Isogeometric Analysis of Wrinkling

Verhelst, H.M.

DOI
10.4233/uuid:0e4c3644-31a4-4157-983d-bd001d91b8ca

Publication date
2024

Document Version
Final published version

Citation (APA)
Verhelst, H. M. (2024). *Isogeometric Analysis of Wrinkling*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:0e4c3644-31a4-4157-983d-bd001d91b8ca

Isogeometric Analysis of Wrinkling

Isogeometric Analysis of Wrinkling

Proefschrift

ter verkrijging van de graad van doctor aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op donderdag 18 januari 2024 om 12:30 uur

door

Hugo Maarten VERHELST

Master of Science in Maritieme Techniek en Toegepaste Wiskunde,
Technische Universiteit Delft, Nederland,
geboren te Wijchen, Nederland.

Dit proefschrift is goedgekeurd door de

promotor: Dr. M. Möller
copromotor: Dr. ir. J.H. Den Besten

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Dr. M. Möller,	Technische Universiteit Delft, Nederland
Dr. ir. J.H. Den Besten,	Technische Universiteit Delft, Nederland
Onafhankelijke leden:	
Prof. dr. ir. A. Van Keulen	Technische Universiteit Delft, Nederland
Prof. dr. J. Kiendl	Bundeswehr Universität München, Duitsland
Prof. dr. A. Reali	Università di Pavia, Italië
Prof. dr. D. Vella	Oxford University, Verenigd Koninkrijk
Prof. dr. ir. C. Vuik,	Technische Universiteit Delft, Nederland, *reservelid*
Overige leden:	
Dr. A. Mantzaflaris	Université Côte d'Azur, Frankrijk

Dr. A. Mantzaflaris heeft in belangrijke mate aan de totstandkoming van het proefschrift bijgedragen.

Keywords: Arc-Length Methods, Adaptive Meshing, Buckling Analysis, Isogeometric Analysis, Hyperelasticity, Tension-Field Theory, Post-Buckling Analysis, Unstructured Splines, Wrinkling Analysis

Printed by: Ridderprint | `www.ridderprint.nl`

Front & Back: The cover of this thesis is the result of a numerical simulation of an aluminium membrane of length × width × thickness = $360 \times 240 \times 10^{-3}$ [mm^3] subject to a horizontal displacement of the top boundary of 10 [mm], in `G+Smo`. This simulation can be reproduced by running `example_ShearWrinkling` with default options. The result is styled into a cover by Studio MES | `https://studiomes.nl`

ISBN 978-94-6483-729-2

An electronic version of this dissertation is available at
`http://repository.tudelft.nl/`.

Preface

Dear reader,

This dissertation is the result of a collaboration between an engineering science and a mathematics department. It is the result of four years of PhD research in collaboration with – again – mathematicians and engineers from all over the world. Therefore, this dissertation is neither written for only engineers nor for only mathematicians. Instead, it provides concepts applied across a broad spectrum between engineering concepts and mathematical test cases. To facilitate the engineer, the mathematician, and anyone else with a thorough background reading this dissertation, a chapter with mathematical and physical preliminaries has been included. This chapter provides basic concepts and ideas related to the core of this dissertation, with a broad range of references to use for further reading. I hope, whoever you are, wherever you are, whenever you read this, and whatever your background is, that you will learn something from this work.

Hugo Maarten Verhelst
Delft, January 2024

Voorwoord

Beste lezer,

Dit proefschrift is een resultaat van een samenwerking tussen een afdeling in ingenieurswetenschappen en een afdeling in de wiskunde. Het is een resultaat van vier jaar doctoraalonderzoek, in samenwerking met – alweer – wiskundigen en ingenieurs van over de hele wereld. Daarom is dit proefschrift niet puur geschreven voor ingenieurs, en ook niet puur voor wiskundigen. In plaats daarvan levert dit proefschrift concepten toegepast op een breed spectrum tussen ingenieursconcepten en wiskundige test-cases. Om de ingenieur, de wiskundige en ieder ander van een diepgaande achtergrond te voorzien om dit proefschrift te lezen is een *preliminaries* hoofdstuk toegevoegd. Dit hoofdstuk bevat basale concepten en ideeën gerelateerd aan de kern van dit proefschrift met een breed scala aan referenties voor verdere verdieping. Ik hoop, wie je ook bent, waar je je ook bevindt, wanneer je dit ook leest en wat je achtergrond ook mag zijn, dat je iets leert van dit werk.

Hugo Maarten Verhelst
Delft, januari 2024

Summary

Wrinkles are ubiquitous in the world around us. In our daily lives, we encounter wrinkles in various forms, whether in our clothes or on our skin. Wrinkles emerge as a result of a delicate interplay between bending, membrane, and foundation stiffness contributions within membranes. While experimental investigations provide insights into the physics underlying wrinkling, numerical investigations find their purpose in the design, analysis, and optimisation of membranes subjected to wrinkling. Nevertheless, the numerical simulation of membrane wrinkling presents several challenges. Firstly, wrinkling constitutes a buckling phenomenon in membranes with low bending stiffness. Wrinkles have the potential to evolve into folds, creases, or other wrinkling patterns as loads or displacements increase. Secondly, the wavelengths of wrinkling can be orders of magnitude smaller than the overall geometry, requiring a small resolution of the numerical simulation and hence increasing computational costs. Overall, the question arises of how to design robust and accurate numerical models for the analysis of wrinkled membranes. This dissertation is subdivided into four parts and aims to provide answers to this question.

Constitutive Modelling
The first theme considers hyperelastic material modelling, with a focus on developing wrinkling models under large strains. The shell model employed in this dissertation is based on the isogeometric analysis paradigm. Specifically, the Kirchhoff–Love shell model is used, which leverages the higher-order continuity of underlying spline spaces. Chapter 3 extends hyperelastic material formulations to stretch-based materials, enabling the use of the isogeometric analysis paradigm for rubber-like shells. Since the modelling of wrinkling patterns imposes physical scales limiting element mesh sizes, chapter 4 introduces a hyperelastic isogeometric membrane element that incorporates an implicit wrinkling model, thus avoiding explicit modelling of wrinkling amplitudes.

Adaptive Simulation
The second theme addresses adaptive methods. On the one hand, spatial adaptivity enhances the local detail in a numerical simulation. Chapter 5 presents an adaptive isogeometric analysis framework based on intuitive goal functions, such as wrinkling amplitudes, to guide adaptive meshing routines. On the other hand, temporal or quasi-temporal adaptivity serves to enhance the efficiency of dynamic or quasi-static simulations. Chapter 6 introduces an adaptive parallel arc-length method. The method's adaptivity arises as a by-product of parallelisation efforts aimed at reducing computational times for quasi-static simulations.

Multi-Patch Modelling
The advantage of the smoothness inherent in the spline spaces used in isogeometric analysis is limited to simple topologies. To benefit from this smoothness in complex geometries,

the third theme of this dissertation focuses on complex domain modelling. Chapter 7 presents a qualitative and quantitative comparison of unstructured spline constructions for multi-patch modelling using isogeometric analysis. This chapter offers insights and suggestions for future developments related to unstructured spline constructions.

Result Reproduction
The final theme of this dissertation concerns the reproducibility of the developed methods. In this section, design considerations are presented for an open-source software library, along with small examples, aimed at ensuring easy reproducibility and supporting future research in the three themes mentioned earlier.

In summary, this dissertation offers a wide range of methods for the isogeometric analysis of structural instabilities in thin-walled structures, including the modelling of wrinkling. The concepts developed in terms of hyperelasticity expand the applicability of wrinkling models to encompass large strains. The concepts developed in terms of adaptivity provide intuitive error estimators that drive local refinement in space, as well as a novel continuation method that eliminates the inherently serial arc-length methods. Through the use of unstructured splines, complex domains become accessible for the analysis of structural stabilities. By creating an open-source, forward-compatible software library, these concepts are made available for future developments in the field of isogeometric analysis of wrinkling.

Samenvatting

Kreukels zijn alomvertegenwoordigd in de wereld om ons heen. In ons dagelijks level nemen we kreukels waar in verschillende vormen en maten: bijvoorbeeld in onze kleding of in onze huid. Kreukels ontstaan ten gevolge van een delicaat samenspel tussen buigstijfheid, membraanstijfheid in membranen, en de stijfheid van de fundering onder membranen. Waar experimenteel onderzoek inzichten geeft in de fysica achter het vormen van kreukels, wordt numeriek onderzoek toegepast in het onderwerp, de analyse en de optimalisering van kreukelende membranen. Toch zijn er verschillende uitdagingen gerelateerd aan numerieke simulatie van kreukels. Ten eerste ontstaan kreukels als een knikprobleem in membranen met een lage buigstijfheid. Kreukels kunnen zich verder evolueren tot plooien, vouwen of andere kreukelpatronen als belastingen of verplaatsingen toenemen. Ten tweede kunnen de golflengtes van kreukels ordes van grootte kleiner zijn dan het formaat van het membraan zelf, wat vraagt om veel detail in de numerieke simulatie, en daardoor de numerieke kosten ophoogt. Over het algemeen is de vraag hoe robuuste en nauwkeurige numerieke modellen voor de analyse van kreukels moeten worden ontworpen. Dit proefschrift is onderverdeeld in vier delen en beoogt deze vraag te beantwoorden.

Constitutief modelleren
Het eerste thema van dit proefschrift beschouwt het modelleren van hyperelastische materialen, met nadruk op het ontwikkelen van kreukelmodellen voor grote rekken. Het schalenmodel wat gebruikt wordt in fit proefschrift is gebaseerd op het isogeometricsche analyse-paradigma. Meer specifiek wordt het Kirchhoff-Love schalenmodel gebruikt, welke gebruik maakt van de hoger-orde continuïteit van de onderliggende *spline*ruimtes. Hoofdstuk 3 breidt hyperelastische materiaalformuleringen uit tot *stretch*-gebaseerde materialen, zodat het gebruik van het isogeometrische analyse-paradigma gebruikt kan worden voor rubberachtige schalen. Omdat fysieke lengteschalen de maasgrootte kunnen bepalen bij het modelleren van kreukelpatronen introduceert Hoofdstuk 4 een hyperelastisch isogeometrisch membraanelement met een impliciet kreukelmodel. Daardoor wordt het expliciet modelleren van kreukelamplitudes vermeden.

Adaptief Simuleren
Het tweede thema adresseert adaptieve methoden. Aan de ene kant kan ruimtelijke adaptiviteit het lokale detail van een numerieke simulatie bevorderen. Hoofdstuk 5 presenteert een adaptief isogeometrisch framework gebaseerd op intuïtieve doelfuncties, bijvoorbeeld kreukelamplitudes, om adaptieve maasmodificatieroutines aan te sturen. Aan de andere kant kan tijds- of quasi-tijdsadaptiviteit de efficiëntie van dynamische of quasistatisch simulaties bevorderen. Hoofdstuk 6 introduceert een adaptieve parallelle booglengtemethode. De adaptiviteit van de methode komt voort als een bijproduct van de parallellisatie die als doel heeft om de rekentijden voor quasistatisch simulaties te reduceren.

Multi-Patch Modelleren

Het voordeel van de gladheid inherent aan de *spline*ruimtes die gebruikt worden in isogeometrische analyse is beperkt tot eenvoudige topologieën. Om voordeel te halen uit deze gladheid voor complexe geometrieën, focust het derde thema van dit proefschrift op het modelleren van complexe domeinen. Hoofdstuk 7 presenteert een kwalitatieve en kwantitatieve vergelijking van ongestructureerde *spline*constructies voor *multi-patch* modellering met isogeometrische analyse. Dit hoofdstuk geeft inzichten en suggesties voor verdere ontwikkelingen gerelateerd aan ongestructureerde *spline*-constructies.

Resultaten Reproduceren

Het laatste thema van dit proefschrift omvat de reproduceerbaarheid van de ontwikkelde methoden. In hoofdstuk 8 worden ontwerpoverwegingen gepresenteerd voor een openbare softwarebibliotheek, samen met kleine voorbeelden. Het doel is om eenvoudige reproduceerbaarheid en toekomstig onderzoek rondom de drie eerdergenoemde thema's te garanderen.

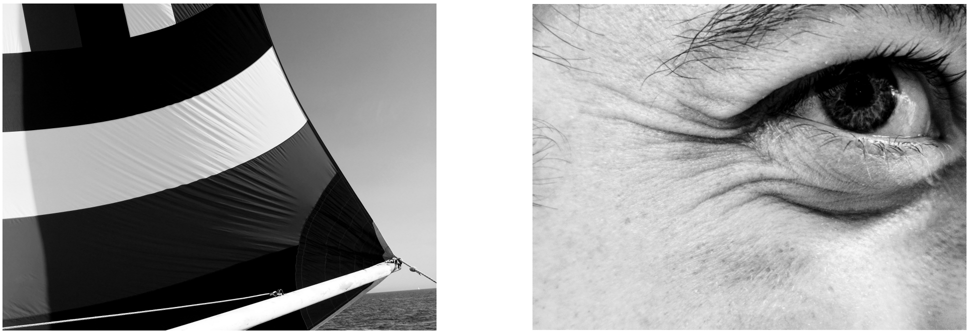
Samenvattend biedt dit proefschrift een breed scala aan methoden voor het isogeometrisch modelleren van constructie-insabiliteiten in dunwandige structuren, inclusief het modelleren van kreukels. De concepten die ontwikkeld zijn aangaande hyperelasticiteit, breiden de toepasbaarheid van kreukelmodellen uit richting grote rekken. De concepten die ontwikkeld zijn gerelateerd aan adaptiviteit geven intuïtieve foutschatters die lokale verfijning kunnen leiden, en ook een nieuwe continuatiemethode die seriële booglengtemethoden elimineert. Via het gebruik van ongestructureerde *spline*constructies komen complexe domeinen beschikbaar voor de analyse van structurele instabiliteiten. Door een openbare, voorwaarts compatibele softwarebibliotheek te creëren, zijn de bovenstaande concepten beschikbaar voor toekomstige ontwikkelingen op het gebied van isogeometrische analyse van kreukels.

Contents

1

Introduction

Wrinkles are an essential part of us. Wrinkles, for example, show up on our skin when laughing (figure 1.1b) or in fruits and vegetables, in case they lay around for too long (figure 1.1d). Wrinkles are all around us in different fields of applications. Wrinkles show up in sails of ships (figure 1.1a) or spacecraft membrane propulsors. Wrinkles shape inflatables like airbags (figure 1.1c)and have a vital role in the stability of parachutes. The thickness of skin tissue, sail canvas, wing foil, airbag sheet and parachute fabric is typically very small, all referring to a membrane type of element. Wrinkles appear at different scales, ranging from layers of graphene at nano-metre scale up to floating solar membranes at kilometre-scale, and at the same time the wrinkle wave lengths are potentially small. Last but not least, the wrinkling behaviour in time introduces a structural stability problem. Patterns suddenly appear, disappear and transform, meaning that numerical analysis of wrinkling membranes is a challenging topic from computational science perspective.



(a) Wrinkles in the corner of the spinnaker of the *Andante*, a classical example of wrinkling under in-plane loading.

(b) Wrinkling in a the corner of the author's eye, a classical example of wrinkling of a membrane on an elastic foundation.

Figure 1.1: (Caption on next page.)

(c) Wrinkling in inflatable neck pillow, a classical example of wrinkling in a pressurised membrane.

(d) Wrinkling of an aged apple, a classical example of growth- or shrinkage-induced wrinkling on an elastic foundation.

Figure 1.1: Everyday examples of different forms of wrinkling.

With the advent of Isogeometric Analysis (IGA), a new computational framework entered the realm of computational science and engineering. IGA is a computational framework enabling a one-to-one workflow of Computer-Aided Design (CAD) and Computer-Aided Engineering (CAE). Using splines as a mathematical basis allows for exactly the same geometry in CAD and CAE, providing high efficiency in numerical analysis of structural instabilities like wrinkling of membranes. The Kirchhoff–Love shell formulation – containing membrane properties – is a key element in that respect. In addition, the geometric representation inherited from CAD provides an intuitive parametrisation for shape optimisation. Despite the momentum of IGA since the introduction in 2005, applications in the field of structural stability problems like wrinkling and membranes are limited. A dedicated framework is considered to be essential for the design, analysis as well as optimisation.

1.1 Research Goals

This dissertation aims to provide a general modelling framework for the modelling of shell instabilities using isogeometric analysis. In particular, the models in this dissertation enable the analysis of membrane wrinkling in a robust and efficient manner.

Develop robust and efficient models for structural stability analysis using isogeometric analysis, with a focus on wrinkling.

Since a computational analysis of stability problems involves multiple aspects, different subjects related to the numerical analysis of wrinkling are covered. First of all, membrane type of materials are sometimes made of rubber and require advanced hyperelastic models rather than a linear elastic one. Opposite to explicit modelling of wrinkling patterns, such models can be modified such that their behaviour changes where the membrane is in a wrinkled state; also known as implicit wrinkling modelling. Therefore, *hyperelastic* models and their application to membrane wrinkling are part of the first sub-goal of this dissertation:

(i) *Extend the isogeometric Kirchhoff-Love shell formulation with constitutive models required for implicit and explicit wrinkling analysis.*

A second aspect involves efficient allocation of computational resources wherever more detail is needed, introducing the concept of adaptivity. This concept of *adaptivity* is covered in the second sub-goal of this dissertation:

(ii) *Develop methods for adaptive meshing and adaptive load stepping tailored to stability analysis.*

In case of more *complex geometries*, for example because of holes, sub-domains have to be patched, introducing a third aspect:

(iii) *Provide a qualitative and quantitative comparison of unstructured spline constructions for IGA.*

In a broad sense, robustness and efficiency can be reflected in software design as well. Robust software can be applied to several problems or when it can be extended, without crashing. Efficiency, particularly for users or developers, is achieved by modularity and an easy-to-use interface. The last sub-goal of this dissertation relates to the tools that are developed through goals (i) to (iii):

(iv) *Develop an efficient, robust and future proof open-source computational tool for isogeometric wrinkling analysis.*

1.2 Dissertation Outline

This dissertation is the result of a study on computational modelling techniques for stability problems using isogeometric analysis. To support the reader with mathematical and physical preliminaries combined with a broad literature overview, **chapter 2** provides preliminaries for the remainder of this dissertation. Thereafter, this dissertation consists of four parts, aligned with goals (i) to (iv):

(i) **Constitutive Modelling**
Constitutive modelling entails the modelling of the relationship between stress and strain in materials. This part focuses on constitutive models for hyperelastic material models, i.e., material models that are accurate for large strains. **Chapter 3** presents a framework to include stretch-based hyperelastic material models in the isogeometric Kirchhoff-Love shell formulation. Stretch-based hyperelastic material models are particularly relevant for the modelling of rubber-like materials. Thereafter, **chapter 4** provides a modification scheme for hyperelastic material models, where the constitutive model of the material model is locally modified depending on the tension field of a membrane at a material point, i.e., whether the membrane is in a taut, slack, or wrinkled state.

(ii) **Adaptive Simulation**
Adaptivity in numerical simulations is typically used to allocate computational resources or detail more efficiently in parts of the domain where it is needed. A domain could be spatial, but it can also be related to the variation of other parameters.

Chapter 5 provides a framework for spatially adaptive simulations for isogeometric Kirchhoff-Love shells. This framework is driven by goal-adaptive error estimators, meaning that the error is defined in terms of a goal function (e.g., predicting the stress in a part of the domain), steering the computational mesh towards regions with the highest error contributions. On the other hand, **chapter 6** provides a novel method to parallelise arc-length methods, which are typically used to find solutions in terms of varying load paths, e.g., in structural stability computations. The approach is referred to as the Adaptive Parallel Arc-Length Method and is inherently adaptive, besides providing computing parallelism.

(iii) **Multi-Patch Modelling**
Since isogeometric shell models are typically defined on tensor-product surfaces, any geometry that can be mapped onto a square can be modelled straightforwardly. More complex topologies, e.g., surfaces with holes, require trimming or multi-patch domain approaches. The latter approach includes unstructured splines, where a smooth basis is constructed over a multi-patch domain. In **chapter 7**, unstructured spline constructions for isogeometric analysis are reviewed and compared qualitatively and quantitatively.

(iv) **Result Reproduction**
Aiming for full reproducibility of the results in this dissertation, as well as compatibility with future research, the last part of this dissertation provides background on the open-source software developed in this research. In particular, **chapter 8** provides a background on the design considerations of the software developed along with this dissertation, along with the software perspective on benchmarks presented in other chapters. In addition, every chapter is concluded with a result reproduction section as an appendix, giving instructions on the reproduction of the figures in the respective chapters.

Each part is indicated on the outsides of the pages of the corresponding chapters using the part title and the colour of the thumb indices. Chapters not belonging to a part have black thumb indices and are untitled. After the core of this dissertation, **chapter 9** provides conclusions about the main goal of this dissertation, and **chapter 10** provides directions for future research.

2

Preliminaries

Physics and mathematics play an important role in this dissertation. To provide the reader with background knowledge as well as a mathematically sound notation, this chapter provides mathematical and physical preliminaries. In section 2.1, a brief overview of the mathematical notations used in this dissertation is provided. Section 2.2 provides a background in isogeometric analysis: This section briefly introduces B-splines (section 2.2.1), refinement splines (section 2.2.4) and unstructured splines (section 2.2.5). Thereafter, section 2.3 focusses on isogeometric shell modelling: It reviews the state-of-the-art in isogeometric shell analysis (section 2.3.1), and provides a detailed derivation of the isogeometric Kirchhoff–Love shell formulation (section 2.3.2), which will be recurring in less detail throughout the other chapters. Given a discrete shell model, routines for numerical structural analysis can be defined. This is done in section 2.4 for various analysis types, for example, arc-length methods (section 2.4.4). Lastly, section 2.5 provides a broad overview of the wrinkling mechanics. This overview starts with the essential physical principles behind wrinkling (section 2.5.1), after which a review of experimental investigations related to wrinkling in various contexts is given (section 2.5.2). Since this dissertation presents numerical methods for the analysis of wrinkling, section 2.5.3 provides a review of the state-of-the-art of the numerical analysis of wrinkles and section 2.5.4 provides a brief overview of techniques to minimise wrinkles in membranes, e.g. by optimisation. The chapter ends with concluding remarks; see section 2.6.

2.1 Notations

This dissertation relies on mathematical derivations related to geometric modelling and numerical analysis for mechanical problems. In this section, the mathematical notations used in this dissertation are briefly outlined.

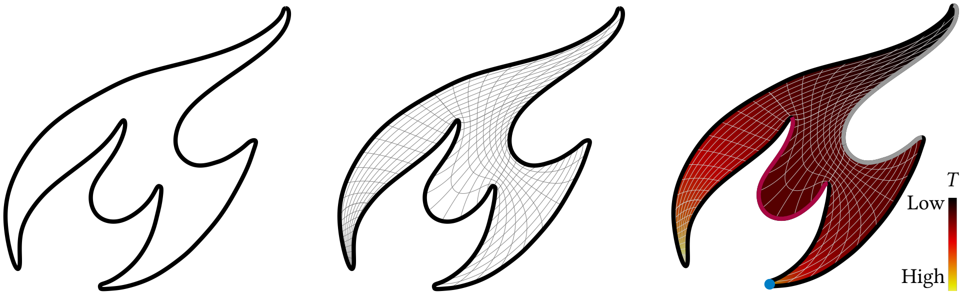
Firstly, scalar quantities are denoted by mostly lower-case Latin or Greek characters, e.g., a, ϕ. Upper-case italic Latin scalars with a subscript or superscript are used to denote elements of a matrix or a tensor, e.g., A_{ij} to denote the elements of a second-order tensor $\mathbf{A}$. As an exception, upper-case Latin scalars can also denote common material or geometric parameters, e.g., E for Young's modulus and L for length. Their definition will follow from the context. Scalar functionals are represented by calligraphic upper-case letters, e.g., $\mathscr{W}_{\text{ext}}$ represents external work. Variations or functional derivatives are typically denoted by a δ in front of the functional, e.g., $\delta_{\boldsymbol{v}}\mathscr{W}_{\text{ext}}$. The optional subscript indicates the variable introduced when taking the variation. Blackboard-bold upper-case letters are used to denote function spaces, e.g., $\mathbb{B}$, $\mathbb{H}$ and $\mathbb{T}$ are spaces of, respectively, B-spline, HB-spline and THB-spline basis functions and $\mathbb{S}$ denotes an arbitrary function space. Lastly, Latin letters used as subscript or superscript are indices and typically take values $1,\dots 3$ and Greek letters used a subscript or superscript typically take values $1,2$.

Vectors and vector fields are denoted by italic and bold Latin or Greek letters. Lower-case quantities typically denote vectors. In addition, vectors with a subscript denote covariant basis vectors and vectors with a superscript denote contravariant basis vectors, e.g., $\boldsymbol{a}_i$ and $\boldsymbol{a}^i$ are covariant and contravariant basis vectors. Vectors decorated with a hat, e.g., $\hat{\boldsymbol{a}}_i$, denote unit vectors with the norm $\|\hat{\boldsymbol{a}}_i\| = 1$. The vector $\boldsymbol{u}$ is always recurring as the displacement vector. Although often omitted, the $\boldsymbol{u}_h$ denotes the discrete displacement vector. An inner product between two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is denoted by $\boldsymbol{a} \cdot \boldsymbol{b}$, with repeated indices denoting summation. A cross-product between the same vectors is denoted by $\boldsymbol{a} \times \boldsymbol{b}$, and the dyadic vector product is denoted by $\boldsymbol{a} \otimes \boldsymbol{b}$.

Second-order tensors are mostly used for continuum mechanics derivations. In this dissertation, second-order tensors are denoted by non-italic upper-case bold Latin letters, e.g., $\mathbf{E}$ for the strain tensor. Matrices are denoted by upper-case italic letters, e.g. K is a stiffness matrix. The double contraction of second-order tensors is denoted by $\mathbf{S} : \mathbf{E}$, resulting in a scalar. In the context of tensors, summation over repeated indices is often used, e.g. $\mathbf{S} : \mathbf{E} = \sum_{i=1}^{3}\sum_{j=1}^{3} S_{ij}E_{ij} = S_{ij}E_{ij}$. Tensors are defined using a matrix of coefficients and a basis, e.g., $\mathbf{E} = E_{ij}\,\boldsymbol{g}^i \otimes \boldsymbol{g}^j$, with the superscript denoting a contravariant basis. Lastly, boldface calligraphic Latin letters indicate fourth-order tensors, e.g., $\boldsymbol{\mathscr{C}} = \mathscr{C}^{ijkl}\,\mathring{\boldsymbol{g}}_i \otimes \mathring{\boldsymbol{g}}_j \otimes \mathring{\boldsymbol{g}}_k \otimes \mathring{\boldsymbol{g}}_l$ denoting the material tensor. The inner product of a fourth-order tensor with a second-order tensor is a second-order tensor, e.g., $\mathbf{S} = \boldsymbol{\mathscr{C}} : \mathbf{E}$, providing coefficients $S_{ij} = \mathscr{C}^{ijkl}E_{kl}$.

