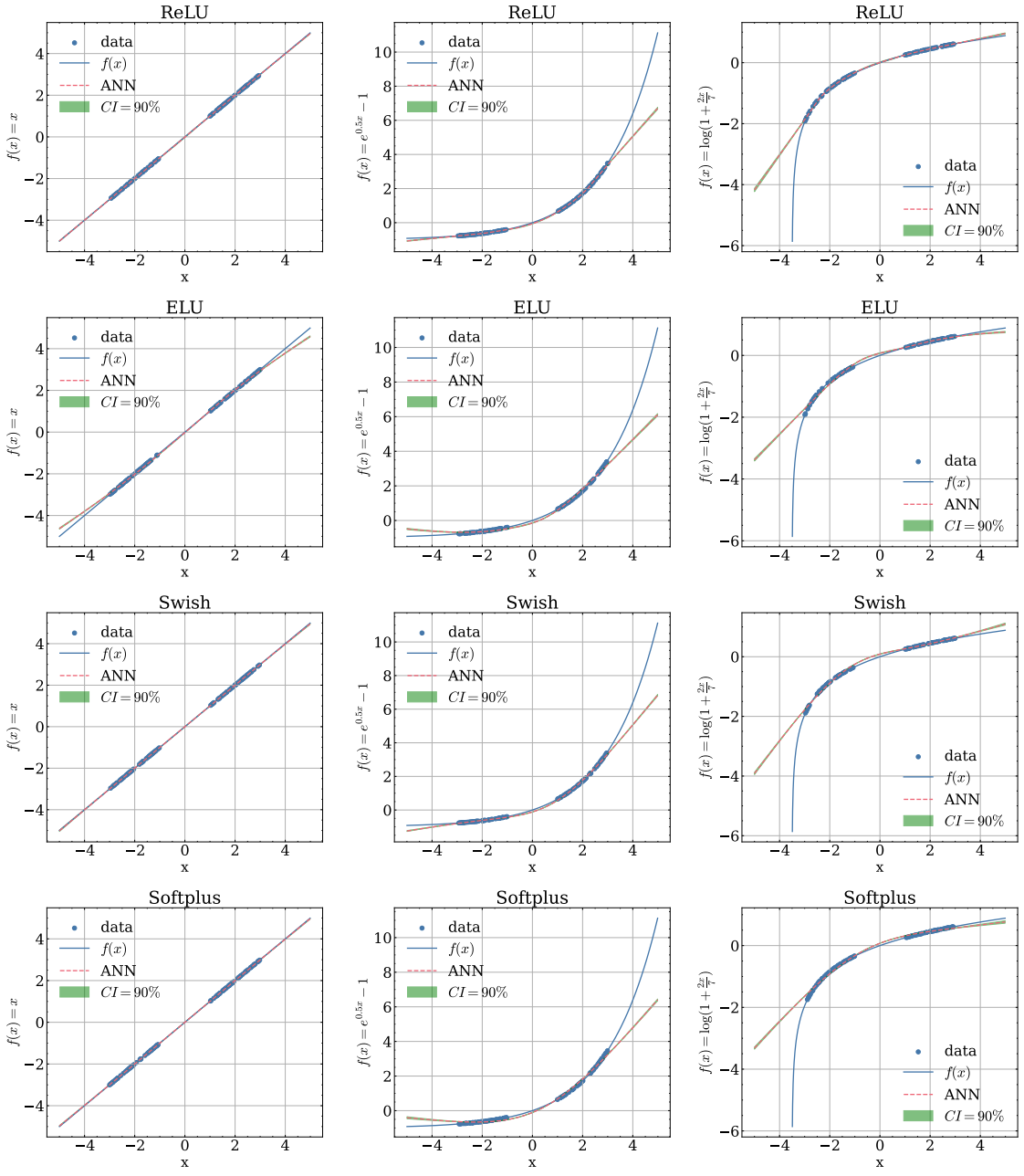
Investigating the influence of using different activation functions in a general setting can be challenging. Nevertheless, it is possible to hypothesize that the characteristics of activation functions are carried over to the property of the neural networks. For instance, an `ELU` network is smooth and can extrapolate to a constant function, whereas the `ReLU` is piecewise-linear and can extrapolate in a linear manner. In fact, the experiments can be theoretically proven in a more general form, as demonstrated by Ziyin, Hartwig, and Ueda (2020). These observations provide insight into the behaviour of neural networks, which can be valuable for designing more efficient and effective machine learning models.

A small experiment is conducted using a fully connected ANN with one hidden layer containing 512 neurons. Training data are collected by sampling from 6 different analytical basic functions in the interval $[-5, 5]$ with a gap in the range $[-1, 1]$. This setup enables us to study the inter-and-extrapolation behaviour of 4 different activation functions. The experiments can be seen in Fig. 4.6 and Fig. 4.7. Data points are represented by blue dots, while the analytical ground truth functions are shown as black lines. The mean values of 21 runs of the predictions of the ANNs are plotted with red dashed lines, and the 90% confidence interval (CI) is displayed as a green-shaded area.

The results suggest that the extrapolation behaviour of the ANNs is dictated by their analytical form of the activation function. Fig. 4.6 illustrates that ReLU and Swish diverge to $\pm\infty$, while ELU and Softplus level off towards a constant value. Notably, none of the activation functions can capture the asymptotic behaviour of the logarithmic function ($\log(1 + \frac{2x}{7})$).

Fig. 4.7 demonstrates the ability of the activation functions to capture the convexity or monotonicity of the target functions. Monotonic activation functions, ReLU, ELU, perform

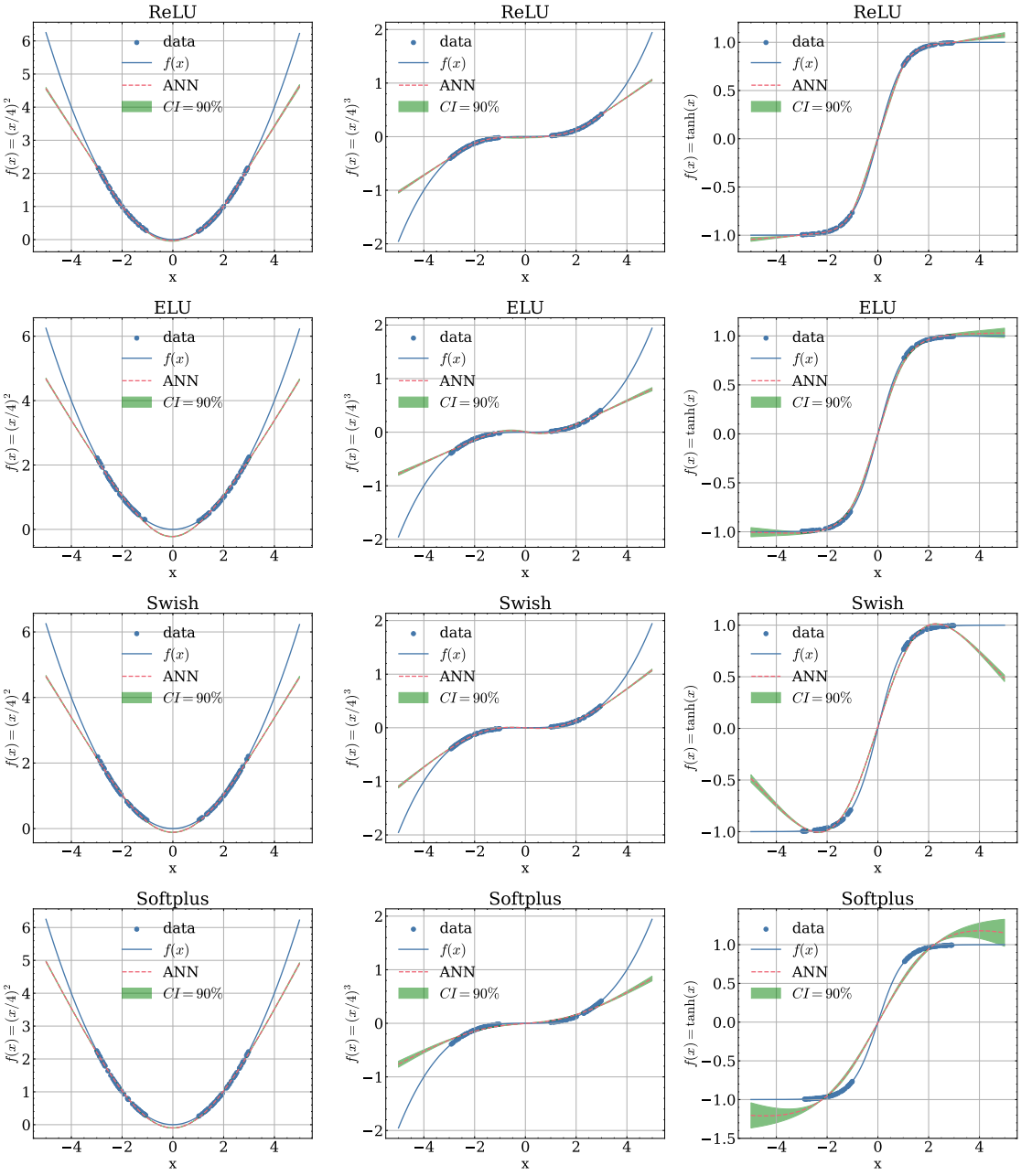
Figure 4.6 Exploration into the generalization of different activation functions for various basic function types.



generally quite well. While Swish, Softplus can also represent these properties but with greater variability, as their confidence interval area is larger. Distinctly, all activation functions exhibit much better interpolation capabilities than extrapolation, especially when sampling data points cover the critical range, such as changes in function curvature.

Despite ANNs being considered as universal approximators, given enough hidden neurons, these experimental observations reveal that they (with 512 neurons) are unsuitable for extrapolation, which is consistent with the discussion in Sect. 4.2.8.

Figure 4.7 Exploration into the generalization of different activation functions for various basic function types (continue).



4.2.8 The power of depth

The main factors to consider when designing the architecture of DNNs are depth (number of layers) and width (number of units per layer). Deeper networks often require fewer neurons per layer and fewer parameters, leading to better generalization to the test set, but they may be harder to train. The ideal network architecture usually has to be determined through experimentation, with the loss values on the validation are used as a guide.

The Universal Approximation Theorem (Cybenko, 1989; Hornik, Stinchcombe, and White, 1989; Leshno et al., 1993) establishes that feedforward neural networks with Sigmoid-family or locally bounded piecewise continuous activation function can approximate any continuous function to any degree of accuracy, as long as the network has a sufficient number of hidden units. This theorem demonstrates that a large MLP is capable of representing any function. However, it is crucial to note that the theorem does not guarantee that the training algorithm can learn the desired function. Learning can fail for two main reasons (Goodfellow, Bengio, and Courville, 2016, p. 199):

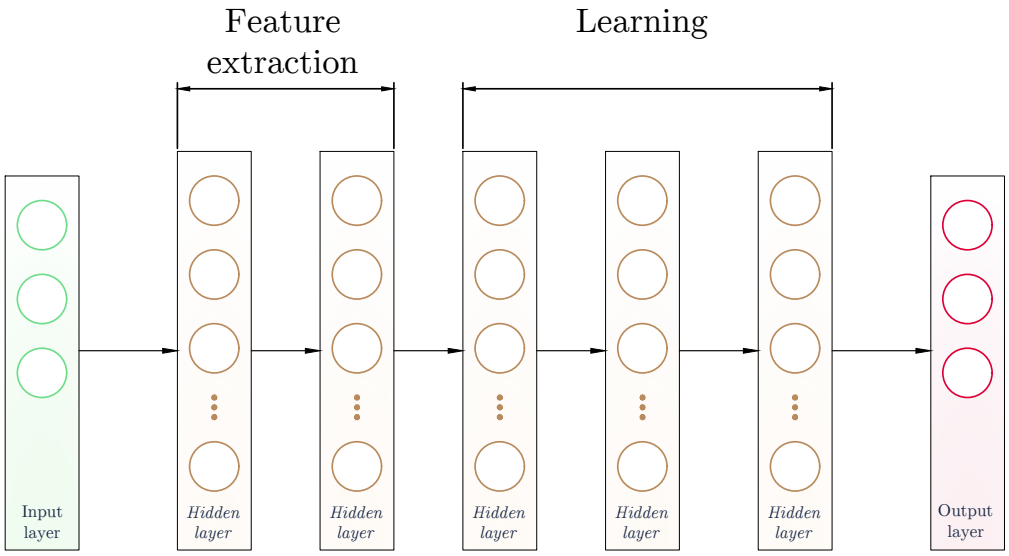
- The optimization algorithm might be unable to find the parameter values corresponding to the desired function such as poor initialization, multiple local minima, etc. (Glorot and Bengio, 2010; Goodfellow, Bengio, and Courville, 2016, Chapter 8);
- The training algorithm could end up at an incorrect function due to overfitting. Despite the universal approximation capabilities of feedforward networks, the "no free lunch theorem" (Wolpert and Macready, 1997) shows that no universally superior machine learning algorithm exists, meaning that there is no universal procedure for choosing a function that generalizes well to points not in the training set.

In practice, employing deep architectures can reduce the number of units required to represent the desired function and decrease generalization error. The use of deep models encodes the belief that the target function involves the composition of several simpler functions, which can be interpreted as discovering a set of underlying factors of variation. Figure 4.8 provides an abstract presentation of how deep neural networks operate (Goodfellow, Bengio, and Courville, 2016; Johnston and Fusi, 2023). The inputs are fed into the network, with the beginning layers learning a higher-level representation of the inputs. As the inputs are processed deeper into the network, each module extracts features and transforms them into higher levels of abstraction. During the learning phase, the network builds up a better understanding of the task, leading to improved generalization after each layer. Ultimately, the network produces helpful predictions for the task at hand. This process is central to the success of deep learning, as evidenced by the extensive literature across a variety of tasks (Brown et al., 2020; Devlin et al., 2019; He et al., 2015; Krizhevsky, Sutskever, and G. E. Hinton, 2017; Szegedy et al., 2014; Tan and Le, 2020; Touvron et al., 2023; Vaswani et al., 2017).

Furthermore, incorporating domain-specific expert knowledge is an effective way to enhance the modelling quality of neural networks in various applications.

In this work, leveraging the understanding of the abstract representation learned by the DNNs, the feature extraction phase is sped up significantly and guided in the physically plausible direction by incorporating physical constraints such as growth and normalization conditions. A new activation function is introduced and applied on the first layer to ensure that the feature presentation step obeys the constraints.

Figure 4.8 Abstract presentation: At a high level, deep networks learn a hierarchy of representations of the inputs, with each layer extracting features and transforming them into a higher level of abstraction. The learning process enables better generalization with each layer, ultimately producing useful predictions for the task.



4.3 Physical constrained deep neural networks based surrogate model

Consider the general supervised learning problem of surrogate material laws using deep neural networks. This DNN-based model is expected to satisfy the physical requirements outlined in Section 2.4. The training set consists of m data pairs, denoted as $\{(\boldsymbol{c}^{(i)}, \boldsymbol{s}^{(i)})\}_{i=1}^{m}$, where $\boldsymbol{c}^{(i)} \in \mathbb{R}^3$ and $\boldsymbol{s}^{(i)} \in \mathbb{R}^3$ are the vectors of squared principal stretches and principal stresses, respectively (c.f. subsection 2.4.8). Using these input-output pairs ensures that the constitutive requirements (2.4.1) and (2.4.2) are fulfilled.

This section introduces a novel approach for incorporating the remaining physical constraints into deep neural networks. The number of accounted physical constraints is considerably large, namely stress-free condition (2.4.3), growth conditions (2.4.4), the Jacobian matrix of the outputs with respect to the inputs being positive definite (2.103), and partially monotonicity of outputs with respect to the inputs (BE-inequalities).

Again, (2.103) implies (BE-inequalities), it is not necessary to incorporate both requirements at once. This work employs partial monotonicity constraint only for generation purposes for which the proposed methodologies can be applied to non-isotropic materials. Existing approaches for effectively enforcing monotonicity into deep neural networks can generally be classified into two groups:

- *Hand-designed monotonic network architecture*: Involves designing a specialized network's architecture such that it guarantees monotonicity through its construction (Daniels and Velikova, 2010; Kitouni, Nolte, and M. Williams, 2023; Milani Fard et al., 2016; Sill, 1997; Wehenkel and Louppe, 2021; You et al., 2017). However, these specialized architectures often impose considerable restrictions or complexity, making practical implementation and training challenging.
- *Heuristic Monotonic Regularization*: An alternative approach aim to enforce monotonicity for pre-existing arbitrary DNN by training it with a heuristically designed regularization (Gupta et al., 2019; X. Liu et al., 2022; Sill and Abu-Mostafa, 1996; Sivaraman et al., 2020). Although this method provides greater flexibility and easier implementation, there is no guarantee that the learned models will ensure the monotonic properties over selected features. Consequently, the monotonicity constraint may be violated on some data points.

Indeed, each line of those methods comes with its own advantages and disadvantages. When opting for the hand-designed architecture approach, the DNN's structure is constrained not only by the requirement to uphold the monotonic property but also by the need to meet growth and normalization conditions. This results in an intricate architecture that can introduce challenges during both the training phase and the subsequent implementation of its Unified Form Language (UFL) version (see Section 4.4). In case of only heuristic monotonic regularization is considered, it is very unlikely the model will satisfy all physical constraints.

To address these inherent challenges, the approach leverages the benefits associated with both methods. The adaptation encompasses both the architecture of the DNN and its training process:

- For *normalization condition, growth condition*: The architecture of DNN is designed by adding a novel activation function called "Logarithmic Linear Unit" (LOGLU) (c.f. 4.3.1).
- For *separately monotone and positive definiteness* of the Jacobian matrix of the outputs with respect to inputs: these point-wise losses (Gupta et al., 2019) act as additional constraints during the training process, ensuring that the learned models satisfy monotonicity and positive definiteness conditions.

4.3.1 Proposed Logarithmic Linear Unit (LOGLU)

Section 4.2.8 describes the characteristics of the depth of DNN layers, which consists of feature extraction and learning phases. The first several layers serve as feature extraction layers, responsible for learning the high-level features from input data. In the context of continuum mechanics, Section 2.2.4 recommends choosing a strain measure that effectively captures the nonlinear effect of deformation, thereby reducing the need to explicitly represent the nonlinear characteristics in constitutive equations.

Interestingly, this intersection of ideas between machine learning and continuum mechanics suggests that the generalised strain tensor, specifically $\boldsymbol{E}^{(0)} = \frac{1}{2}\ln \boldsymbol{C}$, may be a promising candidate for presenting the feature abstraction of the DNNs. Motivating by this alignment, a novel activation is proposed, to be integrated after the input layer, acting as the features extraction layers. The proposed novel activation is as follows:

$$\phi_{\text{LOGLU}}(x) = \begin{cases} \alpha^2 \ln(x+\epsilon) & \text{if} \quad 0 < x < 1\,, \\ \alpha^2(x-1.0) & \text{if} \quad x \geq 1. \end{cases} \; ; \quad \phi'_{\text{LOGLU}}(x) = \begin{cases} \alpha^2 \frac{1}{x+\epsilon} & \text{if} \quad 0 < x < 1\,, \\ \alpha^2 & \text{if} \quad x \geq 1. \end{cases} \tag{4.27}$$

where x is the input to a neuron, α : learnable parameter and ϵ is to avoid divided by zero in its derivative. LOGLU is designed to be monotonic, and it satisfies both growth and normalization conditions making it physically meaningful (mechanically speaking).

The piece-wise structure of LOGLU allows it to separate two cases: extension and compression when the input presents the principal stretch or principal strain. The squaring of α as α^2 ensures that LOGLU $\in (-\infty, 0)$ when $0 < x < 1$ (compression regime), and LOGLU $\in [0, +\infty)$ for $x \geq 1.0$ (extension regime). The strength of asymptotic behaviour is controlled by the learnable parameter α.

Figure 4.9 displays the LOGLU activation and its first derivative for different values of α. Then plots show that LOGLU is monotonically increasing with respect to the inputs, and its asymptotic behaviour is influenced by the parameter α.

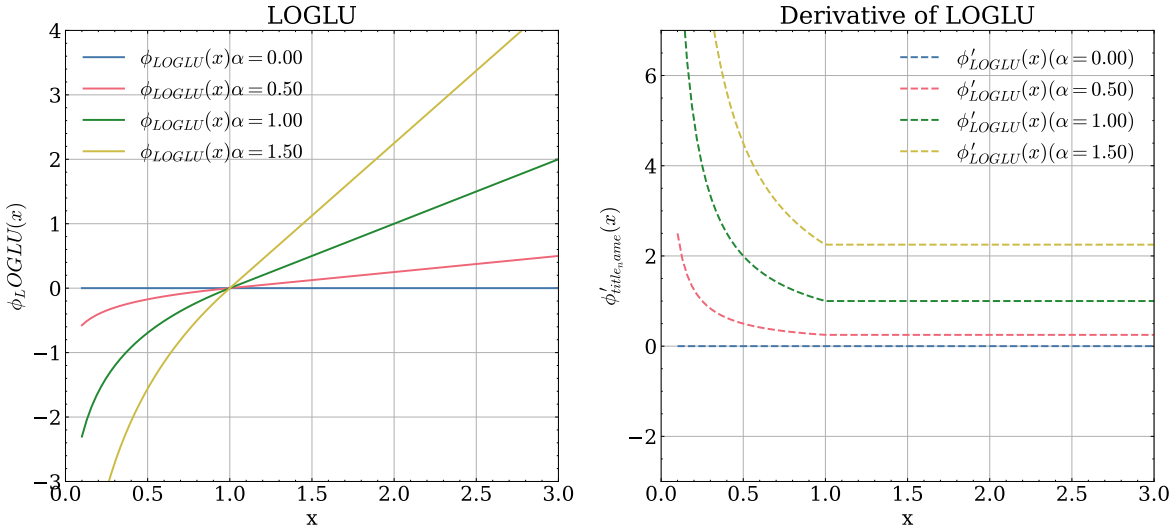
Requirements 2.4.3 and 2.4.4 are satisfied by applying LOGLU. Let LOGNet be the generalized DNNs-based surrogate model with L-hidden layers, denotes as $\hat{\boldsymbol{s}} = f(\boldsymbol{c};\boldsymbol{\theta})$, $\boldsymbol{\theta} = \{\boldsymbol{W}^{(l)}\}_{l=1}^{L}$, of the isotropic non-linear elastic material are defined as:

$$\begin{aligned} \widehat{\boldsymbol{s}}(\boldsymbol{c};\boldsymbol{\theta}) &= \boldsymbol{h}^{(L+1)} \circ \boldsymbol{h}^{(L)} \circ \cdots \circ \boldsymbol{h}^{(1)} \circ \text{LOGLU}(\boldsymbol{c}) \qquad (4.28) \\ &= \phi^{(L+1)}\left(\boldsymbol{W}^{(L+1)\top}\phi^{(l)}\left(\boldsymbol{W}^{(l)\top}\phi^{(L-1)}\left(\ldots\phi^{(1)}\left(\boldsymbol{W}^{(1)\top}\text{LOGLU}(\boldsymbol{c})\right)\right)\right)\right). \quad \text{(LOGNet)} \end{aligned}$$

Here, the employed activation functions satisfy the following:

$$\phi^{(l)}(\boldsymbol{0}) = \boldsymbol{0}\,, \tag{4.29}$$

Figure 4.9 Logarithmic Linear Unit (LOGLU) and its first derivative at different α.



and all the biases $\{\boldsymbol{b}^{(l)}\}_{l=1}^{L}$ are needed to be turned off, such that:

$$\widehat{\boldsymbol{s}}(\boldsymbol{c} = \boldsymbol{1}, |\boldsymbol{\theta}) = \boldsymbol{0}\,. \tag{4.30}$$

For instance, a simple one-hidden layer LOGNet is as follows:

$$\widehat{\boldsymbol{s}} = \boldsymbol{h}^{(2)} \circ \boldsymbol{h}^{(1)} \circ \boldsymbol{h}^{(0)} \tag{4.31}$$

$$= \boldsymbol{W}^{(2)\top}\Big[\phi^{(1)}\Big(\boldsymbol{W}^{(1)\top}\mathrm{LOGLU}(\boldsymbol{c})\Big)\Big]^{\top}. \tag{4.32}$$

Since the problem at hand is the continuous variable regression, $\phi^{(2)}$ is just a linear mapping. With this setting, the derivative of the outputs with respect to the inputs reads:

$$\frac{\partial \widehat{\boldsymbol{s}}}{\partial \boldsymbol{c}} = \frac{\partial \widehat{\boldsymbol{s}}}{\partial \boldsymbol{h}^{(1)}} \frac{\partial \boldsymbol{h}^{(1)}}{\partial\Big(\boldsymbol{W}^{(1)\top}\mathrm{LOGLU}(\boldsymbol{c})\Big)} \frac{\partial\Big(\boldsymbol{W}^{(1)\top}\mathrm{LOGLU}(\boldsymbol{c})\Big)}{\partial\Big(\mathrm{LOGLU}(\boldsymbol{c})\Big)} \frac{\partial\Big(\mathrm{LOGLU}(\boldsymbol{c})\Big)}{\partial \boldsymbol{c}} \tag{4.33}$$

$$= \boldsymbol{W}^{(2)\top}\Big(\phi^{(1)}\Big)'\boldsymbol{W}^{(1)\top}\Big(\mathrm{LOGLU}(1)\Big)'. \tag{4.34}$$

This derivative can be interpreted as the "stiffness" of the materials at a stress-free state, therefore, $\frac{\partial \widehat{\boldsymbol{s}}}{\partial \boldsymbol{c}} \neq \boldsymbol{0}$ for physical plausibility. Equation (4.34) indicates that activation functions for hidden layers of LOGNet must be a function such that $\phi' \neq 0$.