2.2 Isogeometric Analysis

Computer-Aided Design (CAD) and Computer-Aided Analysis (CAA) are closely related in the Computer-Aided Engineering (CAE) workflow. Inside the engineering workflow,



(a) CAD-representation of the flame of the TU Delft.

(b) Discretisation $\mathcal{T}(\Omega)$. The mesh is created using the method of Ji *et al.* [287], optimising the position of the surface control points for the "mesh quality", keeping the boundaries fixed.

(c) Solution of the stationary heat equation $\Delta u = f(t)$ on the domain Ω. A non-dimensional temperature of $T = 20\,[-]$ is applied on the blue point in the bottom right, and a non-dimensional temperature of $T = 10\,[-]$ is applied on the red boundary, both via a Dirichlet boundary condition. The grey boundary is insulated via a Neumann boundary condition.

Figure 2.1: A simple pipeline for Isogeometric Analysis (IGA). Figure (a) provides the design of the flame of the *Technische Universiteit Delft* (TU Delft), for example, provided by Computer-Aided Design (CAD). Using this geometry, figure (b) provides a discretisation of the surface of the flame, referred to as a *mesh*. Lastly, figure (c) provides the solution of the heat equation solved on the mesh of the TU Delft flame, as an example of Computer-Aided Engineering (CAE).

CAD typically uses a Graphical User-Interface (GUI), where lines, surfaces, and volumes can be sculptured *in-silico*. Provided a geometry from CAD, CAA, on the other hand, entails the analysis of physical phenomena such as fluid flow, thermodynamics, magnetism, or elasticity, all described by Partial Differential Equations (PDEs), which can be solved by Finite Element Analysis (FEA). Classically, the CAE workflow handles CAD and CAA as two separate entities connected via a mesh: a decomposition of the domain consisting of elements on which the CAA computations are performed. For complex geometries, the mesh is typically an approximation of the original CAD geometry.

Isogeometric Analysis (IGA) [268] aims to facilitate a smooth CAE workflow by bridging the CAD and CAA routines via splines: mathematical objects that are being used in CAD but traditionally not in CAA. Since its introduction in (2005), IGA employs splines as a mathematical basis both for the geometry and for solving the Partial Differential Equations (PDEs) in CAA, following on the idea of having the same (*iso*) geometry in CAD and CAA. As indicated in figure 2.1, a geometry specified in CAD (see figure 2.1a) is naturally discretised into a mesh (see figure 2.1b) such that CAA can be performed (see figure 2.1). This workflow is naturally facilitated by using splines as the basis of FEA.

Since the advent of IGA (2005), many new properties of the method have been discovered, and it has been applied to several problems in computational mechanics. The book of Cottrell *et al.* [120] provides a textbook overview of the method in its early years (2009), and the book of Bazilevs *et al.* [38] (2013) provides an overview of Fluid-Structure Interaction with a focus on Isogeometric Analysis. Besides these books, few reviews on IGA have been written, for instance the review by Nguyen *et al.* [405], by Gupta *et al.* [226] (2023), by Marussig & Hughes [379] (2018), by De Lorenzis *et al.* [143] (2014), and by Kunoth *et al.* [327] (2018), the latter three primarily focusing on trimming, contact, and linear algebra, respectively. Other publications providing complete literature reviews are the works by Buffa *et al.* [78] on the mathematics of adaptive IGA and by Paul *et al.* [433] on domain coupling methods for fourth-order problems.

The continuity of the spline basis functions employed in IGA provides advantages and disadvantages of the method. Among the advantages of IGA are the following points, relevant to or closely related to this dissertation:

- **High accuracy per degree-of-freedom**
  In classic FEA, pure degree elevation (p-refinement) provides a way to increase the speed of convergence of solvers while performing mesh refinements (h-refinements). IGA does not only provide p-refinement and h-refinements but also a procedure to elevate the degree while preserving maximum continuity across elements, so-called k-refinement [268]. The potential of the k-refinement was demonstrated in many works and showed higher accuracy per degree of freedom than classical FEA. Mathematical justification for this effect was provided by Sande *et al.* [488].

- **Accurate eigenvalue approximations**
  In the context of eigenvalue problems, for example, structural vibrations or buckling problems, IGA provides accurate approximations of eigenvalue spectra compared to classic C^0 FEA, thanks to the k-refinement property. That is, a full eigenvalue spectrum from FEA shows spikes and outliers, whereas the optical branches are eliminated, and the full spectrum converges for IGA with high continuity, besides a few outliers. This effect was identified by Cottrell *et al.* [122] and further investigated by Hughes *et al.* [270] and Hughes *et al.* [269], showing that this advantage is not only limited to structural vibrations but holds for elliptic eigenvalue problems in general.

- **Geometrically exact (structural) stability analysis**
  The exact geometry description in IGA due to its relation with CAD not only has the potential to unify CAD and CAA routines, but it can also affect problems sensitive to geometric imperfections. This, as a consequence, eliminates the geometric error, making its application in structural stability problems advantageous [218, 343]. A recent analysis of Oesterle *et al.* [414] indeed shows the significance of geometric exactness for pre-buckling analysis.

- **Seamless control for shape and topology optimisation**
  Another advantage of directly using the spline discretisation from CAD is the inherent concise geometric parametrisation that can be used as design variables for shape optimisation [194, 596]. Compared to FEA, any mesh modification or generation step is eliminated from the optimisation framework, and geometric exactness is maintained. Examples of isogeometric shape optimisation include stiffened shell optimisation [248], auxetic meta-material optimisation [607], optimisation of beam structures [614], or thermal extruders [666]. In addition, splines offer efficient control of level-set [597, 598] or

density [197] fields in topology optimisation, providing natural parametrisation of the topology and the shape for optimisation of auxetic meta-materials [198, 409] or heat conduction systems [275]. The frameworks of isogeometric topology and shape optimisation were recently combined by Cai *et al.* [82].

The geometric exactness and flexibility with respect to continuity provide advantages of IGA over FEA. However, these aspects also have disadvantages:

- **Computational costs of continuity**
  Since higher continuity of basis functions (see section 2.2.1) comes with a larger overlap (i.e., support) between basis functions, the bandwidth of matrices assembled for CAA typically increases, decreasing the performance of direct [108] and iterative [107] solvers. However, by employing multi-grid solvers, systems of equations can be solved more efficiently [249, 250, 517, 518, 532, 565]. In addition, using weighted-quadrature rules [83, 489] or sum factorisation [12], matrices can be assembled faster. Lastly, it is worth noting that the efficiency of k-refinement compared to conventional h and p-refinement strategies generally reduces mesh sizes, hence increasing critical time steps in explicit dynamic simulations drastically, yielding a decrease in overall computational costs [394].

- **Analysis-aware and analysis-suitable geometric modelling**
  The idea behind IGA to bridge CAD and CAA seamlessly by employing the same mathematical basis requires *analysis-aware* geometric modelling when used in practice [106]. Overall, this CAD-CAA-pipeline requires additional geometric pre-processing to achieve seamlessness. For example, CAD volumes are often represented as a Boundary Representation (BRep), i.e., as a set of surfaces surrounding the geometry. To perform CAA, mesh generation and parametrisation are required, e.g., as presented by Hinz *et al.* [245, 246], Ji *et al.* [282, 284, 287], Shamanskiy *et al.* [499] and others. Furthermore, CAD geometries are often represented by boolean geometric operations and trimming, which could result in so-called *non-watertight* geometric models. To make these CAD geometries *analysis-suitable*, non-watertight holes can be filled using T-splines, for example [36]. In addition, small features in CAD models can lead to meshing problems while being irrelevant to the results of the CAA. To overcome this, *de-featuring* of CAD models provides automatic selection of features in CAD, relevant for CAA [76, 77]. Overall, developments towards seamless CAD-CAA modelling have been made for parametric models [206, 239, 260] and for BReps [558, 559] and trimmed geometries [657].

In the remainder of this section, an overview of some spline constructions for isogeometric analysis will be given. Section 2.2.1 provides a brief background on B-splines and their properties. Then, sections 2.2.4 and 2.2.5 provide the basics of refinement splines and unstructured splines, respectively.

2.2.1 Splines

Geometries in computer-aided design and solutions in isogeometric analysis are described using splines. Splines are piece-wise polynomial mathematical functions, in general characterised by being smooth. The origin of splines can be found in shipbuilding, where wooden strips with attached weights were used to draw smooth curves [183]. The mathematical counterpart of these wooden strips with weights is referred to as basis splines (B-splines), coined by Schoenberg [493]. Nowadays, these mathematical objects lay the foundation of modern Computer-Aided Geometric Design (CAGD). In the following, a brief introduction to B-spline bases and geometries is given. Thereafter, a brief background on advanced techniques for refinement and unstructured splines is provided as an introduction for chapter 5 on adaptive meshing and chapter 7 on unstructured splines. For more information on geometric modelling with splines, the reader is referred to the books of De Boor [141], Piegl & Tiller [437], Böhm & Prautzsch [55], and Lyche & Morken [369], among other reference works.

2.2.2 B-Spline Basis

A B-spline object consists of a B-spline basis and a set of control points. The B-spline basis is typically defined using the Cox-De Boor recursion formula [123, 140] from a knot vector. In general, a knot vector is a sequence of non-decreasing real numbers:

$$\Xi = \left[\xi_1, \xi_2, \dots, \xi_{n+p+1}\right], \quad \xi_i \leq \xi_{i+1}, \quad \forall i = 1, 2, \dots, n+p. \tag{2.1}$$

The above can be used to define n B-splines of degree $p \geq 0$. Provided the knot vector Ξ, the Cox-De Boor recursion formula starts by defining basis functions φ_i^0 of degree $p = 0$

$$\varphi_i^0(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

Given the basis functions of degree 0, the basis functions of higher degrees are recursively defined as

$$\varphi_i^p(\xi) = \frac{\xi - \xi_i}{\xi_{i+d} - \xi_i}\varphi_i^{p-1}(\xi) + \frac{\xi_{i+d+1} - \xi}{\xi_{i+d+1} - \xi_{i+1}}\varphi_{i+1}^{p-1}(\xi). \tag{2.3}$$

Here, quantities are defined to be 0 in the case of division by zero [437]. Although the Cox-De Boor recursion formula provides a straightforward definition of B-splines, it is a rather inefficient way of constructing B-splines. Instead, a B-spline basis can be represented using matrices, providing efficient implementation of the evaluation of the B-spline basis using common linear algebra routines [369].

Having defined the B-spline basis, several properties of these functions can be identified [437]. For the analysis of PDEs, the following properties are of interest:

- **Local support**
  The basis functions have *local support*, meaning that the basis function i with degree p

Parts of section 2.2.1 are based on:
[283] Y. Ji, M. Möller & **H. M. Verhelst**, “Design Through Analysis”, (accepted for publication)

is identically zero outside of the domain $[\xi_i, \xi_{i+p+1}]$. Furthermore, within the knot span $[\xi_j, \xi_{j+1}]$, at most $p+1$ basis functions are non-zero, more precisely, only the functions $\varphi_{j-p,p}, \dots, \varphi_{j,p}$. Typically, the non-zero functions at a point ξ are referred to as the *active functions.*

- **Non-negativity**
  From the definition in equation (2.2), it follows that the basis functions of degree 0, $\varphi_{i,0}$, are non-negative over ξ, i.e., $\varphi_{i,0}(\xi) \geq 0$. By induction over the degree p, it can be shown that the non-negativity property holds for all degrees p, i.e., $\varphi_{i,p}(\xi) \geq 0$ holds for all i, p, and ξ.

- **Partition of unity**
  The basis functions possess the partition of unity property, meaning that for all points in the domain $[\xi_i, \xi_{i+1})$ and all i, the sum of the basis functions is unity. Since the functions have local support, this property implies that the sum of the active functions (i.e., the functions $i-p$ up to i) at any $\xi \in [\xi_i, \xi_{i+1})$ is unity:

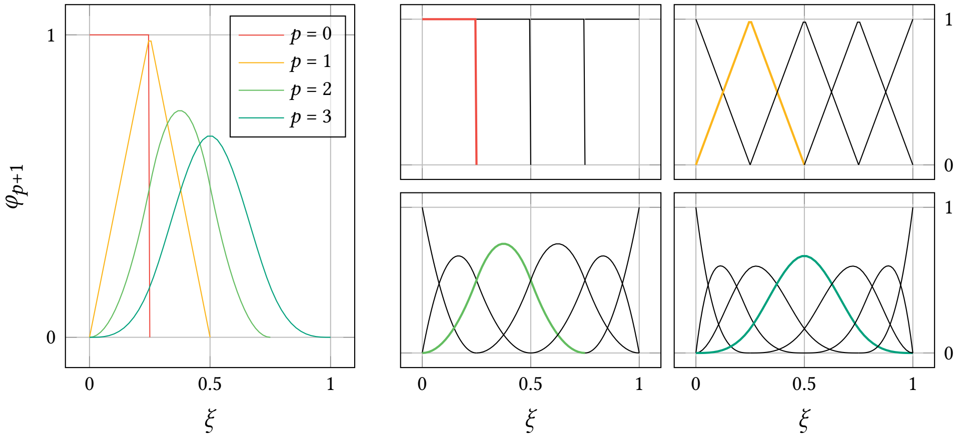
$$\sum_{j=i-p}^{i} \varphi_{j,p}(\xi) = 1. \tag{2.4}$$

- **Regularity**
  All derivatives of $\varphi_{i,p}(\xi)$ with respect to ξ exist in the interior of a knot span (ξ_j, ξ_{j+1}). Moreover, given the multiplicity k of the knot $\xi \in \{\xi_{i+1}, \dots, \xi_{i+p}\}$ with multiplicity k, $\varphi_{i,p}$ is $p-k$ times differentiable at ξ. As a consequence, an increase in the degree increases the continuity, and an increase in the knot multiplicity decreases the continuity.

The combination of the aforementioned properties has implications for discretisations where these basis functions are used. Since the basis functions have local support, a system of equations assembled with this basis is sparse. Furthermore, the non-negativity and partition of unity properties provide positive definiteness of the system matrix and boundedness of errors. Lastly, the arbitrary regularity of the basis provides continuity across elements. The latter property is useful for the modelling of PDEs requiring higher-order derivatives, such as the Kirchhoff–Love shell.

Example 2.2.1 (B-Spline Basis)**.** *Figure 2.2 provides examples of B-spline basis functions given knot vectors with three interior knots on the knot span* $[0,1]$*. The bases have different polynomial degrees p and an interior multiplicity of 1. As a result, the knot vectors are* $\Xi = \{0,0,0.25,0.5,0.75,1,1\}$ *for degree* $p = 1$ *and* $\Xi = \{0,0,0,0.25,0.5,0.75,1,1,1\}$ *for degree* $p = 2$ *onwards.*

A special case of a B-spline basis is the basis with knot vector $\Xi = \{0,\dots,0,1,\dots,1\}$ where the knots 0 and 1 are repeated $p+1$ times, hence its degree is p. This basis is referred to as the *Bernstein basis* and forms the basis of Bézier curves.



(a) Basis function φ_{p+1} for different bases of degree p.

(b) Spline bases for unique knot vector $\Xi = \{0, 0.25, 0.5, 0.75, 1\}$ and degrees $p = 0, 1, 2, 3$, from top-left to bottom-right.

Figure 2.2: B-spline basis functions φ_i on the domain $\xi \in [0, 1]$ with unique knot vector $\Xi = \{0, 0.25, 0.5, 0.75, 1\}$ for degrees $p = 0, 1, 2, 3$. Figure 2.2a provides a canonical representation of the basis functions, i.e., the functions φ_{p+1} are depicted. Figure 2.2b provides all functions of each basis, with the coloured function being the one represented in figure 2.2a.

2.2.3 B-Spline Geometry

In order to represent a geometry, solutions, or other functions using splines, a coefficient $\boldsymbol{b}_i$ is assigned to each basis function φ_i. A spline curve $\boldsymbol{c}(\xi_1) : [0, 1] \longrightarrow \mathbb{R}^d$ in a d-dimensional space is defined by

$$\boldsymbol{c}(\xi_1) = \sum_{i=1}^{N} \varphi_i \boldsymbol{b}_i \tag{2.5}$$

Where $\boldsymbol{b}_i \in \mathbb{R}^d$ denotes the i^{th} control point in the geometric context. The parameter ξ_1 is the first parametric coordinate, typically but not necessarily defined on the unit-interval $[0, 1]$. Given the properties of the basis functions $\varphi_i(\xi)$ mentioned in the previous subsection (see page 10), the properties of a B-spline curve can be formulated [437]:

- **Control point locality**
  Following from the local support of the underlying B-spline basis, control points only have a local effect on the curve. Moving control point $\boldsymbol{b}_i$ only influences the interval $[\xi_i, \xi_{i+p+1})$. This is illustrated in figure 2.3.
- **Convex hull property**
  As a consequence of the partition of unity and the non-negativity of the B-spline basis, a B-spline curve is contained in the *convex hull* of its control polygon. Furthermore, the control polygon interpolates the end points of the curve, i.e., the curve intersects with its control points at its ends.
- **Continuity**
  Due to the regularity of the B-spline basis, a B-spline curve has the same continuity as its basis.

Similar to B-spline curves, a B-spline surface $\boldsymbol{s}(\xi_1,\xi_2) : [0,1]^2 \longrightarrow \mathbb{R}^d$ is defined as

$$\boldsymbol{s}(\xi_1,\xi_2) = \sum_{k=1}^{N} \varphi_k \boldsymbol{b}_k. \tag{2.6}$$

Here, the basis functions $\varphi_k(\xi_1,\xi_2)$ are conveniently numbered basis functions. For tensor-product bases, the basis functions are defined as:

$$\varphi_k(\xi_1,\xi_2) = \varphi_i(\xi_1)\varphi_j(\xi_2), \tag{2.7}$$

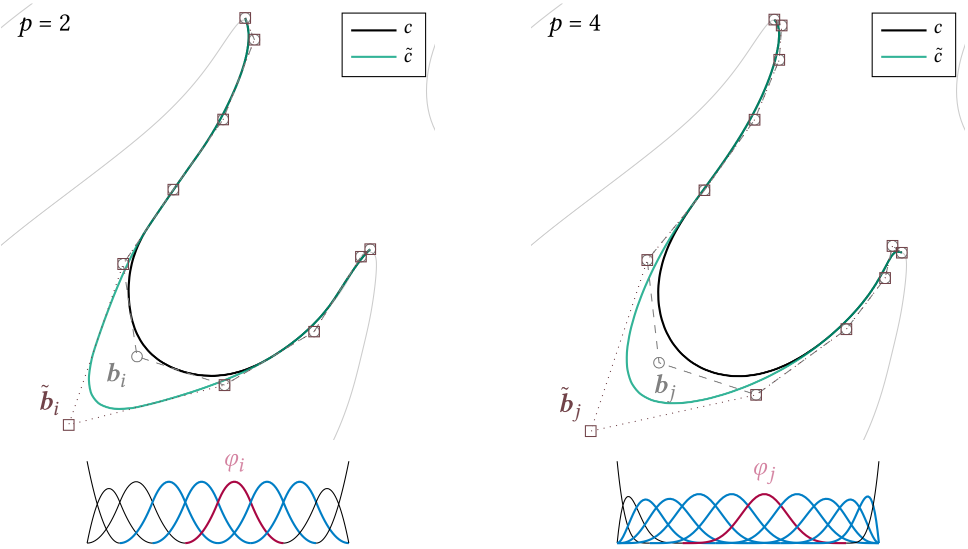
where the index k is, for example, based on a lexicographic ordering of indices i and j. The construction of tensor B-spline geometries with higher domain dimensions, e.g., $\boldsymbol{v}(\xi_1,\xi_2,\xi_3) : [0,1]^3 \longrightarrow \mathbb{R}^3$ for volumes, works in the same way. In general, any parametric domain D can be mapped onto a geometric domain with dimension d as $\boldsymbol{v}(\xi_1,\dots,\xi_D) : [0,1]^D \longrightarrow \mathbb{R}^d$; for example, $(D,d) = (2,3)$ can provide a surface, and $(D,d) = (4,3)$ can provide a time-dependent volume. Although B-spline geometries provide great flexibility in geometric modelling by simply moving control points, they are not capable of modelling exact circular segments, which are commonly used in industrial applications. By weighting every basis function, Non-Uniform Rational B-Splines (NURBSs) provide great flexibility in geometric modelling. Backgrounds on NURBSs can be found in the book of Piegl & Tiller [437].

Example 2.2.2 (B-Spline Curve)**.** *Figure 2.3 provides two examples of B-spline curves. The black curve is an original curve segment from the TU Delft flame in figure 2.1. Using bases with different knot vectors Ξ and degrees p (given below the curve), a B-spline curve is fitted using L_2-fitting, resulting in the red curve c_0. Given curve c, the curve $\tilde{c}$ is constructed by moving one control point $\boldsymbol{b}_i$, denoted by the red, to a new location $\tilde{\boldsymbol{b}}_i$. As a consequence, the locality of the B-spline curve can be observed by the influence of the movement of $\boldsymbol{b}_i$ on the shape of $\tilde{c}$.*

Example 2.2.3 (B-Spline Surfaces)**.** *Figure 2.4 shows a B-spline surface spanned by two outer curve segments of the TU Delft flame and two interface curves inside the domain. The B-spline surface has parametric dimension $q = 2$ and geometric domain dimension $d = 3$. The mesh of the surface is plotted, represented by iso-lines for the knot values of the knot vectors of the surfaces in both directions ξ_1 and ξ_2. The knot vectors are $\Xi_1 = \{0, 1/9, 2/9, \dots, 1\}$ and $\Xi_2 = \{0, 1/11, 2/11, \dots, 1\}$ with degrees $p_1 = p_2 = 2$. The control net is omitted for the sake of representation.*

Furthermore, figure 2.5 shows a multi-patch surface of the full TU Delft flame (see figure 2.1) represented by 5 patches, ω_k, $k = 1,\dots,5$. The multi-patch geometry Ω is obtained by splitting the boundary curves into multiple segments. Afterwards, the method of Ji et al. [287] is used for the planar parametrisation of all patches together, with a C^1 constraint.

Since the parametric domain is typically defined as a simple domain, e.g., a line for a curve or a square for a surface, only simple geometries can be represented with tensor-product splines. These simple geometries are referred to as patches, and by joining different patches, complex geometries can be constructed through so-called *multi-patches.*



(a) $\Xi = \{0, 0, 0, 1/8, 2/8, \ldots, 7/8, 1, 1, 1\}$, $p = 2$

(b) $\Xi = \{0, 0, 0, 0, 0, 1/8, 2/8, \ldots, 7/8, 1, 1, 1, 1, 1\}$, $p = 4$

Figure 2.3: Two B-spline curves with knot vector Ξ and degree p fitted through a segment of the TU Delft flame from figure 2.1 in red. The B-Spline curves c are obtained by L_2 fitting, and the curves $\tilde{c}$ are obtained by moving control points $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ to locations $\tilde{\boldsymbol{b}}_i$ and $\tilde{\boldsymbol{b}}_j$. The basis functions coloured in red indicate the basis functions φ_i and φ_j corresponding to control points $\boldsymbol{b}_i$, $\boldsymbol{b}_j$, $\tilde{\boldsymbol{b}}_i$, and $\tilde{\boldsymbol{b}}_j$, and the basis functions coloured in blue correspond to the basis functions that share the support with φ_i and φ_j.

Unlike the interior of a domain formed by a B-spline geometry, multi-patches do not have high continuity across patch interfaces. Using straight-forward degree of freedom matching, only C^0 continuity can be achieved. Using so-called *unstructured splines*, see section 2.2.5, continuity across patch interfaces can be (partially) recovered.

The concept of refinement of spline geometries by refining the basis is a relevant feature when aiming for accurate geometric modelling or simulation. Given a knot vector Ξ, a new knot ξ_k can be inserted anywhere in the interior of the knot vector to modify the basis without changing the parametric domain. If ξ_k is inserted with a value equal to an existing interior knot ξ_i, the *knot multiplicity* of ξ_k is increased, decreasing local continuity of the basis. If ξ_k is inserted in an interval $[\xi_i, \xi_{i+1}]$, a new basis function is added within the interval, and the interval is *refined*. Furthermore, the degree of the basis (and geometry) can be increased by increasing the multiplicity of the end knots. In general, such *knot insertion* routines can be performed while preserving the geometry. On the other hand, *knot removal* routines can be used to remove knot intervals (i.e., coarsen elements), to reduce multiplicity, or to decrease the degree. Knot removal is therefore not geometry-preserving. For more details about knot insertion, knot removal, and degree modifications, the reader is again referred to the book of Piegl & Tiller [437], but also to the original works of Böhm [52–55, 444] on this topic.

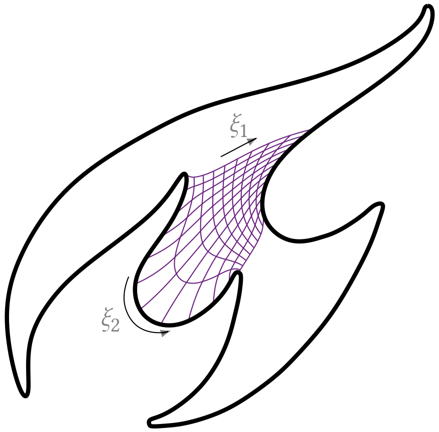


Figure 2.4: Part of the TU Delft flame (see figure 2.1) represented by a tensor-product B-spline. The basis has a degree $p = 2$ knot vector $\Xi = \{0, 1/9, 2/9, \dots, 1\}$ in ξ_1-direction and a degree $p = 2$ knot vector $\Xi = \{0, 1/11, 2/11, \dots, 1\}$ in ξ_2-direction.