The Jacobian matrix of the predicted principal stresses with respect to the squared principal stretches reads:

$$\widehat{\boldsymbol{H}}(\boldsymbol{c};\boldsymbol{\theta}) = \begin{bmatrix} \frac{\partial \hat{s}_1}{\partial c_1} & \frac{\partial \hat{s}_1}{\partial c_2} & \frac{\partial \hat{s}_1}{\partial c_3} \\ \frac{\partial \hat{s}_2}{\partial c_1} & \frac{\partial \hat{s}_2}{\partial c_2} & \frac{\partial \hat{s}_2}{\partial c_3} \\ \frac{\partial \hat{s}_3}{\partial c_1} & \frac{\partial \hat{s}_3}{\partial c_2} & \frac{\partial \hat{s}_3}{\partial c_3} \end{bmatrix}. \tag{4.35}$$

4.3.2 Heuristic monotonic and positive definiteness constraints

This subsection introduces heuristic constraints to enforce monotonicity and positive definiteness of the Jacobian matrix in the deep neural network training process by utilizing a point-wise loss function. This approach draws inspiration from finite element analysis, as described by Gupta (Gupta et al., 2019).

Let $\mathcal{L}^{(PD)}$ be the penalty that characterizes the violation of positive definiteness of $\widehat{\boldsymbol{H}}^{(i)}$. Consider the principal minor test (Prussing, 1986) for semi-definite matrices, which state that "a necessary and sufficient condition for a symmetric matrix $n \times n$ be positive semi-definite is that all *principal minors* are non-negative" (Prussing, 1986). Consequently, to fulfill the principal minor test, the matrix should be

- symmetric matrix,
- three principal minors of $\widehat{\boldsymbol{H}}^{(i)}$ are required to be positive.

However, during the training process, at each iteration, the predicted tensor $\widehat{\boldsymbol{H}}^{(i)}$ may not be symmetric. Therefore, the positive definiteness constraint is incorporated within two steps: symmetric constraint and principal minor test of the $\widehat{\boldsymbol{H}}^{(i)}$.

The symmetric property is applied as follows:

$$\mathcal{L}^{(sym)} = \sum_{i}^{m} \log \left[\cosh \left[\widehat{\boldsymbol{H}}^{(i)^\top} - \widehat{\boldsymbol{H}}^{(i)} \right] \right], \tag{4.36}$$

where $\widehat{\boldsymbol{H}}^{(i)^\top}$ is the transpose of $\widehat{\boldsymbol{H}}^{(i)}$ at the i^{th} row in the dataset.

Subsequently, the principal minor test is applied to symmetric matrix $\widehat{\boldsymbol{G}}^{(i)}$ instead, where

$$\widehat{\boldsymbol{G}}^{(i)} = \left[\widehat{\boldsymbol{H}}^{(i)} \right]^\top + \widehat{\boldsymbol{H}}^{(i)} \tag{4.37}$$

The penalty term $\mathcal{L}^{(PD)}$ is constructed using three principal minors of $\widehat{\boldsymbol{G}}^{(i)}$ and is defined as follows:

$$\mathcal{L}^{(PD)} = \sum_{i}^{m} \max \left(0, \max \left(- \left(\det(\boldsymbol{\Delta}_1), \det(\boldsymbol{\Delta}_2), \det(\boldsymbol{\Delta}_3) \right)^{(i)} \right) \right), \tag{4.38}$$

where m : number of data points. $\{\boldsymbol{\Delta}_k\}_{k=1,2,3}$ denote the principal minors of the symmetric tensor $\widehat{\boldsymbol{G}}^{(i)} = \left[\widehat{\boldsymbol{H}}^{(i)} \right]^\top + \widehat{\boldsymbol{H}}^{(i)}$ at the i^{th} row in the dataset. Similarly to (4.39) case, the minus sign "-" in $\left[\det(\boldsymbol{\Delta}_1), \det(\boldsymbol{\Delta}_2), \det(\boldsymbol{\Delta}_3) \right]^{(i)}$ helps enforce the positive definiteness of Jacobian of the network.

Let $\mathcal{L}^{(mono)}$ be the penalty that characterizes the violation of monotonicity by enforcing non-decreasing behaviour[1]:

$$\mathcal{L}^{(mono)} = \sum_{i}^{m} \max\left(0, \max\left(-\text{diag}\left[\widehat{\boldsymbol{H}}^{(i)}\right]\right)\right), \tag{4.39}$$

where m : number of data points, $\text{diag}\left[\widehat{\boldsymbol{H}}^{(i)}\right]$ is the diagonal components of $\widehat{\boldsymbol{H}}^{(i)}$ at the i^{th} row in the dataset. The inner max operator computes the element-wise maximum, resulting in a vector, and the outer max operator takes the maximum values on that vector. During the training process, the monotonically decreasing trend has a negative value, hence, the minus sign "-" in $-\text{diag}\left[\widehat{\boldsymbol{H}}^{(i)}\right]$ results in positive values, penalizing the gradient.

The final loss function consists of three components: the empirical loss of the neural network (as discussed in Subsection 4.2.4), the monotonic non-decreasing constraint, and the positive definiteness constraint:

$$\mathcal{L}^{(total)} = \gamma_1 \mathcal{L}^{(NN)} + \gamma_2 \mathcal{L}^{(mono)} + \gamma_3 \mathcal{L}^{sym} + \gamma_4 \mathcal{L}^{(PD)}, \tag{4.40}$$

where γ_i is the weight of each loss term. The construction of the loss function in a manner similar to the multi-target regression problems makes it akin to a sub-task of multi-task learning (Ruder, 2017; Y. Zhang and Q. Yang, 2021). To determine the weights γ_i, weighting sub-losses methods in multi-task learning methods (Senushkin et al., 2023) such as Uncertainty Weighting loss (Kendall, Gal, and Cipolla, 2018), or Random Weighting loss (Lin et al., 2022) are employed due to their state-of-the-art efficiency and ease of implementation. The specific method will be selected based on its performance specifically for the considered model.

In this work, `PyTorch` framework (Paszke et al., 2019) is employed to implement and train the neural networks. To seek the optimal parameters $\boldsymbol{\theta}^*$, a variant of SGD algorithm such as Adam (Kingma and Ba, 2014) is utilized:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{total}. \tag{4.41}$$

4.4 Symbolic representation of DNNs-based model inside FEniCSx

The modified DNNs-based model, obtained after training, is integrated into FEM software to model compressible hyperelastic materials. In this work, since `FEniCSx` is employed, and thus a Unified Form Language (UFL) version of the model is implemented. The objective of defining $\hat{\boldsymbol{s}}(\boldsymbol{c}; \boldsymbol{\theta}^*)$ (Eq. (LOGNet)) symbolically is to establish the variational form (3.13), which can be automatically assembled and solve the system inside `FEniCSx`.

[1] For isotropic materials, this constraint can be completely ignore as positive definite constraint of the NN's Jacobian matrix already fulfilled this requirements

To achieve this integration, each component of the network is implemented in UFL, effectively reconstructing the trained network. For example, the following code snippet provides custom implementations of `PyTorch`'s `nn.Linear` and `nn.ReLU` modules, extending their functionality for usage in UFL computations:

```
1  class Linear(nn.Linear):
2      # Perform linear transformation using UFL.
3      def forward_ufl(self, mesh, x):
4          weight_numpy = self.weight.detach().cpu().numpy()
5          y = ufl.as_matrix(fem.Constant(mesh, weight_numpy_array)) * x
6          if self.bias is not None:
7              bias_numpy = self.bias.detach().numpy()
8              y += ufl.as_vector(fem.Constant(mesh, bias_numpy_array))
9          return y
10
11 class ReLU(nn.ReLU):
12     # Apply ReLU activation using UFL.
13     def forward_ufl(self, mesh, x):
14         y = ufl.as_vector([ufl.max(0, v) for v in x])
15         return y
```

Note that UFL lacks built-in support for vectorized activation functions. Thus, when the output is not scalar, one must loop over its components, as shown in line 14 in the code snippet below. This method enables seamless integration of the `PyTorch`-based version with UFL computations.

The final UFL version of ANN's prediction $\hat{\boldsymbol{s}}(\boldsymbol{c};\boldsymbol{\theta}^*)$ is as follows:

```
17 s1, s2, s3 = ANNmodel.forward_ufl([c1,c2,c3])
```

Upon obtaining the UFL version of the predicted outputs $\hat{\boldsymbol{s}}$, they are inserted in the classical variational approach of Eq. (3.13), resulting in:

$$\delta\widehat{W}_{int} = \frac{1}{2}\int_{\Omega_0} \delta\boldsymbol{c}\cdot\hat{\boldsymbol{s}}(\boldsymbol{c};\boldsymbol{\theta}^*)\,dV_0\,. \tag{4.42}$$

The linearization form Eq. (4.42) can be assembled symbolically using the symbolic differentiation capabilities of UFL:

$$D_{\Delta\boldsymbol{u}}\delta\widehat{W}_{int} = \frac{1}{2}\int_{\Omega_0}\left[\delta\boldsymbol{c}(\boldsymbol{u})\cdot\widehat{\boldsymbol{H}}(\boldsymbol{c};\boldsymbol{\theta}^*)\cdot D_{\Delta\boldsymbol{u}}\boldsymbol{c}(\boldsymbol{u})\right]\,dV_0\,. \tag{4.43}$$

It should be noted that, in practice, the assembling process for Eq.(3.17) might not be feasible for very complex DNN architectures, such as those with more than 4 hidden layers, each containing more than 256 neurons.

Chapter 5

DNN-based surrogate models for non-linear isotropic elasticity in principal space

Contents

5.1 Introduction

Typically, using a well-trained standard MLP as a stress-strain response function yields good results, provided that the inputs injected into the surrogate model remain within the data range used for training the model since MLPs are supervised learning models. However, several challenges arise when solving nonlinear boundary-value problems using the Newton-Raphson method, e.g. slow convergence or divergence (Fuhg, Marino, and Bouklas, 2021).

During Newton-Raphson intermediate step, since the physical equilibrium has not yet been reached, the range of inputs provided for the DNN-based model suffers from substantial fluctuations. This may lead the DNN model to extrapolate beyond its training range, potentially resulting in unreliable predictions and convergence issues for Newton-Raphson iterations.

Another challenge is that the Newton-Raphson method relies on the computation of the tangent stiffness matrix, which requires the first-order derivatives of the stress-strain relationship, as described in Eq. (3.17). This derivative of the outputs with respect to the inputs of the surrogate model may not be accurate since the information about it is not presented in the training data set.

Furthermore, the derivatives might exhibit discontinuities and/or their values at the stress-free configuration are undefined due to specific types of employed activation functions. This lack of accuracy and continuity can result in poor convergence properties and inaccurate solutions for boundary-value problems. To address these challenges, this chapter will propose the strategies as follows.

(1) Employing LOGNet-based constitutive model proposed in subsection 4.3.1. This model satisfies the first four general requirements, thanks to the choice of input/output pair and its special architecture, specifically:

1. Principle of material frame indifference (Subsection 2.4.1): The material response is independent of the choice of reference frame, ensuring objectivity in the representation of the material behaviour.
2. Material symmetry transformation (Subsection 2.4.2): The material exhibits symmetry under proper orthogonal transformations $\boldsymbol{Q}$.
3. Stress-free condition for isotropic material (subsection 2.4.3).
4. Growth conditions in large strains (subsection 2.4.4).

By satisfying these assumptions, the LOGNet-based constitutive model can be considered the most general NN-based surrogate model for nonlinear isotropic elastic materials. Its values at stress-free configuration are guaranteed to be zeros and its derivative exists by choosing appropriate activation functions (refer subsection 4.3.1).

Incorporate further mathematical consequences:

5. Consequences of BE-inequalities: Monotonic behaviour of generalized principal stresses s_1, s_2, s_3 with respect to generalized principal strains c_1, c_2, c_3.
6. Consequences of Hill's inequalities: the Jacobian matrix (Eq. (4.35)) is positive definite for all strains.

By altering the training process, the LOGNet-based constitutive model will satisfy all six requirements making it the general NN-based surrogate model for nonlinear isotropic hyperelastic materials.

(2) Employing incremental prescribed Dirichlet boundary conditions, a traditional approach for the principal formulation of isotropic elastic materials, e.g., Connolly, MacKenzie, and Gorash (2019) and Miehe (1993).

(3) A fallback constitutive predictor using the consistency condition between isotropic nonlinear elasticity and isotropic linear elasticity.

These adjustments will encourage the DNN-based model to obey key physical constraints (three items above in the scope of this chapter), ultimately achieving more numerically stable and plausible physical solutions.

This chapter is organized as follows. In Section 5.2, virtual experiments are employed to collect synthetic data for developing a DNN-based surrogate constitutive model for nonlinear isotropic elastic materials. By simulating various deformation modes, a dataset of principal right Cauchy strains and principal 2^{nd} Piola-Kirchhoff stress pairs are generated. The dataset is then partitioned into training, validation, and testing subsets, and the distributions of the strain and stress components are analyzed using box plots and histograms. Section 5.3 presents the training process of LOGNet deep neural network-based surrogate constitutive model for isotropic nonlinear elastic material is presented. The hyperparameter optimization process is employed to determine the optimal architecture for LOGNet. Both synthetic (generated from Ogden-Storåkers model) and experimental (Kossa and Berezvai, 2016) datasets are utilized to validate the performance of the proposed approaches. The validations are presented in Sections 5.4 and 5.5. Next, Section 5.6 present extensive numerical analysis and simulation of the DNN-based model, and their advantage and disadvantage are highlighted. Datasets from the Mooney-Rivlin material model are also used to train and further validate the performance of the surrogate models. Furthermore, the analysis of extreme values and the solutions are also outlined in this section. They offer insights into the numerical instability and a comparison of the standard DNN-based surrogate models and LOGNet.

5.2 Constitutive data collection

In this study, virtual experiments are employed to collect synthetic data for developing a deep neural network-based surrogate constitutive model for nonlinear isotropic Cauchy elastic materials. By simulating various deformation modes, a dataset of squared principal stretches, $\boldsymbol{c} = [c_1, c_2, c_3]^\top$, and corresponding principal 2^{nd} Piola-Kirchhoff stress, $\boldsymbol{s} = [s_1, s_2, s_3]^\top$, pairs are generated and collected at the Gauss points [1]. These simulations are performed on a 3D closed-cell polyethylene foam unit cube using the Ogden-Storåkers model (Eq. (2.126)) with the corresponding set of parameters reported in Table 3.2. This comprehensive dataset serves as a foundation for training our deep learning-based surrogate model, offering a promising avenue for efficient and accurate material characterization.

To generate the datasets, four basic deformation modes are utilized, namely unconstrained uniaxial compression/tension (UT/UC), unconstrained simple shear (SS), and twisting around the $z-$centre line, which connects two points $\big(0.5, 0.5, 0\big)$ and $\big(0.5, 0.5, 1\big)$, of the cube (Tw). The boundary conditions are different from the one in section 3.5 such that the squared principal stretch and stress cover a wide range of values, hence, the name "unconstrained" is used.

Dirichlet boundary conditions on the unit cube are as follows.

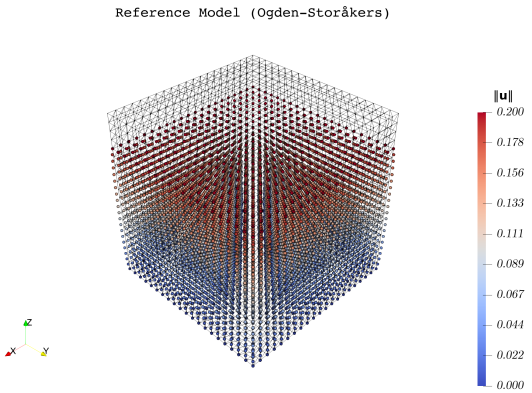
- For unconstrained uniaxial compression/tension, the $z-$component of all nodes on the top face (at $z = 1$) is prescribed, with $u_z = \pm 0.2\,\text{mm}$, two tests are conducted at 20% of compression/tension. At the bottom face (at $z = 0$), all the nodes remain fixed.
- For unconstrained simple shear, the $z-$component of all nodes on the top face (at $z = 1$) is fixed with $u_z = 0$. At the bottom face, all nodes remain fixed. The $x-$component of all nodes in the cube is prescribed with $u_x = ky$, where $k = \pm 0.2\,\text{mm}$, two tests are conducted with 20% is the amount of shear.
- For the case of twisting around the $z-$centre line: all nodes are fixed at the bottom face (at $z = 0$). All nodes in the cube are twitted to an angle clockwise and counterclockwise: $\varphi = \pm 30°$.

The simulations were conducted on a 3D closed-cell polyethylene foam unit cube with dimensions of $1\,\text{mm} \times 1\,\text{mm} \times 1\,\text{mm}$, employing a structured mesh consisting of $20 \times 20 \times 20$ tetrahedron elements, totalling 9261 nodes and 52 808 elements. The shape function is piecewise linear globally continuous functions of order 1.

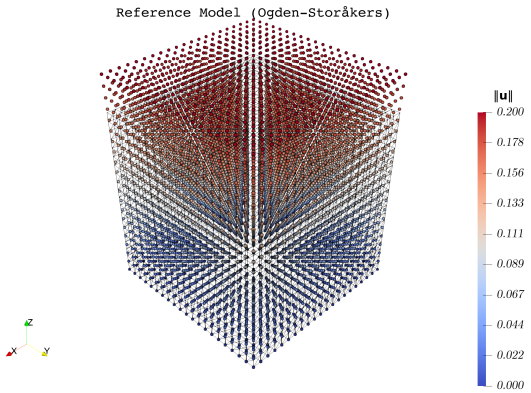
Figure 5.1 illustrates the Gauss points of these deformation modes, Fig. 5.1a and 5.1b shows the UT/UC cases, SS is presented in Fig. 5.1c, and Tw is demonstrated in Fig. 5.1d.

[1] Gauss points, or Gaussian points, are specific locations within a finite element where numerical integration is computed using Gaussian quadrature. In the finite element method, these points coincide with the roots of Legendre polynomials, and they are chosen alongside their associated weights. The selection of Gauss points depends on the type and order of the finite element used. Further details in (Bathe, 2006, section 5.5.3), (Zienkiewicz and Taylor, 2000, Chapter 9).

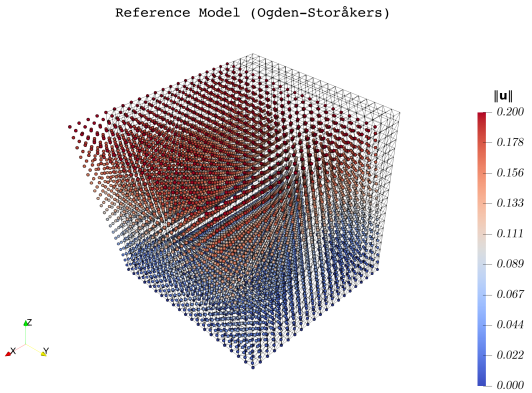
Figure 5.1 Visualization of deformation modes: unconstrained uniaxial compression/tension, simple shear, and twisting around the symmetry-axis of the cube. These fundamental deformation modes were employed to generate synthetic datasets for the study of isotropic hyperelasticity.



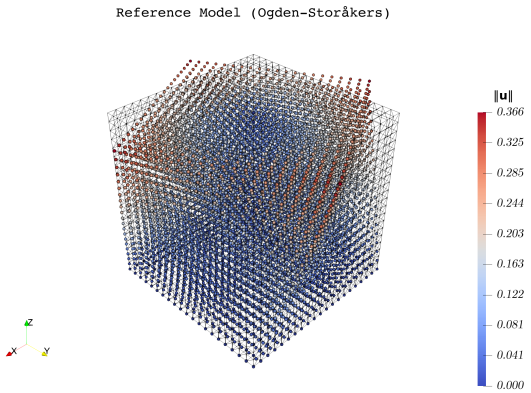
(a) Unconstrained uniaxial compression of 20%.



(b) Unconstrained uniaxial tension of 20%.



(c) Simple shear of 20% amount of shear.



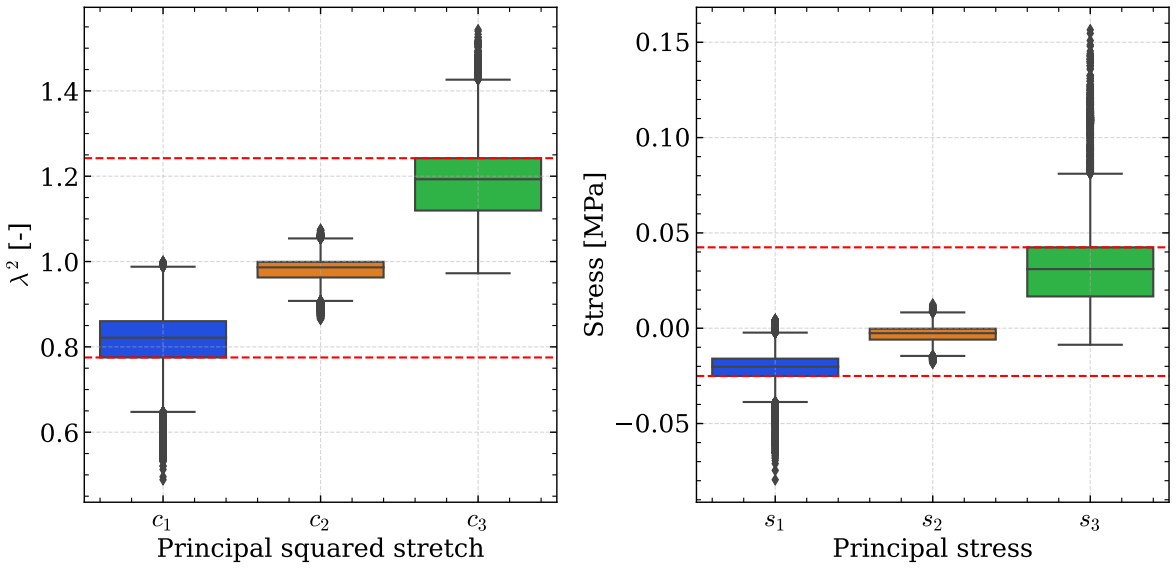
(d) Twisted 30° about the $z-$center line.

Initially, a total of 4 320 000 (six tests) data pairs were collected at the quadrature points during the simulations. However, a significant portion of the data was found to be duplicated, hence, a cleaning process was conducted to remove duplicate values of $\mathbf{c}^{(i)}$ and their corresponding $\mathbf{s}^{(i)}$ pairs. As a result, a total of 51 656 unique data points remained after the cleaning process. These data points were then combined to form the dataset $\mathcal{X} \in \mathbb{R}^{m\times 3}$ and $\mathcal{Y} \in \mathbb{R}^{m\times 3}$, where m represents the number of data points:

$$\mathcal{X} = \begin{bmatrix} \boldsymbol{c}^{(0)} \\ \boldsymbol{c}^{(1)} \\ \vdots \\ \boldsymbol{c}^{(m)} \end{bmatrix} = \begin{bmatrix} c_1^{(0)} & c_2^{(0)} & c_3^{(0)} \\ c_1^{(1)} & c_2^{(1)} & c_3^{(1)} \\ & \vdots & \\ c_1^{(m)} & c_2^{(m)} & c_3^{(m)} \end{bmatrix}; \quad \mathcal{Y} = \begin{bmatrix} \boldsymbol{s}^{(0)} \\ \boldsymbol{s}^{(1)} \\ \vdots \\ \boldsymbol{s}^{(m)} \end{bmatrix} = \begin{bmatrix} s_1^{(0)} & s_2^{(0)} & s_3^{(0)} \\ s_1^{(1)} & s_2^{(1)} & s_3^{(1)} \\ & \vdots & \\ s_1^{(m)} & s_2^{(m)} & s_3^{(m)} \end{bmatrix}. \tag{5.1}$$

To create subsets for training, validation, and testing, 80% of the total data points (i.e., 37 568 points) were assigned to the training set, whereas 10% (i.e., 4696 points) was allocated to the validation set and also 10% (i.e., 4696 points) for the test set. Figure 5.2 displays the

Figure 5.2 The boxplot of the distribution of training dataset: c_1, c_2, c_3, s_1, s_2, and s_3. The inner area defined by the red lines is the training range.



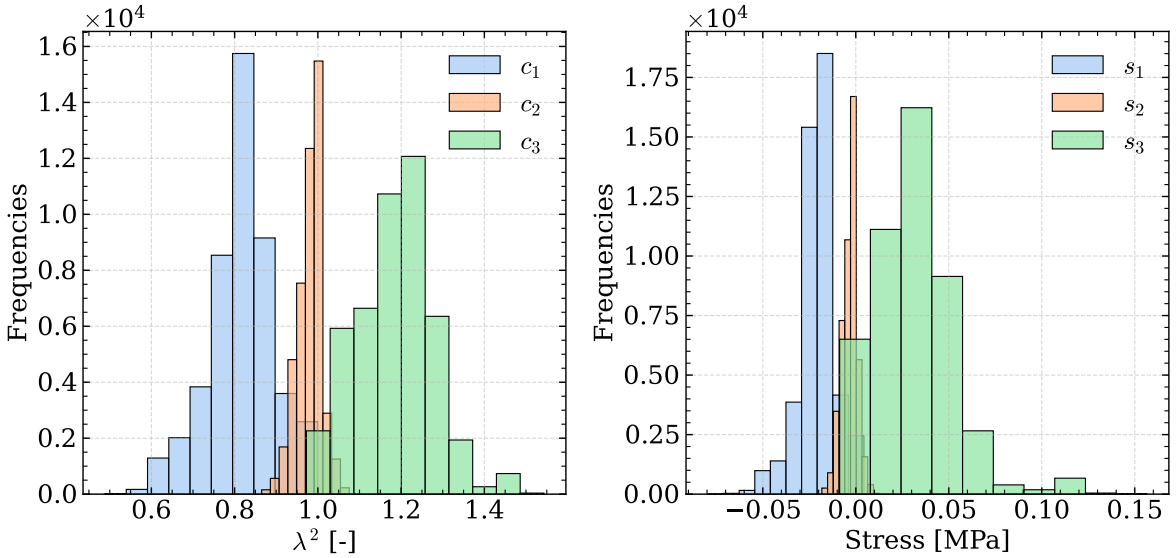
distribution of the generated values for c_1, c_2, c_3, s_1, s_2, and s_3. The median values for c_1, c_2, and c_3 are approximately 0.81, 0.98, and 1.18, respectively. The interquartile ranges (IQR) for these variables span from approximately 0.77 to 0.86 for c_1, 0.96 to 0.99 for c_2, and 1.12 to 1.24 for c_3.

The number of outliers in this synthetic data might seem unnatural at first glance. However, they are actually the extreme values of $\boldsymbol{c}$ and $\boldsymbol{s}$ at the Gaussian points of elements located at the edges of the mesh. Another instance when outliers occur is when there are highly skewed[1] or abnormal elements. These extreme values are unexpected due to the effects of boundary conditions, and concentrated deformations or stresses. Therefore, such outliers should not be considered during the training process of the DNNs. A window of IQR values is created (the inner areas bounded by the red lines on Fig. 5.2) for this purpose, which range from the 25^{th} percentile of c_1 to the 75^{th} percentile of c_3. This window will be used to monitor the input range behaviour of the DNN-based surrogate model during the intermediate step of the Newton-Raphson process for solving the nonlinear boundary value problems.

The histogram plot, Fig 5.3, represents a visual representation of the distributions of the data $\mathcal{X}$ and $\mathcal{Y}$. For c_1, the histogram shows that the values are concentrated around the range of 0.8 to 0.9, with a peak around 0.82. There is a gradual decrease in frequency as the values move towards the extremes. Similarly, for c_2, the histogram indicates a distribution centered around 0.98, with a peak near that value, and the frequencies decrease as the values deviate from the central point. Lastly, for c_3, the histogram displays a distribution centered around 1.18, with a peak near that value, and the frequencies decrease gradually as the values move away from the central point.

[1]Element skewness is the deviation between the optimal cell size to the existing cell size.

Figure 5.3 The histogram of the distribution of training dataset: c_1, c_2, c_3, s_1, s_2, and s_3.



5.3 LOGLU model structure and training process

This section presents the training process of a LOGNet-based surrogate constitutive model for nonlinear isotropic elastic material. The dataset presented in subsection 5.2 is clearly admitted to the aforementioned assumptions, making it suitable for training the DNN. Specifically, the inputs consist of principal strains, while the outputs comprise principal stresses, satisfying the first and second conditions. Additionally, the virtual dataset generated from the experimental of closed-cell polyethylene foam, hence it satisfies the third requirement.

5.3.1 Tuning for LOGLU deep neural networks

The synthetic dataset used for training the DNNs consists of $\mathcal{X} \in \mathbb{R}^{m\times 3}$ and $\mathcal{Y} \in \mathbb{R}^{m\times 3}$, where $m = 37\,568$ represents the total number of data points (Section 5.2). The architecture of LOGNet begins with the LOGLU activation function after the input layer. The rest of the hyperparameters of the DNN are determined using the HPO framework (Subsection 4.2.4).

To perform the HPO efficiently, it is advisable to avoid adding unimportant hyperparameters since the complexity of HPO grows exponentially with the number of parameters. This work employs three algorithm pairs, parameter-sampling and pruning algorithms, to seek the architecture of LOGNet. These are (1) TPE algorithm (Bergstra et al., 2011) combined with Hyperband (L. Li et al., 2018), (2) CMA-ES (Hansen and Ostermeier, 2001) paired with median stopping rule (Golovin et al., 2017), and (3) Random Search sampler (Bergstra and Bengio, 2012) also with Median pruner. These pairs are constructed based on the benchmark result of Optuna's documentation. The search space for the LOGNet consists of three hyperparameters, their names, and ranges are shown in Table 5.1.

Table 5.1 The search space of the hyperparameters of the LOGNet.

Hyperparameter	**Type**	**Range**
Number of hidden layers (L)	`int`	$[2, \ldots, 4]$
Number of neurons (l_n)	`int`	$[9, \ldots, 150]$ (`step` $= 3$)
Activation function ($\phi(x)$)	`category`	[SELU, Swish, Softplus, Snake]

The first hyperparameter is the number of hidden layers (L), which should not exceed four, to make it feasible to implement the resulting network in `FEniCSx`. In each hidden layer, the number of neurons (l_n) is searched in the range from 9 to 150 neurons, with the step of 3. It means that at each trial, the HPO algorithm samples one value from the sequence $\{9 + \texttt{step}, 9 + 2 \times \texttt{step}, \ldots, 9 + k \times \texttt{step} \leq 150\}$, where k is an integer.

Similarly, one of the activation function in the list SELU, Swish, Softplus, Snake will be selected. Only activation functions, $\phi(x)$, that satisfy $\phi'(0) \neq 0$ (derivative at $x = 0$ is not zero) are accepted (Subsection 4.3.1).

In total, 900 trials have been run for three studies on the HPO search. Each algorithm pair (a study) is executed in 300 trials with every trial subjected to 50 training epochs or pruned based on the value of the validation set error, which is the objective function to be optimised. `Adam` optimizer is employed with the setting as: $\eta = 0.001, \beta_1 = 0.9, \beta_2 = 0.99$. Table 5.2 presents the top five validation set losses achieved by each study across the range of hyperparameter configuration (Table 5.1).

Overall, the TPE pair gives the highest-performing architectures with moderate complexity, $L = 3$. This suggests a good balance between complexity and performance, as these models achieve low validation set losses. The CMA-ES with Median pruner also produces competitive architectures, although the validations losses are higher on average compared to TPE. However, it results give less complex models ($L = 2$ or $L = 3$) while maintaining moderate performance. The Random pair demonstrates a mixed performance, with some architecture archiving good results, but also some with higher values of losses. The well-performing structures have a higher hidden layer $L = 4$, and vice versa for $L = 2$. Again, this demonstrates a trade-off between model complexity and validation performance.

It is worth noting that, among four activation functions, the best-performance models are all employed by either Swish or Snake activation functions. Swish appears in a wider range of architectures and frequently achieves competitive performance. This suggests that Swish is a versatile activation function that can be effectively utilized in various models. While less frequently used compared to Swish, Snake still achieves consistently low validation set error, in the TPE case, Snake achieves the best results twice. It may be because of its monotonic and periodic behaviours (Ziyin, Hartwig, and Ueda, 2020). Its architectures tend to have moderate complexity ($L = 3$ or $L = 4$). This indicates that Snake is a promising choice, especially when combined with appropriate hyperparameters.

Table 5.2 The HPO results of LOGNet's architecture. Three HPO algorithms, namely TPE + Hyperband, Random + Median pruner, and CMA-ES + Median pruner, were employed. Every study underwent 300 trials, with each trial trained within 50 epochs or pruned. The top 5 validation set losses achieved by each algorithm are listed in increasing order.

HPO algorithms	$\phi(x)$	L	l_1	l_2	l_3	l_4	$\mathcal{L}^{NN}$ (validation set)
TPE + Hyperband	Snake	3	144	81	81	–	8.77×10^{-7}
	Snake	3	144	84	84	–	9.61×10^{-7}
	Swish	3	123	72	147	–	1.14×10^{-6}
	Swish	3	51	132	60	–	1.36×10^{-6}
	Swish	2	150	129	–	–	1.50×10^{-6}
Random Search + Median pruner	Swish	4	99	27	144	84	1.31×10^{-6}
	Swish	4	78	72	90	96	1.99×10^{-6}
	Snake	3	150	132	81	–	2.30×10^{-6}
	Swish	2	93	90	–	–	2.61×10^{-6}
	Swish	2	147	99	–	–	2.75×10^{-6}
CMA-ES + Median pruner	Swish	3	81	78	78	–	1.62×10^{-6}
	Swish	2	84	75	–	–	1.66×10^{-6}
	Snake	4	96	84	84	144	1.86×10^{-6}
	Snake	2	72	69	–	–	2.05×10^{-6}
	Snake	3	78	81	81	–	2.36×10^{-6}

It is evident that the configuration combines a Snake activation function with 3 hidden layers $\Big[144, 81, 81\Big]$ is the most promising architecture for LOGNet. This architecture achieves the smallest loss value at 8.77×10^{-7} among the trials across all three studies and strikes a favourable balance between complexity and performance. The full formulation is given by:

$$
\begin{aligned}
\hat{\boldsymbol{s}} &= \boldsymbol{h}^{(4)} \circ \boldsymbol{h}^{(3)} \circ \boldsymbol{h}^{(2)} \circ \boldsymbol{h}^{(1)} \circ \boldsymbol{h}^{(0)} \\
\boldsymbol{h}^{(0)} &= \text{LOGLU}(\boldsymbol{c}) \\
\boldsymbol{h}^{(1)} &= \text{Snake}\Big(\boldsymbol{h}^{(0)}\boldsymbol{W}^{(1)\top}\Big) \\
\boldsymbol{h}^{(2)} &= \text{Snake}\Big(\boldsymbol{h}^{(1)}\boldsymbol{W}^{(2)\top}\Big) \\
\boldsymbol{h}^{(3)} &= \text{Snake}\Big(\boldsymbol{h}^{(2)}\boldsymbol{W}^{(3)\top}\Big) \\
\hat{\boldsymbol{s}} &= \boldsymbol{h}^{(4)} = \boldsymbol{h}^{(3)}\boldsymbol{W}^{(4)\top}.
\end{aligned}
$$

The first derivative of the outputs of LOGNet with respect to its inputs reads:

$$\frac{\partial \hat{\boldsymbol{s}}}{\partial \boldsymbol{c}} = \frac{\partial \hat{\boldsymbol{s}}}{\partial \boldsymbol{h}^{(l)}} \left(\prod_{l=1}^{3} \frac{\partial \boldsymbol{h}^{(l)}}{\partial \boldsymbol{h}^{(l-1)}} \right) \frac{\partial \boldsymbol{h}^{(0)}}{\partial \boldsymbol{c}} = \frac{\partial \hat{\boldsymbol{s}}}{\partial \boldsymbol{h}^{(3)}} \frac{\partial \boldsymbol{h}^{(3)}}{\partial \boldsymbol{h}^{(2)}} \frac{\partial \boldsymbol{h}^{(2)}}{\partial \boldsymbol{h}^{(1)}} \frac{\partial \boldsymbol{h}^{(1)}}{\partial \boldsymbol{h}^{(0)}} \frac{\partial \boldsymbol{h}^{(0)}}{\partial \boldsymbol{c}} , \tag{5.2}$$

with

$$\frac{\partial \boldsymbol{h}^{(l)}}{\partial \boldsymbol{h}^{(l-1)}} = \frac{\partial \boldsymbol{h}^{(l)}}{\partial \left(\boldsymbol{h}^{(l-1)} \boldsymbol{W}^{(l)\top} \right)} \frac{\partial \left(\boldsymbol{h}^{(l-1)} \boldsymbol{W}^{(l)\top} \right)}{\partial \boldsymbol{h}^{(l-1)}} = \text{Snake}' \left(\boldsymbol{h}^{(l-1)} \right) \boldsymbol{W}^{(l)\top} . \tag{5.3}$$

At the undeformed configuration, $\boldsymbol{c} = \boldsymbol{1} = [1, 1, 1]$, the predicted values of LOGNet and its derivative are as follows:

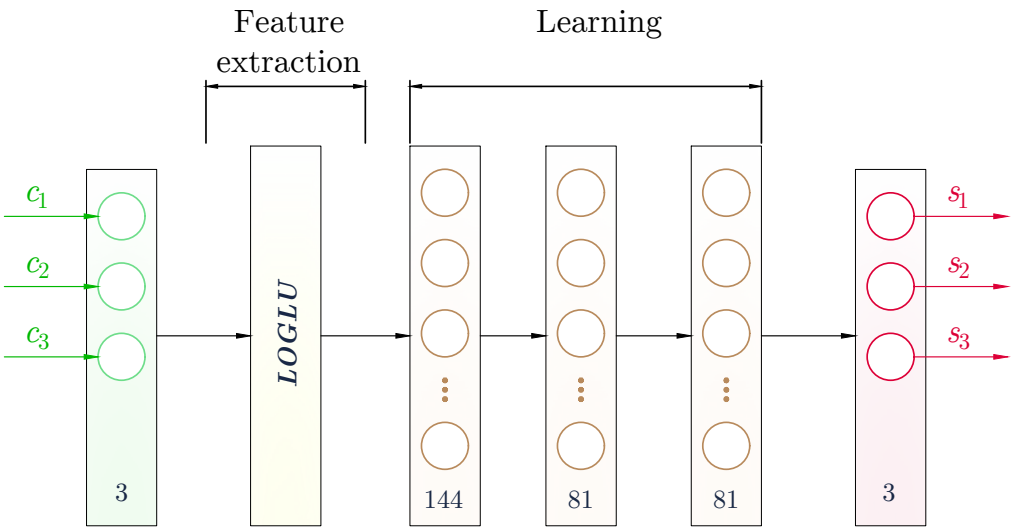
$$\hat{\boldsymbol{s}}(\boldsymbol{1}) = \boldsymbol{0} ; \tag{5.4}$$

$$\frac{\partial \hat{\boldsymbol{s}}(\boldsymbol{1})}{\partial \boldsymbol{c}} = \boldsymbol{W}^{(4)\top} \boldsymbol{1} \boldsymbol{W}^{(3)\top} \boldsymbol{1} \boldsymbol{W}^{(2)\top} \boldsymbol{1} \boldsymbol{W}^{(1)\top} \boldsymbol{1} \, \alpha^2 \neq \boldsymbol{0} . \tag{5.5}$$

Therefore, it satisfies 2.4.3 requirement.

The final LOGNet architecture (Fig. 5.4) follows a 3-hidden layer structure, denoted as $3 - 144 - 81 - 81 - 3$ LOGNet. The input layer and output layer of the DNN correspond to the principal squared stretches $\boldsymbol{c}$ and principal stresses $\boldsymbol{s}$.

Figure 5.4 The final LOGNet architecture



Based on this architecture, three DNNs are created for analysis and investigation of the methods proposed in Chapter 4. These are:

1. A standard multilayer perception with the same architecture $3-144-81-81-3$, denotes as MLP, is used as a based line for comparison.
2. A standard $3-144-81-81-3$ LOGNet, denotes as LOGNet is trained without the heuristic regularization.
3. The $3-144-81-81-3$ LOGNet combines with the monotonicity, symmetry, and positive definiteness regularization, denote as constrained LOGNet or LOGNet (SPD).

5.3.2 Training process of three models

Loss functions: The Log-cosh Loss function serves as the primary training criterion for the DNN models, inheriting the robustness to outliers of the Huber loss while maintaining smoothness and quadratic differentiability. It is utilized as the total loss function for training for both the vanilla MLP and vanilla LOGNet. The vectorized representation is as follows:

$$\mathcal{L}_{MLP} = \mathcal{L}_{\text{LOGNet}} = \mathcal{L}^{(NN)} = \sum_{i}^{m} \log\left[\cosh\left[\widehat{\boldsymbol{s}}(\boldsymbol{c}) - \boldsymbol{s}(\boldsymbol{c})\right]\right], \tag{5.6}$$

In the case of constrained LOGNet, which combines with the regularizations, the total loss takes the form:

$$\mathcal{L}^{(total)} = \gamma_1\mathcal{L}^{(NN)} + \gamma_2\mathcal{L}^{(mono)} + \gamma_3\mathcal{L}^{(sym)} + \gamma_4\mathcal{L}^{(PD)}. \tag{5.7}$$

where $\{\gamma_i\}_{i=1,2,3,4}$ represent the weights assigned to each sub-loss.

The determination of these weights relies on the individual values of the sub-losses (on the validation dataset) resulted from the training of the constrained LOGNet for 100 epochs. Note that, the $\mathcal{L}^{(total)}$ should not be used as a metric for comparing the models, since it is affected differently by each weighing method.

Table 5.3 presented the outcomes of various weighting strategies for the $3-144-81-81-3$ LOGNet model trained over 100 epochs. The model's performance is evaluated based on different validation sub-losses, where lower values indicate better performance. The weighting methods for sub-losses are considered as follows: unweighted losses, hand-tuned weights, Uncertainty Weighting Loss (UWL), and Random Weighting Loss (RWL). The unweighted losses just sum all sub-losses without assigning weights, whereas, the hand-tuned method manually assigns the weights for each sub-losses. The RWL is tested using three different distributions: the standard normal distribution (referred to as RWL-Normal), a uniform distribution over $[0, 1]$ (denoted as RWL-Uniform), and Dirichlet distribution (denotes as RWL-Dirichlet). The function $f_{RWL} = \text{softmax}$ for RWL-Normal and RWL-Uniform, while the identity function for RWL-Dirichlet. The results reveal that the monotone and positive

Table 5.3 Performance comparison of various weighting schemes. The performance of the individual validation sub-losses is presented, with lower values indicating better model performance. Experiments are conducted by training $3-144-81-81-3$ LOGNet over 100 epochs.

Methods	$\mathcal{L}^{(NN)} \downarrow$	$\mathcal{L}^{(sym)} \downarrow$	$\mathcal{L}^{(mono)} \downarrow$	$\mathcal{L}^{(PD)} \downarrow$
Unweighted losses: $\{\gamma_i = 1\}_{1,2,3,4}$	5.90×10^{-4}	4.15×10^{-7}	0.0	0.0
Hand-tuned: $\boldsymbol{\gamma} = \{0.6, 0.3, 0.05, 0.05\}$	3.12×10^{-6}	2.93×10^{-7}	0.0	0.0
UWL	$\mathbf{2.13 \times 10^{-6}}$	$\mathbf{1.26 \times 10^{-7}}$	0.0	0.0
RWL-Normal	3.38×10^{-6}	3.66×10^{-7}	0.0	0.0
RWL-Uniform	3.39×10^{-6}	2.31×10^{-7}	0.0	0.0
RWL-Dirichlet	3.36×10^{-6}	2.29×10^{-7}	0.0	0.0

definiteness criteria are quickly satisfied with $\mathcal{L}^{(mono)} = 0$ and $\mathcal{L}^{(PD)} = 0$, indicating successful handling constraints of the LOGNet model regardless of the weighting methods. Also, the sub-loss $\mathcal{L}^{(sym)}$ is almost one order lower than the $\mathcal{L}^{(NN)}$ loss. Hence, the values of primary sub-losses, $\mathcal{L}^{(NN)}$, and symmetric constraint $\mathcal{L}^{(sym)}$ are under consideration.

Among all the weighting schemes, UWL achieves the best performance with the lowest values for both $\mathcal{L}^{(NN)}$ and $\mathcal{L}^{(sym)}$. The values are $\mathcal{L}^{(NN)} = 2.13 \times 10^{-6}, \mathcal{L}^{(sym)} = 1.26 \times 10^{-7}$, respectively.

Following at the second place is the hand-tune weights with values of $\boldsymbol{\gamma} = [0.6, 0.3, 0.05, 0.05]$. Its performance are $\mathcal{L}^{(NN)} = 3.12 \times 10^{-6}, \mathcal{L}^{(sym)} = 2.93 \times 10^{-7}$. This custom set of values is chosen based on the experience of the author, it is shown that the contributions of $\mathcal{L}^{(NN)}$ and $\mathcal{L}^{(sym)}$ are usually dominated over the monotone and positive definiteness constraints.

The variations in loss values for RWL methods (Normal, Uniform, and Dirichlet) are relatively close to the hand-tuned method, suggesting that they provide comparable results. This method introduces randomness at each epoch by assigning random weights. The Dirichlet distribution achieves the best performance among the three distributions with a small margin. The consistent results across the random weighted schemes imply the randomization type may not significantly impact the overall performance of the model for this specific dataset.

The unweighting losses scheme assigns equal weights for all sub-losses. The results show that its performance is the lowest in comparison with the remaining weighting techniques.

In conclusion, UWL and hand-tuned methods emerge as the most effective approaches for achieving optimal model's performance. In practice, the UWL may introduce numerical instability and fluctuations in the total loss value. These issues can be managed by using different seeds for initialling the DNNs' parameters. Both the hand-tuned method and RWL yield compatible results and stable numerical stability during the training process. Nevertheless, the effectiveness of the hand-tuned weighting method varies depending on the specific case and demands substantial expertise from the machine learning practitioner to obtain an optimal set of weights.

The UWL method is selected for training the constrained LOGNet model based on the results from Table 5.3. The final form of the loss is as follows:

$$\begin{aligned}\mathcal{L}^{(total)}_{\text{LOGNet}\,(PD)} &= \sum_{k=1}^{4} \frac{1}{2\vartheta_k^2}\mathcal{L}^k + \log(1+\vartheta_k^2) \\ &= \frac{1}{2\vartheta^2_{(NN)}}\mathcal{L}^{(NN)} + \frac{1}{2\vartheta^2_{(sym)}}\mathcal{L}^{(sym)} + \frac{1}{2\vartheta^2_{(mono)}}\mathcal{L}^{(mono)} + \frac{1}{2\vartheta^2_{(PD)}}\mathcal{L}^{(PD)} \\ &+ \log(1+\vartheta^2_{(NN)}) + \log(1+\vartheta^2_{(sym)}) + \log(1+\vartheta^2_{(mono)}) + \log(1+\vartheta^2_{(PD)})\,.\end{aligned} \tag{5.8}$$

Here, ϑ_k is trainable parameters, which are determined during the training process.

Training setup: Three models undergo the exact same setup. This entails utilizing a batch size of 2048 and a total of 6000 epochs. The optimization adopts the Adam optimization algorithm (Kingma and Ba, 2017), initialized with a learning rate of 1×10^{-3}, which is subsequently reduced to 1×10^{-4} at epoch 3000, and further to 1×10^{-5} at epoch 5400[1].

In practice, the behaviour of the training process of the three models may exhibit differently due to their different architectures and loss functions. To ensure a fair performance comparison, several additional additional techniques were applied, including:

- The model's parameters were initialized five times with different random seeds, and each model was then trained for 200 epochs. The seed leading to the best performance and the least fluctuation in validation loss values among the five was selected.
- The training dataset size was gradually increased: 50% of the training data samples were used during the first 500 epochs, at the 800$^{\text{th}}$ epoch, 70% of the data points were utilized; and at the 1000$^{\text{th}}$ epoch, the entire training set, comprising 100% of the data, was employed.
- Gradient clipping (Goodfellow, Bengio, and Courville, 2016, section 10.11.1) was applied to address issues related to exploding gradients. Through trial and error, it was determined that setting the gradients at $\|\nabla\mathcal{L}\| \leq 3.0$ greatly improved the training stability and accelerated the empirical convergence[2].

5.3.3 The performance of MLP, LOGLU, and LOGLU (SPD) models

In this subsection, the detailed performance comparison of three models: vanilla MLP, vanilla LOGNet, and the contained LOGNet (SPD), utilized for hyperelastic materials modelling is presented. Although the MLP and LOGNet are not trained with the symmetric, monotonic, and positive definite constraints, they are also assessed with all individual sub-losses on both training and testing datasets for a clear picture of their capacity. The values of the predicted

[1] Although various learning rate scheduling schemes exist, this setup is simple and demonstrates comparative results.