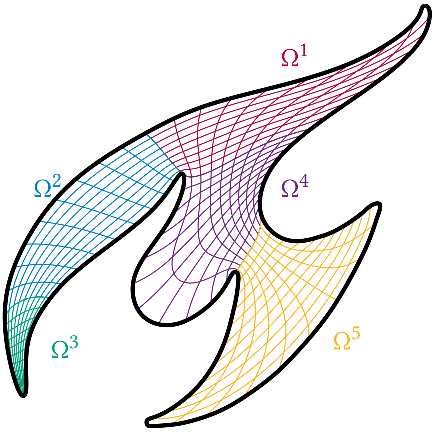


Figure 2.5: The TU Delft flame (see figure 2.1) represented by 5 tensor B-spline patches Ω^i, $i = 1, \dots, 5$.

2.2.4 Refinement Splines

B-spline bases are formulated through a tensor product of bases in each dimension. When refining one element using tensor-product techniques, the refinement also extends to multiple elements in a singular direction (as illustrated in figure 2.7a). This approach establishes a quasi-local strategy for adaptive refinement, which, while advantageous for enhancing the efficiency of multi-scale simulations, offers only partial enhancements. To obtain more localised refinement of spline bases, the scientific literature introduces several constructions. Popular constructions are hierarchical B-splines (HB-splines) [189], truncated hierarchical B-splines (THB-splines) [202, 203], T-splines and its variations [299, 350–352, 360, 498, 616], polynomial splines over hierarchical T-meshes (PHT-splines) [149], locally refined (LR) splines [158], and splines over unstructured meshes (U-splines) [562]. It is worth noting that “U-splines” here should not be confused with the unstructured splines discussed in the upcoming subsection on unstructured splines (see section 2.2.5). This review primarily focuses on (T)HB splines, while additional insights into alternative spline constructions can be found in the aforementioned references.

The construction of the truncated (hierarchical) B-spline basis $(\mathbb{H})\ \mathbb{T}$ is defined recursively as given by Giannelli *et al.* [203]:

1. Initialise $\mathbb{T}^0 = \mathbb{H}^0 = \{\varphi \in \mathbb{B}^0 \,:\, \operatorname{supp} \varphi \neq \varnothing\}$, with the superscript denoting level 0, $\mathbb{B}^0$ a tensor B-spline basis on level 0, and φ a basis function with non-empty support from basis.

2. Recursively define $\mathbb{T}^{\ell+1} = \mathbb{T}_A^{\ell+1} \cup \mathbb{T}_B^{\ell+1}$ or $\mathbb{H}^{\ell+1} = \mathbb{H}_A^{\ell+1} \cup \mathbb{H}_B^{\ell+1}$ for $\ell = 0, \dots, N-2$ with N the maximum level. The truncated basis $\mathbb{T}_A^{\ell+1}$ is defined as

$$\mathbb{T}_A^{\ell+1} = \{\operatorname{trunc}^{\ell+1} \tau \,:\, \tau \in \mathbb{T}^{\ell} \wedge \operatorname{supp} \tau \not\subseteq \Omega^{\ell+1}\}$$

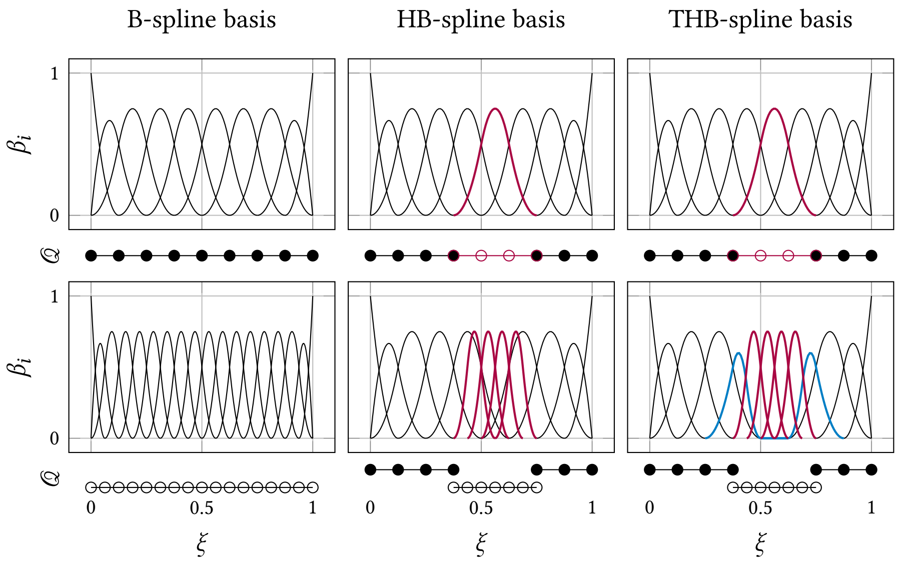


Figure 2.6: Principles of refinement for different spline bases. The top plots represent the basis on level 0, optionally with (to be) refined functions given in red color. The bottom plots illustrate the refined bases: uniform refinement (left), hence level 1; HB-refinement (middle); and THB-refinement (right) with truncated basis functions in blue color. The line $\mathcal{Q}$ represents the elements of the basis. The unrefined unique knot vector in all cases is $\Xi = \{0, 1/8, 2/8, \dots, 7/8, 1\}$, and the degree of the basis is 2. All bases are generated with the open-source IGA library G+Smo [294].

and the hierarchical basis $\mathbb{H}_A^{\ell+1}$

$$\mathbb{H}_A^{\ell+1} = \{\varphi \in \mathbb{H}^\ell \;:\; \operatorname{supp}\beta \not\subseteq \Omega^{\ell+1}\}.$$

Furthermore, the basis $\mathbb{T}_B^{\ell+1} = \mathbb{H}_B^{\ell+1}$ is given by

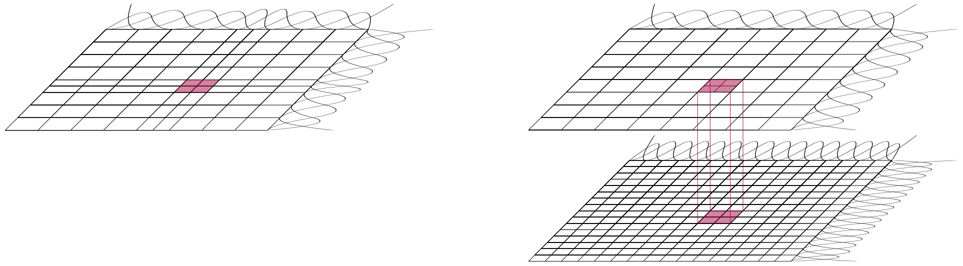
$$\mathbb{H}_B^{\ell+1} = \{\varphi \in \mathbb{B}^{\ell+1} \;:\; \operatorname{supp}\beta \subseteq \Omega^{\ell+1}\},$$

With $\Omega^{\ell+1} \subseteq \Omega^\ell$ nested domains, $\mathbb{B}^\ell$ the B-spline basis on level ℓ and $\operatorname{trunc}^\ell \tau$ the truncation of τ with respect to $\mathbb{B}^{\ell+1}$ and $\Omega^{\ell+1}$.

3. Then the final THB-spline basis is defined as $\mathbb{T} = \mathbb{T}^{N-1}$ and the final HB-spline basis is defined as $\mathbb{H} = \mathbb{H}^{N-1}$

Figure 2.6 illustrates the concept of local refinement employing B-splines, employing (truncated) hierarchical B-splines (HB- and THB-splines, respectively). In the top row of this figure, an initial uniform degree 2 B-spline basis with a uniform knot vector $\Xi = \{0, 1/8, 2/8, ..., 7/8, 1\}$ is presented. The bottom row presents uniform refinement as well as (T)HB refinements applied to the designated functions.

The potential of refinement splines compared to knot insertions for local is illustrated in Figure 2.7. Upon executing a knot insertion within a tensor B-spline basis to refine a



(a) Refinement of a tensor-product B-spline basis. To diadically refine the basis in an element, the knot 0.5625 is inserted in both knot vectors.

(b) Refinement of a (T)HB-spline basis. To refine the basis in an element, the basis functions with support on this element from the finer level (below) are inserted in the original basis (above) following the procedure from figure 2.6.

Figure 2.7: Refinement of a 2-dimensional tensor B-spline basis (a) and a (T)HB-spline basis (b) for a marked element with corners $(0.5, 0.5)$ and $(0.625, 0.625)$. The original B-spline basis has degree 2 and a unique knot vector $\Xi = \{0, 1/8, 2/8, \ldots, 7/8, 1\}$ in both directions.

designated element, the refinement inherently introduces refinement of other elements in the knot line (see figure 2.7a). Conversely, within hierarchical splines, the basis functions are localised in their insertion, thereby introducing degrees of freedom solely within the specified element, as illustrated in figure 2.7b.

2.2.5 Unstructured Splines

Although B-spline bases provide high continuity within a patch, only C^0 continuity (i.e., continuity of the values of the basis functions but not of their derivatives) can easily be obtained on the interfaces; hence, it would be beneficial to preserve at least part of the $C^{\mathrm{d}-1}$ continuity of higher-order B-splines when coupling multiple patches to a multi-patch object. To achieve higher continuity over the interface of multi-patch bases, so-called *unstructured splines* can be constructed. Unstructured splines are splines with higher-order smoothness over patch interfaces.

In the case of 1-dimensional bases, the concept of patch smoothing is trivial but illustrative for higher dimensions. The concept of interface smoothing is illustrated in figure 2.8 and can be interpreted as a construction where basis functions $\varphi \in \mathbb{S}^1$ of a spline space $\mathbb{S}^1$ with interface smoothness 1 are represented by a linear combination of functions $\psi \in \mathbb{S}^0$ from a space $\mathbb{S}^0$. For example, the basis function φ_k is represented by all basis functions ψ_l weighted with coefficient A_{kl}:

$$\varphi_k = A_{kl}\psi_l, \tag{2.8}$$

Or, given a vector of evaluations of ψ_l, $\boldsymbol{\psi}$, the vector of evaluations of φ_k, $\boldsymbol{\varphi}$, is transformed by a matrix A

$$\boldsymbol{\varphi} = A\boldsymbol{\psi}. \tag{2.9}$$

Example 2.2.4 (1-Dimensional interface smoothing)**.** *Consider two bases of degree 2 with unique knot vectors* $\Xi^{(1)} = \{0, 1/8, 2/8, ..., 7/8, 1\}$ *and* $\Xi^{(2)} = \{1, 9/8, 10/8, ..., 15/8, 2\}$*. The basis functions with non-zero value on the interface are denoted by* $\psi^{(1)}_{10}$ *and* $\psi^{(2)}_{11}$*, and the basis*

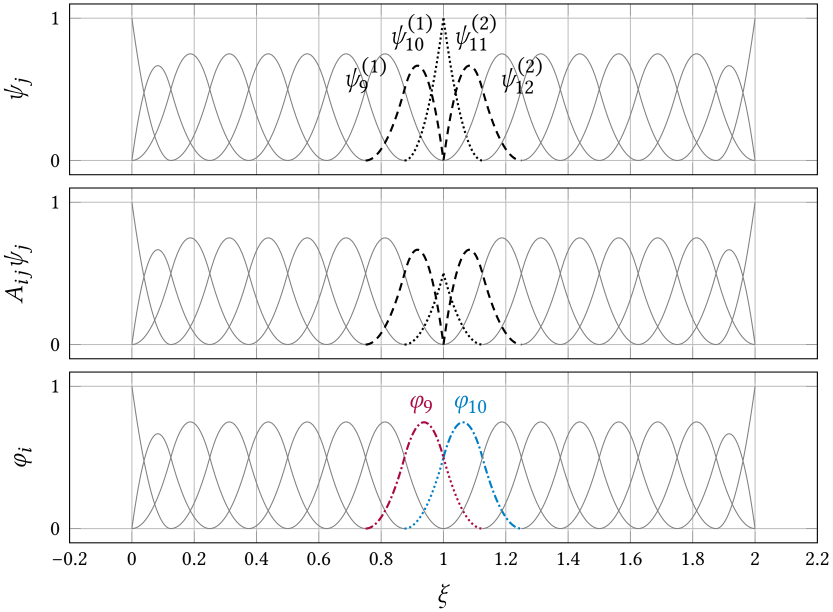


Figure 2.8: The concept of interface smoothing between two bases of degree 2 with unique knot vector $\Xi^{(1)} = \{0, 1/8, 2/8, ..., 7/8, 1\}$ (left) and $\Xi^{(2)} = \{1, 9/8, 10/8, ..., 15/8, 2\}$ (right). The top figure plots the basis functions of the two bases, where the dashed functions have zero value but non-zero derivative on the interface. The dotted functions are non-zero on the boundary. The middle row presents scaled basis functions $A_{ij}\psi_j$. Here, all functions are scaled by a factor of 1, except the dotted functions, which are scaled by a factor of 1/2. The bottom row presents the basis $\varphi_i = A_{ij}\psi_j$, where the sum is evaluated over the repeated index j. The red and blue functions, respectively φ_9 and φ_{10}, are constructed by taking the sum of the dashed and dotted functions in their support on the one side (resulting in the dash-dotted line) and taking the dotted line on the other side.

function with non-zero derivative on the interface are denoted by, $\psi_9^{(1)}$ *and* $\psi_{12}^{(2)}$*. When scaling* $\psi_{10}^{(1)}$ *and* $\psi_1^{(2)}$ *by a factor of* 1/2 *and all the other functions with a factor of 1 in their support,* C^1 *smoothing over the interface is achieved, illustrated in Figure 2.8. The local bases in this example have* 10 *basis functions each; hence,* 20 *in total. The global basis consists of* 18 *functions; hence, the matrix* A *is a* 18×20 *matrix. For interface basis function* φ_9*, the non-zero coefficients in matrix* A *are* $A_{9,9} = 1$*,* $A_{9,10} = A_{9,11} = 1/2$ *and for interface basis function* φ_{10}*, the non-zero coefficients are* $A_{10,12} = 1$*,* $A_{10,10} = A_{9,11} = 1/2$.

In higher dimensions, interface smoothing, as illustrated in figure 2.8, can be performed to construct interface basis functions. However, the increased parametric dimension (see, e.g., figure 2.5) introduces vertices where the smoothing of basis functions is non-trivial. Spline constructions that provide smoothing mappings like the matrix A are referred to as *unstructured splines*, providing bases with higher smoothness than C^0 over patch interfaces and vertices. Chapter 7 of this dissertation provides a comparison of unstructured spline approaches for multi-patch surfaces.

Examples of unstructured spline constructions include the D-Patch [464, 569], the

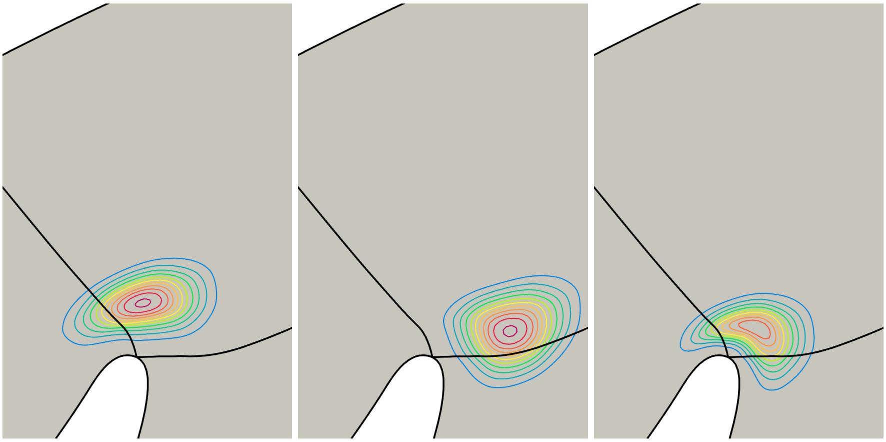


Figure 2.9: A sample of Almost-C^1 [534] basis functions on the corner of the TU Delft flame where 3 patches join (see figure 2.1a). The basis functions are represented by contour lines with an interval of 0.05, starting from 0.

Almost-C^1 construction [534], the Approximate C^1 basis [618, 619], the Analysis-Suitable G^1 construction [109, 181, 182], polar spline constructions [568] and constructions based on subdivision surfaces [30, 378].

Example 2.2.5 (Unstructured splines)**.** *As shown in figure 2.1a, the flame of TU delft can be represented by a multi-patch geometry using five patches. Given this multi-patch segmentation, interface smoothing can be applied across all interfaces. The geometry has only one extraordinary vertex: the boundary vertex between Ω^1, Ω^2 and Ω^4, located on a domain boundary. On this vertex, special spline constructions to ensure C^1 continuity need to be applied. In figure 2.9, three basis functions following from the Almost-C^1 construction around the vertex are plotted.*

2.3 Isogeometric Analysis for Thin-Walled Structures

Structural mechanics plays an essential role in various engineering disciplines. Thin-walled structures, characterised as assemblies of (curved) plates with a small thickness compared to their in-plane dimensions, are of particular interest in various engineering disciplines (e.g., automotive, offshore, or aeronautical engineering) because of their low weight per unit volume. The interest in thin-walled structures calls for the numerical analysis of such structures in design and analysis. In this section, a background on isogeometric analysis for thin-walled structures is provided. Section 2.3.1 provides a literature overview of developments in isogeometric structural analysis, mainly focussed on shells. Section 2.3.2 provides a detailed derivation of the isogeometric Kirchhoff–Love shell model, supporting the remaining chapters in this dissertation.

2.3.1 Isogeometric Membrane, Shell and Solid Models

As the field of computational methods for thin-walled structures is extensive, overviews of models are provided in many works, e.g., in the work of Bischoff *et al.* [49]. For the sake of brevity, this section highlights the most important theories for the modelling of thin-walled structures. A comprehensive literature overview on the isogeometric Kirchhoff–Love shell formulation is provided first, since it is most relevant for this dissertation. Thereafter, a brief overview of membrane, solid and other shell formulations is given for the sake of completeness, and the reader is referred to the general overview of shell models provided by Bischoff *et al.* [49], and specifically on isogeometric shell formulations by Loibl [361]. Furthermore, the book by Hughes [267] provides a good overview of shell elements in Finite Element Analysis.

Kirchhoff–Love Shell Formulation

In finite element methods, the Kirchhoff–Love shell theory is assumed in, for example, the SHEBA shell [16, 49]. Since splines provide higher-degree element continuity, the Kirchhoff–Love shell theory is relatively straightforwardly implemented in the isogeometric Kirchhoff–Love shell [319]. Here, the curvature of the surface can directly be used to model the bending stiffness, eliminating the need for degrees of freedom for rotations and hence yielding rotation-free shells. Isogeometric Kirchhoff–Love shells with rotational degrees of freedom are referred to as mixed formulations [326, 453, 454], but these elements have a higher number of degrees of freedom compared to the formulation from [319] employing C^1 continuity. Since its introduction in 2009, many contributions have been made to the isogeometric Kirchhoff–Love shell formulations. In the following, developments and applications of the isogeometric Kirchhoff–Love shell relevant to this dissertation are highlighted.

- **Constitutive Relations**
  In order to ensure a wide application of the Kirchhoff–Love shell, several constitutive relations have been developed. For composite materials, laminate theory is used in the work of Herrema *et al.* [238], but more advanced models, including plasticity and damage, were developed by Alaydin *et al.* [4]. Furthermore, an anisotropic material model with arbitrarily curved fibre directions was developed by Wu *et al.* [635]. For hyperelastic models and biological tissues, formulations for non-linear elastic (i.e., hyperelastic models) constitutive relations have been developed by Tepole *et al.* [555] for a selection of biological material models, by Kiendl *et al.* [320] for general hyperelastic models, and as a part of this dissertation, [587] for stretch-based hyperelastic models. Furthermore, Roohbakhshan & Sauer [480] provides strategies to prevent through-thickness integration, and Sauer & Duong [490] provides formulations for solid-like and liquid-like shells. Beyond elasticity, formulations for modelling plasticity are provided by Ambati *et al.* [8], and formulations for modelling brittle and ductile fracture using phase-field models are respectively provided by Proserpio *et al.* [446] and Proserpio *et al.* [447], whereas Nguyen-Thanh *et al.* [411] provides extended isogeometric analysis (XIGA) for the Kirchhoff–Love shell to model fracture. Furthermore, Behzadinasab *et al.* [40] presents formulations for peridynamic modelling using isogeometric Kirchhoff–Love shells. Lastly, homogenisation techniques for modelling micro-structures on a macro-structural level for Kirchhoff–Love shells are given by Do *et al.* [157].

- **Complex Geometries**
  The isogeometric Kirchhoff–Love shell presented by Kiendl *et al.* [319] requires continuous derivatives throughout the domain, i.e., C^1 continuity. Many works have focused on the modelling of complex geometries since multi-patch domains are C^0 by default. Since chapter 7 is devoted to the modelling of complex domains for isogeometric analysis using smooth basis constructions, a detailed overview of techniques for complex domain modelling for isogeometric analysis is given there. In brief, multiple approaches for complex domain modelling can be distinguished: trimming or immersed methods, variational coupling, and coupling through unstructured splines. In the case of trimming or immersed methods, the geometry is defined by a geometry and a set of trimming curves or surfaces defining the actual domain and the parts of the domain to be omitted. Trimmed isogeometric Kirchhoff–Love shells are considered in [116], among others, and patch-wise quadrature rules are provided by Loibl *et al.* [362]. Variational coupling is an approach where multi-patch domains are coupled by adding equations to the variational problem to solve. For the Kirchhoff–Love shell, this can be done through penalty methods [5, 119, 238, 342, 657] or Nitsche's method [47], or via mortar methods [247, 386]. These methods can also be used to couple trimmed domains [118, 225, 448]. Lastly, unstructured spline bases provide smooth splines over multi-patch geometries. These constructions have not been applied extensively to isogeometric Kirchhoff–Love shell problems [89, 182, 463, 534]. In the work of Liu *et al.* [355], an unstructured spline construction is used on a local level, and weak coupling is used to couple these unstructured spline patches. Besides complex domain modelling for shells, coupling approaches for shells with other elements to model realistic geometries have also been studied. For example, Raknes *et al.* [458] develop an isogeometric cable model coupled to Kirchhoff–Love shells; Guo & Ruess [222] and Liu *et al.* [357] develop models where Kirchhoff–Love shells are coupled with solid-like patches; and Benson *et al.* [45] develop a shell formulation that utilises the Kirchhoff–Love shell formulation or the Reissner–Mindlin shell formulation where applicable. In the work of Xiao *et al.* [640], lattice skin structures are modelled using isogeometric Kirchhoff–Love shells for the skin and isogeometric truss elements for the lattice structure. Another approach for the modelling of complex structures entails the embedding of stiffeners in shells. This includes the works of Bauer *et al.* [35], Hirschler *et al.* [248], Wang *et al.* [612], and Wang *et al.* [602].

- **Mesh Adaptivity**
  Adaptive meshing routines for isogeometric Kirchhoff–Love shells to localise degrees of freedom for computational efficiency have been developed in recent years. Antolin *et al.* [13] developed an *a-posteriori* error estimator for linear Kirchhoff–Love shells and Kirchhoff plates based on a so-called *bubble space* and using hierarchical B-splines, applied to trimmed domains in [116], benefiting from element refinement near small geometric features [117]. Furthermore, Casquero *et al.* [88] used T-splines for local mesh refinement in isogeometric Kirchhoff–Love shells. In the context of fracture modelling in Kirchhoff–Love shells, mesh adaptivity plays an important role since fracture is a local phenomenon. In the works of Proserpio *et al.* [446] and Paul *et al.* [433], mesh adaptivity is applied to the problems of brittle fracture, and in the work of Proserpio *et al.* [447], it is applied to ductile fracture in isogeometric Kirchhoff–Love shells, all using

LR NURBS. In chapter 5 of this dissertation, a goal-adaptive refinement procedure is presented, allowing to refine the isogeometric Kirchhoff–Love shell using a pre-defined goal function, e.g., based on displacements, stresses, or strains.

- **Locking and Computational Efficiency**
  Important developments for the isogeometric Kirchhoff–Love shell have been made to reduce membrane locking effects and improve computational efficiency. Locking is the phenomenon where pure displacement shell formulations underestimate deformations or give stress oscillations, particularly for low thickness compared to the element size. In the works of Oesterle *et al.* [415] and Loibl [361], a good overview of locking in isogeometric shell formulations is given. Although the locking effects in isogeometric Kirchhoff–Love shells are generally small due to high inter-element continuity [344], investigations of the locking effects have been performed by Leonetti *et al.* [344], showing that patch-wise reduced integration [289] can eliminate locking effects in Kirchhoff–Love shells. In the work by Loibl *et al.* [362], such patch-quadrature schemes are extended for trimmed Kirchhoff–Love shells. As an alternative to the patch-quadrature scheme, Zou *et al.* [663] propose a new quadrature scheme to alleviate locking in Kirchhoff–Love shells, removing oscillations in displacement and stress fields. Another important development for the acceleration of the convergence of displacement-based shell elements is the Mixed Integration Point (MIP) method [340, 372]. Here, stress and displacement are decoupled in the constitutive relations, improving the Newton convergence and allowing larger step sizes in quasi-static computations. The MIP has been applied to Kirchhoff–Love shells by Leonetti *et al.* [340, 344] and to solid shells by Leonetti *et al.* [341].

- **Applications**
  Besides theoretical developments on the applicability, accuracy, and performance of the isogeometric Kirchhoff–Love shell, its application is wide-spread. For example, the isogeometric Kirchhoff–Love shell is successfully applied to the modelling of heart valves and arteries [27, 258, 259, 290, 297, 320, 634, 635, 644, 646], towards enabling patient-specific cardiology. Furthermore, the shell formulation is increasingly used for crash simulations of cars [334, 335, 501, 503, 504], since typical mesh sizes of Kirchhoff–Love shells can be an order of magnitude larger than their finite element counterpart, increasing the time step size in explicit dynamics simulations and therefore increasing the overall efficiency of the simulation [394]. Lastly, more related to the contents of this dissertation is the application of isogeometric Kirchhoff–Love shells for cloth simulation [507] and the modelling of buckling imperfections [220] in instability problems.

Membrane Formulation

Membrane elements are not characterised as shell elements because the bending stiffness of a membrane is assumed to be zero. Therefore, they in fact represent a limit case of shells where the thickness approaches zero. In that case, the only stiffness that is considered in membranes is the membrane stiffness; hence, the curvature terms are omitted in equation (2.36). As a consequence, the derivation of the isogeometric membrane element formulation can be done similarly to the Kirchhoff–Love shell derivation, omitting bending contributions. Since the bending stiffness contribution in the isogeometric Kirchhoff–Love shell imposes the C^1 continuity constraint of this shell formulation, this constraint vanishes for isogeometric membrane elements. Therefore, membrane elements provide a light-weight alternative to other shell formulations, especially for explicit dynamics computations Chen *et al.* [97]. In the context of wrinkling, membrane models are also finding their use, e.g., in combination with Tension-Field Theory [402] (see equation (2.109)). However, as bending energy plays a vital role in wrinkling physics (see section 2.5.1), membranes are restrictive when explicit modelling of wrinkling amplitudes is needed.