[2] A similar recommendation is shown in (Mikolov, 2012, section 3.2.2), and theoretical justification can be found in J. Zhang et al. (2020).

principal stress vector, $\hat{\boldsymbol{s}}$, at the undeformed state, $\boldsymbol{c} = \Big[1.0, 1.0, 1.0\Big]$ are also examined for validating with the normalisation requirement.

The results of the training process are presented in Table 5.4. Overall, all models exhibit good generalization capacity, which can be seen from their gap between training sub-losses and their corresponding test sub-losses are small. Furthermore, all models successfully adhere to the monotonicity constraint as indicated by the zero values in the $\mathcal{L}^{(mono)}$ column. Similarly, they all satisfy the positive definiteness constraint with zero values in the $\mathcal{L}^{(PD)}$ column. In terms of the normalisation requirement, the MLP achieves a reasonable performance with $\hat{\boldsymbol{s}} = \Big[-9.85 \times 10^{-5}, -2.93 \times 10^{-6}, -2.36 \times 10^{-4}\Big]^{\top}$. The LOGNet and LOGNet (SPD) seamlessly satisfy it by construction.

The LOGNet model achieves the lowest $\mathcal{L}^{(NN)}$ values on the test set with $\mathcal{L}^{(NN)}_{MLP} = 1.60 \times 10^{-9}$, while the constrained LOGNet gets en par performance with the MLP model with the values of $\mathcal{L}^{(NN)}_{\text{LOGNet}} = 3.23 \times 10^{-9}$ and $\mathcal{L}^{(NN)}_{MLP} = 1.60 \times 10^{-9}$, respectively. On the symmetry preservation

Table 5.4 Performance comparison of three models: vanilla MLP, vanilla LOGNet, and constrained LOGNet (SPD). The performance of the individual sub-losses on both the training dataset and test dataset is presented, with lower values indicating better model performance. The final column shows the values of the predicted principal stress vector, $\hat{\boldsymbol{s}}(\boldsymbol{c} = \mathbf{1})$, at the undeformed state, $\boldsymbol{c} = \big[1.0, 1.0, 1.0\big]$.

Models	Dataset	$\mathcal{L}^{(NN)} \downarrow$	$\mathcal{L}^{(sym)} \downarrow$	$\mathcal{L}^{(mono)} \downarrow$	$\mathcal{L}^{(PD)} \downarrow$	$\hat{\boldsymbol{s}}(\boldsymbol{c} = \mathbf{1})$
MLP	Training	1.55×10^{-9}	1.90×10^{-6}	0.0	0.0	$\begin{bmatrix} -9.85 \times 10^{-5} \\ -2.93 \times 10^{-6} \\ -2.36 \times 10^{-4} \end{bmatrix}$
	Test	1.60×10^{-9}	1.94×10^{-6}	0.0	0.0	
LOGNet	Training	$\mathbf{4.88 \times 10^{-10}}$	1.21×10^{-7}	0.0	0.0	$\begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$
	Test	$\mathbf{4.78 \times 10^{-10}}$	1.22×10^{-7}	0.0	0.0	
LOGNet (SPD)	Training	3.28×10^{-9}	$\mathbf{1.44 \times 10^{-9}}$	0.0	0.0	$\begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$
	Test	3.23×10^{-9}	$\mathbf{1.50 \times 10^{-9}}$	0.0	0.0	

capability, the LOGNet (SPD) model demonstrated its dominance through its smallest $\mathcal{L}^{(sym)}$ values compared to both the vanilla MLP and LOGNet models. Specifically, for the test dataset, the LOGNet (SPD) model achieved $\mathcal{L}^{(sym)}_{\text{LOGNet}(SPD)} = 1.50 \times 10^{-9}$, while the vanilla MLP and LOGNet models achieved $\mathcal{L}^{(sym)}_{MLP} = 1.94 \times 10^{-6}$ and $\mathcal{L}^{(sym)}_{\text{LOGNet}} = 1.22 \times 10^{-7}$, respectively. This suggests that the LOGNet (SPD) model is better equipped to preserve the symmetry of the material's behaviour.

In summary, the LOGNet (SPD) model stands out as a promising choice due to its good performance in terms of symmetry preservation and competitive performance in terms of primary loss $\mathcal{L}^{(NN)}$. The results suggest that the LOGNet (SPD) model is well-suited for hyperelastic materials modelling tasks, with the ability to capture compatible complex material behaviours while satisfying important physical constraints.

5.4 Numerical validation of LOGLU-based constitutive models

To evaluate the performance of the models, synthetic data for both uniaxial (UA) and biaxial (BA) tests is generated using the Ogden-Storåkers strain-energy (2.126). Recall that the interquartile range (IQR) of the training dataset for c_1 is between 0.77 and 0.86 (section 5.2), and that of c_3 is 1.12 and 1.24. Therefore, the effective training range is defined by the IQR of c_1 and c_3, which belongs to the range of $\Big[0.77, 1.24\Big]$. It is reasonable to generate the synthetic datasets such that those IQRs are covered, hence, the range for the principal squared stretches is chosen from 0.2 to 1.6, i.e. $c^{(UA)} = c^{(BA)} \in \Big[0.2, 1.6\Big]$. By this range of inputs, both interpolation and extrapolation capabilities of the DNN-based constitutive models can be investigated. The transverse principal squared stretches for both cases can be approximated as follows (Kossa and Berezvai, 2016; Storåkers, 1986):

$$\lambda_T^{(UA)} = \lambda^{-\beta/(2\beta+1)}\,; \quad c_T^{(UA)} = c^{-2\beta/(2\beta+1)} \tag{5.9}$$

$$\lambda_T^{(BA)} = \lambda^{-2\beta/(\beta+1)}\,; \quad c_T^{(BA)} = c^{-4\beta/(\beta+1)}\,. \tag{5.10}$$

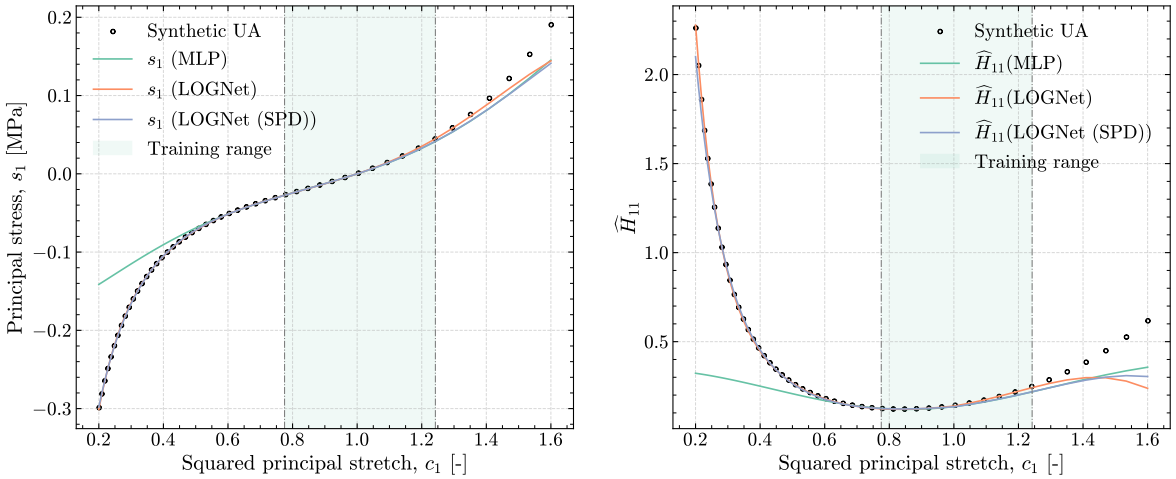
Figure 5.5 presents the comparison between the virtual Ogden-Storåkers synthetic data (considered as ground truth) and the predictions of three DNN-based constitutive non-linear hyperelastic material laws: the baseline MLP, LOGNet, and LOGNet (SPD). The evaluation covers both uniaxial (UA) and biaxial (BA) cases. The predictive performance of each model is plotted in coloured solid lines: green for the baseline MLP model, orange for the LOGNet, and blue for LOGNet (SPD). Synthetic data points are denoted by circle markers, while the training range is highlighted by the green shaded area. The UA test case is depicted in Figure 5.5a, while the BA scenario is illustrated in Figure 5.5b.

In the left-hand plots of the figures, the reference principal value, s_1, of the 2$^{\text{nd}}$ Piola-Kirchhoff stresses are plotted against the principal squared stretches (c_1) in the loading direction (x-axis). Notably, all DNN-based material models exhibit an impressive fit with the synthetic data within the training region (green area), for both the UA and BA cases.

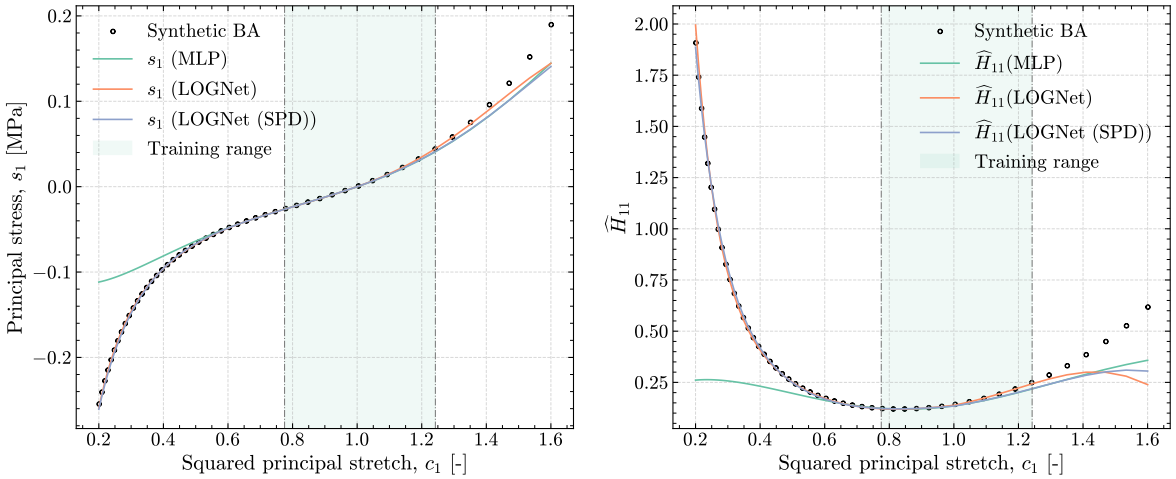
For the extrapolation capability of the models, it is evident that both LOGNet and LOGNet (SPD) models exhibit dominance over the baseline MLP model in the compression region, $c_1 < 1.0$. In this regime, the two LOGNet-based models align well with the ground truth data, demonstrating excellent accuracy. Moreover, their ability to accurately capture the asymptotic and monotonic behaviours highlights their proficiency in fulfilling growth requirements. On the contrary, the MLP model struggles to closely follow the asymptotic trend.

Within the tension regime, $c_1 \leq 1.0$, all constitutive models remain relatively accurate in capturing the upward trend, however, slightly less accurate compared to their performance in the compression region. LOGNet slightly slightly better performance than its counterparts

Figure 5.5 Performance comparison between the Ogden-Storåkers model and the predictions of three DNN-based constitutive non-linear hyperelastic material laws: the baseline MLP, LOGNet, and LOGNet (SPD). The left-hand plots depict the 2nd Piola-Kirchhoff stress (s_1) and the component $H_{11} = \frac{\partial \hat{s}1}{\partial c_1}$ is on the right-hand plot, whereas the squared principal stretch (c_1) is in x-axis for both uniaxial (UA) and biaxial (BA) cases. The green shaded area represents the training range. The overall R^2 scores for predicted principal stresses are $R^2_{MLP} = 0.794$, $R^2_{\text{LOGNet}} = 0.995$ and $R^2_{\text{LOGNet}(SPD)} = 0.993$.



(a) Performance comparison between Ogden-Storåkers model on the virtual uniaxial test.



(b) Performance comparison between Ogden-Storåkers model on the virtual equibiaxial test.

in this regard. The overall R^2 scores of the predicted principal stress of three models are $R^2_{MLP} = 0.794$, $R^2_{\text{LOGNet}} = 0.995$ and $R^2_{\text{LOGNet}(SPD)} = 0.993$.

The right-hand plots of the figures present the comparison between the component $H_{11} = \frac{\partial \hat{s}_1}{\partial c_1}$ of the derivative of the principal stress with respect to the squared principal stretch. It is evident that in both tests, all models continue to perform well inside the training regime with LOGNet performing slightly better than the other two.

In the compression regime, both LOGNet-based models, again, achieve great accurate extrapolation compared to the synthetic data in both cases, outperforming the MLP with a large margin. Both LOGNet capture significantly well the exponentially increasing trend of H_{11} in this region, again confirming their ability to satisfy the growth requirement. In the tension regime, all three models are comparable in extrapolation performance which is still well until the extension is larger than 1.4. All the values of $H_{11} > 0$ indicate the monotonic requirement is satisfied.

Additionally, the stresses are zero $s = 0$ at $c_1 = 1$ confirming the growth and normal requirements of hyperelasticity as discussed in Sect. 2.4.9. The overall R^2 scores of the predicted H_{11} of three models are $R^2_{MLP} = -0.171$, $R^2_{\text{LOGNet}} = 0.983$ and $R^2_{\text{LOGNet}(SPD)} = 0.985$.

In summation, both LOGNet-based material models consistently excel within the testing regime, displaying remarkable performance across both uniaxial and biaxial cases. Their proficiency in accurate extrapolation, asymptotic, monotonic behaviour, and overall fitting with the hyperelastic material properties.

5.5 Experimental validation of LOGLU-based constitutive models

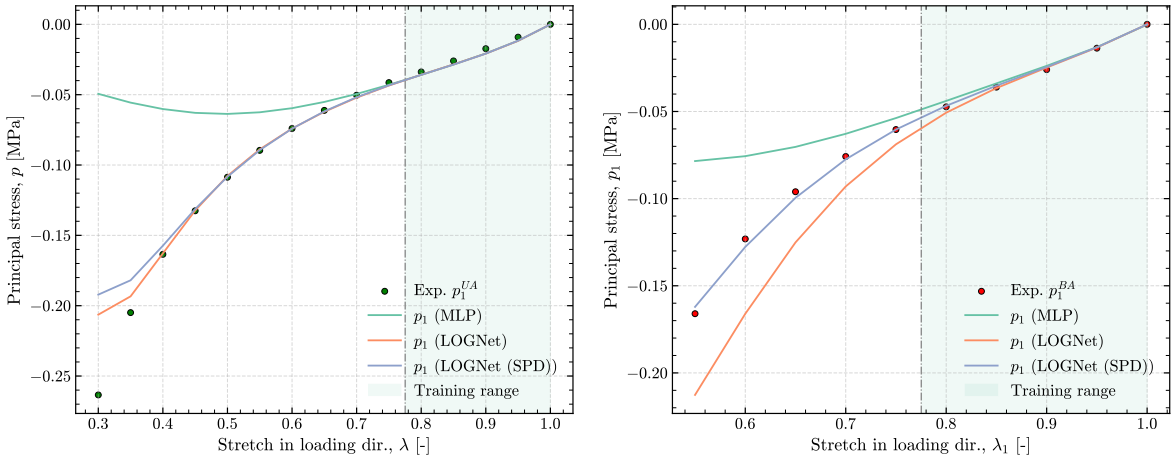
In this subsection, DNN-based constitutive models is validated using experimental data obtained from the uniaxial and equibiaxial compression (Kossa and Berezvai, 2016). Figure 5.6 provides a visual comparison between the predictions generated by three DNN-based constitutive non-linear hyperelastic material models—namely, the baseline MLP, LOGNet, and LOGNet (SPD)—and experimental data. The left subplot presents the UA case, while the right subplot illustrates the BA one. The predicted principal 2$^{\text{nd}}$ Piola-Kirchhoff stresses, $\hat{s}_i$, are converted to the principal 1$^{\text{st}}$ Piola-Kirchhoff stresses, $\hat{p}_i$ by:

$$\hat{p}_i = \lambda_i \hat{s}_i \,. \tag{5.11}$$

The predicted principal stresses for both UA and BA (p_1^{UA} and p_1^{BA}) are represented using distinct solid colours: green for the baseline MLP model, orange for LOGNet, and blue for LOGNet (SPD). Additionally, the experimental data points are depicted using filled markers, and the region of the training data is shaded in green. The graph indicates a good fit between the three models and the experimental data within the training regime for the UA case. However, for the BA case, it is evident that the LOGNet (SPD) outperforms its counterparts. The model predictions start at zero stress for $\lambda = 1$ and follow a similar trend for both cases, with decreasing stresses as λ decreases.

Beyond the training region, for both cases, the asymptotically decreasing trend of stresses is captured well by the extrapolation capability of both LOGNet models. Conversely, the MLP model displays limitations in effectively extrapolating this property. In the UA test, both

Figure 5.6 Performance comparison between the Ogden-Storåkers model and the predictions of three DNN-based constitutive non-linear hyperelastic material laws: the baseline MLP, LOGNet, and LOGNet (SPD). The left-hand plots depict the UA experimental data and the BA case is on the right-hand plot, whereas the squared principal stretch (c_1) is in the x-axis. The green shaded area represents the training range.



LOGNet models manage to extrapolate predictions adeptly until $\lambda_1 < 0.35$, beyond which a slight deviation from the trend is observed. Notably, in the BA case, the LOGNet (SPD) surpassed all other models to maintain significantly high accuracy in the stress predictions.

Overall, the graph suggests that the model predictions are a good approximation of the experimental data and satisfy all the constitutive requirements proposed in Sect. 2.4. The LOGNet (SPD) impressively captures the material behaviour well not only within the training range but also outside it, across all virtual and experimental tests. Furthermore, the LOGNet (SPD) demonstrates its capability to satisfy all proposed physical requirements. This indicates that the LOGNet (SPD) model exhibits high robustness against various test cases and serves as a reliable surrogate model for non-linear isotropic hyperelastic materials.

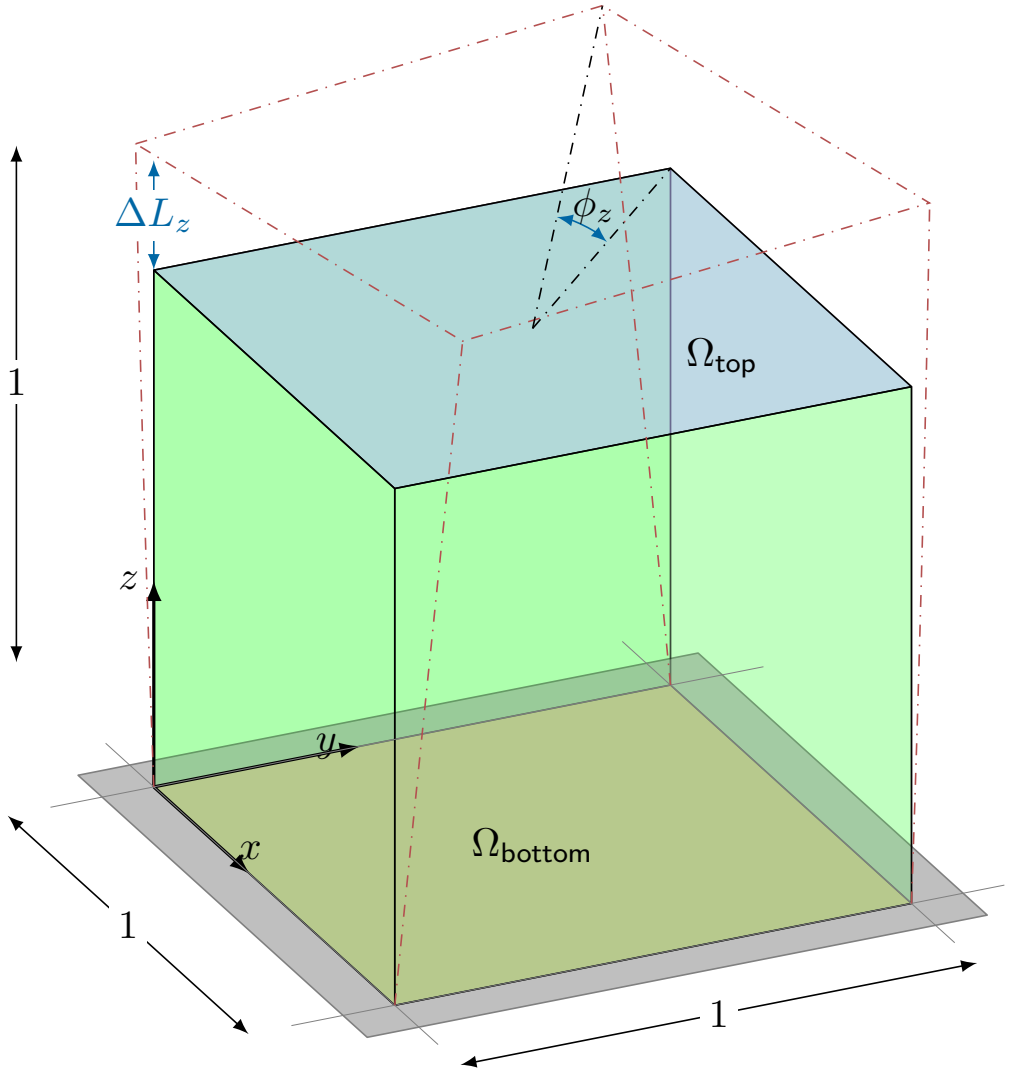
5.6 Finite element implementation and numerical simulation results

This section further examines the DNN-based models, namely standard MLP and LOGNet (SPD), through numerical examples of modelling a compressible 3D closed-cell polyethylene foam unit cube. Employing DNNs to completely replace traditional material law in the NR iterations may present side effects of low convergence or divergence. Therefore, the performance of these models is evaluated by the numerical results in terms of both the difference of the principal stresses and the numerical stability of the Newton-Raphson iterative process. The examples are conducted using `FEnicSx` following the procedure described in Section 4.4.

5.6.1 Numerical example setup

To evaluate the performance of the baseline MLP and LOGNet (SPD) surrogate model, a previously unseen (by the DNNs) large deformed benchmark simulation, referred to as "Tw-UT", was employed. The "Tw-UT" test involved simultaneous unconstrained uniaxial extension at 10% ($\Delta L_z = 0.1\,\text{mm}$) and twisting around the centered axis at an angle $\phi_z = 10°$ on the top face, while the bottom face was fixed. The 3D closed-cell polyethylene foam

Figure 5.7 Boundary conditions of simultaneously uniaxial and twisted test. On the top face, Ω_{top}, is subjected to 10% uniaxial displacement of $\Delta L_z = 0.1\,\text{mm}$ and twisting around the centered axis at an angle $\phi_z = 10°$. On the bottom face, Ω_{bottom}, all nodes are fixed.



unit cube with dimensions $1\,\text{mm} \times 1\,\text{mm} \times 1\,\text{mm}$ is used, and the boundary conditions are illustrated in Fig 5.7. The numerical model is implemented on a mesh with a $15 \times 15 \times 15$ grid, comprising 4096 nodes and 22958 elements. The "Tw-UT" benchmark is of interest as it combines two deformation modes simultaneously, while the dataset used for training the DNN only includes single-mode deformations.

To evaluate the DNN model's performance, 303 750 data points are extracted from each model and used to compute the displacements $\boldsymbol{u}$ and the 2^{nd} Piola-Kirchhoff stresses $\boldsymbol{s}$. Therefore, the number of data points is $m = 303\,750$, which is nearly 10 times the size of the training set. Two metrics are employed to assess the performance of the DNN model: the coefficient of determination, R^2 (Eq. (3.33)), and the relative error (Eq. (3.34)).

The simulation is executed without incremental loading and uses the Newton-Raphson method (NR) to seek the solutions of the boundary value problem. As the NR algorithm iterates towards a minimum for the solutions, the iterative process is stopped at two stages, at absolute residual values of $\texttt{atol} = 1 \times 10^{-6}$ and $\texttt{atol} = 1 \times 10^{-12}$ for numerical stability investigation.

It is important to note that the NR algorithm is sensitive to the initial guess, hence, poor initial values may lead to numerical instability and divergence in the NR solver. In this work, before commencing the NR procedure, the initial guess of the principal squared stretch vector is adjusted such that $\boldsymbol{c}_0 = [0.81, 0.98, 1.18]$, which matches the mean values of $\boldsymbol{c}$ in the training dataset. This modification ensures that computations using the DNN-based material laws start from a favourable point, leading to improved convergence behaviour in the NR iterations[1]. An alternative approach to handling the initial guess is to employ stepping load or stepping-prescribed Dirichlet boundary conditions, which is a standard technique for BVPs formulated in principal space (Miehe, 1993).

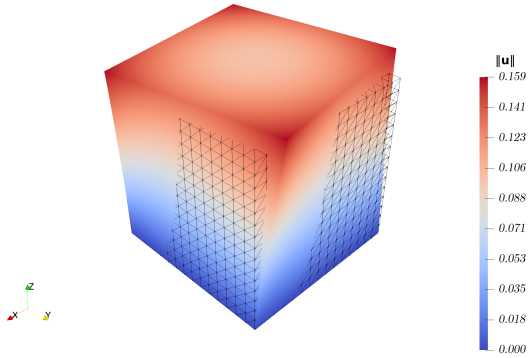
5.6.2 Numerical simulation results

Figure 5.8 presents the numerical results of the "Tw-UT" test, with the termination criterion of the NR iterative process set to $\texttt{atol} = 1 \times 10^{-6}$. The first row shows the magnitude of the 3D view of the predicted displacement fields, $\|\hat{\boldsymbol{u}}\|$, and the $X - Z$ plane at $x = 0.5$. The second row presents the norm of the difference between the principal stress vectors, $\|\boldsymbol{s} - \hat{\boldsymbol{s}}\|$, of LOGNet (SPD) and the reference model, Ogden-Storåkers. Since the MLP's performance, in this case, is close to that of LOGNet (SPD), its figures are omitted. The deformed cube is illustrated in colour, and the black wire-frame represents the undeformed cube for reference.

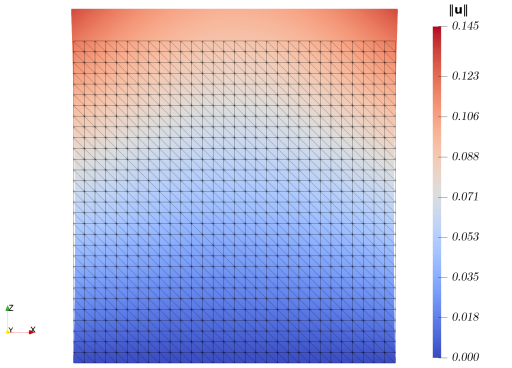
It is important to note that distorted or nonphysical displacement fields will occur if the employed DNN-based surrogate models are not well-trained. In this case, the 3D view and $X - Z$ plane at $x = 0.5$ figures reveal that the magnitude of the predicted displacement field of LOGNet (SPD), $\|\hat{\boldsymbol{u}}\|$, do not exhibit any nonphysical deformations. Excellent performance is achieved on the $\|\hat{\boldsymbol{u}}\|$ with an $R^2_u = 1.0$ and the error percentage is $err_u = 6.65 \times 10^{-2}\,\%$ over the whole domain.

[1]The mean values of the input vector are usually where the DNN models reach their peak performance, hence, it is reasonable to choose mean values as the initial guess. Several experiments conducted by the author have confirmed that this choice works.

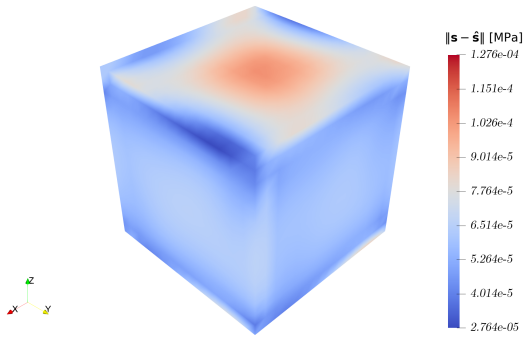
Figure 5.8 Performance of the DNN-based material law for a polyethylene foam unit cube subjected to the "Tw-UT", 10° twitted and 10 % extended test. The DNN-based model achieves: $R_u^2 = 1.0$ and $err_u = 6.65 \times 10^{-4}$ % on the displacement field and $R_s^2 = 1.0$ and $err_s = 2.23 \times 10^{-3}$ % on the stress field.



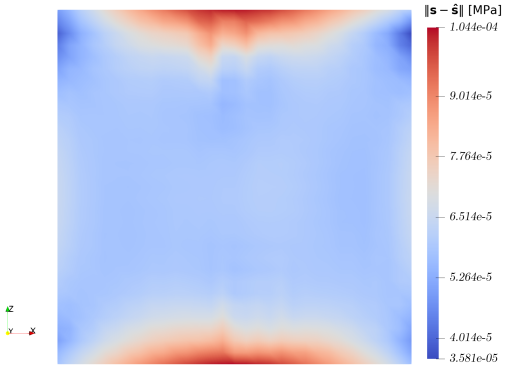
(a) Magnitude of displacement field: 3D view.



(b) Magnitude of displacement field: $X - Z$ plane at $x = 0.5$.



(c) Magnitude of the principal stress difference: 3D view.

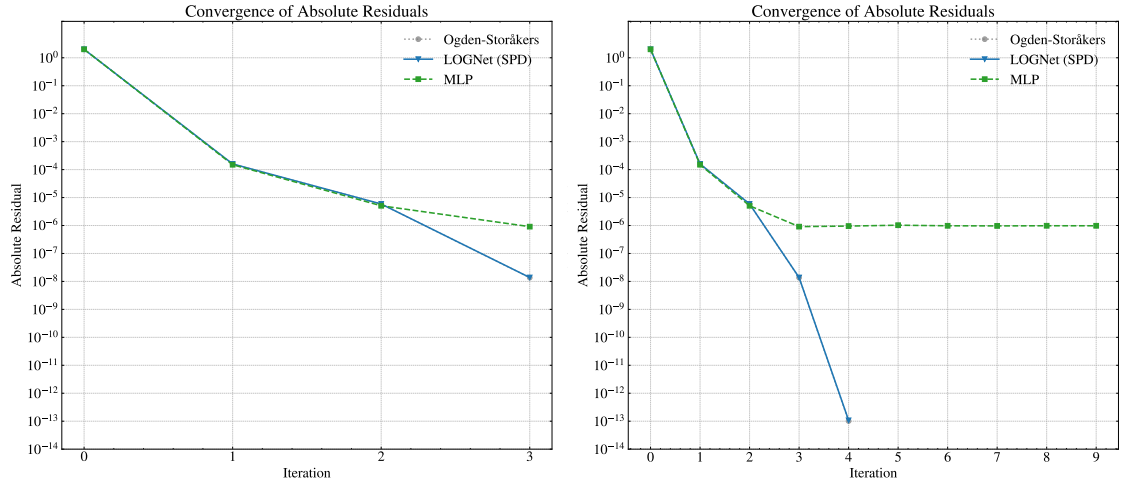


(d) Magnitude of the principal stress difference: $X - Z$ plane at $x = 0.5$.

A close examination of Figures 5.8c and 5.8d reveals nearly identical stress fields between the DNN-based predictions and that of the reference model. Overall, the LOGNet (SPD) also captures well the stress fields. The highest values of the magnitude of the principal stress difference are in the vicinity where the boundary conditions are applied. The maximum value of the norm is only $\max(\|\boldsymbol{s} - \hat{\boldsymbol{s}}_{\text{LOGNet}-SPD}\|) = 1.044 \times 10^{-4}$, which suggests a high level of agreement between the two approaches. Indeed, the adjusted coefficient of determination and the error percentage are $R_s^2 = 1.0$ and $err_s = 0.223\,\%$, respectively.

This similarity in performance underscores the effectiveness of both DNN-based surrogate models, even when exposed to complex, combined deformation modes not found in the single-mode training dataset.

Figure 5.9 Benchmark case: "Tw-UT". Comparison of the convergence behavior of the Newton-Raphson iterative process is conducted with three models, MLP (green dashed line), LOGNet (SPD) model (blue solid line), and the reference Ogden-Storåkers model (grey dashed dot line). Two termination criteria in terms of the absolute residual, $\texttt{atol} = 1 \times 10^{-6}$ (left) and $\texttt{atol} = 1 \times 10^{-12}$ (right) are presented.



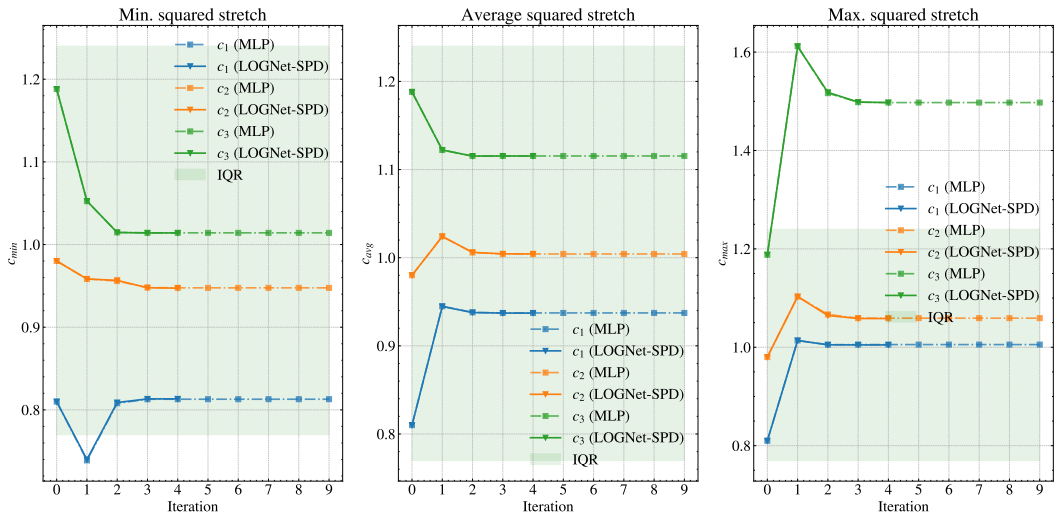
(a) Termination criteria: $\texttt{atol} = 1 \times 10^{-6}$ **(b)** Termination criteria: $\texttt{atol} = 1 \times 10^{-12}$

Further investigation of the behavior of NR's iterations for both the DNN-based material laws against the reference Ogden-Storåkers model is conducted. Two stages of the NR iterations are shown in Fig. 5.9. The left figure presents the termination criteria set to $\texttt{atol} = 1 \times 10^{-6}$ and $\texttt{atol} = 1 \times 10^{-12}$ shown on the right graph. The reference model is denoted by the grey dotted line, whereas the green dashed line corresponds to the MLP, and the blue solid line presents the LOGNet (SPD).

In the case of a termination criterion with $\texttt{atol} = 1 \times 10^{-6}$, LOGNet (SPD) matches nearly identically the reference model's performance. Although lacking behind the LOGNet (SPD), the baseline MLP model still demonstrates high accuracy and consistency, closely approaching the performance of the other counterparts.

Continuing the NR iterative process until the termination criteria reach $\texttt{atol} = 1 \times 10^{-12}$. The absolute residual value of the MLP-based model fluctuates around the absolute value of the residual $\epsilon_{abs} = 9.7 \times 10^{-7}$ until it is forced to be terminated after 10 iterations. This behaviour may originate from the inaccurate predictions of the MLP model, during the iterative process, at Gauss points that stand outside of the training regime. In contrast, the LOGNet (SPD) model maintains a performance closely aligned with the reference model, reaching a converged absolute value of the residual is $\epsilon_{abs} = 1.07 \times 10^{-13}$.

Figure 5.10 Benchmark case: "Tw-UT". The range of values of the squared stretch vector $\boldsymbol{c}$ during the Newton iteration. Comparison of the progression between MLP and LOGNet (SPD) models versus the IQR $IQR = [0.77, 1.24]$.



The updating progress of the range of inputs $\boldsymbol{c}$ is shown in Figure 5.10, covering the case of $\texttt{atol} = 1 \times 10^{-12}$. At each iteration, new values of $\boldsymbol{c}$ are updated and fed into the material models. The values of the MLP are presented using rectangle dash-dotted line whereas triangle solid lines for LOGNet (SPD). The colours assigned for each component of the inputs are blue for c_1, orange for c_2, and green is c_3. The average values of each component $\{c_i\}_{i=1,2,3}$ are illustrated in the middle graph, whereas the extreme values, minimum and maximum, of the components are plotted in the left and right graphs, respectively. Most values of the Gauss points in the domain are around the average values, and some points near the critical locations such as corners of the cube have extreme values. The effective training range is defined by the IQR of c_1 and c_3, which belongs to the range of $\left[0.77, 1.24\right]$ and denotes by the green-shaded area.

In the first four iterations, the average squared stretch graph indicates that most of the values of the input vector belong to the training range. Starting from the initial guess of $\boldsymbol{c}_0$, during the iteration, they remain within the training range. On the other hand, the extreme values of components c_1 and c_3 are outside the range, especially component c_3 with the peak at about 1.62, then gradually reducing to 1.48. Due to the extrapolation ability, both

DNN-based models still achieve good performance, clearly, LOGNet (SPD) has extrapolated better. However, they differ by small margins.

After four iterations, although with small margins of error in updating the values of inputs, the residual value of the MLP cannot reach the termination criterion and keeps oscillating. Note that these results are conducted with double-precision floating-point format (`float64`); therefore, these small error margins can be revealed. It suggests that surrogate models should achieve very high accuracy in order to maintain the numerical stability of the NR iterative process. Further investigations, where the influence of the extreme values is looked at in more detail, are conducted in Subsection 5.6.5.

5.6.3 Convergence analysis of various grid discretisations

Additional convergence analyses were conducted for the DNN-based models using various discretisations (mesh sizes) of the unit cube, as summarized in Table 5.5. Three different grid sizes were investigated: $10 \times 10 \times 10$, $15 \times 15 \times 15$, and $20 \times 20 \times 20$. Recall that the training data is generated from $20 \times 20 \times 20$ grid; therefore, at other grid sizes, the DNN-based models are subjected to extrapolation. The table reports the values of relative residual (ϵ_{rel}), absolute residual (ϵ_{abs}), and the R_s^2 score for the stress field, comparing the results between the reference model and the DNN-based models at each grid size.

Overall, the LOGNet (SPD) model consistently demonstrates convergence behaviour that closely mirrors the reference model across all three grid sizes, while the MLP model exhibits larger residual values. Both models provide robust stress predictions across different mesh sizes with $R_s^2 \geq 0.9992$ for the MLP, and archive $R_s^2 = 1.0$ for the LOGNet (SPD). All models share the same initial guess, $\boldsymbol{c}_0$. As the iterations progress to the second iteration, a similar convergence trend is observed across all three grid sizes, with both DNN-based material laws consistently approaching the reference model in terms of ϵ_{rel} and ϵ_{abs}. However, at the final iteration, the MLP model ends with residuals differing by one order of magnitude. The level of agreement between the LOGNet (SPD) model and the reference model demonstrates its reliability and robustness in capturing the material behaviours, especially in scenarios extrapolating to different resolution discretisation.

5.6.4 Extended numerical analysis of the LOGNet

This section provides further numerical analysis of the LOGNet (SPD) based model, aiming to extend the validation scope of its capabilities. Throughout this section, the benchmark scenario of the "Cantilever beam - uniformly distributed load" is utilized. The setup and boundary conditions are kept the same as shown in Subsection 3.5.2, except for the magnitude of the distributed loads, q, will be changed. The termination criterion of the NR iterative process is set to $\texttt{atol} = 1 \times 10^{-7}$ for all the tests. The Mooney-Rivlin strain-energy (2.129) is also employed as an extension to validate the robustness of the LOGNet model.

Table 5.5 Convergence behaviour of DNN-based models (MLP and LOGNet (SPD)) alongside the reference model on varying grid sizes.

Grid size	Iteration	ϵ_{abs}			ϵ_{rel}			R_s^2	
		Ogden-Storåkers	MLP	LOGNet (SPD)	Ogden-Storåkers	MLP	LOGNet (SPD)	MLP	LOGNet (SPD)
$10 \times 10 \times 10$	0	1.39	1.39	1.39	1.00	1.00	1.00	0.9993	1.0
	1	1.59×10^{-4}	1.59×10^{-4}	1.61×10^{-4}	1.14×10^{-4}	1.14×10^{-4}	1.16×10^{-4}		
	2	3.36×10^{-6}	3.79×10^{-6}	3.42×10^{-6}	2.41×10^{-6}	2.72×10^{-6}	2.45×10^{-6}		
	3	2.87×10^{-9}	8.72×10^{-7}	3.21×10^{-9}	2.05×10^{-9}	6.25×10^{-7}	2.30×10^{-9}		
$15 \times 15 \times 15$	0	2.00	2.00	2.00	1.00	1.00	1.00	0.9992	1.0
	1	1.57×10^{-4}	1.49×10^{-4}	1.59×10^{-4}	7.86×10^{-5}	7.45×10^{-5}	7.93×10^{-5}		
	2	5.87×10^{-6}	5.08×10^{-6}	5.88×10^{-6}	2.92×10^{-6}	2.53×10^{-6}	2.93×10^{-6}		
	3	1.36×10^{-8}	9.10×10^{-7}	1.38×10^{-8}	6.82×10^{-9}	4.53×10^{-7}	6.88×10^{-9}		
$20 \times 20 \times 20$	0	2.62	2.62	2.62	1.00	1.00	1.00	0.9991	1.0
	1	1.66×10^{-4}	1.45×10^{-4}	1.67×10^{-4}	6.34×10^{-5}	5.56×10^{-5}	6.38×10^{-5}		
	2	8.74×10^{-6}	6.61×10^{-6}	8.72×10^{-6}	3.33×10^{-6}	2.52×10^{-6}	3.32×10^{-6}		
	3	3.97×10^{-8}	3.64×10^{-7}	3.93×10^{-8}	1.51×10^{-8}	1.39×10^{-7}	1.50×10^{-8}		

5.6.4.1 Evaluation of LOGNet vs Mooney-Rivlin: numerical validation

The same numerical validation procedure outlined in Section 5.4 is replicated using synthetic data for both UA and BA test cases. The process of data generation for the reference Mooney-Rivlin model is maintained consistent with that of the Ogden-Storåkers model.

Characteristics of the training dataset of the Mooney-Rivlin reference model are as follows: the interquartile range for c_1 falls between 0.77 and 0.90, while that of c_3 ranges from 1.09 to 1.30. Consequently, the effective training range is defined by the IQR of c_1 and c_3, encompassing the values $\left[0.77, 1.30\right]$, displaying minimal variance compared to the training dataset of the Ogden-Storåkers strain-energy. Therefore, the range for the principal squared stretches is maintained similarly, from 0.2 to 1.6, i.e. $c^{(UA)} = c^{(BA)} \in \left[0.2, 1.6\right]$. This consistent input range facilitates investigation into both interpolation and extrapolation capabilities of the DNN-based constitutive models.

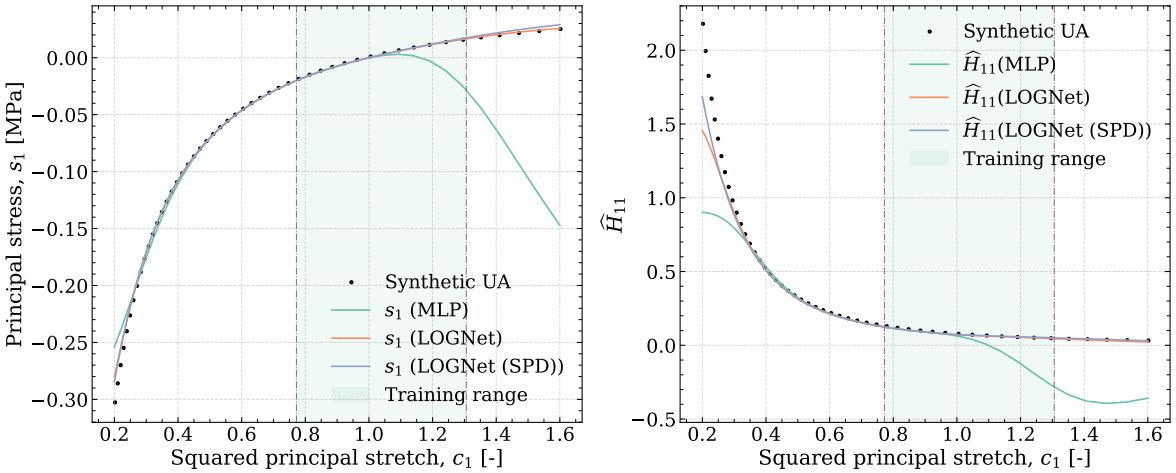
Given the similarity in the ranges of the training datasets, the values from Table 5.2 can be conveniently reused. The architecture $3-72-69-33$ is selected as the foundational structure for all three DNN-based models, including the standard DNN, LOGNet, and LOGNet (SPD). The whole process of training these three surrogate models is kept the same as described in Subsection 5.3.2.

Fig. 5.11 demonstrates the comparison between the Mooney-Rivlin synthetic dataset (reference) and the predictions generated by three DNN-based hyperelastic material models: the baseline MLP, LOGNet, and LOGNet (SPD). Both uniaxial (UA) and biaxial (BA) cases are covered. This evaluative process includes both uniaxial and biaxial test cases. Each model's prediction is visualised by distinct colored lines, where the baseline MLP model is denoted in green, LOGNet in orange, and LOGNet (SPD) in blue. Synthetic data points are visually marked with circle indicators, and the training range is demarcated by the shaded green area. Specifically, the UA test case is depicted in Figure 5.11a, while the BA scenario is portrayed in Figure 5.11b.

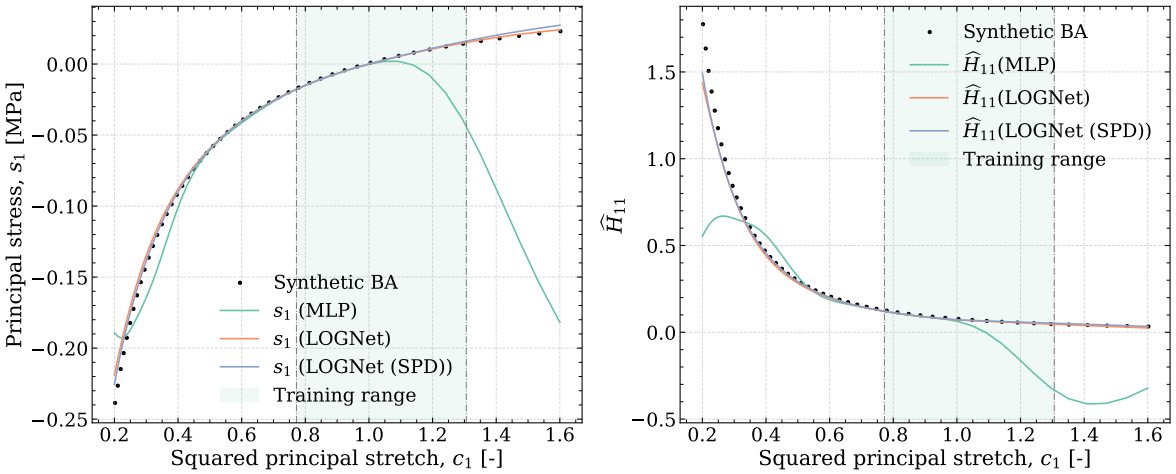
The left-hand plots in the figures depict the relationship between the reference principal value, s_1, of the 2^{nd} Piola-Kirchhoff stresses and the principal squared stretches (c_1) along the loading direction (on the x-axis). Both LOGNet-based material models demonstrate an impressive alignment with the synthetic data within the designated training region (highlighted in green), evident in both the UA and BA cases. In contrast, the MLP model shows worse performance in the extension regime ($c_1 > 1.1$).

Regarding the extrapolation capability, it is evident that both LOGNet and LOGNet (SPD) models outperform the baseline MLP model in the compression region, $c_1 < 0.5$. In this regime, the two LOGNet-based models closely match the ground truth data, demonstrating excellent accuracy, and capturing asymptotic and monotonic behaviours. Conversely, in the BA test case, the MLP model struggles to replicate the asymptotic trend.

Figure 5.11 Performance comparison between the Ogden-Storåkers model and the predictions of three DNN-based constitutive non-linear hyperelastic material laws: the baseline MLP, LOGNet, and LOGNet (SPD). The left-hand plots depict the 2nd Piola-Kirchhoff stress (s_1) and the component $H_{11} = \frac{\partial \hat{s}1}{\partial c_1}$ is on the right-hand plot, whereas the squared principal stretch (c_1) is in x-axis for both uniaxial (UA) and biaxial (BA) cases. The green shaded area represents the training range. The overall R^2 scores for predicted principal stresses are $R^2_{MLP} = 0.812$, $R^2_{\text{LOGNet}} = 0.996$ and $R^2_{\text{LOGNet}(SPD)} = 0.997$.



(a) Performance comparison between Mooney-Rivlin model and DNN-based models on the virtual uniaxial test.



(b) Performance comparison between Mooney-Rivlin model and DNN-based models on the virtual equibiaxial test.

In the tension regime, $c_1 \leq 1.0$, LOGNet-based constitutive models maintain relative accuracy in capturing the upward trend. They continue to closely follow the reference data, emphasizing LOGNet's dominance over its counterparts in this region. On the other hand, the predictions of the MLP model appear pretty "arbitrary", and fail to sustain the upward trends. Overall, the R^2 scores for the predicted principal stress of three models are $R^2_{MLP} = 0.812$, $R^2_{\text{LOGNet}} = 0.996$ and $R^2_{\text{LOGNet}(SPD)} = 0.997$.