Reissner–Mindlin Shell Formulation

The Reissner–Mindlin shell formulation allows for transverse shear deformations, contrary to the Kirchhoff–Love shell formulation. Therefore, this shell formulation is applicable for applications where the slenderness of the structure is lower than for the Kirchhoff–Love shell. However, due to the transverse shear contribution, transverse shear locking effects become more pronounced as the thickness decreases, thus when the slenderness increases [49]. Within isogeometric analysis, the first isogeometric Reissner–Mindlin shell was presented by Benson *et al.* [44], showing accuracy and robustness for higher degrees without an attempt to investigate reduced quadrature to reduce shear locking. Later, the Reissner–Mindlin formulation was extended towards a rotation-free formulation, assuming higher-order continuity of the basis [43]. A method to compute the director vectors exactly was proposed by Dornisch *et al.* [162], and Dornisch *et al.* [163] presented a robust method using continuous rotations. In terms of locking, the aforementioned works employed the potential of higher-order bases to minimise locking effects, and classical methods to reduce locking effects are used by Adam *et al.* [1] and Caseiro *et al.* [86] by using reduced integration and the Assumed Natural Strain method, respectively. By employing a new mixed formulation, Zou *et al.* [665] alleviate shear and membrane locking. Alternatively, Beirão da Veiga *et al.* [41] and Kikis *et al.* [321] use adjusted approximation spaces for rotations and displacements to treat transverse shear locking in Reissner–Mindlin plates and shells, respectively. Furthermore, Oesterle *et al.* [415] developed a locking-free Reissner–Mindlin shell and extended it to large deformations [413] by using a technique closely related to the hierarchical shell formulations by Echter *et al.* [175]. Alternative formulations for the isogeometric Reissner–Mindlin shell include one based on tangential differential calculus Schöllhammer & Fries [494] and a collocation approach by Kiendl *et al.* [317]. In order to facilitate multi-patch analysis, Dornisch & Klinkel [161] developed a methodology to model shells with kinks, and Dornisch *et al.* [165] developed mortar-based coupling for Reissner–Mindlin shells. Regarding constutitive modelling, the Reissner–Mindlin shell has been applied to plasticity problems [44] as well as peridynamic modelling for crack simulations [502, 637–639].

Solid–Shell Formulations

Solid–shell formulations are shell formulations that do not only include transverse shear, like the Reissner–Mindlin shell, but also include load-induced thickness changes [361]. In the isogeometric analysis framework, several solid-like shell elements have been developed, which are reviewed in this paragraph. First, Bouclier *et al.* [59] developed a solid–shell element with a modified interpolation scheme of stresses and strains through-thickness and based on a so-called $\bar{B}$ projection to deal with locking effects, extended to geometrically non-linear analysis by Bouclier *et al.* [60]. In the same period, Hosseini *et al.* [256] provided a solid shell with a linear Lagrange shape function through-thickness, where the through-thickness deformations are quadratically dependent on the displacement field, based on the model of Parisch [430], making the element insensitive to shear and membrane locking. This formulation was extended to a geometrically non-linear model using B-splines through thickness by Hosseini *et al.* [257], providing a locking-free element for length over thickness ratios up to 400. By using B-splines through thickness, laminate layers are easily modelled through thickness by using knot insertion. Another isogeometric solid–shell element was presented by Caseiro *et al.* [86], employing the Assumed Natural Strain concept to alleviate shear and membrane locking effects. Their formulation was extended by Caseiro *et al.* [87] for large deformations and elasto-plasticity. Lastly, Leonetti *et al.* [341] presented a solid–shell model for geometrically non-linear analyses with linear through-thickness interpolation of geometry and displacements in a total Lagrangian setting, used for Koiter stability analysis in Leonetti *et al.* [343]. By using reduced integration, locking effects are minimised, and using the Mixed Integration Point method, Newton convergence is improved [341]. In the work of Liguori *et al.* [353], this model was used for thermoelastic modelling of solid shells.

Hierarchical Shell Formulations

Echter *et al.* [175] presented a family of isogeometric shell formulations called the *hierarchic shells*. The hierarchical shell family consists of a 3-parameter (3p) Kirchhoff–Love formulation, which is extended to a 5-parameter (5p) Reissner–Mindlin shell and a 7-parameter (7p) solid-like shell by adding displacement-based director vectors to the Kirchhoff–Love director vector (see equation (2.13)). An alternative approach to the 5p-hierarchic shell is presented by Oesterle *et al.* [415], where the director vector is defined using displacement degrees of freedom instead of hierarchic rotations. Both approaches are extended to a geometrically non-linear framework by Oesterle *et al.* [416].

Solids

Besides shell and membrane formulations, where geometries are considered a surface and the thickness is a parameter, the mechanics of structures can also be performed by using general elasticity equations. In that case, a thin-walled geometry is modelled as a volume instead of a surface. Compared to shell models, solid models solved using Galerkin methods require a 3D meshed domain, making volumetric parametrisation methods more relevant for solids than for shells, e.g., see the work of Shamanskiy *et al.* [499]. On the other hand, solid models can be solved using the Boundary Element Method (BEM), reducing the problem to the boundary domain rather than the interior domain, as shown in the works of Chasapi *et al.* [94] and Nguyen *et al.* [410], among others.

2.3.2 The Isogeometric Kirchhoff–Love Shell

Since the isogeometric Kirchhoff–Love shell formulation will play an important role in the remainder of this dissertation, a derivation of the theory is provided in this section.

Although the derivation can be found in the contributions of this dissertation in chapters 3 and 5, it is in short form. Therefore, for the sake of completeness, the following derivation aims to provide extra detail to the derivation of the isogeometric Kirchhoff–Love shell presented in the following chapters of this dissertation.

Geometry

Kirchhoff–Love shell theory describes the deformation of surfaces. Hence, let $\boldsymbol{s}(\theta^1, \theta^2) : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$ be a surface. For this surface, the *covariant basis vector* $\boldsymbol{a}_\alpha$ is defined by taking the derivatives of the surface with respect to the parametric coordinate θ^α, i.e.

$$\boldsymbol{a}_\alpha = \frac{\partial \boldsymbol{s}}{\partial \theta^\alpha}, \quad \alpha = 1, 2, \tag{2.10}$$

Using the covariant basis, the *covariant metric tensor*, or *first fundamental form*, is defined by

$$a_{\alpha\beta} = \boldsymbol{a}_\alpha \cdot \boldsymbol{a}_\beta. \tag{2.11}$$

Using the first fundamental form of the surface, the *contravariant metric tensor* is defined using the inverse of $[a_{\alpha\beta}]$, as $a^{\alpha\beta} = [a_{\alpha\beta}]^{-1}$. Furthermore, the *contravariant basis* $\boldsymbol{a}^\alpha$ is defined by

$$\boldsymbol{a}^\alpha = a^{\alpha\beta} \boldsymbol{a}_\beta. \tag{2.12}$$

Using the covariant basis vectors from equation (2.10), the *surface unit normal vector* is defined by

$$\hat{\boldsymbol{a}}_3 = \frac{\boldsymbol{a}_1 \times \boldsymbol{a}_2}{|\boldsymbol{a}_1 \times \boldsymbol{a}_2|}. \tag{2.13}$$

In addition to the surface gradients, the curvature of the surface is a quantity of interest, typically related to bending. In the present derivation of the Kirchhoff–Love shell theory, the curvature is included via the *second fundamental form*, as

$$b_{\alpha\beta} = \hat{\boldsymbol{a}}_3 \cdot \boldsymbol{a}_{\alpha,\beta} = -\hat{\boldsymbol{a}}_{3,\beta} \cdot \boldsymbol{a}_\alpha \tag{2.14}$$

Here, $\boldsymbol{a}_{\alpha,\beta}$ denotes the second derivative or Hessian of the surface, and $\hat{\boldsymbol{a}}_{3,\alpha}$ denotes the derivative of the unit normal vector with respect to the parameter θ^α. Via Weingarten's formula [617], it holds that $\hat{\boldsymbol{a}}_{3,\alpha} = -b_\alpha^\beta \boldsymbol{a}_\beta$ with $b_\alpha^\beta = a^{\alpha\gamma} b_{\gamma\beta}$. Since the second fundamental form $b_{\alpha\beta}$ depends on the surface Hessian $\boldsymbol{a}_{\alpha,\beta}$, second derivatives of the surface description $\boldsymbol{s}(\theta^1, \theta^2)$ are required.

Assuming the *Kirchhoff hypothesis*, i.e., no shear of the shell cross-section, orthogonality of orthogonal vectors after deformation, and no thickness change, the Kirchhoff–Love shell formulation assumes that any point in the shell can be described by its position on the surface $\boldsymbol{s}(\theta^1, \theta^2)$ and its position along the surface normal $\boldsymbol{a}_3$ as

$$\boldsymbol{x}(\theta^1, \theta^2, \theta^3) = \boldsymbol{s}(\theta^1, \theta^2) + \theta^3 \hat{\boldsymbol{a}}_3 \tag{2.15}$$

The derivatives of the coordinate system $\boldsymbol{x}$ with respect to the parametric coordinates θ^i $(i = 1, 2, 3)$ provide the full basis of the coordinate system used for the Kirchhoff–Love shell element. The covariant basis of $\boldsymbol{x}$ is given by

$$\begin{aligned} \boldsymbol{g}_\alpha &= \frac{\partial \boldsymbol{x}}{\partial \theta^\alpha} = \boldsymbol{a}_\alpha + \theta^3 \boldsymbol{a}_{3,\alpha}, \\ \boldsymbol{g}_3 &= \frac{\partial \boldsymbol{x}}{\partial \theta^\alpha} = \hat{\boldsymbol{a}}_3. \end{aligned} \tag{2.16}$$

Following from the covariant basis, the first fundamental form $g_{ij} = \boldsymbol{g}_\alpha \cdot \boldsymbol{g}_\beta$ is defined using the first and second fundamental forms as

$$\begin{aligned} g_{\alpha\beta} &= (\boldsymbol{a}_\alpha + \theta^3 \boldsymbol{a}_{3,\alpha}) \cdot (\boldsymbol{a}_\beta + \theta^3 \boldsymbol{a}_{3,\beta}), \\ &= a_{\alpha\beta} - 2\theta^3 b_{\alpha\beta} + (\theta^3)^2 \boldsymbol{a}_\alpha \cdot \boldsymbol{a}_\beta, \\ g_{33} &= 1, \\ g_{i3} = g_{3i} &= 0. \end{aligned} \tag{2.17}$$

The last term, quadratic in θ^3, can be neglected for thin or moderately thick shells [49]. The contravariant metric tensor g^{ij} and the contravariant basis $\boldsymbol{g}^i$ are derived, like for the surface $\boldsymbol{s}$. Using the shell coordinate system equation (2.15) and the covariant basis equation (2.17), the kinematic equation for the Kirchhoff–Love shell can be derived.

Kinematic Equation

The kinematic equation relates shell displacements to strains. Let $\mathring{\boldsymbol{x}}(\theta^1, \theta^2, \theta^3)$ denote the undeformed configuration of the shell, and let $\boldsymbol{x}(\theta^1, \theta^2, \theta^3)$ denote the deformed configuration of the shell. Then, the displacement $\boldsymbol{u}(\theta^1, \theta^2, \theta^3)$ of a material point is defined as

$$\boldsymbol{u}(\theta^1, \theta^2, \theta^3) = \boldsymbol{x}(\theta^1, \theta^2, \theta^3) - \mathring{\boldsymbol{x}}(\theta^1, \theta^2, \theta^3) \tag{2.18}$$

Additionally, the *deformation gradient* $\mathbf{F}$ is a tensor that maps between the undeformed basis $\mathring{\boldsymbol{g}}_i$ and the deformed basis $\boldsymbol{g}_i$, meaning that an infinitesimal line element $\mathrm{d}\mathring{\boldsymbol{x}}$ in the deformed configuration is defined as $\mathrm{d}\boldsymbol{x} = \mathbf{F} \cdot \mathrm{d}\mathring{\boldsymbol{x}}$ in the undeformed configuration [24]. Accordingly, the deformation gradient $\mathbf{F}$ is defined as

$$\mathbf{F} = \boldsymbol{g}_i \otimes \mathring{\boldsymbol{g}}^i. \tag{2.19}$$

Indeed, the deformation gradient maps $\mathring{\boldsymbol{g}}_i$ onto $\boldsymbol{g}_i$ via $\boldsymbol{g}_i = \mathbf{F}\mathring{\boldsymbol{g}}$ [24]. Using the deformation gradient, the Green-Lagrange strain tensor $\mathbf{E} = E_{ij}\, \mathring{\boldsymbol{g}}^i \otimes \mathring{\boldsymbol{g}}^j$ relates the non-linear relation between deformations and strains

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^\top \mathbf{F} - \mathbf{I}\right) = \frac{1}{2}(\mathbf{C} - \mathbf{I}), \tag{2.20}$$

where $\mathbf{C}$ is the *deformation tensor.* Using the definition of the deformation gradient and the fact that the identity tensor $\mathbf{I}$ is equal to the metric tensor G^{ij} on $\mathring{\boldsymbol{g}}^i \otimes \mathring{\boldsymbol{g}}^j$ yields

$$E_{ij} = \frac{1}{2}\left(g_{ij} - G_{ij}\right) \tag{2.21}$$

Using the definition of the metric tensor from equation (2.17), the coefficients of the strain tensor can be expressed in terms of the surface metric and the curvature:

$$\begin{aligned} E_{\alpha\beta} &= \frac{1}{2}\left(a_{\alpha\beta} - 2\theta^3 b_{\alpha\beta} - \mathring{a}_{\alpha\beta} + 2\theta^3 \mathring{b}_{\alpha\beta}\right) \\ &= \frac{1}{2}\left(a_{\alpha\beta} - \mathring{a}_{\alpha\beta}\right) + \theta^3\left(\mathring{b}_{\alpha\beta} - b_{\alpha\beta}\right) = \varepsilon_{\alpha\beta} + \kappa_{\alpha\beta}. \end{aligned} \tag{2.22}$$

The shear strains E_{i3}, E_{3i}, and the normal strain E_{33} vanish because of the orthogonality and unity of the basis vector $\mathbf{g}_3$ in deformed and undeformed configurations. This indeed shows that the shell formulation following from the assumed coordinate system in equation (2.15) yields a formulation free of cross-sectional shear and thickness change. Hence, the shell can be represented by its mid-surface only, and the strain tensor is represented with respect to the first two components of the basis, i.e., $\mathbf{E} = E_{\alpha\beta}\, \mathring{\mathbf{g}}^{\alpha} \otimes \mathring{\mathbf{g}}^{\beta}$. The coefficients ε_{ij} and $\kappa_{\alpha\beta}$ relate to the *membrane strain tensor* $\boldsymbol{\varepsilon} = \varepsilon_{\alpha\beta}\, \mathring{\mathbf{g}}^{\alpha} \otimes \mathring{\mathbf{g}}^{\beta}$ and the *bending strain tensor* $\boldsymbol{\kappa} = \kappa_{\alpha\beta}\, \mathring{\mathbf{g}}^{\alpha} \otimes \mathring{\mathbf{g}}^{\beta}$.

Constitutive Relation

In general continuum mechanics, the *second Piola-Kirchhoff* stress tensor $\mathrm{S} = S^{ij}\, \mathring{\mathbf{g}}_i \otimes \mathring{\mathbf{g}}_j$ is energetically conjugate to the Green-Langrange strain tensor $\mathbf{E} = E_{ij}\, \mathring{\mathbf{g}}^i \otimes \mathring{\mathbf{g}}^j$[24]. For a 3D continuum, the coefficients of the second Piola-Kirchhoff stress tensor can be defined using a *strain energy density function* Ψ:

$$S^{ij} = 2\frac{\partial \Psi}{\partial C_{ij}} \tag{2.23}$$

In addition, the *material tensor* or *elasticity tensor* $\boldsymbol{\mathcal{C}} = \mathcal{C}^{ijkl}\, \mathring{\mathbf{g}}_i \otimes \mathring{\mathbf{g}}_j \otimes \mathring{\mathbf{g}}_k \otimes \mathring{\mathbf{g}}_l$ is a fourth-order tensor that relates the total differentials of the second Piola-Kirchhoff stress S and the Green-Lagrange strain $\mathbf{E}$. Its coefficients are defined by

$$\mathcal{C}^{ijkl} = \frac{\partial S_{ij}}{\partial E^{kl}} = 4\frac{\partial^2 \Psi}{\partial C_{ij} \partial C_{kl}}, \tag{2.24}$$

Such that the coefficients of the total differential of the second Piola-Kirchhoff stress tensor, $\mathrm{d}S^{ij}$, relates to the total differential of the Green-Lagrange strain tensor, $\mathrm{d}E_{ij}$ via

$$\mathrm{d}S^{ij} = \mathcal{C}^{ijkl}\, \mathrm{d}E_{ij} \tag{2.25}$$

For linear elastic materials, stress and strain are linearly dependent, such that $\boldsymbol{\mathcal{C}}$ has constant coefficients according to equation (2.24). Therefore, the following identity is valid for linear materials:

$$S^{ij} = \mathcal{C}^{ijkl} E_{kl}. \tag{2.26}$$

Furthermore, assuming small strains, the through thickness deformation is neglected and $C_{33} = g_{33} = 1$ holds. The latter allows to use 2D constitutive models. However, when strains are large, for example in hyperelastic material models, the plane-stress assumption that

$S^{33} = 0$ is typically violated [320], hence $C_{33} \neq 1$. To use the in-plane components of the stress tensor $S^{\alpha\beta}$ in the Kirchhoff–Love shell model, *static condensation* of the material tensor $\boldsymbol{\mathscr{C}}$ needs to be performed to satisfy the plane stress condition. The formulations for the hyperelastic stress and material tensors for Kirchhoff–Love shells are provided in [320]. Chapter 3 of this dissertation elaborates more on hyperelastic formulations for Kirchhoff–Love shells [587].

Variational Formulation

The variational formulation for the Kirchhoff–Love shell is derived based on *Hamilton's Principle*, which states that

$$\mathscr{H}(\boldsymbol{u},\dot{\boldsymbol{u}}) = \int_{\tau_1}^{\tau_2} \mathscr{T}(\dot{\boldsymbol{u}}) - \mathscr{W}^{\text{int}}(\boldsymbol{u}) + \mathscr{W}^{\text{ext}}(\boldsymbol{u})\,\mathrm{d}\tau\,, \tag{2.27}$$

must be stationary for the displacement field $\boldsymbol{u}$ and velocity field $\dot{\boldsymbol{u}}$. Here, τ_1 and τ_2 are the beginning and the end of a time interval with $\tau \in [\tau_1, \tau_2]$, $\mathscr{T}(\dot{\boldsymbol{u}})$ is the kinetic energy of the system, $\mathscr{W}^{\text{int}}(\boldsymbol{u})$ is the internal potential energy of the system, and $\mathscr{W}^{\text{ext}}(\boldsymbol{u})$ is the external work acting on the system. To find the extremum of equation (2.27), its variation $\delta_{\boldsymbol{v}}\mathscr{H}(\boldsymbol{u},\dot{\boldsymbol{u}},\boldsymbol{v},\dot{\boldsymbol{v}})$ with respect to $\boldsymbol{u}$ and $\dot{\boldsymbol{u}}$ must be zero. This implies that

$$\delta_{\boldsymbol{v}}\mathscr{H}(\boldsymbol{u},\dot{\boldsymbol{u}},\boldsymbol{v},\dot{\boldsymbol{v}}) = \int_{\tau_1}^{\tau_2} \delta_{\boldsymbol{v}}\mathscr{T}(\dot{\boldsymbol{u}},\dot{\boldsymbol{v}}) - \delta_{\boldsymbol{v}}\mathscr{W}^{\text{int}}(\boldsymbol{u},\boldsymbol{v}) + \delta_{\boldsymbol{v}}\mathscr{W}^{\text{ext}}(\boldsymbol{u},\boldsymbol{v})\,\mathrm{d}\tau = 0 \tag{2.28}$$

The kinetic energy $\mathscr{T}(\dot{\boldsymbol{u}})$ and its variation $\delta\mathscr{T}(\dot{\boldsymbol{u}},\dot{\boldsymbol{v}})$ are expressed in terms of the body velocity $\dot{\boldsymbol{x}}$, which depends on the time-derivative of the displacements $\dot{\boldsymbol{u}}$, having variation $\dot{\boldsymbol{v}}$:

$$\mathscr{T}(\dot{\boldsymbol{x}}) = \frac{1}{2}\int_{\Omega^\star} \rho\dot{\boldsymbol{u}}\cdot\dot{\boldsymbol{u}}\,\mathrm{d}\Omega^\star\,, \quad \delta_{\boldsymbol{v}}\mathscr{T}(\dot{\boldsymbol{u}},\dot{\boldsymbol{v}}) = \int_{\Omega^\star} \rho\dot{\boldsymbol{u}}\cdot\dot{\boldsymbol{v}}\,\mathrm{d}\Omega^\star\,, \tag{2.29}$$

Where $\Omega^\star$ is the volume of the body. Applying partial integration with respect to the temporal domain on the kinetic energy term and using (i) the fact that the virtual velocities $\dot{\boldsymbol{v}}$ are zero on the time-domain boundary τ_1 and τ_2, and (ii) the fact that undeformed geometry $\mathring{\boldsymbol{x}}$ is time-independent, hence $\ddot{\mathring{\boldsymbol{x}}} = \ddot{\boldsymbol{u}}$ [208], equation (2.28) can be written as:

$$\delta_{\boldsymbol{v}}\mathscr{H}(\boldsymbol{u},\ddot{\boldsymbol{u}},\boldsymbol{v}) = \int_{\tau_1}^{\tau_2} \delta_{\boldsymbol{v}}\mathscr{T}(\ddot{\boldsymbol{u}},\boldsymbol{v}) - \delta_{\boldsymbol{v}}\mathscr{W}^{\text{int}}(\boldsymbol{u},\boldsymbol{v}) + \delta_{\boldsymbol{v}}\mathscr{W}^{\text{ext}}(\boldsymbol{u},\boldsymbol{v})\,\mathrm{d}\tau = 0 \tag{2.30}$$

Since this must hold for all functions $\boldsymbol{v}$, the integrand must be equal to zero, i.e.

$$\delta_{\boldsymbol{v}}\mathscr{T}(\ddot{\boldsymbol{u}},\boldsymbol{v}) - \delta_{\boldsymbol{v}}\mathscr{W}^{\text{int}}(\boldsymbol{u},\boldsymbol{v}) + \delta_{\boldsymbol{v}}\mathscr{W}^{\text{ext}}(\boldsymbol{u},\boldsymbol{v}) = 0. \tag{2.31}$$

The result of equation (2.31) is similar to the principle of virtual work as used in the derivation of the Kirchhoff–Love shell model in [319], with an additional term for the kinetic energy. In the sequel, the kinetic energy is omitted unless specified otherwise.

Since the variation of the potential energy functional, $\delta_{\boldsymbol{v}}\mathscr{W}(\boldsymbol{u},\boldsymbol{v}) = \delta_{\boldsymbol{v}}\mathscr{W}^{\text{ext}}(\boldsymbol{u},\boldsymbol{v}) - \delta_{\boldsymbol{v}}\mathscr{W}^{\text{int}}(\boldsymbol{u},\boldsymbol{v})$ can be non-linear, the displacements $\boldsymbol{u}$ can be found using the Newton-Raphson method by solving

$$\delta_{\boldsymbol{v}}\mathscr{W} + \delta^2_{\boldsymbol{v}\boldsymbol{w}}\mathscr{W}\Delta\boldsymbol{u} = 0, \tag{2.32}$$

where $\delta_{\boldsymbol{v}}\mathscr{W} = \delta\mathscr{W}(\boldsymbol{u},\boldsymbol{v})$ and $\delta^2_{\boldsymbol{v}\boldsymbol{w}}\mathscr{W} = \delta^2\mathscr{W}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ is the second variation of the energy in the system using virtual displacements $\boldsymbol{v}$ and $\boldsymbol{w}$. The term $\Delta\boldsymbol{u}$ is the incremental update of the displacements.

The equation for the external virtual work is rather straightforward. Given the body force vector $\boldsymbol{f}$, the boundary force vector $\boldsymbol{g}$, and the point loads $\boldsymbol{p}_i$, $i = 1,\dots,N_p$ are independent of the deformation field $\boldsymbol{u}$, the first variation of the external work $\mathscr{W}^{\text{ext}}$ simply yields

$$\begin{aligned}\delta_{\boldsymbol{v}}\mathscr{W}^{\text{ext}} &= \int_{\Omega^\star} \boldsymbol{f}\cdot\boldsymbol{v}\,\mathrm{d}\Omega^\star + \int_{\partial\Omega^\star} \boldsymbol{g}\cdot\boldsymbol{v}\,\mathrm{d}\Gamma + \sum_{i=1}^{N_p}\boldsymbol{p}_i\cdot\boldsymbol{v}\\ &= \int_\tau\int_\Omega \boldsymbol{f}\cdot\boldsymbol{v}\,\mathrm{d}\Omega\,\mathrm{d}\theta^3 + \int_\tau\int_{\partial\Omega} \boldsymbol{g}\cdot\boldsymbol{v}\,\mathrm{d}\Gamma\,\mathrm{d}\theta^3 + \sum_{i=1}^{N_p}\boldsymbol{p}_i\cdot\boldsymbol{v}\end{aligned} \tag{2.33}$$

Where $\Omega^\star = \theta^3\times\Omega$ is the volume of the body with θ^3 the thickness domain $\theta^3 = [-t/2, t/2]$ of the shell and Ω the surface domain. For the internal virtual work, the first variation with respect to the displacements $\boldsymbol{u}$ is given by:

$$\delta_{\boldsymbol{v}}\mathscr{W}^{\text{int}} = \int_{\Omega^\star} \mathrm{S} : \delta_{\boldsymbol{v}}\mathrm{E}\,\mathrm{d}\Omega^\star = \int_\tau\int_\Omega \mathrm{S} : \delta_{\boldsymbol{v}}\mathrm{E}\,\mathrm{d}\Omega = \int_\Omega \mathrm{N} : \delta_{\boldsymbol{v}}\boldsymbol{\varepsilon} + \mathbf{M} : \delta_{\boldsymbol{v}}\boldsymbol{\kappa}\,\mathrm{d}\Omega \tag{2.34}$$

Here, the definition of the strain tensors $\boldsymbol{\varepsilon}$ and $\boldsymbol{\kappa}$ from equation (2.22) is used, and the *membrane force tensor* N and the bending moment tensor $\mathbf{M}$ are defined as moments of the stress tensor through thickness:

$$\begin{aligned}\mathrm{N} &= \int_\tau \mathrm{S}\,\mathrm{d}\theta^3,\\ \mathbf{M} &= \int_\tau \theta^3\mathrm{S}\,\mathrm{d}\theta^3.\end{aligned} \tag{2.35}$$

The second variation of the energy of the system, required for solving the non-linear system of equations using the Newton-Raphson iterations (see equation (2.32)), solely depends on the second variation of the internal energy, assuming deformation-independent body forces[1]. Taking the variation of the internal energy with respect to $\boldsymbol{u}$, the second variation becomes:

$$\delta^2_{\boldsymbol{v}\boldsymbol{w}}\mathscr{W}(\boldsymbol{u}) = \int_\Omega \delta_{\boldsymbol{w}}\mathrm{N} : \delta_{\boldsymbol{v}}\boldsymbol{\varepsilon} + \mathrm{N} : \delta^2_{\boldsymbol{v}\boldsymbol{w}}\boldsymbol{\varepsilon} + \delta_{\boldsymbol{w}}\mathbf{M} : \delta_{\boldsymbol{v}}\boldsymbol{\kappa} + \mathbf{M} : \delta^2_{\boldsymbol{v}\boldsymbol{w}}\boldsymbol{\kappa}\,\mathrm{d}\Omega. \tag{2.36}$$

The variations of N and $\mathbf{M}$ can be obtained using the total differential of S, δS (see equation (2.25)). First, since

$$\delta\mathrm{E} = \delta\big(\boldsymbol{\varepsilon} + \theta^3\boldsymbol{\kappa}\big) = \delta\boldsymbol{\varepsilon} + \delta\big(\theta^3\boldsymbol{\kappa}\big) = \delta\boldsymbol{\varepsilon} + \boldsymbol{\kappa}\delta\theta^3 + \theta^3\delta\boldsymbol{\kappa}, \tag{2.37}$$

[1] The case with deformation-dependent body forces, e.g., a follower-pressure $\boldsymbol{f} = p\hat{\boldsymbol{a}}_3$, is discussed in chapter 4.

And using the total differential of the strain, $\delta\mathbf{E}$, and integrating $\delta\mathrm{S}$ through the thickness, the total differentials of $\mathbf{N}$ and $\mathbf{M}$ are obtained:

$$
\begin{aligned}
\delta\mathrm{N} &= \int_{\theta^3} \delta\mathrm{S}\,\mathrm{d}\theta^3 &&= \int_{\theta^3} \left(\boldsymbol{\mathcal{C}} : \delta\boldsymbol{\varepsilon} + \theta^3\boldsymbol{\mathcal{C}} : \delta\boldsymbol{\kappa} + \delta\theta^3\boldsymbol{\mathcal{C}} : \boldsymbol{\kappa}\right)\mathrm{d}\theta^3 \\
\delta\mathbf{M} &= \int_{\theta^3} \theta^3\delta\mathrm{S}\,\mathrm{d}\theta^3 &&= \int_{\theta^3} \left(\theta^3\boldsymbol{\mathcal{C}} : \delta\boldsymbol{\varepsilon} + \theta^3\boldsymbol{\mathcal{C}} : \delta\boldsymbol{\kappa} + \delta\theta^3\boldsymbol{\mathcal{C}} : \boldsymbol{\kappa}\right)\mathrm{d}\theta^3
\end{aligned}
\tag{2.38}
$$

From the first and second variations of the internal energy, respectively equation (2.34) and equation (2.36), it can be seen that the first and second variations of the membrane strain and bending strain tensors, together with the first variation of the membrane force and bending moment tensors, are needed.

For non-linear time-dependent problems, the second variation of the kinetic energy $\mathcal{T}(\boldsymbol{u})$ provides the inertia of the system. Following from the first variation, $\delta_{\boldsymbol{v}}\mathcal{T}(\boldsymbol{u},\boldsymbol{v})$ in equation (2.29), the second variation of the kinetic energy simply becomes:

$$
\delta_{\boldsymbol{uv}}\mathcal{T}(\boldsymbol{u},\boldsymbol{v},\ddot{\boldsymbol{w}}) = \int_{\Omega} t\rho\ddot{\boldsymbol{w}}\cdot\boldsymbol{v} + \boldsymbol{\mathcal{O}}(t^3)\,\mathrm{d}\Omega
\tag{2.39}
$$

Given that the density ρ is constant over the thickness. Furthermore, given the coordinate system definition in equation (2.15), equation (2.39) only includes translational inertia, since rotational inertia is neglected due to the small thickness assumption of the Kirchhoff–Love shell [208]. Lastly, from equation (2.39), it can be seen that the second variation of the kinetic energy is independent of the solution field $\boldsymbol{u}$, making the operator linear.

Lastly, the boundary conditions of the shell problem are partially incorporated as Neumann conditions through the function $\boldsymbol{g}$ in equation (2.33). The Dirichlet boundary conditions can be imposed strongly by selecting a suitable function space for $\boldsymbol{u}$ or weakly using, for example, penalty methods [238].

Discretisation

The principle of virtual work derived in equation (2.31) is valid for any variation of the unknown displacement field $\boldsymbol{u}(\theta^1,\theta^2,\theta^3)$. In order to discretise the principle of virtual work, it is assumed that the undeformed and deformed configurations $\mathring{\boldsymbol{x}}$ and $\boldsymbol{x}$, respectively, are represented by a finite sum of basis functions $\varphi_k(\theta^1,\theta^2)$ weighted by coefficients $\mathring{\boldsymbol{x}}_k^h$ and $\boldsymbol{x}_k^h$, i.e.

$$
\begin{aligned}
\mathring{\boldsymbol{x}}^h(\theta^1,\theta^2) &= \sum_k \varphi_k(\theta^1,\theta^2)\mathring{\boldsymbol{x}}_k^h, \\
\boldsymbol{x}^h(\theta^1,\theta^2) &= \sum_k \varphi_k(\theta^1,\theta^2)\boldsymbol{x}_k^h.
\end{aligned}
\tag{2.40}
$$

Here, the superscript h indicates discrete approximations of $\mathring{\boldsymbol{x}}$ or $\boldsymbol{x}$, and the index k indicates the k^{th} component of this representation. Since the displacement field $\boldsymbol{u}$ is defined as the difference between $\mathring{\boldsymbol{x}}$ and $\boldsymbol{x}$, it can similarly be expressed as a discrete field $\boldsymbol{u}^h$, and the variations in the principle of virtual work are represented by virtual displacements $\boldsymbol{u}_k^h$. As a consequence, all variations in the virtual work equation are represented by derivatives with respect to components of the virtual nodal displacements $\boldsymbol{u}_k^h$. In the following, all quantities are referred to in the discrete setting, hence the superscript h is omitted.

In the sequel, r denotes the global index of the degree of freedom, and u_r represents a component of one of the nodal displacement vectors. For the sake of brevity, the shorthand notation $(\cdot)_{,r} = \frac{\partial (\cdot)}{\partial u_r}$ is used to represent derivatives with respect to u_r. Using equation (2.18), the variation of the deformed configuration is

$$\boldsymbol{x}_{,r} = \sum_k \left(\mathring{\boldsymbol{x}}_{k,r} + \boldsymbol{u}_{k,r} \right) = \sum_k \varphi_k \boldsymbol{u}_{k,r} = \boldsymbol{u}_{,r} \tag{2.41}$$

The last equality follows from the fact that the undeformed configuration is trivially independent of the deformation field $\boldsymbol{u}$. Similarly, the derivatives of the covariant basis vectors $\boldsymbol{a}_\alpha$ of the discrete deformed configuration $\boldsymbol{x}$, see equation (2.10), are:

$$\boldsymbol{a}_{\alpha,r} = \left(\frac{\partial \boldsymbol{x}_k}{\partial \theta^\alpha} \right)_{,r} = \sum_k \frac{\partial \varphi_k}{\partial \theta^\alpha} \boldsymbol{u}_{k,r} \tag{2.42}$$

As a consequence, the variation of the surface metric tensor of the deformed configuration, $a_{\alpha\beta}$ (see equation (2.11)), becomes

$$a_{\alpha\beta,r} = \left(\boldsymbol{a}_\alpha \cdot \boldsymbol{a}_\beta \right)_r = \boldsymbol{a}_{\alpha,r} \cdot \boldsymbol{a}_\beta + \boldsymbol{a}_\alpha \cdot \boldsymbol{a}_{\beta,r} \tag{2.43}$$

Since the undeformed configuration is invariant to the deformation field $\boldsymbol{u}$, the first variation of the membrane strain tensor $\boldsymbol{\varepsilon}$ from equation (2.22) becomes

$$\varepsilon_{\alpha\beta,r} = \frac{1}{2} a_{\alpha\beta,r} \tag{2.44}$$

Similarly, the second variation of the deformed configuration, the deformed surface metric tensor, and the membrane strain can be derived. Starting with the first variation of the deformed configuration from equation (2.41), the second variation becomes

$$\boldsymbol{x}_{,rs} = \sum_k \varphi_k \boldsymbol{u}_{k,rs} = 0. \tag{2.45}$$

The second variation of $\boldsymbol{u}_k$ is zero since the components of these nodal weights are linear in u_r. Similarly, $\boldsymbol{a}_{\alpha,rs} = 0$. As a consequence, the second variation of the surface metric tensor in the deformed configuration, $a_{\alpha\beta}$, becomes

$$a_{\alpha\beta,rs} = \boldsymbol{a}_{\alpha,rs} \cdot \boldsymbol{a}_\beta + \boldsymbol{a}_{\alpha,r} \cdot \boldsymbol{a}_{\beta,s} + \boldsymbol{a}_{\alpha,s} \cdot \boldsymbol{a}_{\beta,s} + \boldsymbol{a}_\alpha \cdot \boldsymbol{a}_{\beta,rs}, = \boldsymbol{a}_{\alpha,r} \cdot \boldsymbol{a}_{\beta,s} + \boldsymbol{a}_{\alpha,s} \cdot \boldsymbol{a}_{\beta,s}. \tag{2.46}$$

Again, since the undeformed configuration is invariant to the deformation field $\boldsymbol{u}$, the second variation of the membrane strain tensor becomes

$$\varepsilon_{\alpha\beta,rs} = \frac{1}{2} a_{\alpha\beta,rs} \tag{2.47}$$

To derive the variations of the curvature tensor, the variations of the second fundamental form $b_{\alpha\beta}$ are needed, hence requiring variations of $\boldsymbol{a}_{\alpha,\beta}$ and $\hat{\boldsymbol{a}}_3$; see equation (2.14). Firstly, the variation of $\boldsymbol{a}_{\alpha,\beta}$ with respect to u_r is

$$\boldsymbol{a}_{(\alpha,\beta),r} = \left(\frac{\partial^2 \boldsymbol{x}}{\partial\theta^\alpha\partial\theta^\beta}\right)_{,r} = \sum_k \frac{\partial^2\varphi_k}{\partial\theta^\alpha\partial\theta^\beta}\boldsymbol{u}_{k,r} \tag{2.48}$$

Furthermore, using $\boldsymbol{a}_3 = \boldsymbol{a}_1 \times \boldsymbol{a}_2$, the variation of the unit normal vector $\hat{\boldsymbol{a}}_3$ is

$$\hat{\boldsymbol{a}}_{3,r} = \left(\frac{\boldsymbol{a}_3}{|\boldsymbol{a}_3|}\right)_{,r} = \frac{|\boldsymbol{a}_3|\boldsymbol{a}_{3,r} - \boldsymbol{a}_3\big(|\boldsymbol{a}_3|\big)_{,r}}{|\boldsymbol{a}_3|^2} \tag{2.49}$$

Here, the variation of the non-unit normal vector $\boldsymbol{a}_3$ is obtained by

$$\boldsymbol{a}_{3,r} = \boldsymbol{a}_{1,r} \times \boldsymbol{a}_2 + \boldsymbol{a}_1 \times \boldsymbol{a}_{2,r}, \tag{2.50}$$

And since $|\boldsymbol{a}_3| = \sqrt{\boldsymbol{a}_3 \cdot \boldsymbol{a}_3}$, the variation of the normalisation $|\boldsymbol{a}_3|$ is

$$\big(|\boldsymbol{a}_3|\big)_{,r} = \frac{\boldsymbol{a}_3 \cdot \boldsymbol{a}_{3,r}}{|\boldsymbol{a}_3|}, \tag{2.51}$$

Such that the variation of the unit surface normal vector of the undeformed configuration, $\hat{\boldsymbol{a}}_3$, can be obtained. Together with the variation of the surface hessian, $\boldsymbol{a}_{(\alpha,\beta),r}$ from equation (2.48), the variation of the second fundamental form becomes

$$b_{\alpha\beta,r} = \hat{\boldsymbol{a}}_{3,r} \cdot \boldsymbol{a}_{\alpha,\beta} + \hat{\boldsymbol{a}}_3 \cdot \boldsymbol{a}_{(\alpha,\beta),r} \tag{2.52}$$

From the definition of the bending strain tensor $\boldsymbol{\kappa}$ in equation (2.22) and the fact that the undeformed configuration is invariant to the deformation field $\boldsymbol{u}$, the coefficients of the first variation of the bending strain tensor become:

$$\kappa_{\alpha\beta,r} = -b_{\alpha\beta,r} \tag{2.53}$$

To obtain the second variation of the bending strain tensor $\boldsymbol{\kappa}$, the second variations of $\boldsymbol{a}_{\alpha,\beta}$ and $\hat{\boldsymbol{a}}_3$ need to be obtained in order to compute the second variation of $b_{\alpha\beta}$. Firstly, from equation (2.48), it follows that $\boldsymbol{a}_{(\alpha,\beta),rs} = 0$ since the second variation of $\boldsymbol{u}_k$ is zero. Secondly, for the second variation of the unit normal vector $\hat{\boldsymbol{a}}_3$, the second variation of the non-unit normal vector $\boldsymbol{a}_3$ and its length $|\boldsymbol{a}_3|$ are needed. The second variation of the non-unit normal vector follows from the first variation in equation (2.49) and from $\boldsymbol{a}_{(\alpha,\beta),rs}$:

$$\boldsymbol{a}_{3,rs} = \boldsymbol{a}_{1,rs} \times \boldsymbol{a}_2 + \boldsymbol{a}_{1,r} \times \boldsymbol{a}_{2,s} + \boldsymbol{a}_{1,s} \times \boldsymbol{a}_{2,r} + \boldsymbol{a}_1 \times \boldsymbol{a}_{2,rs} = \boldsymbol{a}_{1,r} \times \boldsymbol{a}_{2,s} + \boldsymbol{a}_{1,s} \times \boldsymbol{a}_{2,r} \tag{2.54}$$

Furthermore, the second variation of $|\boldsymbol{a}_3|$ is

$$\big(|\boldsymbol{a}_3|\big)_{,rs} = \frac{|\boldsymbol{a}_3|(\boldsymbol{a}_3 \cdot \boldsymbol{a}_{3,r})_{,s} - (\boldsymbol{a}_3 \cdot \boldsymbol{a}_{3,r})\big(|\boldsymbol{a}_3|\big)_{,s}}{|\boldsymbol{a}_3|^2} = \frac{\boldsymbol{a}_{3,s} \cdot \boldsymbol{a}_{3,r} + \boldsymbol{a}_3 \cdot \boldsymbol{a}_{3,rs}}{|\boldsymbol{a}_3|} - \frac{(\boldsymbol{a}_3 \cdot \boldsymbol{a}_{3,r})(\boldsymbol{a}_3 \cdot \boldsymbol{a}_{3,s})}{|\boldsymbol{a}_3|^3} \tag{2.55}$$

Using the second variations of the non-unit normal $\boldsymbol{a}_3$ (see equation (2.54)) and its length $|\boldsymbol{a}_3|$ (see equation (2.55)), the second variation of the unit normal vector $\hat{\boldsymbol{a}}_3$ can be derived:

$$\begin{aligned}\hat{\boldsymbol{a}}_{3,rs} &= \left(\frac{|\boldsymbol{a}_3|\boldsymbol{a}_{3,r} - \boldsymbol{a}_3\big(|\boldsymbol{a}_3|\big)_{,r}}{|\boldsymbol{a}_3|^2}\right)_{,s}\\ &= \frac{\Big(|\boldsymbol{a}_3|\boldsymbol{a}_{3,r} - \boldsymbol{a}_3\big(|\boldsymbol{a}_3|\big)_{,r}\Big)_{,s}}{|\boldsymbol{a}_3|^2} - \frac{\Big(|\boldsymbol{a}_3|\boldsymbol{a}_{3,r} - \boldsymbol{a}_3\big(|\boldsymbol{a}_3|\big)_{,r}\Big)2|\boldsymbol{a}_3|\big(|\boldsymbol{a}_3|\big)_{,s}}{|\boldsymbol{a}_3|^4}\\ &= \frac{\boldsymbol{a}_{3,rs}}{|\boldsymbol{a}_3|} - \frac{\big(|\boldsymbol{a}_3|\big)_{,s}\boldsymbol{a}_{3,r}}{|\boldsymbol{a}_3|^2} - \frac{\boldsymbol{a}_{3,s}\big(|\boldsymbol{a}_3|\big)_{,r}}{|\boldsymbol{a}_3|^2} - \frac{\boldsymbol{a}_3\big(|\boldsymbol{a}_3|\big)_{,rs}}{|\boldsymbol{a}_3|^2} + 2\frac{\boldsymbol{a}_3\big(|\boldsymbol{a}_3|\big)_{,r}\big(|\boldsymbol{a}_3|\big)_{,s}}{|\boldsymbol{a}_3|^3}\end{aligned} \tag{2.56}$$

Additionally, taking the variation of $b_{\alpha\beta,r}$ and using the first and second variations of $\boldsymbol{a}_{\alpha,\beta}$ and $\hat{\boldsymbol{a}}_3$, the second variation of the second fundamental form $b_{\alpha\beta}$ can be obtained:

$$\begin{aligned}b_{\alpha\beta,rs} &= \hat{\boldsymbol{a}}_{3,rs} \cdot \boldsymbol{a}_{\alpha,\beta} + \hat{\boldsymbol{a}}_{3,r} \cdot \boldsymbol{a}_{(\alpha,\beta),s} + \hat{\boldsymbol{a}}_{3,s} \cdot \boldsymbol{a}_{(\alpha,\beta),r} + \hat{\boldsymbol{a}}_3 \cdot \boldsymbol{a}_{(\alpha,\beta),rs},\\ &= \hat{\boldsymbol{a}}_{3,rs} \cdot \boldsymbol{a}_{\alpha,\beta} + \hat{\boldsymbol{a}}_{3,r} \cdot \boldsymbol{a}_{(\alpha,\beta),s} + \hat{\boldsymbol{a}}_{3,s} \cdot \boldsymbol{a}_{(\alpha,\beta),r}.\end{aligned} \tag{2.57}$$

From equation (2.57) and equation (2.22), it directly follows that the coefficients of the second variation of the bending strain tensor are:

$$\kappa_{\alpha\beta,rs} = -b_{\alpha\beta,rs}. \tag{2.58}$$

Besides the first and second variations of the membrane strain tensor $\boldsymbol{\varepsilon}$ and the bending strain tensor $\boldsymbol{\kappa}$, the first variations of the membrane force tensor $\mathbf{N}$ and the bending moment tensor $\mathbf{M}$ also need to be obtained. Using the total differentials $\mathrm{d}\mathbf{N}$ and $\mathrm{d}\mathbf{M}$ (see equation (2.35)), the coefficients of the first variations of $\mathbf{N}$ and $\mathbf{M}$ with respect to u_r are

$$\begin{aligned}N^{\alpha\beta}_{,r} &= \left(\int_\tau \mathscr{C}^{\alpha\beta\gamma\delta}\,\mathrm{d}\theta^3\right)\varepsilon_{\gamma\delta,r} + \left(\int_\tau \theta^3\mathscr{C}^{\alpha\beta\gamma\delta}\,\mathrm{d}\theta^3\right)\kappa_{\gamma\delta,r}\\ M^{\alpha\beta}_{,r} &= \left(\int_\tau \theta^3\mathscr{C}^{\alpha\beta\gamma\delta}\,\mathrm{d}\theta^3\right)\varepsilon_{\gamma\delta,r} + \left(\int_\tau (\theta^3)^2\mathscr{C}^{\alpha\beta\gamma\delta}\,\mathrm{d}\theta^3\right)\kappa_{\gamma\delta,r}\end{aligned} \tag{2.59}$$

Note that the last term of equation (2.37) drops out because the variation of θ^3 with respect to u_r is zero. Using the variations with respect to the nodal displacement components u_r, the first and second variations of the energy equation in the shell following from the virtual work statement in equation (2.31) can be defined for each component u_r. Firstly, the first variation of the energy statement provides the components of the residual vector $\boldsymbol{R}$ as

$$R_r(\boldsymbol{u}) = \int_\Omega \mathrm{N}(\boldsymbol{u}) : \boldsymbol{\varepsilon}_{,r}(\boldsymbol{u}) + \mathrm{M}(\boldsymbol{u}) : \boldsymbol{\kappa}_{,r}(\boldsymbol{u})\,\mathrm{d}\Omega - \int_\Omega \boldsymbol{f} \cdot \boldsymbol{u}_{,r}\,\mathrm{d}\Omega - \int_{\partial\Omega} \boldsymbol{g} \cdot \boldsymbol{u}_{,r}\,\mathrm{d}\Gamma. \tag{2.60}$$

Secondly, the second variation of the energy statement from equation (2.32) provides the stiffness matrix for the Newton-Raphson iterations, also known as the *(tangential) stiffness matrix K*, with coefficients:

$$K_{rs} = \int_\Omega \mathrm{N}_{,s}(\boldsymbol{u}) : \boldsymbol{\varepsilon}_{,r}(\boldsymbol{u}) + \mathrm{N}(\boldsymbol{u}) : \boldsymbol{\varepsilon}_{,rs}(\boldsymbol{u}) + \mathrm{M}_{,s}(\boldsymbol{u}) : \boldsymbol{\kappa}_{,r}(\boldsymbol{u}) + \mathrm{M}(\boldsymbol{u}) : \boldsymbol{\kappa}_{,rs}(\boldsymbol{u})\,\mathrm{d}\Omega. \tag{2.61}$$

In the case of zero displacements, i.e., $\boldsymbol{u} = 0$, the deformation gradient $\mathbf{F}$ is an identity map, and the deformation tensor $\mathbf{C}$ is the identity tensor. Therefore, the stress tensor becomes the null tensor, $\mathbf{S} = \mathbf{0}$, making the tensors $\mathbf{N}$ and $\mathbf{M}$ vanish as well. In this case, the first integral of the residual vector $\boldsymbol{R}$ is zero, and the second and fourth terms drop out of the stiffness matrix K. This gives the *external force vector* $\boldsymbol{P}$ and the *linear stiffness matrix* K^L, with coefficients:

$$
\begin{aligned}
P_r &= -R_r(0) = \int_{\Omega} \boldsymbol{f} \cdot \boldsymbol{u}_{,r} \,\mathrm{d}\Omega + \int_{\partial\Omega} \boldsymbol{g} \cdot \boldsymbol{u}_{,r} \,\mathrm{d}\Gamma, \\
K^L_{rs} &= K_{rs}(0) = \int_{\Omega} \mathrm{N}_{,s}(0) : \boldsymbol{\varepsilon}_{,r}(0) + \mathbf{M}_{,s}(0) : \boldsymbol{\kappa}_{,r}(0) \,\mathrm{d}\Omega .
\end{aligned}
\tag{2.62}
$$

Discretising the second variation of the kinetic energy, see equation (2.39), gives the coefficients of the mass matrix:

$$
M_{rs} = \int_{\Omega} t\rho \boldsymbol{u}_{,r} \cdot \boldsymbol{u}_{,s} \,\mathrm{d}\Omega \tag{2.63}
$$

Up to this point, all quantities have been defined to be used in the variational formulation, except for the basis functions φ_k to define the undeformed and deformed configurations of the shell surface as well as the displacement field $\mathring{\boldsymbol{x}}$, $\boldsymbol{x}$, and $\boldsymbol{u}$, respectively (see equation (2.40)). Since the Hessian of the metric tensor $a_{\alpha,\beta}$ is used in the definition of the second fundamental form and its variations (see equations (2.14), (2.52) and (2.57)), the basis functions φ_k need to be differentiable up to the second derivative. Due to the higher-order continuity that can be achieved using splines, they provide a suitable basis for the Kirchhoff–Love shell. In the paradigm of using the same splines for the representation of the geometry $\boldsymbol{s}(\theta^1, \theta^2)$ as well as for the discrete solution of the displacement field $\boldsymbol{u}^h(\theta^1, \theta^2, \theta^3)$, this choice of the basis introduces the *isogeometric Kirchhoff–Love shell.*

2.4 Numerical Structural Analysis

In the previous section on isogeometric shell analysis (see section 2.2), the derivation for the discretisation of the Isogeometric Kirchhoff–Love shell is presented. As a result of discretisation, the residual vector (see equation (2.60)) and the stiffness matrix (see equation (2.61)) form the building blocks for structural analysis. Without loss of generality, this section provides a mathematical background on the solution procedures to perform numerical structural analysis given operators like the residual vector and the stiffness matrix. The procedures presented in the current section are independent of the discretisation method (e.g., FEA, IGA) and independent of the element type used (e.g., solids, shells, beams). Although the section primarily focuses on structural analysis, some numerical procedures are also applicable in other fields of computational physics, as will be detailed in the description of these methods.

In section 2.4.1, numerical procedures for static analysis of structures are provided. These analyses involve constant (non-linear) loading and time independence. Static analysis is relevant in chapters 3, 5, 7 and 8 of dissertation. In section 2.4.2, the numerical procedure for linear modal analysis is provided. This procedure regards vibration analysis of structures and is relevant for chapters chapters 5, 7 and 8. Section 2.4.3 provides the numerical procedures for computing structural stability and linear buckling analysis. These procedures are relevant for chapters 3, 5 and 8. Following linear buckling analysis, section 2.4.4 provides the basics for numerical analysis of post-buckling analysis. This involves structural stability computations and numerical procedures with incremental loads or displacements in a quasi-static fashion, meaning that dynamic effects are omitted. Algorithms for post-buckling analysis are fundamental for the numerical analysis of wrinkling and are relevant for chapters 3, 5, 6 and 8.