The right-hand plots illustrate the comparison between the component $H_{11} = \frac{\partial \hat{s}_1}{\partial c_1}$ vs squared principal stretch, c_1. In both tests, it's evident that the LOGNet models maintain strong performance within the training regime, with LOGNet (SPD) exhibiting a slightly superior performance.

In the compression regime, both LOGNet-based models, again, both LOGNet-based models once again showcase exceptional extrapolation accuracy, outperforming the MLP model by a significant margin. Both LOGNet capture well the exponentially increasing trend of H_{11} in this region, again confirming their ability to satisfy the growth requirement.

In the tension regime, the MLP model fails to sustain the monotonic requirement, evident from the negative values of its H_{11} components. On the other hand, both LOGNet's models demonstrate comparable and excellent extrapolation performances, closely aligning with the data. The positive values of H_{11} across the graphs indicate fulfillment of the monotonic requirement. Additionally, the stresses are zero $s_1 = 0$ at $c_1 = 1$ confirming the growth and normal requirements of hyperelasticity as discussed in Sect. 2.4.9.

The overall R^2 scores of the predicted H_{11} of three models are $R^2_{MLP} = 0.614$, $R^2_{\text{LOGNet}} = 0.921$ and $R^2_{\text{LOGNet}(SPD)} = 0.958$. In summary, both LOGNet-based material models consistently demonstrate exceptional performance within the testing regime, displaying remarkable accuracy across both uniaxial and biaxial cases.

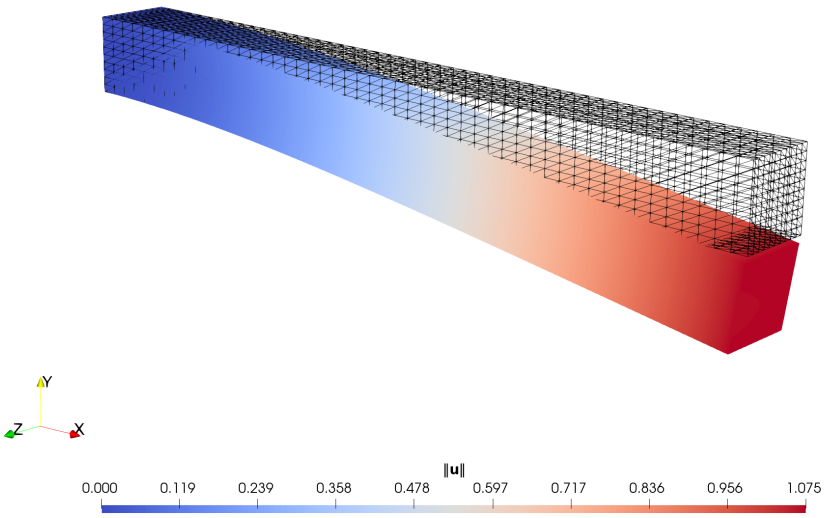
5.6.4.2 Cantilever Beam benchmarks

Evaluation of LOGNet **(SPD) vs Mooney-Rivlin**: given the slightly better performance of the LOGNet (SPD), it is employed for conducting the benchmark of the Cantilever beam under uniformly distributed load. Figure 5.12 compares the numerical results between the LOGNet (SPD) model and Mooney-Rivlin as the reference model. The magnitude of the 3D view of the predicted displacement field, $\|\hat{\boldsymbol{u}}\|$, and the $X - Z$ plane at $x = 0.5$ are shown in figures 5.12a and 5.12b, respectively. It is revealed that numerical simulation achieved good results without any nonphysical behaviour over the whole domain. The performance evaluation for displacement and stress fields of the LOGNet (SPD) model is showcased in Fig. 5.13. This figure illustrates both the norm of the difference between the displacement field, $\|\boldsymbol{u} - \hat{\boldsymbol{u}}\|$, and the principal stress vectors, $\|\boldsymbol{s} - \hat{\boldsymbol{s}}\|$.

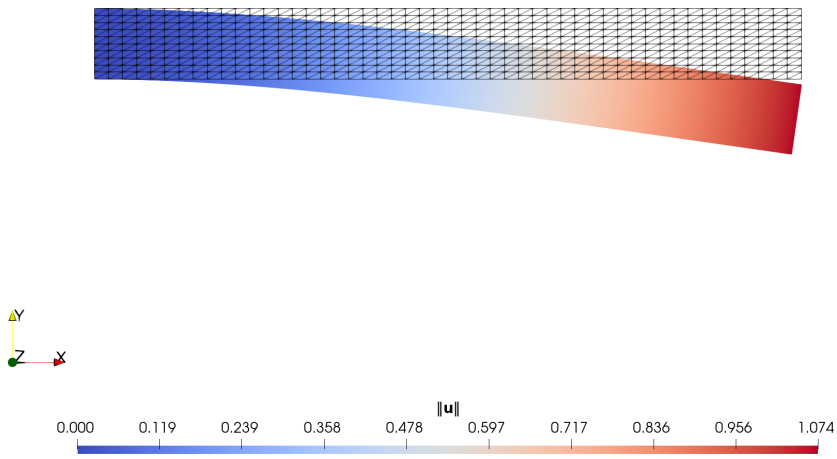
Figures 5.13a and 5.13b illustrate similar displacement and stress fields between the predictions generated by the LOGNet (SPD) and those of the Mooney-Rivlin model. The maximum values for the norm of the difference, $\max(\|\boldsymbol{u} - \hat{\boldsymbol{u}}\|) = 4.7 \times 10^{-3}$ and $\max(\|\boldsymbol{s} - \hat{\boldsymbol{s}}\|) = 1.2 \times 10^{-5}$, confirm a high level of agreement between the two models. Notably, the highest values of the principal stress difference are localized near the fixed support.

Comparative Assessment: LOGNet **(SPD) vs. Ogden-Storåkers**: in this scenario, the distributed load is changed to $q = 20\,\text{Pa}$. The performance of the LOGNet (SPD) model maintains its excellence even when applied to the Ogden-Storåkers dataset. The 3D

Figure 5.12 Cantilever beam - uniformly distributed load, the displacement field. Performance of the LOGNet (SPD) material law (trained on Mooney-Rivlin data) for a polyethylene foam cantilever beam subjected to an uniformly distributed load, $q = 10\,\text{Pa}$.

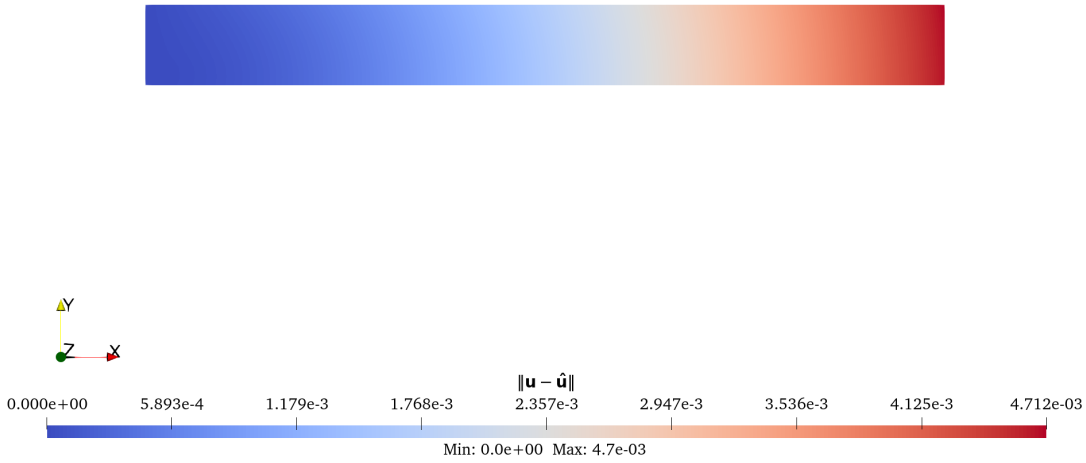


(a) Magnitude of displacement field: 3D view.

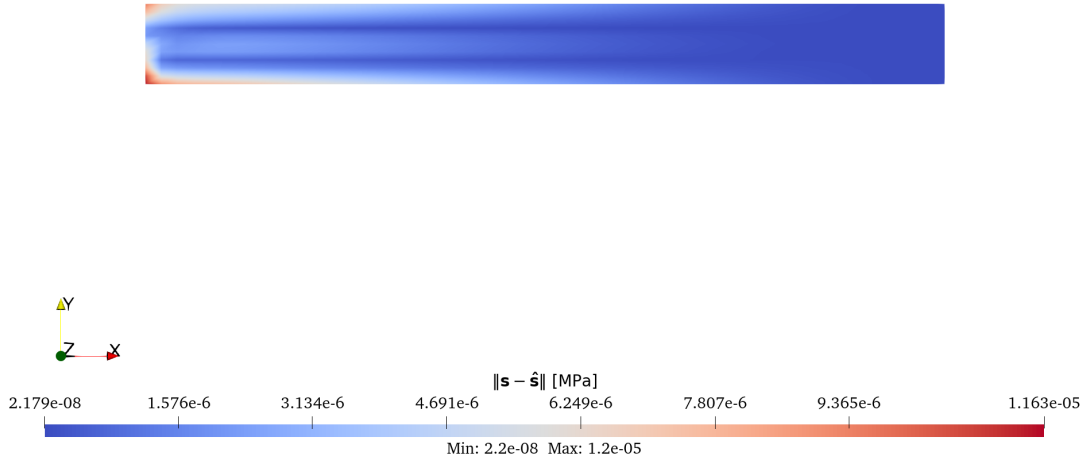


(b) Magnitude of displacement field:3D view.

Figure 5.13 Cantilever beam - uniformly distributed load, the norm of difference of the displacement and stress fields. Performance of the LOGNet (SPD) material law (trained on Mooney-Rivlin data) for a polyethylene foam cantilever beam subjected to an uniformly distributed load, $q = 10\,\mathrm{Pa}$.



(a) Magnitude of the difference of the displacement field: $X - Y$ plane at $z = 0.5$.



(b) Magnitude of the difference of the principal stress: $X - Y$ plane at $z = 0.5$.

representation of the displacement field of the cantilever beam benchmark is similar to that of the Mooney-Rivlin case and is therefore omitted. The norm of the difference between the two fields is illustrated in Fig. 5.14. For the case of training on the Ogden-Storåkers dataset, Figures 5.14a and 5.14b depict nearly identical displacement and stress fields between the DNN-based predictions and that of the reference model. The maximum values of the norm of the difference, $\max(\|\boldsymbol{u} - \hat{\boldsymbol{u}}\|) = 4.8 \times 10^{-3}$ and $\max(\|\boldsymbol{s} - \hat{\boldsymbol{s}}\|) = 1.1 \times 10^{-4}$, confirms a high level of agreement between the two models.

The results, again, demonstrate the great capabilities of the LOGNet (SPD) over the complex benchmark, underscoring its robustness and effectiveness in capturing the material constitutive behaviour from data.

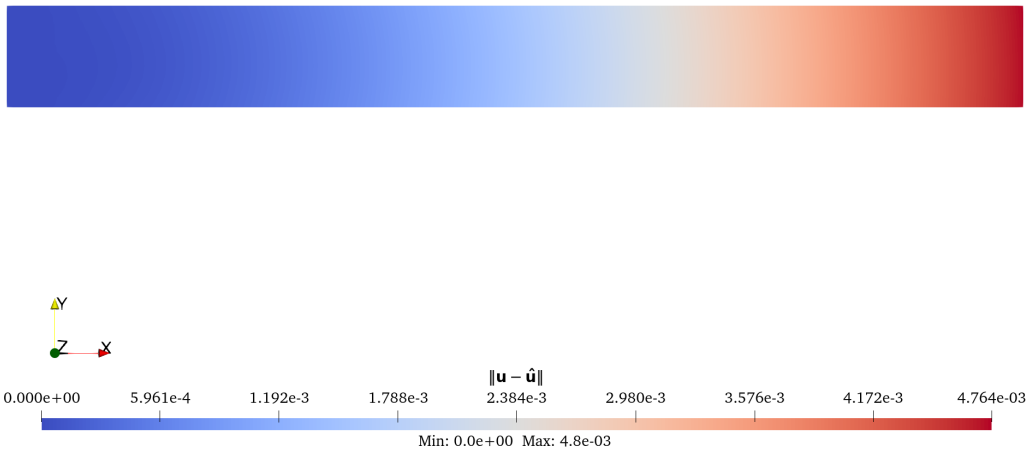
5.6.5 The influence of extreme values

In some cases of a testing scenario, the inputs injected into the DNN-based surrogate models are beyond the training range, leading to inaccurate predictions, and the Newton-Raphson procedure may diverge. Consider an extremely large deformed benchmark simulation, the "Tw-UC" test. Simultaneously applying unconstrained uniaxial compression at 20% and twisting around the centered axis at an angle 20° on the top face, while the bottom face is fixed. The 3D closed-cell polyethylene foam unit cube with the characteristic is exactly the same as the previous section. The numerical simulation is also conducted in the same setting. Fig. 5.15 compares the behaviour of NR's iterations between both the DNN-based material laws: the baseline MLP and LOGNet (SPD) versus the reference Ogden-Storåkers model. The grey dotted line represents the reference model, while the green dashed line indicates the MLP and the blue solid line depicts the LOGNet (SPD). The left and right figures present two results of the NR process terminated at $\texttt{atol} = 1 \times 10^{-6}$ and $\texttt{atol} = 1 \times 10^{-12}$, respectively.

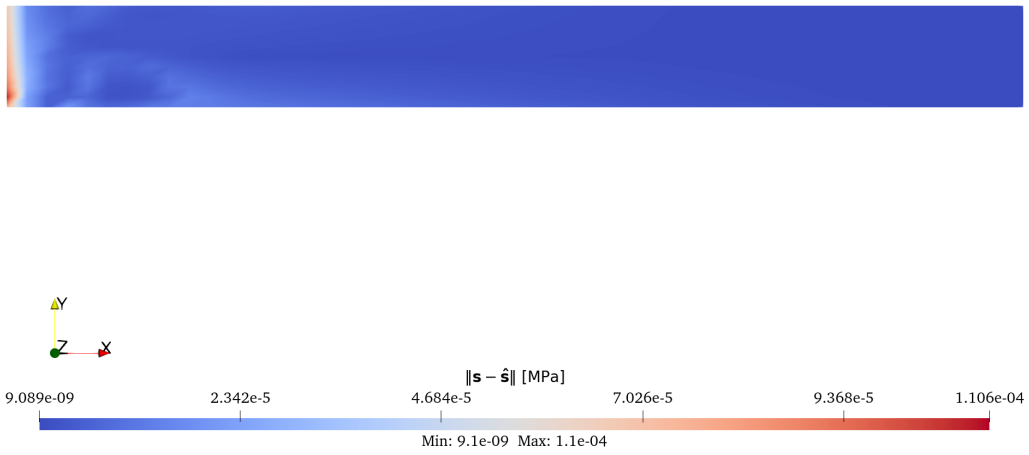
Overall, both the DNN-based models cannot approach the performance of the Ogden-Storåkers model. They perform well in the first three iterations and then struggle to pass the convergence criteria. The LOGNet (SPD) based process succeeds in seeking the final solution and passes the criteria at 4^{th} step for the $\texttt{atol} = 1 \times 10^{-6}$ case, 7^{th} step for the case of $\texttt{atol} = 1 \times 10^{-12}$. The Ogden-Storåkers model finished at 4^{th} and 5^{th} step for the same criteria. The computations of the MLP model struggle to seek better solutions and keep oscillating around $\texttt{rtol} = 1.0079 \times 10^{-6}$ and $\texttt{atol} = 4.041 \times 10^{-6}$ until the procedure is forced to stop at the 10^{th} iteration. This behaviour will become more severe in more difficult test cases and may lead to divergence of NR.

The root of this problem is the values range of the input vector injected into the DNN-based models during solving the BVP using NR. Figure 5.16 shows the components of the squared stretch vector at each iteration of the NR procedure for the "Tw-UC" test case. The source of the success of the LOGNet (SPD) model lies in its ability to extrapolate outside its training

Figure 5.14 Cantilever beam - uniformly distributed load, the norm of difference of the displacement and stress fields. Performance of the LOGNet (SPD) material law (trained on Ogden-Storåkers data) for a polyethylene foam cantilever beam subjected to an uniformly distributed load, $q = 20\,\mathrm{Pa}$.

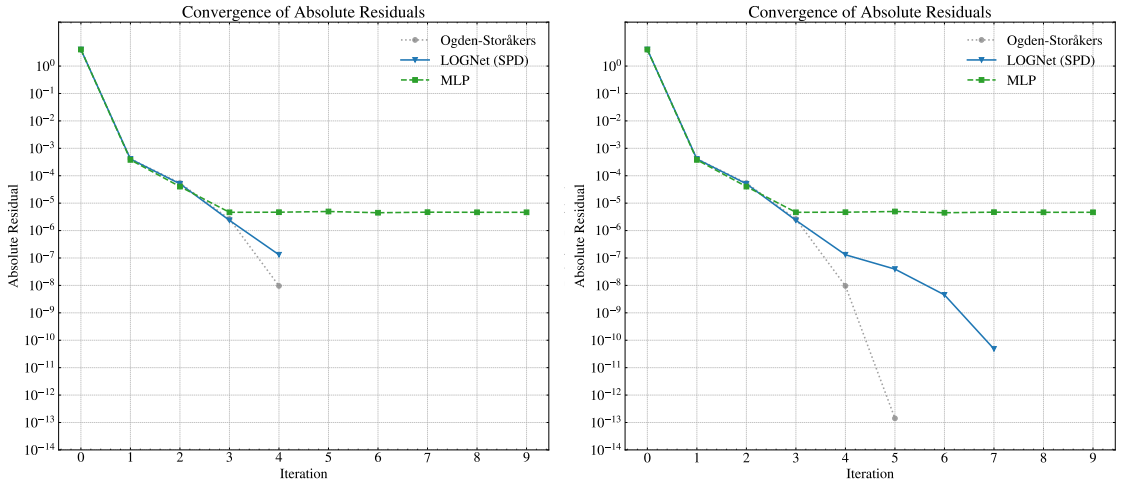


(a) Magnitude of the difference of the displacement field: $X - Y$ plane at $z = 0.5$.



(b) Magnitude of the difference of the principal stress: $X - Y$ plane at $z = 0.5$.

Figure 5.15 Benchmark case: "Tw-UC". Comparison of the convergence behavior of the Newton-Raphson iterative process is conducted with three models, MLP (green dashed line), LOGNet (SPD) model (blue solid line), and the reference Ogden-Storåkers model (grey dashed dot line). Two termination criteria in terms of the absolute residual, $\texttt{atol} = 1 \times 10^{-6}$ (left) and $\texttt{atol} = 1 \times 10^{-12}$ (right) are presented.



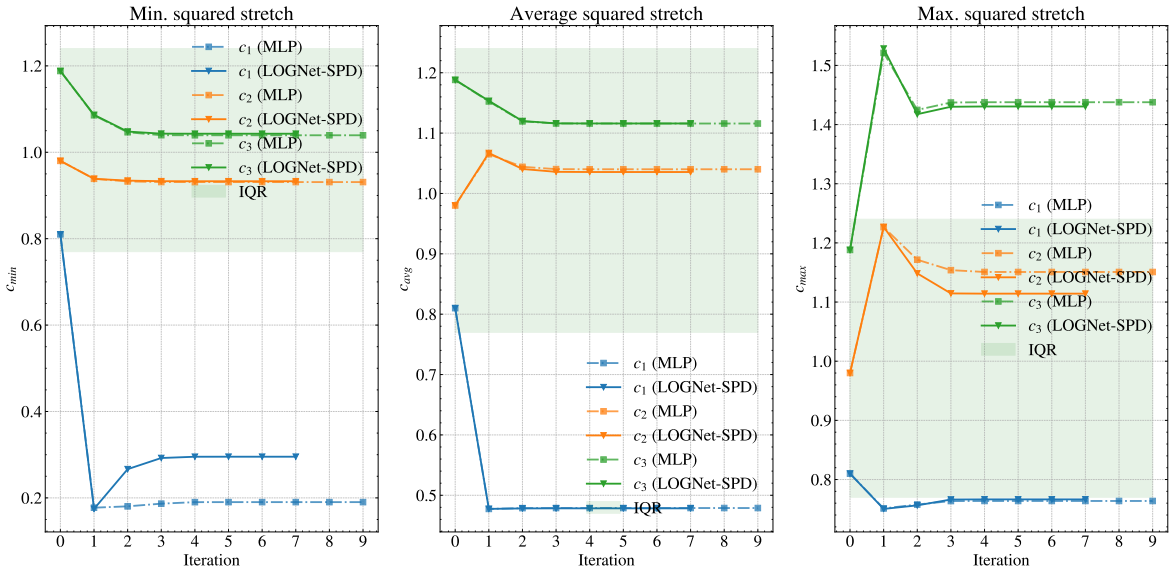
(a) Termination criteria: $\texttt{atol} = 1 \times 10^{-6}$ **(b)** Termination criteria: $\texttt{atol} = 1 \times 10^{-12}$

range. The values of the MLP are presented using solid lines whereas dashed lines for LOGNet (SPD).

It is evident that the extreme values of the inputs are not inside the training range. The values of the first component, c_1, for the whole domain and during the whole iteration steps, lie beyond the training range. The third component c_3 has only the maximum values exhibiting the same behaviour as c_1. In contrast, the component c_2 remains inside the range for the whole process.

Compression is the dominant property in this test, hence, it challenges the extrapolation abilities of the surrogate material models in this region. Both DNN-based models handle the tension regime very well, starting from the initial guess, $c_0^{max} = 1.18$, both models overshoot at the first iteration, then convert to the correct solution near $c_3^{max} = 1.43$. However, compression handling is not the strength of the baseline MLP model (which has been investigated in section 5.5). The MLP manages most of the Gauss point in the domain, the average value of $c_1^{avg} = 0.478$ and the maximum value of $c_1^{max} = 1.43$. It is the minimum value that breaks the prediction, meaning that computations at only a few Gauss points are around this value. The left graph gives evidence that MLP cannot produce correct predictions at the iteration 1, leading to wrongly updating new values for c_1 at the next iteration at some Gauss points. Hence, the NR's iteration using the MLP model updated the value of c_1 oscillates about 0.19, whereas LOGNet (SPD), with its extrapolation power, arrives near the correct value at 0.29 within 7 steps.

Figure 5.16 Benchmark case: "Tw-UC". The range of values of the squared stretch vector $\boldsymbol{c}$ during the Newton iteration. Comparison of the progression between MLP and LOGNet (SPD) models versus the IQR $IQR = [0.77, 1.24]$.



5.6.6 The solution of extreme values

The influence of the extreme values may also apply to the LOGNet (SPD) model as in the case of the MLP in the previous subsection if a "difficult" and untrained test case appeared. Obviously, since DNN-based surrogate material models are supervised learning models, they are not supposed to deploy on untrained problems. Nevertheless, this is the drawback of these types of models.

The solution for these scenarios is to employ a fallback predictor, which is an analytical function that can prevent the NR's algorithm diverted, and guide the updated solutions back to the training regime. One other property that can be exploited in this case is the consistency conditions between isotropic hyperelasticity and isotropic linear elasticity. The response function, in the infinitesimal strain, should have the form as follows:

$$\boldsymbol{\sigma} = \bar{\lambda} \operatorname{Tr} \boldsymbol{\varepsilon} \boldsymbol{I} + 2\mu\boldsymbol{\varepsilon}\,, \tag{5.12}$$

where $\bar{\lambda}, \mu$ are the Lamé constants. The Lamé constants can be determined using the following conditions for a stretch-based strain energy $\Psi(\lambda_1, \lambda_2, \lambda_3)$ (R. W. Ogden, 1997, chapter 7):

$$\frac{\partial^2 \Psi}{\partial \lambda_i \partial \lambda_j}(1,1,1) = \bar{\lambda} + 2\mu\delta_{ij}\,. \tag{5.13}$$

Hence, the Lamé constants are derived as follows:

$$\bar{\lambda} = 2\widehat{H}_{12}(1,1,1)\,; \tag{5.14}$$

$$\mu = \widehat{H}_{11}(1,1,1) - \widehat{H}_{12}(1,1,1)\,. \tag{5.15}$$

Here $\widehat{H}_{ij}$ is the component of the Jacobian matrix of the DNN at $\boldsymbol{c} = \Big[1,1,1\Big]$.

5.7 Conclusion

This chapter presents the implementation of the novel approach for incorporating constitutive requirements into the DNN-based material model, as proposed in Chapter 4. After detailing the data acquisition technique, the training process for the LOGNet model is presented. This encompasses a comprehensive account of employing hyperparameter optimization frameworks to meticulously fine-tune the architecture of the LOGNet model. Notably, this architecture serves as the foundation for both the baseline MLP and the LOGNet (SPD) variant. Furthermore, the meticulous selection of optimal weight loss algorithms for the LOGNet (SPD) model is explored.

Afterward, the performance evaluation of all three models takes place, along with rigorous validation against synthetic and experimental data from the UA and BA tests. It is evident that the LOGNet (SPD) model demonstrates robust performance across all tests, exhibiting its competence not only within the training domain but also in terms of extrapolation capabilities. Given that the performance of the vanilla LOGNet is slightly less than the LOGNet (SPD), the latter is chosen for further investigation.

The implementation of both models, the baseline MLP and LOGNet (SPD), within a `FEniCSx` framework, is also carried out, followed by a comparative analysis of their respective numerical solutions. Notably, the superiority of the LOGNet (SPD) model remains consistent across all test cases, outperforming the baseline MLP.

However, it is essential to acknowledge the drawbacks that DNN-based surrogate material models, being supervised learning models, exhibit limitations when applied to untrained problems. This inherent constraint underscores an important aspect of these models. Through the strategic integration of *prior* physical constraints into the architecture and the training process, the LOGNet (SPD) model showcases reduced sensitivity when confronted with scenarios involving extreme value presentations.

Chapter 6

Conclusion and perspectives

The present dissertation has explored the application of *machine learning approach in computational mechanics*. Specifically, it focuses on presenting the constitutive response function via Deep Neural network-based surrogate models to effectively model non-linear compressible elastic materials.

To ensure the creation of consistent and physically meaningful material laws, six fundamental physical requirements for non-linear isotropic elastic materials are investigated and successfully incorporated into the DNN-based models. Addressing challenging tasks involves formulating the material in principal space. Additionally, it requires altering both the architecture of the DNNs and their training process to incorporate these physical requirements effectively. The presented methodology is validated with both numerical simulations and experimental data.

Before presenting in detail the methodology and results, the dissertation, as detailed in Chapter 1, highlighted the research direction for modelling constitutive governing equations for the non-linear elastic materials. In particular, three potential directions have been summarized, namely model-free computing approach (Conti, S. Müller, and Ortiz, 2018; T. Kirchdoerfer and Ortiz, 2016), machine learning surrogate modelling method (Hoerig, Ghaboussi, and Insana, 2017; Wehenkel and Louppe, 2021), and symbolic regression method (Schmidt and Lipson, 2009).

Among these, the model-free approach refrains from presenting explicit models or hypotheses for the interpretation of outcomes. Conversely, symbolic regression presents much better interpretability but tends to yield lengthy expressions, posing considerable challenges, notably within high-dimensional problem spaces. Plus, the integration of established prior knowledge into symbolic regression remains an unexplored topic in current research endeavors. Both of the methods require an extensive amount of data.

In contrast, the machine learning-based approach offers the potential for indirect interpretation and the incorporation of physical prerequisites. As such, it becomes the core focus of this thesis,

revolving around the refinement of a modified Deep Neural Network capable of extracting insights from experimental data while adhering to prior constitutive requisites.

Six fundamental requirements for non-linear elastic materials are presented in Chapter 2 to which a surrogate model should respect. They are the principle of material frame indifference, material symmetry transformation, undistorted and stress-free configurations, growth conditions in large strains, and Hill's constitutive inequalities. Satisfying the material frame indifference and material symmetry transformation (in the case of isotropic materials) criteria can be accomplished through various methods. One straightforward approach involves employing the principal strains and principal stresses as input/output pairs for DNNs-based models. For that, Chapter 3 provides background for the formulation of boundary value problems of non-linear isotropic elastic materials in principal space. Hyperelastic materials are chosen as reference models for illustrative purposes.

A proposed procedure for parameter identification for hyperelastic models simultaneously from multiple experimental datasets. Furthermore, numerical simulations are conducted to verify the simplified formulation of boundary value problems in principal space. The results are validated and achieve high accuracy with the experimental data. This formulation is, then, utilized for testing DNNs-based surrogate material laws in subsequent chapters.

Chapter 4 introduces fundamentals on deep neural networks, their components, and related theories. The process of developing new architecture and training process of Deep neural networks such that they satisfy constitutive conditions are presented in this chapter.

In Chapter 2, boundary value problems of non-linear isotropic elasticity with modified DNN-based material laws are demonstrated and analyzed. Several challenges arise when solving nonlinear boundary-value problems using the Newton-Raphson method, such as slow convergence or divergence and inaccuracy of the tangent matrix, etc. During the Newton-Raphson intermediate steps, as the physical equilibrium has not yet been reached, the range of inputs provided for the DNN-based model suffers from substantial fluctuations. This may cause the DNN model to extrapolate beyond its training range, potentially resulting in unreliable predictions and convergence issues for Newton-Raphson iterations. However, a systematic and detailed analysis of the root causes is not reported. Therefore, numerical stability is also a main focus of the chapter, with a deeper analysis and investigation of the root causes of these unfavorable behaviors being presented. Examples of the out-of-training-range problem leading to instability of the Newton-Raphson iterative process are shown, illustrating that the proposed LOGNet model is robust in dealing with these issues.

The main contributions are highlighted as follows:

- Parameters fitting of hyperelastic models using constrained optimization method.
- Simplifying the formulation of the virtual work expression, which now only necessitates principal strains and stresses instead of the full tensorial forms.

- Introduction of novel activation function named "Logarithmic Linear Unit" LOGLU.
- Proposing the LOGNet architecture, altering the MLP by integrating LOGLU before the first hidden layer to fulfill growth and normalization conditions.
- Applying Heuristic Regularization to encourage separately monotone and positive definiteness conditions.
- Investigation into the underlying causes of numerical instability of the NR process when employing DNN-based material laws.

This work revolves around addressing three key challenges: the simplification of isotropic non-linear elastic materials in principal space, the integration of the constitutive requirements into the deep neural network, and the analysis of numerical instability when solving the BVP using DNN-based surrogate models.

For future works, given the substantial data demand of supervised machine learning models and the scarcity of real-world experimental data, future research directions should emphasize the treatment and augmentation of real experimental data. The method outlined in Weber, Geiger, and Wagner (2021, Appendix C) provides a means to transform and enrich basic modes of experimental data. For instance, rare experimental data, such as brain and fat tissue data as presented by (Mihai et al., 2017), can be enriched using this method. Following that, anisotropic non-linear elastic materials can be assumed for brain and fat tissues. However, modifying the type of constitutive requirements, outlined in this work, is necessary for such cases. Specifically, Hill's inequalities may not hold for anisotropic materials, hence, incorporating other anisotropic requirements becomes essential.

Subsequently, the core architecture of LOGNet proposed in this work could still hold relevance if the input/output pairs represent the stretch/stress pairs along the fiber direction. Nevertheless, the heuristic monotonic regularization and positive definite constraints mentioned in Section 4.3 should be replaced accordingly to adapt to the new anisotropic requirements.

Currently, the workflow involves three primary steps:

1. data collection performed in the `FEniCSx` environment,
2. the training process carried out using `PyTorch`,
3. the conversion of the trained model back to `FEniCSx`.

An alternative unified approach, proposed by Xue et al. (2023), suggests streamlining these steps using a single library like `JAX`. This unified approach could potentially simplify the process.

Additionally, using large DNN-based constitutive models introduce difficulties for the FEM packages that do not support vectorised operators. This limitation demands significant computation resources, and even more severely when computing the derivatives of the DNNs

via automatic differentiation. To address these challenges, network compression techniques such as pruning and quantization could be employed. Compressing the network frequently results in minimal accuracy loss and, in certain situations, might even enhance accuracy as discussed by (Liang et al., 2021).

Expanding the existing framework to address a multi-scale computational problem using the FE^2 homogenization algorithm. The FE^2 homogenisation algorithm for multi-scale modeling involves iterative interactions between the macroscale and the microscale, utilizing a representative volume element. The process continues until convergence is attained during each step of the macroscale loading or prescribed displacement. The microscale problem is addressed in real-time using finite elements, making the algorithm computationally intensive. By employing the current framework, inputs from the macroscale are fed into the LOGNet model. Subsequently, the outputs of the LOGNet model are sent back to the macroscale. Therefore, the FE processes at the microscale are illuminated, which will significantly accelerate the FE^2 procedure.

In the long term, the potential of symbolic regression trained on an extensive dataset of formulas, as demonstrated by Vastl et al. (2022), offers a promising avenue for discovering constitutive material laws. Nonetheless, with the introduction of every new method, the challenge persists in finding effective ways to incorporate *prior* knowledge and maintain the physical consistency of the new technique.

References

Agrawal, A. and A. Choudhary (2016). "Perspective: Materials Informatics and Big Data: Realization of the "Fourth Paradigm" of Science in Materials Science". In: *APL Materials* 4.5, p. 053208.

Akhtar, N. et al. (2021). *Advances in adversarial attacks and defenses in computer vision: A survey.*

Akiba, T. et al. (2019). "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *arXiv:1907.10902 [cs, stat].*

Alnaes, M. S. et al. (2014). "Unified Form Language: A domain-specific language for weak formulations of partial differential equations". In: *ACM Transactions on Mathematical Software* 40.

Anssari-Benam, A. and C. O. Horgan (2022). "New constitutive models for the finite deformation of isotropic compressible elastomers". In: *Mechanics of Materials* 172, p. 104403.

Antman, S. S. (1983). "Regular and singular problems for large elastic deformations of tubes, wedges, and cylinders". In: *Archive for Rational Mechanics and Analysis* 83.1, pp. 1–52.

Ayensa-Jiménez, J. et al. (2018). "A New Reliability-Based Data-Driven Approach for Noisy Experimental Data with Physical Constraints". In: *Computer Methods in Applied Mechanics and Engineering* 328, pp. 752–774.

Baker, M. and J. L. Ericksen (1954). "Inequalities restricting the form of the stress-deformation relations for isotropic elastic solids and Reiner-Rivlin fluids". In: *Journal of the Washington Academy of Sciences* 44.2. Publisher: Washington Academy of Sciences, pp. 33–35.

Ball, J. M. (1976). "Convexity conditions and existence theorems in nonlinear elasticity". In: *Archive for Rational Mechanics and Analysis* 63.4, pp. 337–403.

Bathe, K.-J. (2006). *Finite Element Procedures.* New Jersey: Prentice-Hall, Inc. 1056 pp.

Beatty, M. F. (1987). "Topics in Finite Elasticity: Hyperelasticity of Rubber, Elastomers, and Biological Tissues—With Examples". In: *Applied Mechanics Reviews* 40.12, pp. 1699–1734.

Becker, G. F. (1893). "The finite elastic stress-strain function". In: *American Journal of Science* s3-46.275. Publisher: American Journal of Science Section: Structural geology, pp. 337–356.

Beex, L. A. A. (2019). "Fusing the Seth–Hill strain tensors to fit compressible elastic material responses in the nonlinear regime". In: *International Journal of Mechanical Sciences* 163, p. 105072.

Bell, G., T. Hey, and A. Szalay (2009). "Beyond the Data Deluge". en. In: *Science* 323.5919, pp. 1297–1298.

Bergstra, J. and Y. Bengio (2012). "Random Search for Hyper-Parameter Optimization". In: *Journal of Machine Learning Research* 13.10, pp. 281–305.

Bergstra, J. et al. (2011). "Algorithms for Hyper-Parameter Optimization". In: *Advances in Neural Information Processing Systems*. Vol. 24. Curran Associates, Inc.

Bertram, A. (2021). *Elasticity and Plasticity of Large Deformations: Including Gradient Materials*. Springer Nature. 423 pp.

Bhat, H. S. (2019). "Learning and Interpreting Potentials for Classical Hamiltonian Systems". In: *arXiv:1907.11806 [physics, stat]*.

Biot, M. A. (1965). *Mechanics of incremental deformations*. New York: John Wiley & Sons, Inc. 516 pp.

Bischl, B. et al. (2023). "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges". In: *WIREs Data Mining and Knowledge Discovery* 13.2. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1484, e1484.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Information science and statistics. New York: Springer.

Blatz, P. J. and W. L. Ko (1962). "Application of Finite Elastic Theory to the Deformation of Rubbery Materials". In: *Transactions of The Society of Rheology* 6.1, pp. 223–252.

Blondel, M. et al. (2022). *Efficient and Modular Implicit Differentiation*.

Bongard, J. and H. Lipson (2007). "Automated Reverse Engineering of Nonlinear Dynamical Systems". en. In: *Proceedings of the National Academy of Sciences* 104.24, pp. 9943–9948.

Borne, K. D. (2010). "Astroinformatics: Data-Oriented Astronomy Research and Education". en. In: *Earth Science Informatics* 3.1, pp. 5–17.

Bottou, L., F. E. Curtis, and J. Nocedal (2018). *Optimization Methods for Large-Scale Machine Learning*.

Bradbury, J. et al. (2018). *JAX: composable transformations of Python+NumPy programs*. Version 0.2.5.

Breiman, L. (2017). *Classification and Regression Trees*. en. Routledge.

Brown, T. B. et al. (2020). *Language Models are Few-Shot Learners*.

Bruhns, O. T., H. Xiao, and A. Meyers (2001). "Constitutive inequalities for an isotropic elastic strain-energy function based on Hencky's logarithmic strain tensor". In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 457.2013, pp. 2207–2226.

Brunton, S. L., J. L. Proctor, and J. N. Kutz (2016). "Discovering Governing Equations from Data by Sparse Identification of Nonlinear Dynamical Systems". en. In: *Proceedings of the National Academy of Sciences* 113.15, pp. 3932–3937.

Carroll, M. (2009). "Must Elastic Materials be Hyperelastic?" In: *Mathematics and Mechanics of Solids* 14.4. Publisher: SAGE Publications Ltd STM, pp. 369–376.

Chakrabarty, J. (2012). *Theory of plasticity*. Elsevier.

Champion, K. et al. (2019). "Data-Driven Discovery of Coordinates and Governing Equations". In: *arXiv:1904.02107 [stat]*.

Chan, K. H. V. (2017). "Some Artificial Neural Network Rule Extraction Algorithms for Nonlinear Regression Problems and Their Application". en. PhD thesis. Faculty of Graduate Studies and Research, University of Regina.

Chinesta, F. et al. (2017). "Data-Driven Computational Plasticity". In: *Procedia Engineering*. International Conference on the Technology of Plasticity, ICTP 2017, 17-22 September 2017, Cambridge, United Kingdom 207, pp. 209–214.

Ciarlet, P. G., ed. (1988). *Mathematical elasticity: Three-dimensional elasticity*. 1st. Vol. 20. Elsevier. 456 pp.

Clevert, D.-A., T. Unterthiner, and S. Hochreiter (2016). *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. version: 5.

Cole, G. W. and S. A. Williamson (2019). *Avoiding Resentment Via Monotonic Fairness*.

Connolly, S. J., D. MacKenzie, and Y. Gorash (2019). "Isotropic hyperelasticity in principal stretches: explicit elasticity tensors and numerical implementation". In: *Computational Mechanics* 64.5, pp. 1273–1288.

Conti, S., S. Müller, and M. Ortiz (2018). "Data-Driven Problems in Elasticity". en. In: *Archive for Rational Mechanics and Analysis* 229.1, pp. 79–123.

Curnier, A. and L. Rakotomanana (1991). "Generalized Strain and Stress Measures: Critical Survey and New Results". In: *Engineering Transactions* 39.3-4. Number: 3-4, pp. 461–538.

Cybenko, G. (1989). "Approximation by superpositions of a sigmoidal function". In: *Mathematics of Control, Signals and Systems* 2.4, pp. 303–314.

Daniels, H. and M. Velikova (2010). "Monotone and Partially Monotone Neural Networks". In: *IEEE Transactions on Neural Networks* 21.6. Conference Name: IEEE Transactions on Neural Networks, pp. 906–917.

Devlin, J. et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*.

Dhar, V. (2013). "Data Science and Prediction". In: *Commun. ACM* 56.12, pp. 64–73.

Domhan, T., J. T. Springenberg, and F. Hutter (2015). "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves". In: *Proceedings of the 24th International Conference on Artificial Intelligence*. IJCAI'15. Buenos Aires, Argentina: AAAI Press, pp. 3460–3468.

Draxl, C. and M. Scheffler (2019). "The NOMAD Laboratory: From Data Sharing to Artificial Intelligence". en. In: *Journal of Physics: Materials* 2.3, p. 036001.

Duchi, J., E. Hazan, and Y. Singer (2011). "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In: *The Journal of Machine Learning Research* 12 (null), pp. 2121–2159.

Dugas, C. et al. (2000). "Incorporating Second-Order Functional Knowledge for Better Option Pricing". In: *Advances in Neural Information Processing Systems*. Vol. 13. MIT Press.

Eggersmann, R. et al. (2019). "Model-Free Data-Driven Inelasticity". In: *Computer Methods in Applied Mechanics and Engineering* 350, pp. 81–99.

Farabet, C. et al. (2013). "Learning Hierarchical Features for Scene Labeling". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1915–1929.

Farahani, K. and R. Naghdabadi (2000). "Conjugate stresses of the Seth–Hill strain tensors". In: *International Journal of Solids and Structures* 37.38, pp. 5247–5255.

Feelders, A. (2000). "Prior Knowledge in Economic Applications of Data Mining". In: *Principles of Data Mining and Knowledge Discovery*. Ed. by D. A. Zighed, J. Komorowski, and J. Żytkow. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 395–400.

Fosdick, R. and M. Šilhavý (2006). "Generalized Baker–Ericksen Inequalities". In: *Journal of Elasticity* 85.1, pp. 39–44.

Fuhg, J. N., M. Marino, and N. Bouklas (2021). "Local approximate Gaussian process regression for data-driven constitutive laws: Development and comparison with neural networks". In: *arXiv:2105.04554 [cs, stat]*.

Fukushima, K. (1969). "Visual Feature Extraction by a Multilayered Network of Analog Threshold Elements". In: *IEEE Transactions on Systems Science and Cybernetics* 5.4. Conference Name: IEEE Transactions on Systems Science and Cybernetics, pp. 322–333.

Fukushima, K. (1975). "Cognitron: A self-organizing multilayered neural network". In: *Biological Cybernetics* 20.3, pp. 121–136.

Ghaboussi, J. et al. (1998). "Autoprogressive Training of Neural Network Constitutive Models". en. In: *International Journal for Numerical Methods in Engineering* 42.1, pp. 105–126.

Ghaboussi J., Garrett J. H., and Wu X. (1991). "Knowledge-Based Modeling of Material Behavior with Neural Networks". In: *Journal of Engineering Mechanics* 117.1, pp. 132–153.

Glorot, X. and Y. Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. ISSN: 1938-7228. JMLR Workshop and Conference Proceedings, pp. 249–256.

Golovin, D. et al. (2017). "Google Vizier: A Service for Black-Box Optimization". In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and*

Data Mining. KDD '17: The 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Halifax NS Canada: ACM, pp. 1487–1495.

González, D., F. Chinesta, and E. Cueto (2019). "Thermodynamically Consistent Data-Driven Computational Mechanics". en. In: *Continuum Mechanics and Thermodynamics* 31.1, pp. 239–253.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. MIT Press. 801 pp.

Gupta, A. et al. (2019). *How to Incorporate Monotonicity in Deep Networks While Preserving Flexibility?*

Gurtin, M. E. (1981). *An Introduction to Continuum Mechanics*. New York: Academic Press. 280 pp.

Habera, M. and A. Zilian (2021). *Symbolic spectral decomposition of 3x3 matrices*.

Haj-Ali Rami et al. (2001). "Simulated Micromechanical Models Using Artificial Neural Networks". In: *Journal of Engineering Mechanics* 127.7, pp. 730–738.

Hansen, N. and A. Ostermeier (2001). "Completely Derandomized Self-Adaptation in Evolution Strategies". In: *Evolutionary Computation* 9.2, pp. 159–195.

Hashash, Y. M. A., S. Jung, and J. Ghaboussi (2004a). "Numerical Implementation of a Neural Network Based Material Model in Finite Element Analysis". en. In: *International Journal for Numerical Methods in Engineering* 59.7, pp. 989–1005.

Hashash, Y. M. A., S. Jung, and J. Ghaboussi (2004b). "Numerical implementation of a neural network based material model in finite element analysis". In: *International Journal for Numerical Methods in Engineering* 59.7, pp. 989–1005.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. en. 2nd ed. Springer Series in Statistics. New York: Springer-Verlag.

Haykin, S. S. (2009). *Neural Networks and Learning Machines*. en. 3. ed. OCLC: 857737780. New York: Pearson.

He, K. et al. (2015). *Deep Residual Learning for Image Recognition*.

Hencky, H. (1928). "Uber die Form des Elastizitatsgesetzes bei ideal elastischen Stoffen". In: *Zeit. Tech. Phys* 9, pp. 215–220.

Hesthaven, J. S. and S. Ubbiali (2018). "Non-Intrusive Reduced Order Modeling of Nonlinear Problems Using Neural Networks". In: *Journal of Computational Physics* 363, pp. 55–78.

Hey, T., S. Tansley, and K. Tolle (2009a). *Jim Gray on eScience: A Transformed Scientific Method*. en.

Hey, T., S. Tansley, and K. Tolle (2009b). "The Fourth Paradigm: Data-Intensive Scientific Discovery". In: *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research.

Higham, N. J. and V. Noferini (2016). "An algorithm to compute the polar decomposition of a 3 × 3 matrix". In: *Numerical Algorithms* 73.2, pp. 349–369.

Hill, R. (1968). "On constitutive inequalities for simple materials—I". In: *Journal of the Mechanics and Physics of Solids* 16.4, pp. 229–242.

Hill, R. (1970). "Constitutive inequalities for isotropic elastic solids under finite strain". In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 314.1519. Publisher: Royal Society, pp. 457–472.

Hill, R. (1979). "Aspects of Invariance in Solid Mechanics". In: *Advances in Applied Mechanics*. Ed. by C.-S. Yih. Vol. 18. Elsevier, pp. 1–75.

Hinton, G. et al. (2012). "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups". In: *IEEE Signal Processing Magazine* 29.6. Conference Name: IEEE Signal Processing Magazine, pp. 82–97.

Hoerig, C., J. Ghaboussi, and M. F. Insana (2019). "Data-Driven Elasticity Imaging Using Cartesian Neural Network Constitutive Models and the Autoprogressive Method". In: *IEEE Transactions on Medical Imaging* 38.5, pp. 1150–1160.

Hoerig, C., J. Ghaboussi, and M. F. Insana (2017). "An Information-Based Machine Learning Approach to Elasticity Imaging". en. In: *Biomechanics and Modeling in Mechanobiology* 16.3, pp. 805–822.

Hoerig, C., J. Ghaboussi, and M. F. Insana (2018). "Cartesian Neural Network Constitutive Models for Data-Driven Elasticity Imaging". In: *arXiv:1809.04121 [cs, stat]*.

Hoger, A. (1987). "The stress conjugate to logarithmic strain". In: *International Journal of Solids and Structures* 23.12, pp. 1645–1656.

Hogeweg, P. (31-Mar-2011). "The Roots of Bioinformatics in Theoretical Biology". en. In: *PLOS Computational Biology* 7.3, e1002021.

Holzapfel, G. A. (2000). *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. 1st edition. Chichester Weinheim: John Wiley & Sons LTD. 470 pp.

Holzapfel, G. A. (2002). "Nonlinear Solid Mechanics: A Continuum Approach for Engineering Science". en. In: *Meccanica* 37.4, pp. 489–490.

Hornik, K., M. Stinchcombe, and H. White (1989). "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5, pp. 359–366.

Hruschka, E. R. and N. F. F. Ebecken (2006). "Extracting Rules from Multilayer Perceptrons in Classification Problems: A Clustering-Based Approach". In: *Neurocomputing*. Neural Networks 70.1, pp. 384–397.

Huber, P. J. (1964). "Robust Estimation of a Location Parameter". In: *The Annals of Mathematical Statistics* 35.1. Publisher: Institute of Mathematical Statistics, pp. 73–101.

Humphrey, J. D., R. K. Strumpf, and F. C. Yin (1990). "Determination of a constitutive relation for passive myocardium: II. Parameter estimation". In: *Journal of Biomechanical Engineering* 112.3, pp. 340–346.

Huysmans, J., B. Baesens, and J. Vanthienen (2006). "ITER: An Algorithm for Predictive Regression Rule Extraction". eng. In: *Lecture Notes in Computer Science*. Vol. 4081. Springer, pp. 270–279.

Ibañez, R. et al. (2017). "Data-Driven Non-Linear Elasticity: Constitutive Manifold Construction and Problem Discretization". en. In: *Computational Mechanics* 60.5, pp. 813–826.

Ibañez, R. et al. (2016). "A Manifold Learning Approach to Data-Driven Computational Elasticity and Inelasticity". en. In: *Archives of Computational Methods in Engineering* 25.1, pp. 47–57.

Ibáñez, R. et al. (2018). "A Multidimensional Data-Driven Sparse Identification Technique: The Sparse Proper Generalized Decomposition". en. In: *Complexity*.

Ibáñez, R. et al. (2019). "Hybrid Constitutive Modeling: Data-Driven Learning of Corrections to Plasticity Models". en. In: *International Journal of Material Forming* 12.4, pp. 717–725.

Ibáñez Pinillo, R. (2019). "Advanced Physics-Based and Data-Driven Strategies". eng. Ph.D. Thesis. Universitat Politècnica de Catalunya.