2.4.1 Static Analysis

The main goal of static analysis is to obtain a time-independent response of a structure subject to a time-independent load. Due to the time independence of the analysis, the kinetic energy is assumed to be zero and the load is constant. Therefore, static analysis equilibrates the internal potential energy due to the deformation of the structure with the external work exerted on the structure. For the isogeometric Kirchhoff–Love shell, this implies finding the solution $\boldsymbol{u}$ that minimises the virtual work equation equation (2.31).

The Newton-Raphson Method

The most commonly used way to find the solution $\boldsymbol{u}$ for non-linear static analysis is to use Newton-Raphson iterations. For the Kirchhoff–Love shell, this entails solving equation (2.32) with the Jacobian of the energy statement defined as the variation of equation (2.31) in equation (2.36). This entails that the residual equation

$$\boldsymbol{R}(\boldsymbol{u}) = 0, \tag{2.64}$$

is solved incrementally by

$$K(\boldsymbol{u}^i)\Delta\boldsymbol{u} = -\boldsymbol{R}(\boldsymbol{u}^i), \quad \boldsymbol{u}^{i+1} = \boldsymbol{u}^i + \Delta\boldsymbol{u}, \quad i = 0, 1, \ldots \tag{2.65}$$

Here, $\boldsymbol{u}^i$ is the solution in iteration i, and $\Delta\boldsymbol{u}$ is the solution increment. The Newton-Raphson iterations can be initialised by a linear static analysis:

$$K\boldsymbol{u} = \boldsymbol{P}, \tag{2.66}$$

Where for the initialisation of $\boldsymbol{u}^0$, the linear stiffness matrix K_L is used, which is by definition found by using the stiffness matrix for the undeformed configuration, i.e., $K(0)$.

The Dynamic Relaxation Method

An alternative approach to the Newton-Raphson method for static analysis is to use a pseudo-dynamic system. This method was developed by Otter & Day [421] for the application of tidal computations, and later it was referred to as the Dynamic Relaxation method [419, 420]. The dynamic relaxation method is based on the solution of the structural dynamics equation:

$$M\ddot{\boldsymbol{u}}(t) + C\dot{\boldsymbol{u}}(t) - \boldsymbol{R}(\boldsymbol{u})(t) = 0, \tag{2.67}$$

where M is the mass matrix, $\ddot{\boldsymbol{u}}$ is the vector of discrete accelerations, C is the damping matrix, $\dot{\boldsymbol{u}}$ is the vector of discrete velocities, and $\boldsymbol{R}(\boldsymbol{u})$ is the residual vector. Using central finite differences, the acceleration vector $\ddot{\boldsymbol{u}}$ can be expressed in terms of the velocity vector $\dot{\boldsymbol{u}}$ and a time step Δt:

$$\ddot{\boldsymbol{u}}_t = \frac{\dot{\boldsymbol{u}}_{t+\Delta t/2} - \dot{\boldsymbol{u}}_{t-\Delta t/2}}{\Delta t}, \tag{2.68}$$

where the notation $u_t = u(t)$ is adopted for the sake of clarity. A common assumption in dynamic relaxation methods is to define the damping matrix proportional to the mass matrix, i.e., $C = cM$. Using $\dot{\boldsymbol{u}}_t = \frac{1}{2}(\dot{\boldsymbol{u}}(t + \Delta t/2) + \dot{\boldsymbol{u}}(t - \Delta t/2))$ and substituting the damping matrix and equation (2.68) into equation (2.67) gives:

$$M\left(\frac{\dot{\boldsymbol{u}}_{t+\Delta t/2} - \dot{\boldsymbol{u}}_{t-\Delta t/2}}{\Delta t} + c\frac{\dot{\boldsymbol{u}}_{t+\Delta t/2} + \dot{\boldsymbol{u}}_{t-\Delta t/2}}{2}\right) - \boldsymbol{R}(\boldsymbol{u}_t) = 0. \tag{2.69}$$

This expression can be simplified to obtain the following result:

$$\dot{\boldsymbol{u}}_{t+\Delta t/2} = \frac{(2 - c\Delta t)}{(2 + c\Delta t)}\dot{\boldsymbol{u}}_{t-\Delta t/2} + \frac{2\Delta t}{(2 + c\Delta t)}M^{-1}\boldsymbol{R}(\boldsymbol{u}_t). \tag{2.70}$$

Here, the velocities at time $t = 0$ are initialised by the zero vector, i.e., $\dot{\boldsymbol{u}}_0 = 0$. Similar to equation (2.68), the vector of nodal velocities $\dot{\boldsymbol{u}}$ can be expressed by finite differences as follows:

$$\dot{\boldsymbol{u}}_{t+\Delta t/2} = \frac{\boldsymbol{u}_{t+\Delta t} - \boldsymbol{u}_t}{\Delta t}. \tag{2.71}$$

Rewriting this expression, the deformations at time-step $t + \Delta t$ are updated using the updated velocities from equation (2.70):

$$\boldsymbol{u}_{t+\Delta t} = \boldsymbol{u}_t + \Delta t\dot{\boldsymbol{u}}_{t+\Delta t/2}. \tag{2.72}$$

The equations above can be solved for a damping parameter c and if and only if the mass matrix M is invertible. In the work of Papadrakakis [429], the mass matrix is chosen to be $M = \rho D$, and the parameters ρ and c are automatically determined. In this model, D is a diagonal matrix with the main diagonal terms of the linear stiffness matrix K. Another choice for M is to use a diagonal lumped mass matrix [7, 32, 33, 291, 292, 566] or a column-sum of the stiffness matrix [571]. Different scaling methods of the mass and damping contributions in the dynamic relaxation method are summarised by Rezaiee-Pajand & Estiri [467] and Rodriguez *et al.* [479]. Furthermore, the works of Joldes *et al.* [291, 292] provide algorithms for fast computations of the Dynamic Relaxation method on GPUs.

An alternative approach to using the damping matrix C is to use the so-called *kinetic damping* approach, introduced by Cundall [128]. In this approach, the kinetic energy in the system is traced, and the nodal velocities $\dot{\boldsymbol{u}}$ are set to zero when a peak in kinetic energy is detected. The advantage of this method is that no parameter for damping is required and that it provides robustness [32, 33, 508]. Firstly, the kinetic energy in the system is defined by:

$$E_t^K = \frac{1}{2}\dot{\boldsymbol{u}}^\top M\dot{\boldsymbol{u}}. \tag{2.73}$$

Hence, a peak is detected if $E^K_t > E^K_{K,t+\Delta t}$ for $\Delta t > 0$. Following the work by Topping & Khan [566], it is assumed that a peak occurs in the middle of the interval $[t - \Delta t, t]$, hence at $t - \Delta t/2$ if $E^K_{t-3\Delta t/2} < E^K_{t-\Delta t/2}$ and $E^K_{t-\Delta t/2} > E^K_{t+\Delta t/2}$. In that situation, the displacement vector $\boldsymbol{u}_{t+\Delta t}$ and the velocity vector $\dot{\boldsymbol{u}}_{t+\Delta t/2}$ are known. Using these solutions, the displacements at the peak can be computed by:

$$\boldsymbol{u}_{t^\star} = \boldsymbol{u}_{t+\Delta t} - \frac{3}{2}\dot{\boldsymbol{u}}_{t+\Delta t/2} + \frac{\Delta t}{2} M^{-1}\boldsymbol{R}(\boldsymbol{u}_t), \tag{2.74}$$

Where the peak time is denoted by $t^\star$. Using the displacement vector $\boldsymbol{u}_{t^\star}$, the method is re-initiated using $\boldsymbol{u}_{t^\star}$. Since the velocities are fully damped after a kinetic energy peak, they are set to zero upon re-initialization. Hence, to compute the next step after the restart at a peak on $\boldsymbol{u}_{t^\star}$, the velocity vector for $\boldsymbol{u}_{t^\star+\Delta t/2}$ becomes:

$$\dot{\boldsymbol{u}}_{\boldsymbol{u}_{t^\star+\Delta t/2}} = \frac{\Delta t}{2} M^{-1}\boldsymbol{R}(\boldsymbol{u}_{t^\star}) \tag{2.75}$$

Using equation (2.75), the displacement vector $\boldsymbol{u}_{t^\star+\Delta t}$ can be found using equation (2.72). This kinematic damping procedure is successfully applied by Barnes [32, 33], Taylor *et al.* [551] and Rezaiee-Pajand & Estiri [467], among others, showing the robustness of the method while eliminating the need to determine the damping coefficient c. Furthermore, Lee *et al.* [333] presented an explicit ALM using Dynamic Relaxation with kinetic damping, avoiding solving any linear system.

Example 2.4.1 (Static analysis)**.** *Figure 2.10 illustrates a static analysis on the flame of the TU Delft (see figure 2.1a) subject to a point load in out-of-plane direction, $P_z = 1\,[N]$, on its top point. The material of the flame is steel with a thickness of $t = 1\,[mm]$, a Young's modulus of $E = 210\,[GPa]$ and a Poisson's ratio of $\nu = 0.3\,[-]$. The bounding-box of the flame is $5.5 \times 5.5\,[m^2]$.*

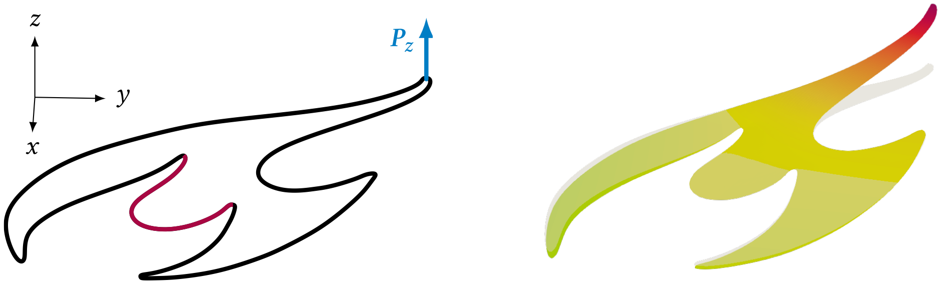
The parametrisation of the flame is as in figure 2.1b and the Almost-C^1 method us used to generate a C^1 basis over the multi-patch domain. By solving the equation $K_L\boldsymbol{u} = \boldsymbol{P}$ (see equation (2.62)), the deformation of the flame can be computed. The solution is provided in figure 2.10b.

2.4.2 Modal Analysis

The main goal of modal analysis is to find the eigenfrequencies and corresponding eigenmodes of a structure. Eigenfrequencies are the resonance frequencies of a structure, and eigenmodes are the shape in which the structure vibrates at a given eigenfrequency. Modal analysis plays an important role not only in the acoustic analysis of structures but also in the analysis of structures in turbulent flows. Moreover, modal analysis can also be used to construct a reduced-order model for dynamic analysis [659].

The starting point for the derivation of the modal analysis equations is the structural dynamics equation as given in equation (2.67). Assuming a damping-free system, no forcing, and a harmonic solution $\boldsymbol{u}(t) = \boldsymbol{u}_A e^{i\omega t}$, the discrete structural dynamics equation simplifies to

$$-\omega^2 M\boldsymbol{u}_A + K\boldsymbol{u}_A = 0 \tag{2.76}$$



(a) Problem setup. The red boundary is fixed in all directions.

(b) The deformed geometry. The colours represent vertical displacements.

Figure 2.10: Static analysis on the flame of the TU Delft (see figure 2.1a). The problem setup is given in (a), with the resulting deformation in (b).

Where $\boldsymbol{u}_A$ are the amplitudes of the harmonic solution. Equation (2.76) represents the so-called free vibration equation. This equation can be written as an eigenvalue problem:

$$\omega^2 M\boldsymbol{\phi} = K\boldsymbol{\phi} \tag{2.77}$$

With ω and $\boldsymbol{\phi}$ the eigenfrequency and discrete mode shape, respectively. Since this generalized eigenvalue problem is composed of discrete operators M and K with N degrees of freedom, the solution to equation (2.77) consists of N eigenpairs $(\omega_k, \boldsymbol{\phi})$. In engineering practice, the number of degrees of freedom N can be large, and finding all eigenpairs of the system becomes costly. In this case, special routines for sparse systems can be used to find a small number of eigenvalues in a given range, e.g., the shifted block Lanczos algorithm [212] or the shift-invert and Cayley transforms [384].

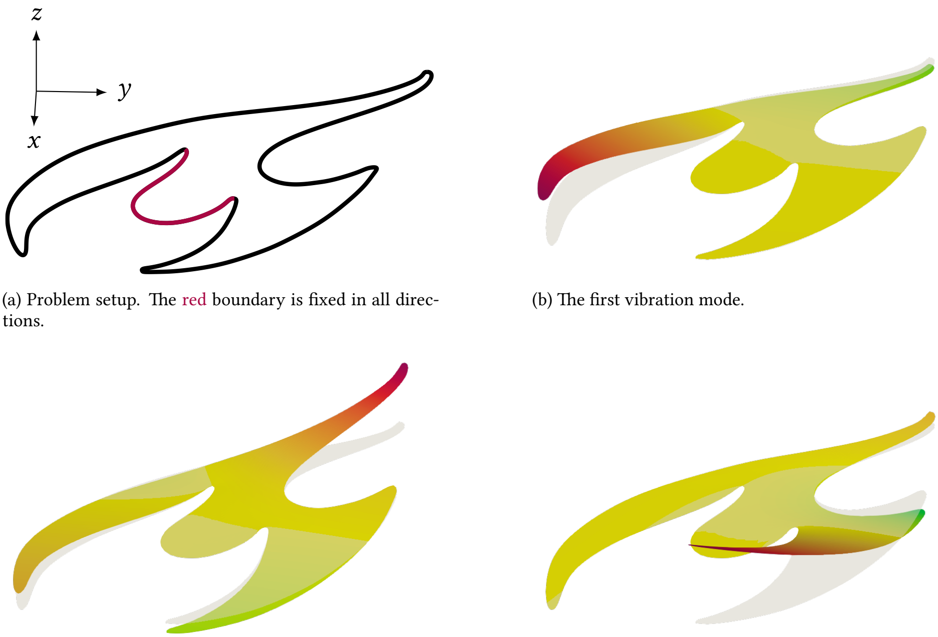
Example 2.4.2 (Modal analysis)**.** *As an example of modal analysis, the flame of TU Delft (see figure 2.1a) is again considered. The problem setup is as in example 2.4.1: The flame is fixed at the same boundary and the material properties are the same, with an additional density of* $\rho = 7850\,[kg/m^3]$*, and no load is applied on the geometry. Moreover, the parametrisation and the unstructured spline basis are as in figure 2.11 as well. The first three vibration modes are given in figures 2.11b to 2.11d, with corresponding eigenfrequencies* $\omega_1 = 0.18\,[rad/s]$*,* $\omega_2 = 0.39\,[rad/s]$ *and* $\omega_3 = 0.88\,[rad/s]$.

2.4.3 Buckling Analysis

The goal of buckling analysis is to find the critical loads in a load configuration for which the structure will be unstable. In the case of linear buckling analysis, a magnification factor λ_{crit} is found for which a scaled external load factor $\lambda_{\text{crit}}\boldsymbol{P}$ causes structural instability in the buckling mode shape $\boldsymbol{\phi}$. Before deriving the procedure to find the critical buckling load magnification factor λ_{crit} and the corresponding buckling mode $\boldsymbol{\phi}$, some notes on structural stability are made.

Structural Stability

The stability of a structure is related to the stiffness matrix $K(\boldsymbol{u})$. In particular, the displacements $\boldsymbol{u}$ for which the stability of a structure changes are such that the determinant of the



(a) Problem setup. The red boundary is fixed in all directions.

(b) The first vibration mode.

(c) The second vibration mode.

(d) The third vibration mode.

Figure 2.11: Modal analysis on the flame of the TU Delft (see figure 2.1a). The problem setup is given in (a) and the resulting vibration mode shapes of the first, second and third vibration modes are given in (b-d)

.

stiffness matrix $K(\boldsymbol{u})$ is zero, i.e., $\det K(\boldsymbol{u}) = 0$. Using Vieta's rule [142], the determinant of $K(\boldsymbol{u})$ can be written as

$$\det K(\boldsymbol{u}) = \prod_k \lambda_k \tag{2.78}$$

With λ_k the k^{th} eigenvalue of the matrix $K(\boldsymbol{u})$. This implies that the condition $\det K(\boldsymbol{u}) = 0$ is satisfied when one or more eigenvalues of $\det K(\boldsymbol{u})$ are zero. When $K(\boldsymbol{u})$ is symmetric positive definite, the determinant of $K(\boldsymbol{u})$ can also be found by the product of the diagonal matrix from a Cholesky decomposition $K(\boldsymbol{u}) = L^\top DL$ [631], i.e.

$$\det K(\boldsymbol{u}) = \prod_k D_{kk} \tag{2.79}$$

Given a deformation vector $\boldsymbol{u}$, the stability of the structure with this deformation vector can be determined by analysing the determinant of $K(\boldsymbol{u})$. As specified above, $\det K(\boldsymbol{u}) = 0$ corresponds to a change in the stability of the structure. Moreover, if $\det K(\boldsymbol{u}) > 0$, a structure is considered stable, and if $\det K(\boldsymbol{u}) < 0$, it is considered unstable [631]. This implies that the stability of a structure can be determined by the sign of the smallest diagonal entry of the Cholesky decomposition of $K(\boldsymbol{u})$, following equation (2.79), or by the sign of the smallest eigenvalue of $K(\boldsymbol{u})$, following equation (2.78).

Given a singular point $\boldsymbol{w}_S = (\boldsymbol{u}_S, \lambda_S)$ such that $\det K(\boldsymbol{u}_S) = 0$, it can either be a *bifurcation point* or a *limit point*; see figure 2.14 and figure 2.13c, respectively. A bifurcation point is an intersection of multiple solution branches and is often associated with buckling instabilities since the deformation patterns of the branches intersecting in $\boldsymbol{w}_S$ are of different nature. A limit point, on the other hand, is a point where the stability of the structure changes without intersecting with other branches. This type of singular point is often associated with snapping instabilities. To identify whether a singular point is a bifurcation point or a limit point, the product of the first eigenvector $\boldsymbol{\phi}_1$ of $K(\boldsymbol{u}_S)$ with the external load vector $\boldsymbol{P}$ is considered. If this product is (close to) zero, a bifurcation point is found; otherwise, it is a limit point [632].

Linear Buckling Analysis

To find the critical load for a structure in a certain load configuration, the problem is to find the solution vector $\boldsymbol{u}$ such that $\det K(\boldsymbol{u}) = 0$, in other words, that the stiffness matrix is singular. This condition is equivalent to solving

$$K(\boldsymbol{u})\boldsymbol{\phi} = 0 \tag{2.80}$$

From [70], it follows that using a split of the stiffness matrix into a linear and a non-linear part, i.e., $K(\boldsymbol{u}) = K_L + K_{NL}(\boldsymbol{u})$, equation (2.80) can be written as

$$(K_L - \lambda K(\boldsymbol{u}_L))\boldsymbol{\phi} = 0. \tag{2.81}$$

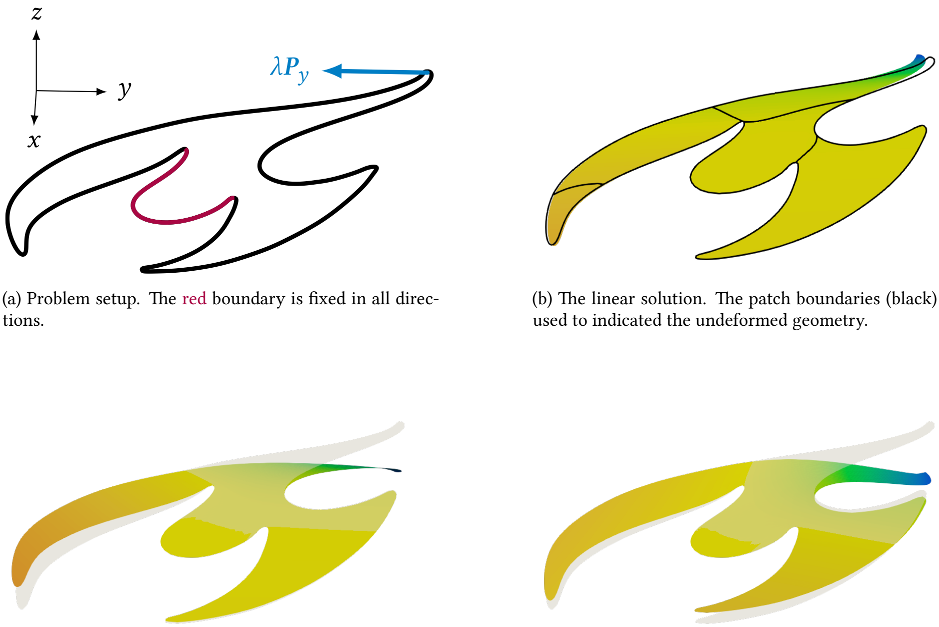
Where λ is the load magnification factor of the load applied in the linear problem $K\boldsymbol{u}_L = \lambda\boldsymbol{P}$ with $\boldsymbol{u}_L$ the solution to this linear problem. From equation (2.81), it can be seen that to obtain the critical load factor λ and the corresponding buckling mode shape $\boldsymbol{\phi}$, the generalised eigenvalue problem

$$K\boldsymbol{\phi}_k = \lambda_k K_{NL}(\boldsymbol{u})\boldsymbol{\phi}_k, \tag{2.82}$$

Has to be solved. Solving this eigenvalue problem gives the critical load factors λ_k and buckling mode shapes $\boldsymbol{\phi}_k$, $k = 1, .., N$ for a system with N degrees of freedom.

Example 2.4.3 (Buckling analysis)**.** *To illustrate a the setup of a buckling problem, the TU Delft flame is exposed to a horizontal load $P_y = 1\,[N]$ scaled by a factor λ on its top point (figure 2.12a) and the marked boundary is fixed in all directions. The material, the parametrisation and the basis are as in examples 2.4.1 and 2.4.2.*

Firstly, figure 2.12b presents the linear solution $\boldsymbol{u}_L$ to the linear shell problem in equation (2.62)*. This solution is used to compute the eigenvalue problem in equation* (2.81)*, of which figures 2.12c and 2.12d are the first two buckling modes, $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ with load factors $\lambda_1 = 4.6 \cdot 10^{-6}\,[-]$ and $\lambda_2 = 5.7 \cdot 10^{-6}\,[-]$.*



(a) Problem setup. The red boundary is fixed in all directions.

(b) The linear solution. The patch boundaries (black) are used to indicated the undeformed geometry.

(c) The first buckling mode.

(d) The second buckling mode.

Figure 2.12: Buckling analysis on the flame of the TU Delft (see figure 2.1a). The problem setup is given in (a) and the linear solution is given in (b). The resulting buckling mode shapes of the first and second buckling mode are given in (c) and (d).

2.4.4 Quasi-static Analysis

Quasi-static analysis is the analysis of structures subject to loads or displacements with a prescribed change. The type of analysis is referred to as *quasi*-static since the resulting structural response is assumed to be time-independent, hence all dynamic effects are negligible. Quasi-static analysis is typically used in applications where the structural response with respect to increasing loads or strains is of interest, for example, when testing limit states with fractures or tests involving structural instabilities. The primary application of quasi-static analysis in this dissertation is wrinkling modelling, where membrane deformations are studied as a consequence of an increasing load or displacement.

Let $\boldsymbol{R}(\boldsymbol{u},\lambda)=0$ be the equilibrium path corresponding to the discrete residual $\boldsymbol{R}(\boldsymbol{u},\lambda)$, on its turn defined by geometry, material properties, and loads acting on a structure. On the equilibrium path, the structure subject to the defined load case is in equilibrium, meaning that the sum of the internal and (λ-scaled) external forces is equal to zero. Figure 2.13 depicts three ways to find the equilibrium path. Firstly, the external load can be incremented using load increments $\Delta\lambda$, resulting in displacement increment $\Delta\boldsymbol{u}$. However, as illustrated in figure 2.13a, this can lead to multiple solutions or large jumps in the solution path. Alternatively, the displacement at a point or on a boundary can be controlled incrementally, yielding the equivalent $\Delta\boldsymbol{u}$ for which a load increment $\Delta\lambda$ can be computed. Similar to load control, this can lead to large jumps on the paths, as shown in figure 2.13b. In order to be able to capture the full equilibrium path, a remedy to load- or displacement-control is to use arc-length control. In this case, the increments $\Delta\lambda$ and $\Delta\boldsymbol{u}$ are combined to ensure they stay on the equilibrium path, provided a constraint equation $f(\Delta\boldsymbol{u},\Delta\lambda)=0$. As illustrated in figure 2.13c, the constraint equation can be a hypersphere of which the intersections with the equilibrium path provide the next point. Given the residual $\boldsymbol{R}(\boldsymbol{u},\lambda)$, the ALMs find the solution $\boldsymbol{w}_i=(\boldsymbol{u}_i,\lambda_i)$ in load step i with $\boldsymbol{u}_i$ and load factor λ_i, using

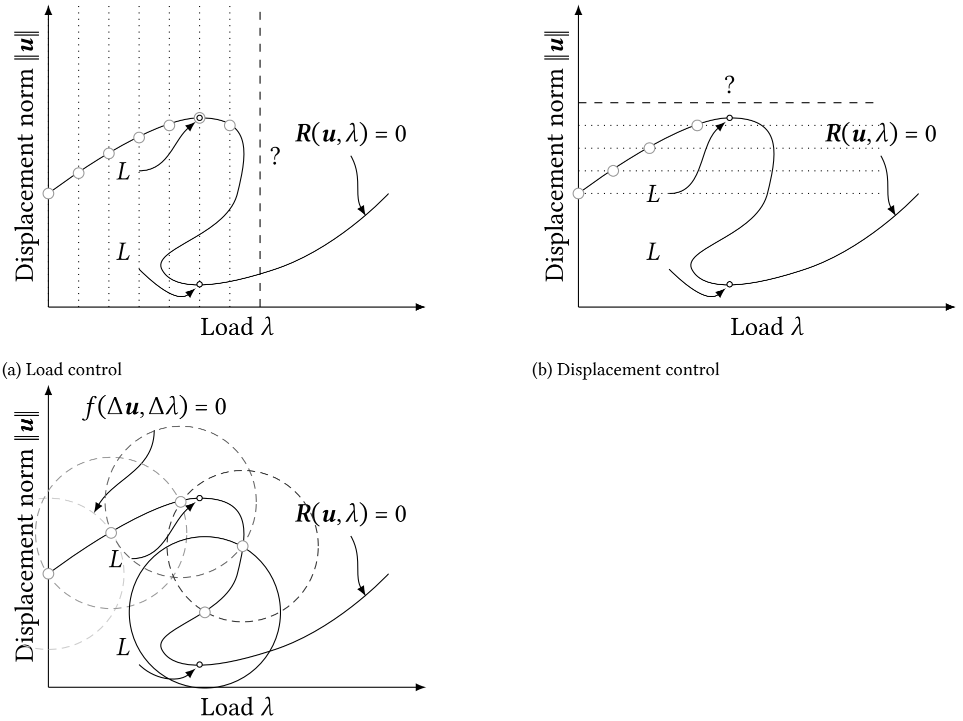
$$\boldsymbol{w}_i=\boldsymbol{w}_{i-1}+\Delta\boldsymbol{w}_i,\quad i=1,2,\dots \tag{2.83}$$

given an initial deformation $\boldsymbol{w}_0$ and with $\Delta\boldsymbol{w}_i$ the solution increment in load-step i, which is iteratively updated in iteration k as

$$\Delta\boldsymbol{w}_k=\Delta\boldsymbol{w}_{k-1}+\delta\boldsymbol{w},\quad \Delta\boldsymbol{w}_0=(0,0),\quad k=1,2,\dots \tag{2.84}$$

In the remainder of this section, a brief introduction to Crisfield's [124] Arc-Length Method (ALM) is given. A commonly used alternative is Riks' method, for which the reader is referred to the works by Riks [469, 470], Wempner [620] and Ramm [459]. For more information on ALMs in general, the reader is referred to the books by Crisfield [126], Crisfield *et al.* [127], de Borst *et al.* [142], and Wriggers [631]. Furthermore, the work by Fafard & Massicotte [178] provides a geometric interpretation of ALMs, Carrera [85] provide a comparison of some methods, and Memon & Su [385] provides historical remarks.

Parts of section 2.4.4 are based on:
[583] **H. M. Verhelst**, M. Möller, J. H. Den Besten, F. J. Vermolen & M.L. Kaminski, "Equilibrium Path Analysis Including Bifurcations with an Arc-Length Method Avoiding A Priori Perturbations", Numerical Mathematics and Advanced Applications, ENUMATH 2019: European Conference, Egmond aan Zee, The Netherlands, September 30-October 4, 1109-1117 (2019)


Displacement norm $\|\boldsymbol{u}\|$
Load λ
$\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$
?
L
L
$f(\Delta\boldsymbol{u},\Delta\lambda) = 0$

(a) Load control

(b) Displacement control

(c) Arc-length control

Figure 2.13: Load (top left), displacement (top right), and arc-length control (bottom left) for structural analysis problems. The question mark (?) indicates the iteration where load and displacement control encounter a point where the next obtained point is typically difficult to find. Limit points are indicated with *L*.

Crisfield's Arc-Length Method

In Crisfield's ALM, the constraint equation is given by

$$f(\Delta\boldsymbol{u},\Delta\lambda) = \Delta\boldsymbol{u}^\top\Delta\boldsymbol{u} + \Psi^2\Delta\lambda^2\boldsymbol{P}^\top\boldsymbol{P} - \Delta\ell^2 = 0, \tag{2.85}$$

with $\Delta\ell$ the arc length, $\boldsymbol{P}$ the vector of external forces, and Ψ a scaling factor providing the spherical constraint for $\Psi = 0$ or the elliptical constraint for $\Psi = 1$. As discussed by Schweizerhof & Wriggers [497] and Bellini & Chulya [42],

$$\Psi_i = \frac{\boldsymbol{u}_{i-1}^\top\boldsymbol{u}_{i-1}}{\lambda_{i-1}^2\boldsymbol{P}^\top\boldsymbol{P}}, \tag{2.86}$$

provides a dimension-independent influence of the load increment $\Delta\lambda\boldsymbol{P}$ in the constraint equation for load step *i*. Together with the system of equations resulting from the nonlinear problem to be solved, the constraint equation adds an extra equation to the system to be solved. If the discretisation of the equation provides a banded system matrix, the system loses its banded nature when the constraint equation is added [631]. Therefore, the system of equations is solved in a segregated way [126]. To this end, the incremental displacement in iteration k is split into two parts:

$$\delta \boldsymbol{u}_k = \beta \delta \bar{\boldsymbol{u}}_k + \delta \lambda_k \delta \hat{\boldsymbol{u}}_k, \tag{2.87}$$

where

$$\begin{aligned} K(\boldsymbol{u}) \delta \bar{\boldsymbol{u}}_k &= \boldsymbol{R}(\boldsymbol{u}_k, \lambda_k), \\ K(\boldsymbol{u}) \delta \hat{\boldsymbol{u}}_k &= \boldsymbol{P}, \end{aligned} \tag{2.88}$$

can be solved to compute the contributions of a standard load-controlled Newton-Raphson iteration, $\delta \bar{\boldsymbol{u}}_k$, and from an incremental update of the load $\boldsymbol{P}$, $\delta \hat{\boldsymbol{u}}_k$, respectively. Furthermore, β is a line-search parameter, which is $\beta = 1$ by default. Using equation (2.84), the constraint equation in equation (2.85) can be written as

$$a \delta \lambda_k^2 + b \delta \lambda_k + c = 0, \tag{2.89}$$

with

$$\begin{aligned} a &= \delta \hat{\boldsymbol{u}}_k^\top \delta \hat{\boldsymbol{u}}_k + \Psi^2 \boldsymbol{P}^\top \boldsymbol{P} = a_0 \\ b &= 2\big(\delta \hat{\boldsymbol{u}}_k^\top \Delta \boldsymbol{u} + \Delta \lambda \Psi^2 \boldsymbol{P}^\top \boldsymbol{P}\big) + 2\beta \delta \hat{\boldsymbol{u}}_k^\top \delta \bar{\boldsymbol{u}}_k = b_0 + \beta b_1 \\ c &= \beta^2 \delta \bar{\boldsymbol{u}}_k^\top \delta \bar{\boldsymbol{u}}_k + 2\beta \delta \bar{\boldsymbol{u}}_k^\top \Delta \boldsymbol{u} + \Delta \boldsymbol{u}^\top \Delta \boldsymbol{u} + \Delta \lambda^2 \Psi^2 \boldsymbol{P}^\top \boldsymbol{P} - \Delta \ell^2 = c_0 + \beta c_1 + \beta^2 c_2. \end{aligned} \tag{2.90}$$

Since equation (2.89) is quadratic, it can have zero to two solutions. In the regular case, two roots are found for equation (2.89), corresponding to two intersections of the constraint equation with the equilibrium path. In this case, a root needs to be selected such that the path following proceeds in the correct direction, e.g., without stepping backwards. Selection of the correct root corresponds to finding the correct $\delta \lambda$ such that the displacement increment from equation (2.87) can be computed. In the original work of Crisfield [124], it is proposed to take the increment that yields the largest positive inner-product with respect to the previous step at iteration k. The root is selected as [472]:

$$\delta \lambda = \max_{\delta \lambda_r,\, r=1,2} \delta \lambda_r \big(\Delta \boldsymbol{u}^\top \delta \hat{\boldsymbol{u}}_k + \Psi^2 \Delta \lambda\big). \tag{2.91}$$

In the case of $b^2 - 4ac$ in equation (2.90), complex roots occur, and the root selection from equation (2.91) fails. As explained by Carrera [85], complex roots occur in regions where the path is strongly curved within one load step, potentially yielding more than two intersections. In this case, the arc length can be bisected until real solutions are found for equation (2.89) [42], or a line-search technique proposed by Lam & Morley [329] or Zhou & Murray [660] can be used. Given the displacement update $\delta \boldsymbol{u} = \beta \delta \bar{\boldsymbol{u}} + \delta \lambda \delta \hat{\boldsymbol{u}}$ and given coefficients a_0, b_0, b_1, c_0, c_1 and c_2 from equation (2.90) such that $b^2 - 4ac < 0$ in equation (2.89) (for $\beta = 1$), the condition $b^2 - 4ac \geq 0$ can be enforced, yielding the following equation in terms of the line-search parameter $\tilde{\beta} \neq 1$ [125]

$$a_s \tilde{\beta}^2 + b_s \tilde{\beta} + c_s \geq 0, \tag{2.92}$$

with [472]

$$\begin{aligned} a_s &= b_1^2 - 4 a_0 c_2, \\ b_s &= 2 b_0 b_1 - 4 a_0 c_1, \\ c_s &= b_0^2 - 4 a_0 c_0. \end{aligned} \tag{2.93}$$

The solutions of equation (2.92) are simply found by $\hat{\beta}_{1,2} = \left(-b_s \pm \sqrt{b_s^2 - 4a_s b_s}\right)/2a_s$. Zhou & Murray [660] argue that the solutions $\beta_{1,2}$ are of opposite sign and that if β is between these roots, the constraint equation is satisfied. In their work, it is proposed to choose $0 < \beta \leq \min(1, \beta_2)$ for $\beta_1 < \beta_2$. In the work of Lam & Morley [329], it is advised to choose β as:

$$\beta = \begin{cases} \beta_2 - \epsilon & \text{if } \beta_2 < 1.0, \\ \beta_2 + \epsilon & \text{if } -\dfrac{b_s}{a_s} < 1.0 < \beta_2, \\ \beta_1 - \epsilon & \text{if } \beta_1 < 1.0 < -\dfrac{b_s}{a_s}, \\ \beta_1 + \epsilon & \text{if } 1.0 < \beta_1, \end{cases} \tag{2.94}$$

with $\epsilon = 0.05|\beta_2 - \beta_1|$. In case β is close to zero, it is advised to bisect the arc length [472, 660].

At the beginning of a new load step of the ALM, the arc-length iterations need to be initialized using a predictor, such that $\Delta\boldsymbol{u}$ and $\delta\lambda$ in equations (2.89) and (2.90) are non-zero. The load increment $\delta\lambda_0$ of the predictor is given by:

$$\delta\lambda_0 = \begin{cases} \dfrac{\Delta\ell}{\sqrt{2\delta\boldsymbol{u}_t^\top \delta\boldsymbol{u}_t}} & \text{if } (\boldsymbol{u}, \lambda) = (0,0), \\ \dfrac{\Delta\ell}{\sqrt{\delta\boldsymbol{u}_t^\top \delta\boldsymbol{u}_t + \Psi^2 \boldsymbol{P}^\top \boldsymbol{P}}} & \text{elsewhere.} \end{cases} \tag{2.95}$$

The sign of $\delta\lambda_0$ is chosen positive in case of the first load step (i.e., in case where $(\boldsymbol{u}, \lambda) = (0,0)$) and it is determined by the previous load step $\Delta\boldsymbol{u}, \Delta\lambda$ otherwise [184–186]:

$$\operatorname{sign}(\delta\lambda_0) = \operatorname{sign}\left(\Delta\boldsymbol{u}^\top \delta\hat{\boldsymbol{u}} + \Delta\lambda\Psi^2 \boldsymbol{P}^\top \boldsymbol{P}\right). \tag{2.96}$$

In the work of Feng *et al.* [184–186], it is noted that the sign of $\delta\lambda_0$ can also be determined by the sign of the determinant of $K(\boldsymbol{u})$ as a by-product in case a Cholesky solver is used, as discussed in section 2.4.3. Furthermore, they also remark in their works that the determination of the travel direction works well with a proper arc-length selection. If the arc length is too large compared to the curvature of the solution path, the predictor can lead to a point where more than two intersections are found and complex roots are encountered. An alternative predictor is presented by Kadapa [295], using the two previous steps for an extrapolation.

Arc-Length Exploration

Quasi-static analysis is not only related to finding solution paths as illustrated in figure 2.14, e.g., snapping problems with limit-point stabilities in structural analysis, but also to problems where the solutions of $\boldsymbol{R}(\boldsymbol{u}, \lambda) = 0$ branches into multiple solution curves in a pitch-fork or bifurcation; see figure 2.14. Finding a network of curves forming the possible equilibrium paths of a non-linear problem $\boldsymbol{R}(\boldsymbol{u}, \lambda) = 0$ can be done using ALMs and is referred to as *arc-length exploration.* Arc-length exploration allows for the study of multiple solution paths in a system, as well as the transition between the paths. Rivetti & Neukirch [474] showed transitions between wrinkling configurations in the case of membrane wrinkling on a foundation by considering multiple bifurcation branches. Furthermore, Draelants *et al.* [169] apply arc-length exploration to hormone transport modelling, Wouters & Vanroose [630] uses it for modelling of superconductors, and Thies *et al.* [561] applies it to solve an equation describing the skin patterning of species.

The first ingredient of an arc-length exploration method is to find bifurcation points. To this end, the detection of singular points can be done using equation (2.79). If a singular point is detected between the points $(\boldsymbol{w}_k, \boldsymbol{w}_{k+1})$, the bifurcation point can be approached using the robust but slowly converging bi-section method [595], being a robust [80] but slow [631] root-finding method. Alternatively, a Newton-Raphson method solving the system:

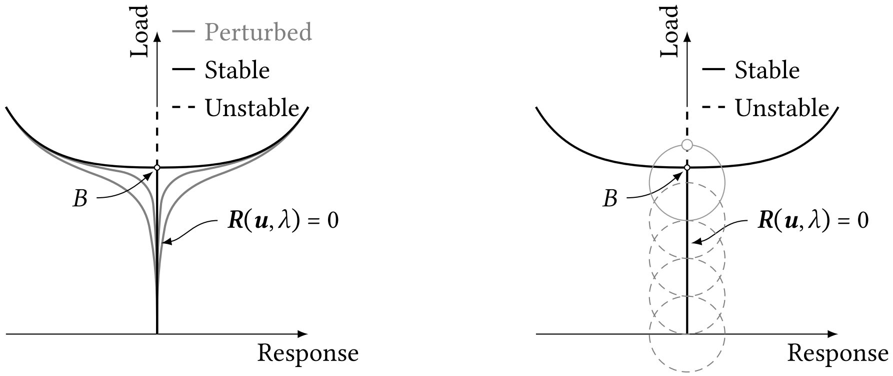
$$\begin{bmatrix} \boldsymbol{R}(\boldsymbol{u},\lambda) \\ K(\boldsymbol{u})\boldsymbol{\phi} \\ l(\boldsymbol{\phi}) \end{bmatrix} = 0. \tag{2.97}$$

Can be used. Here, besides solving the non-linear equation $\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$, the condition $K(\boldsymbol{u})\boldsymbol{\phi} = 0$ is solved, being equivalent to finding $\boldsymbol{\phi}$ such that $\det K(\boldsymbol{u}) = 0$ (see equation (2.79)). Lastly, the equation $l(\boldsymbol{\phi}) = \|\boldsymbol{\phi}\| - 1 = 0$ is solved, preventing the trivial solution $\boldsymbol{\phi} = 0$ from being found. Note that this algorithm is able to find limit points as well, since condition equation (2.79) is valid for limit points and bifurcation points. This method is based on the work of Wagner & Wriggers [595] and Wriggers & Simo [632], and the reader is referred to their work for further implementation details. Furthermore, Shi & Crisfield [506] proposes a semi-direct approach, relaxing the condition to solve the equilibrium equation $\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$ exactly in early iterations, providing a more efficient scheme.

Given a bifurcation point satisfying equation (2.97), branch switching can simply be performed by applying a perturbation in the direction of one of the eigenvectors of the system. In that case, the starting point of a new branch $\boldsymbol{u}_S, \lambda_S$ is given by [505, 631]:

$$\boldsymbol{u}_S = \boldsymbol{u}_B + \zeta\bar{\boldsymbol{\phi}}, \quad \lambda_S = \lambda_B. \tag{2.98}$$

Here, $\bar{\boldsymbol{\phi}}$ is the first eigenvector of $K(\boldsymbol{u}_B)$ and ζ is a perturbation magnitude, chosen large enough to switch the branch but small enough not to influence the solutions in the new branch [631]. This branch-switching is the most straight-forward approach, but it suffices for most pitch-fork bifurcations [632]. Alternative approaches are given by Wriggers & Simo [632], Shi [505], and Wouters & Vanroose [630]. Software packages for automatic exploration include `PyNCT` [168] in `Python`, `BifurcationKit.jl` [582] in `Julia`, or `LOCA` [487] within `Trilinos` [237] in `C++`.


Load
— Perturbed
— Stable
- - Unstable
B
$\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$
Response
Load
— Stable
- - Unstable
B
$\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$
Response

(a) Effect of initial perturbations to avoid passing bifurcation points.

(b) Arc length method applied on pitch-fork bifurcation, without perturbations.

Figure 2.14: A solution path with a bifurcation point B. The path can be *approximated* by perturbing the solution (a). Alternatively, using the extended ALM, point B can be approximated and the original solution branch can be found (b)

2.5 Wrinkling Mechanics

Wrinkling is a phenomenon that is all around us, for example, in sails (figure 1.1a), in our skin (figure 1.1b), in inflatable items (figure 1.1c), or in aged fruit (figure 1.1d). According to the Oxford English Dictionary, a *wrinkle* [422, entry 2] is indeed inspired by the world around us:

> A crease, fold, or ridge, caused by the folding, puckering, or contraction of a fabric, cloth, or other pliant substance.

Here, a *pliant* substance is a substance with high bending flexibility. In physics, a distinction is made between wrinkling, folding, and creasing. Adopting the definitions from Li *et al.*, wrinkling is defined as [345]:

> (..) periodic or chaotic surface undulations appearing on an originally flat surface.

Here, wrinkling is associated with a deformation of a surface that is initially flat, and the undulations imply that the amplitude of the wrinkles is smoothly distributed. Folding, on the other hand, is defined as [345]:

Parts of section 2.5 are inspired by the literature reviews of the following MSc. theses, which where supervised by the author throughout their PhD research:

[331] *E. Lavaerts (2020). "Framework to research and design wrinkle free very large flexible offshore solar platforms by adding permeability". Master's thesis, Technische Universiteit Delft,*

[51] *S. de Bode (2021). "Wrinkling analysis and design of Offshore Flexible Floating Solar Structures". Master's thesis, Technische Universiteit Delft.*

> (...) a buckling-induced surface structure with localised, deep surface valleys.

This definition implies that folds are initiated by buckling (into a wrinkled shape), and as loading progresses, folds start to appear. In this definition, the word *localized* is of great importance. Typically, a wrinkled membrane transforming into a folded membrane experiences a transition from homogeneous to heterogeneous wrinkling wave lengths through the domain. Lastly, the distinction between folding and creasing is made based on the notion of self-contact with the crease:

> (...) when an initially smooth surface forms a self-contacting shape with a sharp ridge or sulci.

As described by Li *et al.* [345], folding occurs on membranes with a relatively high bending stiffness compared to the foundation stiffness, whereas creasing usually occurs at the surface of soft materials without hard skins.

Understanding the mechanisms of wrinkling, folding, and creasing finds relevance in multiple fields of science and engineering. In physics, scaling laws have been developed to describe the physics of wrinkling, inspired by water lilies [643], beach balls [578], apples [91], et cetera. In aeronautical engineering, wrinkling is investigated in the development of solar sails for space propulsion [195, 210, 211, 388, 484–486, 626] or parachutes [25, 26, 298, 417, 536–546, 546–549, 556, 557] and in automotive engineering, wrinkling plays an important role in the deployment of airbags [209, 300, 301, 383]. In nano-engineering, wrinkling plays an important role in the development of thin graphene sheets. This material of atomic thickness has extraordinary properties such as high thermal conductivity and mechanical strength, but its very low thickness makes it prone to wrinkling [10, 148, 349, 600, 604, 606], which as a consequence influences its thermal conductivity [96] among other properties. Furthermore, in the maritime and offshore domains, wrinkling is a topic of interest in sailing [466] or for the development of large floating membrane structures, for example, to enable offshore solar energy generation [583]. Besides the mechanical engineering disciplines, wrinkling has been a topic of interest in the cosmetic industry to reduce skin wrinkling or in the biomedical sciences related to wound healing [527–529].

In this section, background on the mechanics of wrinkling is provided. In section 2.5.1, the physics behind wrinkling are summarized. Here, the interplay between bending, foundation, and membrane energies leading to wrinkling patterns in membranes is discussed. Section 2.5.2 provides a review of pioneering results from wrinkling experiments, and section 2.5.3 provides a review of numerical methods employed in the simulation of membrane wrinkling. Although the major focus of this dissertation is on the numerical analysis of wrinkled membranes, a review of experimental investigations is added to help understand the phenomenon. Lastly, section 2.5.4 provides a brief review of design measures that can be taken to eliminate wrinkling in membranes.

2.5.1 Wrinkling Energy Contributions

When looking at the various applications of wrinkling research, the common denominator is that wrinkling typically concerns thin membranes. Indeed, typical wrinkling patterns can only exist if the bending stiffness of the material is small enough to allow for small radii of curvature, which is naturally related to low thickness. However, low bending stiffness is not the only factor influencing the wrinkling phenomenon. In the potential energy balance of a wrinkled membrane, which is naturally minimised, the energy from *bending*, *membrane forces*, and *foundation stiffness* determine the amplitudes and wave lengths in a wrinkled membrane. Following the works of Cerda & Mahadevan [91] and Pocivavsek *et al.* [442], the energy contributions of bending energy, foundation energy, and stretching energy are discussed. These energy contributions form the foundation for the study of the physics of wrinkling and the assumption of numerical models.

Bending energy

The bending energy stored in a wrinkled membrane depends on the thickness, the bending stiffness, and the radius of curvature of the deformed membrane. In particular, for a line as in figure 2.15, the bending stiffness per unit width is given by Cerda & Mahadevan [91]:

$$\mathcal{U}_B = \frac{1}{2}\int_\Omega B\left(\frac{\partial^2\zeta}{\partial x^2}\right)^2 \mathrm{d}x \tag{2.99}$$

Assuming a two-dimensional setting and small slopes, $\frac{\partial^2\zeta}{\partial x^2}$ is the curvature of the line at position x and B is the bending stiffness. Equation (2.99) shows that the bending energy is proportional to the square of the curvature, meaning that the higher the curvature in the membrane, the higher the bending energy stored in the wrinkle, as illustrated in example 2.5.1. In general, the curvature of a wrinkle increases with decreasing wave length for a fixed amplitude or with increasing amplitude for a fixed wave length. Therefore, it can be stated that *bending energy penalises short waves or high amplitudes.*

Example 2.5.1 (Bending energy of sine waves)**.** *Consider three sine-waves with different frequencies and equal amplitudes, defined by $\zeta(x) = -\zeta_0\sin(a\pi x)$, on the domain $x \in [0,\pi]$ with frequency parameter $a = 1,2,4$. The amplitude ζ_0 is assumed to be sufficiently small, such that the the curvature of these waves – assuming a two-dimensional setting and small slopes – is given by the second derivative of the amplitude function:*

$$\zeta''(x) = \zeta_0 a^2\pi^2\sin(ax). \tag{2.100}$$

As a result, the bending stiffness stored in the function $\zeta(x)$ is given by

$$\mathcal{U}_B = \frac{1}{2}\int_0^1 B(\zeta''(x))^2\,\mathrm{d}x = \frac{\zeta_0^2 B}{4}\left(a^4\pi^4\right). \tag{2.101}$$

For the curves in figure 2.15, this implies that when a doubles, the bending energy increases by a factor of $2^4 = 16$.

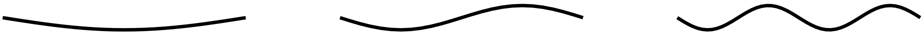


Figure 2.15: Three sine waves with the same amplitude, from left to right: $\zeta(x) = -\zeta_0 \sin(a\pi x)$, $a = 1,2,4$, $x \in [0,1]$, $\zeta_0 = 1/20$.

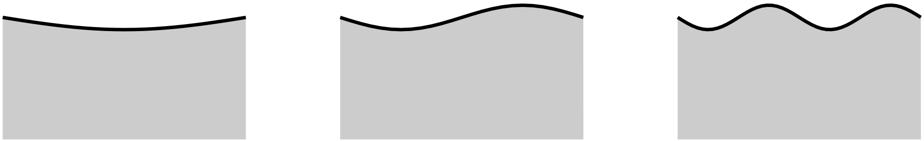


Figure 2.16: Three sine waves with the same amplitude, from left to right: $\zeta(x) = -\zeta_0 \sin(a\pi x)$, $a = 1,2,4$, $x \in [0,1]$, $\zeta_0 = 1/20$, supported by a foundation (gray).

Foundation energy

When a membrane is supported by a substrate or foundation, wrinkling of the membrane implies indentation of the foundation. For an elastic foundation, the potential energy stored due to indentation is simply given by the square of the indentation times the stiffness of the membrane [91], equivalent to the potential energy stored in a spring:

$$\mathcal{U}_F = \frac{1}{2}\int_\Omega K\zeta^2 \,\mathrm{d}x, \tag{2.102}$$

where K is the foundation stiffness. Equation (2.102) shows that the foundation energy increases with increasing indentation area into the foundation, which is not necessarily related to wrinkling wave lengths, as illustrated in example 2.5.2. It should be noted that equation (2.102) assumes that the potential energy stored in the membrane is related to amplitudes into and out of the foundation. When the membrane is supported by the foundation and detachment is possible (e.g., for liquids when surface tension is small or for delaminated composites), only the indenting amplitudes imply potential energy storage in the foundation, slightly modifying equation (2.102). As a general rule of thumb, it can be stated that *foundation energy penalises wrinkles with large indentation areas.*

Example 2.5.2 (Substrate energy of sine waves)**.** *Consider three sine-waves with different frequencies and equal amplitudes, defined by $\zeta(x) = -\zeta_0 \sin(a\pi x)$, on the domain $x \in [0,\pi]$ with parameters $a = 1,2,4$. The potential energy stored in the foundation is given by equation* (2.102)*:*

$$\mathcal{U}_F = \frac{1}{2}\int_0^1 \zeta(x)^2 \,\mathrm{d}x = \frac{\zeta_0^2 K}{4}, \quad a \in \mathbb{Z}. \tag{2.103}$$

This shows that irrespective of the parameter a, the curves in figure 2.16 store the same amount of energy in the foundation.