Jadon, S. (2020). "A survey of loss functions for semantic segmentation". In: *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), pp. 1–7.

Johnston, W. J. and S. Fusi (2023). "Abstract representations emerge naturally in neural networks trained to perform multiple tasks". In: *Nature Communications* 14.1. Number: 1 Publisher: Nature Publishing Group, p. 1040.

Kaiser, E., J. N. Kutz, and S. L. Brunton (2018). "Sparse Identification of Nonlinear Dynamics for Model Predictive Control in the Low-Data Limit". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 474.2219, p. 20180335.

Kaklauskas Gintaris and Ghaboussi Jamshid (2001). "Stress-Strain Relations for Cracked Tensile Concrete from RC Beam Tests". In: *Journal of Structural Engineering* 127.1, pp. 64–73.

Kanno, Y. (2018). "Simple Heuristic for Data-Driven Computational Elasticity with Material Data Involving Noise and Outliers: A Local Robust Regression Approach". en. In: *Japan Journal of Industrial and Applied Mathematics* 35.3, pp. 1085–1101.

Kendall, A., Y. Gal, and R. Cipolla (2018). *Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics*.

Kiefer, J. and J. Wolfowitz (1952). "Stochastic Estimation of the Maximum of a Regression Function". In: *The Annals of Mathematical Statistics* 23.3. Publisher: Institute of Mathematical Statistics, pp. 462–466.

Kingma, D. P. and J. Ba (2014). "Adam: A Method for Stochastic Optimization". In: *arXiv:1412.6980 [cs]*. version: 1.

Kingma, D. P. and J. Ba (2017). *Adam: A Method for Stochastic Optimization*.

Kirby, R. C. and A. Logg (2006). "A compiler for variational forms". In: *ACM Transactions on Mathematical Software* 32.

Kirchdoerfer, T. and M. Ortiz (2016). "Data-Driven Computational Mechanics". In: *Computer Methods in Applied Mechanics and Engineering* 304, pp. 81–101.

Kirchdoerfer, T. and M. Ortiz (2017). "Data Driven Computing with Noisy Material Data Sets". In: *Computer Methods in Applied Mechanics and Engineering* 326, pp. 622–641.

Kirchdoerfer, T. and M. Ortiz (2018). "Data-Driven Computing in Dynamics". en. In: *International Journal for Numerical Methods in Engineering* 113.11, pp. 1697–1710.

Kirchdoerfer, T. T. (2017). "Data Driven Computing". PhD thesis. California Institute of Technology.

Kitouni, O., N. Nolte, and M. Williams (2023). *Expressive Monotonic Neural Networks.*

Klein, D. et al. (2021). "Polyconvex anisotropic hyperelasticity with neural networks". In: *arXiv:2106.14623 [cond-mat].*

Kossa, A. and S. Berezvai (2016). "Novel strategy for the hyperelastic parameter fitting procedure of polymer foam materials". In: *Polymer Testing* 53, pp. 149–155.

Koza, J. R. and J. R. Koza (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* en. MIT Press.

Kraft, D. (1988). "A software package for sequential quadratic programming". In: *Forschungsbericht-Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt.*

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25.* Ed. by F. Pereira et al. Curran Associates, Inc., pp. 1097–1105.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2017). "ImageNet classification with deep convolutional neural networks". In: *Communications of the ACM* 60.6, pp. 84–90.

Lake, B. M., R. Salakhutdinov, and J. B. Tenenbaum (2015). "Human-Level Concept Learning through Probabilistic Program Induction". en. In: *Science* 350.6266, pp. 1332–1338.

Lefik, M. and B. A. Schrefler (2003). "Artificial Neural Network as an Incremental Non-Linear Constitutive Model for a Finite Element Code". In: *Computer Methods in Applied Mechanics and Engineering.* Multiscale Computational Mechanics for Materials and Structures 192.28, pp. 3265–3283.

Leng, Y., S. Calve, and A. B. Tepole (2021). "Predicting the Mechanical Properties of Fibrin Using Neural Networks Trained on Discrete Fiber Network Data". In: *arXiv:2101.11712 [cs, q-bio].*

Leshno, M. et al. (1993). "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural Networks* 6.6, pp. 861–867.

Leygue, A. et al. (2018). "Data-Based Derivation of Material Response". In: *Computer Methods in Applied Mechanics and Engineering* 331, pp. 184–196.

Li, L. et al. (2018). *Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization.*

Li, S. et al. (2019). "Discovering Time-Varying Aerodynamics of a Prototype Bridge by Sparse Identification of Nonlinear Dynamical Systems". In: *Physical Review E* 100.2, p. 022220.

Li, X. et al. (2019). "Sparse Learning of Partial Differential Equations with Structured Dictionary Matrix". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 29.4, p. 043130.

Liang, T. et al. (2021). "Pruning and quantization for deep neural network acceleration: A survey". In: *Neurocomputing* 461, pp. 370–403.

Lin, B. et al. (2022). *Reasonable Effectiveness of Random Weighting: A Litmus Test for Multi-Task Learning.*

Linka, K. et al. (2021). "Constitutive artificial neural networks: A fast and general approach to predictive data-driven constitutive modeling by deep learning". In: *Journal of Computational Physics* 429, p. 110010.

Liu, W. K. et al. (2019). "A Computational Mechanics Special Issue on: Data-Driven Modeling and Simulation—Theory, Methods, and Applications". en. In: *Computational Mechanics* 64.2, pp. 275–277.

Liu, X. et al. (2022). *Certified Monotonic Neural Networks.*

Logg, A. and G. N. Wells (2010a). "DOLFIN: Automated finite element computing". In: *ACM Transactions on Mathematical Software* 37.

Logg, A., G. N. Wells, and J. Hake (2012). "DOLFIN: a C++/Python finite element library". In: *Automated solution of differential equations by the finite element method.* Ed. by A. Logg, K.-A. Mardal, and G. N. Wells. Vol. 84. Lecture notes in computational science and engineering. Section: 10. Springer.

Logg, A. and G. N. Wells (2010b). "DOLFIN: Automated Finite Element Computing". In: *ACM Trans. Math. Softw.* 37.2, 20:1–20:28.

Long, Z. et al. (2017). "PDE-Net: Learning PDEs from Data". In: *arXiv:1710.09668 [cs, math, stat].*

Mangan, N. M. et al. (2016). "Inferring Biological Networks by Sparse Identification of Nonlinear Dynamics". In: *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* 2.1, pp. 52–63.

Mangan, N. M. et al. (2017). "Model Selection for Dynamical Systems via Sparse Regression and Information Criteria". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473.2204, p. 20170009.

Marckmann, G. and E. Verron (2006). "Comparison of Hyperelastic Models for Rubber-like Materials". In: *Rubber Chemistry and Technology* 79.5, pp. 835–858.

Marsden, J. E. and T. J. R. Hughes (1994). *Mathematical Foundations of Elasticity.* Courier Corporation. 576 pp.

Martius, G. and C. H. Lampert (2016). "Extrapolation and Learning Equations". In: *arXiv:1610.02995 [cs]*.

Marzano, S. (1983). "An interpretation of Baker-Ericksen inequalities in uniaxial deformation and stress". In: *Meccanica* 18.4, pp. 233–235.

May-Newman, K. and F. C. Yin (1998). "A constitutive law for mitral valve tissue". In: *Journal of Biomechanical Engineering* 120.1, pp. 38–47.

Miehe, C. (1993). "Computation of isotropic tensor functions". In: *Communications in Numerical Methods in Engineering* 9.11. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cnm.1640091105, pp. 889–896.

Mihai, L. A. et al. (2017). "A family of hyperelastic models for human brain tissue". In: *Journal of the Mechanics and Physics of Solids* 106, pp. 60–79.

Mikolov, T. (2012). "Statistical language models based on neural networks". PhD thesis. Brno, CZ: Brno University of Technology, Faculty of Information Technology.

Milani Fard, M. et al. (2016). "Fast and Flexible Monotonic Functions with Ensembles of Lattices". In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc.

Mohamed, M. H. (2011). "Rules Extraction from Constructively Trained Neural Networks Based on Genetic Algorithms". In: *Neurocomputing* 74.17, pp. 3180–3192.

Montáns, F. J. et al. (2019). *Complex Algorithms for Data-Driven Model Learning in Science and Engineering*. en. https://www.hindawi.com/journals/complexity/2019/5040637/. Research Article.

Montavon, G., W. Samek, and K.-R. Müller (2018). "Methods for Interpreting and Understanding Deep Neural Networks". In: *Digital Signal Processing* 73, pp. 1–15.

Morman Jr., K. N. (1986). "The Generalized Strain Measure With Application to Nonhomogeneous Deformations in Rubber-Like Solids". In: *Journal of Applied Mechanics* 53.3, pp. 726–728.

Nair, V. and G. E. Hinton (2010). "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML'10. Madison, WI, USA: Omnipress, pp. 807–814.

Neff, P., B. Eidel, and R. Martin (2014). *The axiomatic deduction of the quadratic Hencky strain energy by Heinrich Hencky*.

Neff, P., B. Eidel, and R. J. Martin (2016). "Geometry of Logarithmic Strain Measures in Solid Mechanics". In: *Archive for Rational Mechanics and Analysis* 222.2, pp. 507–572.

Neff, P., I.-D. Ghiba, and J. Lankeit (2015). "The Exponentiated Hencky-Logarithmic Strain Energy. Part I: Constitutive Issues and Rank-One Convexity". In: *Journal of Elasticity* 121.2, pp. 143–234.

Neff, P., I. Münch, and R. Martin (2016). "Rediscovering GF Becker's early axiomatic deduction of a multiaxial nonlinear stress–strain relation based on logarithmic strain". In: *Mathematics and Mechanics of Solids* 21.7, pp. 856–911.

Negahban, M. and A. S. Wineman (1989). "Material symmetry and the evolution of anisotropies in a simple material—I. Change of reference configuration". In: *International Journal of Non-Linear Mechanics* 24.6, pp. 521–536.

Nguyen, L. T. K., R. C. Aydin, and C. J. Cyron (2022). "Accelerating the distance-minimizing method for data-driven elasticity with adaptive hyperparameters". In: *Computational Mechanics* 70.3, pp. 621–638.

Nguyen, L. T. K. and M.-A. Keip (2018). "A Data-Driven Approach to Nonlinear Elasticity". In: *Computers & Structures* 194, pp. 97–115.

Nguyen, A.-P. et al. (2023). "MonoNet: Enhancing interpretability in neural networks via Monotonic Features". In: *Bioinformatics Advances*, vbad016.

Nierlich, E. (2005). "An "Empirical Science" of Literature". en. In: *Journal for General Philosophy of Science* 36.2, pp. 351–376.

Noll, W. (2006). "A Frame-Free Formulation of Elasticity". In: *Journal of Elasticity* 83.3, pp. 291–307.

Noll, W. (2009). "Thoughts on Thermomechanics". In: *Journal of Thermal Stresses* 33.1. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01495730903408724, pp. 15–28.

Odajima, K. et al. (2008). "Greedy Rule Generation from Discrete Data and Its Use in Neural Network Rule Extraction". In: *Neural Networks* 21.7, pp. 1020–1028.

Oden, J. T. (2011). *An Introduction to Mathematical Modeling: A Course in Mechanics*. John Wiley & Sons. 348 pp.

Ogden, R. W. (1970). "Compressible isotropic elastic solids under finite strain-constitutive inequalities". In: *The Quarterly Journal of Mechanics and Applied Mathematics* 23.4, pp. 457–468.

Ogden, R. W. (1997). *Non-linear elastic deformations*. Courier Corporation.

Ogden, R. W., G. Saccomandi, and I. Sgura (2004). "Fitting hyperelastic models to experimental data". In: *Computational Mechanics* 34.6, pp. 484–502.

Ogden, R. (1970). "Compressible isotropic elastic solids under finite strain-constitutive inequalities". In: *The Quarterly Journal of Mechanics and Applied Mathematics* 23.4. Publisher: Oxford University Press, pp. 457–468.

Paszke, A. et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: p. 12.

Peherstorfer, B. and K. Willcox (2015). "Dynamic Data-Driven Reduced-Order Models". In: *Computer Methods in Applied Mechanics and Engineering* 291, pp. 21–41.

Peherstorfer, B. and K. Willcox (2016). "Data-Driven Operator Inference for Nonintrusive Projection-Based Model Reduction". In: *Computer Methods in Applied Mechanics and Engineering* 306, pp. 196–215.

Poon, H. and P. Domingos (2011). "Sum-Product Networks: A New Deep Architecture". In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 689–690.

Prussing, J. E. (1986). "The principal minor test for semidefinite matrices". In: *Journal of Guidance, Control, and Dynamics* 9.1. Publisher: American Institute of Aeronautics and Astronautics, pp. 121–122.

Qin, T., K. Wu, and D. Xiu (2019). "Data Driven Governing Equations Approximation Using Deep Neural Networks". In: *Journal of Computational Physics* 395, pp. 620–635.

Quade, M. et al. (2018). "Sparse Identification of Nonlinear Dynamics for Rapid Model Recovery". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 28.6, p. 063116.

Raissi, M., P. Perdikaris, and G. E. Karniadakis (2019). "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations". In: *Journal of Computational Physics* 378, pp. 686–707.

Raissi, M. and G. E. Karniadakis (2018). "Hidden Physics Models: Machine Learning of Nonlinear Partial Differential Equations". In: *Journal of Computational Physics* 357, pp. 125–141.

Raissi, M., P. Perdikaris, and G. E. Karniadakis (2017a). "Machine Learning of Linear Differential Equations Using Gaussian Processes". In: *Journal of Computational Physics* 348, pp. 683–693.

Raissi, M., P. Perdikaris, and G. E. Karniadakis (2017b). "Physics Informed Deep Learning (Part I): Data-Driven Solutions of Nonlinear Partial Differential Equations". In: *arXiv:1711.10561 [cs, math, stat]*.

Raissi, M., P. Perdikaris, and G. E. Karniadakis (2017c). "Physics Informed Deep Learning (Part II): Data-Driven Discovery of Nonlinear Partial Differential Equations". In: *arXiv:1711.10566 [cs, math, stat]*.

Ramachandran, P., B. Zoph, and Q. V. Le (2017). *Searching for Activation Functions*.

Rangwala, M. H. (2006). "Empirical Investigation of Decision Tree Extraction from Neural Networks". en. PhD thesis. Ohio University.

El-Ratal, W. H. and P. K. Mallick (1996). "Elastic response of flexible polyurethane foams in uniaxial tension". In: *Journal of Engineering Materials and Technology* 118.2.

Robbins, H. and S. Monro (1951). "A Stochastic Approximation Method". In: *The Annals of Mathematical Statistics* 22.3. Publisher: Institute of Mathematical Statistics, pp. 400–407.

Rosenblatt, F. (1957). *The perceptron: A perceiving and recognizing automaton*. 85-460-1. Ithaca, New York: Cornell Aeronautical Laboratory.

Ruder, S. (2017). *An Overview of Multi-Task Learning in Deep Neural Networks*.

Rudy, S. H. et al. (2017). "Data-Driven Discovery of Partial Differential Equations". en. In: *Science Advances* 3.4, e1602614.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). "Learning representations by back-propagating errors". In: *Nature* 323.6088. Number: 6088 Publisher: Nature Publishing Group, pp. 533–536.

Sahoo, S. S., C. H. Lampert, and G. Martius (2018). "Learning Equations for Extrapolation and Control". In: *arXiv:1806.07259 [cs, stat]*.

Saito, K. and R. Nakano (2002). "Extracting Regression Rules from Neural Networks". In: *Neural Networks* 15.10, pp. 1279–1288.

Samarasinghe, S. (2016). *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. en. CRC Press.

Schmidt, M. and H. Lipson (2009). "Distilling Free-Form Natural Laws from Experimental Data". en. In: *Science* 324.5923, pp. 81–85.

Scroggs, M. W., I. A. Baratta, et al. (2022). "Basix: a runtime finite element basis evaluation library". In: *Journal of Open Source Software* 7.73, p. 3982.

Scroggs, M. W., J. S. Dokken, et al. (2022). "Construction of arbitrary order finite element degree-of-freedom maps on polygonal and polyhedral cell meshes". In: *ACM Transactions on Mathematical Software* 48.2, 18:1–18:23.

Senushkin, D. et al. (2023). *Independent Component Alignment for Multi-Task Learning*.

Seth, B. R. (1961). *Generalized strain measure with applications to physical problems*. Wisconsin Univ-Madison Mathematics Research Center.

Seth, B. R. (1966). "Measure-concept in mechanics". In: *International Journal of Non-Linear Mechanics* 1.1, pp. 35–40.

Setiono, R. and H. Liu (1997). "NeuroLinear: From Neural Networks to Oblique Decision Rules". In: *Neurocomputing* 17.1, pp. 1–24.

Setiono, R. and J. Y. L. Thong (2004). "An Approach to Generate Rules from Neural Networks for Regression Problems". In: *European Journal of Operational Research* 155.1, pp. 239–250.

Shariff, M. H. B. M. (2000). "Strain Energy Function for Filled and Unfilled Rubberlike Material". In: *Rubber Chemistry and Technology* 73.1, pp. 1–18.

Shen, Y. et al. (2004). "Neural Network Based Constitutive Model for Rubber Material". In: *Rubber Chemistry and Technology* 77.2, pp. 257–277.

Shin, Y. and J. Ghosh (1991). "The Pi-Sigma Network: An Efficient Higher-Order Neural Network for Pattern Classification and Function Approximation". In: *IJCNN-91-Seattle International Joint Conference on Neural Networks*. Vol. i, 13–18 vol.1.

Sill, J. (1997). "Monotonic Networks". In: *Advances in Neural Information Processing Systems*. Vol. 10. MIT Press.

Sill, J. and Y. Abu-Mostafa (1996). "Monotonicity Hints". In: *Advances in Neural Information Processing Systems*. Vol. 9. MIT Press.

Silver, D. et al. (2017). "Mastering the Game of Go without Human Knowledge". en. In: *Nature* 550.7676, pp. 354–359.

Simo, J. C. and R. L. Taylor (1991). "Quasi-incompressible finite elasticity in principal stretches. continuum basis and numerical algorithms". In: *Computer Methods in Applied Mechanics and Engineering* 85.3, pp. 273–310.

Sivaraman, A. et al. (2020). "Counterexample-Guided Learning of Monotonic Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., pp. 11936–11948.

Snoek, J., H. Larochelle, and R. P. Adams (2012). "Practical Bayesian Optimization of Machine Learning Algorithms". In: *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc.

Solomatine, D., L. See, and R. Abrahart (2008). "Data-Driven Modelling: Concepts, Approaches and Experiences". en. In: *Practical Hydroinformatics: Computational Intelligence and Technological Developments in Water Applications*. Ed. by R. J. Abrahart, L. M. See, and D. P. Solomatine. Water Science and Technology Library. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 17–30.

Stoffel, M., F. Bamer, and B. Markert (2019). "Neural Network Based Constitutive Modeling of Nonlinear Viscoplastic Structural Response". In: *Mechanics Research Communications* 95, pp. 85–88.

Storåkers, B. (1986). "On material representation and constitutive branching in finite compressible elasticity". In: *Journal of the Mechanics and Physics of Solids* 34.2, pp. 125–145.

Sutton, R. S. (1992). "Introduction: The Challenge of Reinforcement Learning". en. In: *Reinforcement Learning*. Ed. by R. S. Sutton. The Springer International Series in Engineering and Computer Science. Boston, MA: Springer US, pp. 1–3.

Swersky, K., J. Snoek, and R. P. Adams (2014). *Freeze-Thaw Bayesian Optimization*.

Szegedy, C. et al. (2014). *Going Deeper with Convolutions*.

Tan, M. and Q. V. Le (2020). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*.

Thrun, S. (1995). "Extracting Rules from Artificial Neural Networks with Distributed Representations". In: *Advances in Neural Information Processing Systems 7*. Ed. by G. Tesauro, D. S. Touretzky, and T. K. Leen. MIT Press, pp. 505–512.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1, pp. 267–288.

Touvron, H. et al. (2023). *LLaMA: Open and Efficient Foundation Language Models*.

Towell, G. G. and J. W. Shavlik (1993). "Extracting Refined Rules from Knowledge-Based Neural Networks". en. In: *Machine Learning* 13.1, pp. 71–101.

Truesdell, C. and W. Noll (2004). "The Non-Linear Field Theories of Mechanics". In: *The Non-Linear Field Theories of Mechanics*. Ed. by C. Truesdell, W. Noll, and S. S. Antman. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1–579.

Vapnik, V., S. Golowich, and A. Smola (1996). "Support Vector Method for Function Approximation, Regression Estimation and Signal Processing". In: *Advances in Neural Information Processing Systems*. Vol. 9. MIT Press.

Vastl, M. et al. (2022). *SymFormer: End-to-end symbolic regression using transformer-based architecture*. version: 3.

Vaswani, A. et al. (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc.

Virtanen, P. et al. (2020). "SciPy 1.0: fundamental algorithms for scientific computing in Python". In: *Nature Methods* 17.3. Number: 3 Publisher: Nature Publishing Group, pp. 261–272.

Voss, J. (2021). "Constitutive inequalities in nonlinear elasticity with focus on planar quasiconvexity". PhD thesis. DuEPublico: Duisburg-Essen Publications online, University of Duisburg-Essen, Germany.

Wang, K. and W. Sun (2018). "A Multiscale Multi-Permeability Poroplasticity Model Linked by Recursive Homogenizations and Deep Learning". In: *Computer Methods in Applied Mechanics and Engineering* 334, pp. 337–380.

Wang, K. and W. Sun (2019). "Meta-Modeling Game for Deriving Theory-Consistent, Microstructure-Based Traction–Separation Laws via Deep Reinforcement Learning". In: *Computer Methods in Applied Mechanics and Engineering* 346, pp. 216–241.

Wang, K., W. Sun, and Q. Du (2019). "A Cooperative Game for Automated Learning of Elasto-Plasticity Knowledge Graphs and Models with AI-Guided Experimentation". en. In: *Computational Mechanics* 64.2, pp. 467–499.

Wang, L. M., K. Linka, and E. Kuhl (2023). "Automated model discovery for muscle using constitutive recurrent neural networks". In: *Journal of the Mechanical Behavior of Biomedical Materials*, p. 106021.

Wang, Q. et al. (2022). "A Comprehensive Survey of Loss Functions in Machine Learning". In: *Annals of Data Science* 9.2, pp. 187–212.

Weber, P., J. Geiger, and W. Wagner (2021). "Constrained neural network training and its application to hyperelastic material modeling". In: *Computational Mechanics* 68.5, pp. 1179–1204.

Wehenkel, A. and G. Louppe (2021). "Unconstrained Monotonic Neural Networks". In: *arXiv:1908.05164 [cs, stat]*.

Wolpert, D. and W. Macready (1997). "No free lunch theorems for optimization". In: *IEEE Transactions on Evolutionary Computation* 1.1. Conference Name: IEEE Transactions on Evolutionary Computation, pp. 67–82.

Wu, X. and J. Ghaboussi (1995). *Neural Network-Based Material Modeling*. en. Text. University of Illinois Engineering Experiment Station. College of Engineering. University of Illinois at Urbana-Champaign.

Xiao, H., O. T. Bruhns, and A. Meyers (1997). "Logarithmic strain, logarithmic spin and logarithmic rate". In: *Acta Mechanica* 124.1, pp. 89–105.

Xue, T. et al. (2023). "JAX-FEM: A differentiable GPU-accelerated 3D finite element solver for automatic inverse design and mechanistic data science". In: *Computer Physics Communications* 291, p. 108802.

Yagawa, G. and H. Okuda (1996). "Neural Networks in Computational Mechanics". en. In: *Archives of Computational Methods in Engineering* 3.4, p. 435.

Yang, L. and A. Shami (2020). "On hyperparameter optimization of machine learning algorithms: Theory and practice". In: *Neurocomputing* 415, pp. 295–316.

You, S. et al. (2017). *Deep Lattice Networks and Partial Monotonic Functions*.

Young, W. A. and G. R. Weckman (2010). "Using a Heuristic Approach to Derive a Grey-Box Model through an Artificial Neural Network Knowledge Extraction Technique". en. In: *Neural Computing and Applications* 19.3, pp. 353–366.

Zhang, J. et al. (2020). *Why gradient clipping accelerates training: A theoretical justification for adaptivity*.

Zhang, Y. and Q. Yang (2021). *A Survey on Multi-Task Learning*.

Zhou, Z.-c., Y. Jiang, and S. Chen (2003). "Extracting Symbolic Rules from Trained Neural Network Ensembles". In: *AI Commun.* 16, pp. 3–15.

Zienkiewicz, O. C. and R. L. Taylor (2000). *The Finite Element Method, The Basis*. Fifth edition. Vol. Volume. Oxford: Butterworth-Heinemann. 736 pp.

Ziyin, L., T. Hartwig, and M. Ueda (2020). *Neural Networks Fail to Learn Periodic Functions and How to Fix It*.