As discussed by Pocivavsek *et al.* [442] and Cerda & Mahadevan [91], the combination of foundation and bending energy already illustrates an energy balance. Since the total energy in a system is always minimised, the interplay between the influence of small and large wave lengths in combination with the wrinkling amplitude determines the final wrinkling shape. In example 2.5.3, the bending and foundation energy contributions from examples 2.5.1 and 2.5.2 are combined to illustrate energy minimisation of different numbers of half-waves for a sinusoidal wrinkling shape.

Example 2.5.3 (Interplay between bending an substrate energy)**.** *The wrinkling shape will be determined by minimising the total energy in the system. Under the assumption that the wrinkle shape is sinusoidal with a small amplitude ζ_0, the total energy in the sinusoidal wrinkle follows from examples 2.5.1 and 2.5.2:*

$$\mathcal{U} = \mathcal{U}_B + \mathcal{U}_F = \frac{\zeta_0^2 B}{4}\left(a^4\pi^4\right) + \frac{\zeta_0^2 K}{4}. \tag{2.104}$$

In addition, introducing inextensibility of the wrinkle under end-shortening, i.e. the length of the curve does not change when wrinkled, the following condition needs to be satisfied [91]:

$$\Delta = \frac{1}{2}\int_0^1 \left(\frac{\partial\zeta}{\partial x}\right)^2 \mathrm{d}x = \frac{1}{2}a^2\pi^2\zeta_0^2. \tag{2.105}$$

This condition allows to express the amplitude ζ_a of the wave in terms of the number of half-waves a for a given end-point displacement Δ:

$$\zeta_0 = \frac{1}{a\pi}\sqrt{2\Delta}. \tag{2.106}$$

Substituting this relation into equation (2.104) *gives the total energy stored in the sinusoidal wrinkle in terms of the bending stiffness B, the foundation stiffness K and the number of half-waves a:*

$$\mathcal{U} = \frac{\Delta}{2}\left(B\pi^2 a^2 + \frac{K}{\pi^2 a^2}\right). \tag{2.107}$$

This energy can be minimised with respect to a for any combination of the bending stiffness B and the foundation stiffness K. In figure 2.17 the number of half-waves corresponding to the minimal energy state of the sinusoidal wrinkle is plotted for different ranges of the bending stiffness B and foundation stiffness K. Indeed, this figure shows that lower bending stiffness implies a larger number of half-waves, or, that decreasing foundation stiffness implies less wrinkles.

Stretching energy

Besides bending energy and energy stored in a foundation, the third energy considered for a wrinkled membrane is stretching energy. When a membrane is subject to axial tension or compression, potential energy is stored in the axial (in-plane) deformations of the membrane. More precisely, given a tension $T(x)$ along the x direction [91],

$$U_S = \frac{1}{2}\int_\Omega T(x)\left(\frac{\partial\zeta}{\partial x}\right)^2 \mathrm{d}x, \tag{2.108}$$

Following the work of Cerda & Mahadevan [91], it can be observed that this expression is similar to equation (2.102), with the foundation energy $K \sim T/L^2$, given the domain length L. In other words, the membrane stretching forms an effective elastic foundation. Therefore, a rule of thumb for stretching energy is that *stretching energy penalises wrinkles with large indentations.*

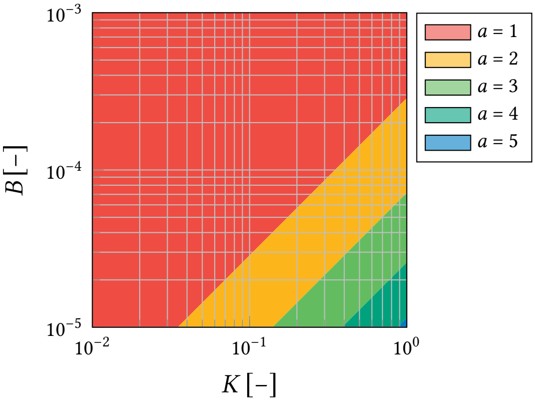


Figure 2.17: The number of wrinkles (coloured) for which the total energy $\mathcal{U} = \mathcal{U}_B + \mathcal{U}_F$ is minimised, plotted against different values of the (dimensionless) bending and foundation stiffnesses, B and K, respectively. This diagram is valid only for wrinkles represented by a sinusoidal wave.

2.5.2 Experimental Investigations

In the last decades, experimental investigations on membrane wrinkling have increased our understanding of the phenomenon. The interplay between the bending, membrane, and foundation energies from section 2.5.1 explains the wrinkling wave length and amplitudes that can be observed in situations like the ones depicted in figure 1.1. In this section, a brief literature on wrinkling experiments and observations is provided. The goal of this review is to provide a general understanding and overview of different wrinkling settings and knowledge about them. The section is structured according to figure 1.1, hence providing an overview of: i) wrinkles in membranes due to in-plane tension or shear; ii) wrinkles in elastically supported membranes; iii) wrinkles in pressurised membranes; and iv) wrinkles due to growth.

Since this section does not provide a complete overview of the wrinkling literature, the reader is referred to the reviews of Wang *et al.* [611] on tension-induced wrinkles, to [550] on wrinkles in curved surfaces, to [370] on wrinkles in membranes with elasticity gradients, to [434] on wrinkles in membranes with low bending stiffness and high membrane stiffness, and to [345] for a complete but less recent review on wrinkling.

Membranes Subject to Shear or Tension

In the pioneering work of Cerda *et al.* [92], the wrinkling behaviour of an elastic sheet under tension was studied. By decomposing the energy in the system into a membrane tension part as well as a bending part (see section 2.5.1), a scaling law is derived for the wrinkling wave length. The pioneering work of Cerda *et al.* [92] was generalised in [91] to be applicable to membranes on elastic foundations as well. For a membrane subject to tension, the minimisation of the bending energy and the stretching energy inside the membrane yield the final wrinkling pattern. More precisely, for wrinkles with amplitudes

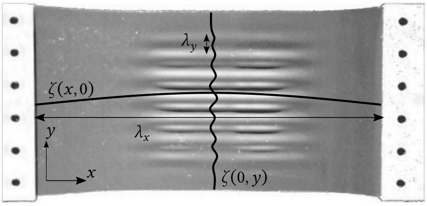


Figure 2.18: Wrinkling of a rectangular membrane subject to a tensile load, adopted from Cerda *et al.* [92]. Two sketches of the wrinkles in the x and y directions are indicated by the lines $\zeta(x,0)$ and $\zeta(0,y)$, with wave lengths λ_x and λ_y, respectively.

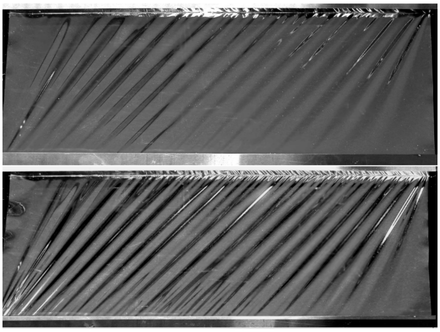


Figure 2.19: Wrinkling of a rectangular membrane subject to a shear load due to horizontal displacement of the top boundary, adopted from Wong & Pellegrino [626]. The top figure represents $\delta = 1.0\,[\mathrm{mm}]$ and the bottom figure $\delta = 3.0\,[\mathrm{mm}]$.

ζ and wave lengths λ_x and λ_y in x and y-direction, respectively, the bending energy of the wrinkles is determined by the wave lengths in transverse y-direction, and the stretching energy is related to the longitudinal half waves with length λ_x; see figure 2.18. For a low number of wrinkles, the wave length λ_y is relatively high, implying relatively high amplitudes ζ and low curvatures ζ'', and consequently low bending energy and high stretching energy. In the opposite case, a large number of wrinkles implies low wave lengths λ_y, relatively low amplitudes ζ and high curvatures ζ'' and therefore relatively high bending energy and low membrane energy. The final wrinkling state of the membrane is a result of this interplay between amplitudes, wave lengths λ_y, amplitudes ζ and curvatures ζ''.

Following up on the works of Cerda *et al.*, many investigations on the classical membrane subject to tension have been carried out. Numerical investigations of the wrinkling re-stabilisation (i.e., the disappearance of wrinkles) in the tension wrinkling case have been performed [233, 346, 403, 551, 601, 658] and Panaitescu *et al.* [427] and Chopin *et al.* [100] conducted simulations and experiments and provided new scaling laws and insights on the role of the singularities in the corners of the membrane. Furthermore, Puntel *et al.* [449] provide insights into the critical wrinkling load based on analytical formulations, and Xin & Davidovitch [641] provide insights on the formation and disappearance of wrinkles based on analytical formulations, motivating that geometric non-linearities play an important role in the non-linear wrinkling response. However, using only linear theory, Silvestre [512] motivates that wrinkling under tension can be initiated due to Poisson's effect. The tension wrinkling case also inspired works on the effects of material orthotropy by Sipos & Fehér [514], Zhu *et al.* [661], and Yuan & Xing [655], and the effect of the micro-structure of nematic elastomers [441]. Furthermore, the effects of holes [404, 650] and embedded stiff elements [349, 649] are studied. Furthermore, the work of Wang *et al.* [608] studies the wrinkling of a twisted sheet, showing great similarities with the tension wrinkling case, and the work of Wang *et al.* [603] studies the effect of curvature on the tension wrinkling behaviour.

Motivated by the wrinkling of solar sails, Wong & Pellegrino published a trilogy on the wrinkling of membranes, providing experiments [626], analytical mathematical models [627], and numerical simulations [628], based on earlier works [624, 625, 629]. In their work, Wong & Pellegrino considered the wrinkling of a membrane subject to shear and of a membrane subject to corner loads; see figure 2.20. In the analytical analysis of both experiments [627], scaling laws for the wrinkling wave lengths and amplitudes and wave lengths are derived based on an energy balance between bending and membrane tension effects similar to the work of Cerda & Mahadevan. In their analysis, they used the so-called *stress fields* describing stress states of bi-axial tension, bi-axial compression, and uni-axial tension, which have a close relation to *tension fields* as described in equation (2.109) in section 2.5.3.

The case of a membrane undergoing shear deformation was studied experimentally before by Jenkins *et al.* [281] and Mansfield [375] performed experiments and provided analytical solutions for the inclination angle of the diagonal shear wrinkles. The shear wrinkling case inspired Iwasa to derive scaling laws for shear wrinkling combined with tension on the top boundary (also present in the experiments of Wong & Pellegrino) and to derive analytical formulations for shear and tension wrinkling using tension fields [272, 273]. Furthermore, the shear wrinkling case provides a challenging case for numerical simulations, as reported by Wong & Pellegrino [627].

Furthermore, the case of a square membrane subject to outward diagonal corner loads (see figure 2.20) was experimentally studied in the work of Wong & Pellegrino [626] and in the works of Blandino *et al.* [50, 150] and Adler *et al.* [2]. Depending on the ratio between the loads over both diagonals, different wrinkling patterns can be found [626], typically involving fan-shaped wrinkles in the corners of the membrane; see figure 2.20. Based on the works by Blandino *et al.* [50], Tessler *et al.* [560] were among the first to numerically model this benchmark, comparing results with the work of Blandino *et al.* [50]. Lastly, Bouzidi & Lecieux [64] performed an experimental and numerical analysis of wrinkling in a cruciform membrane subject to bi-axial loading. The scope of their work is on the reproducibility of wrinkling experiments, showing that different patterns corresponding to different solution branches are found when repeating the experiments.

Lastly, a commonly studied example problem entails the wrinkling of an annular membrane with an applied torque on the inner boundary; see figure 2.21. This problem can be seen as a shear wrinkling case, as in figure 2.19, but axisymmetric. It was studied experimentally and analytically by Mikulas [387], providing closed-form solutions of the torque versus the rotation of the inner boundary. The wrinkling behaviour of this problem was later studied analytically by Coman & Haughton [110–112], providing expressions for the wrinkling onset and the corresponding wrinkling pattern. More experimental results are provided by Miyamura [389] for different materials.

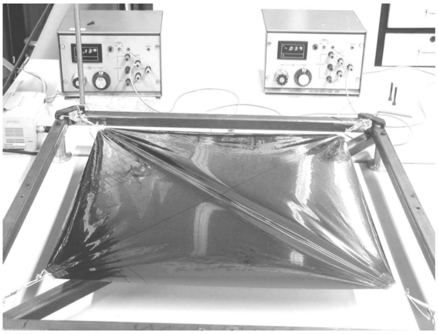


Figure 2.20: Wrinkling of a square membrane subject to outward diagonal corner loads, adopted from Wong & Pellegrino [626].

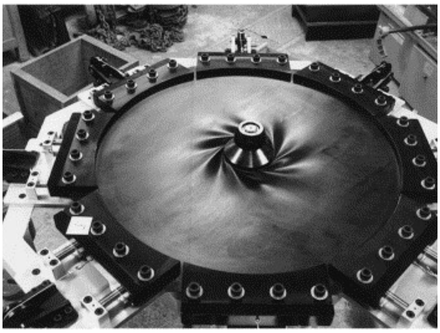


Figure 2.21: Wrinkling of an annular membrane subject to a torque on the internal boundary, adopted from Miyamura [389].

Membranes on a Foundation

In the work of Pocivavsek *et al.* [442], the transition between wrinkles and folds of a compressed membrane supported by a softer solid or fluid is studied. Using the interplay between bending energy and foundation energy (see section 2.5.1), a scaling law for the wrinkling wave length and amplitude was derived. Furthermore, the transition from a wrinkled membrane into a membrane with localised folds can be explained using bending and foundation energies. Following the work of Pocivavsek *et al.* [442], an inextensible membrane on an elastic foundation is considered; see figure 2.22. The in-extensibility assumption can be made if the bending stiffness of the membrane is far lower than the membrane stiffness, which is typically the case when the thickness of the membrane is low. When the horizontal distance Δ is low, the membrane buckles into an equally distributed wrinkling pattern, where elastic energy is stored in the foundation due to the wrinkling amplitude ζ and bending energy is stored in the membrane due to the curvature ζ'' of the membrane. When the distance Δ increases, fold localisation is observed, where the membrane transitions into a state of minimal total energy (due to a higher-order effect, see [442]), where the foundation energy localises into one deep valley with sharp curvatures, and where the rest of the membrane flattens out, i.e., locally reaches a state of low amplitudes ζ and curvatures ζ''. Since the local wrinkling-to-fold transition is a local phenomenon, it can occur repetitively in elastically supported membranes.

Inspired by the work of Pocivavsek *et al.* [442], similar studies were carried out by other researchers to investigate the wrinkling behaviour of the elementary case of the compressed membrane on an elastic support. For example, the experiments by Jambon-Puillet *et al.* [277] investigate the influence of self-weight and compression beyond self-contact on folding behaviour, and the work of Brau *et al.* [68] studies the difference between a viscous or elastic response. Wagner & Vella [594] study the effect of delamination of the membrane from the fluid. Furthermore, the work of Audoly [20] provides insights into the buckling behaviour of a floating elastica and relates the outcomes to the work of Pocivavsek *et al.* [442], whereas [153] studies the folding behaviour of a floating sheet, both based on analytical mathematical models. Analytical mathematical models were also used

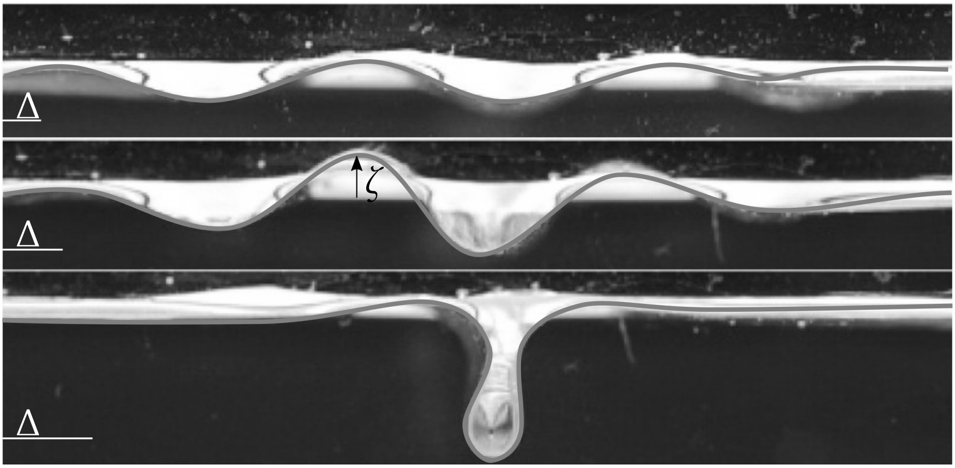


Figure 2.22: Wrinkling of a membrane on a liquid substrate, adopted from Pocivavsek *et al.* [442]. The figure depicts three stages with increasing membrane compression (Δ) from top to bottom. The wrinkling amplitude is indicated by ζ.

in the works [473, 474], where the latter concluded that the smooth transition from wrinkles to folds is a series of bifurcations in theory.

Another example of the wrinkling of membranes on a foundation entails the indentation of a floating elastic sheet. As for the case studied by Pocivavsek *et al.* [442], the relation between the bending energy and the energy stored in the foundation is relevant in this case. For thin, very thin membranes, the effect of surface tension comes into play [251, 581], but in the analysis of Box *et al.* [65], the role of bending stiffness is more important. In the work of Vella & Davidovitch [579], unified scaling relations are obtained for the wrinkling behaviour at its onset and for larger indentations for very thin and thicker membranes. These relations were experimentally verified and extended by Ripp *et al.* [471]. Floating indented circular and annular membranes also show variations in the wrinkling wave length, which was studied in more detail by Paulsen *et al.* [435, 436]. Furthermore, the works of Schroll *et al.* [495] and Huang *et al.* [265], among other studies, provide an analysis of capillary wrinkling of thin floating membranes.

Pressurised Membranes

A third commonly observed mechanism for wrinkling is related to the wrinkling of pressurised membranes. This phenomenon occurs in closed membranes subject to internal pressure and with relatively thin skin. By studying the indentation of a beach ball, i.e., a sphere subject to internal pressure, Vella *et al.* [578] derived scaling relations for the wrinkling wave length and the number of wrinkles around the indentation upon initialisation of the wrinkles. Using a non-dimensional term for the tension due to pressurisation, the wrinkling behaviour of shells subject to low and high internal pressure is studied. This analysis shows that the response for low pressurisation is bending-dominated, whereas

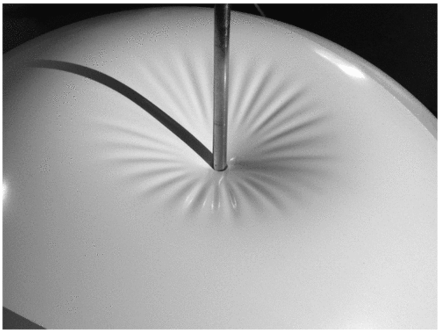


Figure 2.23: Wrinkling of an indented beach ball, adopted from Vella *et al.* [578].

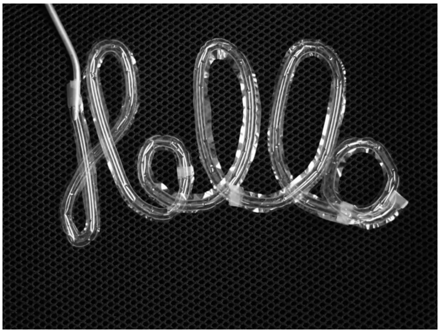


Figure 2.24: Inflation of a curvilinear balloon, adopted from Siéfert *et al.* [511].

the response for high internal pressures becomes membrane-dominated, creating a system where wrinkles are formed as a balance between membrane tension and bending energy, equivalent to the findings of Cerda *et al.* [92]. Following up on the works of Vella *et al.* [577, 578, 580], the work of Taffetani & Vella [531] studies the post-wrinkling behaviour (i.e., *far from threshold behaviour* [139]) of the indentation of a spherical pressurised shell using analytical and Finite Element modelling. In their study, it was found that the number of wrinkles changes as the indentation increases and that a spatial variation in the number of wrinkles is observed when moving away from the centre of the indentation. The latter effect is referred to as a wrinkling hierarchy [574]. Furthermore, the FEM results of Taffetani & Vella [531] show period doubling of the wrinkles for large indentations, similar to the wrinkling-to-fold transition in floating membranes as observed by Pocivavsek *et al.* [442].

Other studies on pressurised membranes have been performed by Siéfert *et al.* [509–511] and Panetta *et al.* [428]. Although these studies do not particularly elaborate on the wrinkling physics but rather focus on the inverse design of inflatable shapes, the effect of wrinkles and folds on the mechanical properties of these inflated structures is emphasised as a topic of study [509, 511]. The global mechanical properties of inflatable beams and panels have been studied by Wielgosz & Thomas [563, 621, 622] for pressurised tubes or so-called *drop-stitch panels* (e.g., the inflatable structure used for *Stand-Up Pedalling* boards) [90], for which a recent literature overview is given by van Engelen [573]. Furthermore, a recent study by Guo *et al.* [221] involved the design of a bi-stable inflatable pillow.

Growth-Induced Wrinkling

The examples on tension wrinkling, wrinkling of elastically supported membranes or pressurised membranes (see figures 1.1a to 1.1c), typically involve load conditions where external loads are applied to the membrane. On the contrary, wrinkling of membranes can also emerge as a consequence of growth of the membrane, induced by chemical reactions, swelling, or thermal expansion. Nevertheless, wrinkling patterns in growth-induced wrinkling are determined by energy balances between membrane tension energy, bending energy, or energy stored in an elastic foundation [613]. For example, dehydration of fruits figure 1.1d or vegetables can show complex wrinkling patterns when drying out [347]. In this situation, shrinkage of the elastic foundation (i.e., the flesh) to which the membrane (i.e., the skin) is attached causes a contraction of the skin, which will buckle as soon as a threshold is exceeded. Such growth-induced wrinkling patterning is studied by Li *et al.* [347], Stoop *et al.* [523], and Xu *et al.* [648] for spheres in particular. Similarly, swelling in hydrogels with different properties in the interior or at the boundary can show structural instabilities due to swelling, showing similarities to the work of Pocivavsek *et al.* [442] when applied to a flat layer [166, 167, 280]. Inspired by nature, Xu *et al.* [643] applied the theory of growth-induced wrinkling to study the morphogenesis of growing aquatic plant leaves, revealing the influence of water as an elastic foundation on the wrinkling wave lengths observed in the leaves. In addition, this study also reveals the influence of the shape of the leaves, whether they are circular (hence confined in a circumferential direction) or fan-shaped (i.e., Pacman-shaped).

2.5.3 Numerical Investigations

In section 2.5.1 and section 2.5.2, the physics behind wrinkling and experimental wrinkling investigations are discussed. The experimental investigations on the wrinkling of membranes and soft materials provide great contributions to the understanding of wrinkling patterns observed in nature and their relation to known mechanics. However, for engineering applications where wrinkling is an important factor , accurate and efficient numerical modelling of wrinkle patterns becomes of great importance. This section provides an overview of previous works on the numerical analysis of wrinkled membranes. In general, the numerical modelling can be done in different ways: by explicitly modelling the wrinkling amplitudes and wave lengths using shell models, by implicitly embedding the effects of wrinkling in element formulations, or by using reduced-order models based on the Föppl-Von Kármán plate equations. In the remainder of this subsection, a literature overview of each approach is provided.

Explicit Modelling

The conceptually most straight-forward approach for the modelling of membrane wrinkling is to use mathematical models that model the physics of thin films, including membrane and bending effects. To this end, plate or shell models will be able to simulate wrinkling patterns as they can be observed in a lab setting or in nature, whereas membrane models without bending stiffness would not be able to model wrinkles.

Complementary to experiments for the investigation of wrinkling, commercial finite element methods have been used for wrinkling modelling in many experimental studies [179, 349, 367, 382, 403, 404, 427, 524, 531, 578, 628, 650, 651]. Since the primary focus of these studies is not to develop new numerical approximation models for the computational study of membrane wrinkling, these works show that off-the-shelf finite element techniques are able to model the wrinkling of membranes in certain conditions. These studies use dynamic relaxation methods (see section 2.4.1) or Arc-Length Methods (ALMs) (see section 2.4.4).

On the contrary, scientists have developed dedicated numerical models for the study of membrane wrinkling. The models are either based on membrane elements without bending stiffness, on plate elements for initially flat surfaces, or on shell elements to study initially curved geometries.

The development of dedicated numerical models for the study of membrane wrinkling, in particular modelling amplitudes and wave lengths explicitly, has gained increasing attention in the last few years. For initially flat geometries, the Föppl-Von Kármán (FvK) model has been adopted. This model includes out-of-plane displacements in the strain definition and assumes linear bending strains. The original FvK model, which was used in [322] to model tension wrinkling with B-spline basis functions, was extended to non-linear strains and applied to the problem of tension wrinkling in [233]. The model from [233] was used by Sipos & Fehér [514] to study the effect of material orthotropy and was later extended for hyperelastic materials in [346] and applied to the tension wrinkling case, from which it was concluded that hyperelastic constitutive modelling is relevant in that particular case. The extended FvK model was adopted in numerous works in the group of Xu *et al.* (see [611] for an overview) to study the tension-wrinkling case for various hyperelastic models [191], to study the effect of orthotropy [359] and general anisotropic anisotropy [652] on the wrinkling initialisation and re-stabilisation. Other models developed in the group of F. Xu include a consistent finite-strain plate theory using Chebychev spectral collocation [190], later applied to anisotropic materials [192], and a doubly curved shell model based on differential geometry [609] used to study the effect of curvature on the initialisation and re-stabilisation of stretched sheets [603] and the effect of damage [610], later extended to study the wrinkling of tori [599].

Another model proposed by Steigmann [520] uses Koiter's non-linear plate theory and has been used in combination with Dynamic Relaxation methods [551] to model complex wrinkling problems such as the shear wrinkling case of Wong & Pellegrino [626] and wrinkling of a membrane with a hole [551], an annulus [552], for the modelling of wrinkles in graphene [451] and in plates with straight and wavy fibre reinforcement [553, 554].

Besides a variation of element assumptions, explicit modelling of wrinkling requires robust algorithms for finding equilibrium solutions. One approach in the explicit modelling of wrinkling is to use a static approach, consisting of a dynamic relaxation method, a Newton-Raphson method, or a combination of both (see section 2.4.1). Wrinkling modelling using static analysis is done by Taylor *et al.* [551]. Another approach is to use continuation methods, for example, the Arc-Length Method (see section 2.4.4), as has been done by Wong & Pellegrino [628], Diaby *et al.* [152], and Healey *et al.* [233], among others. Arc-length methods are also used in chapter 3 of this dissertation. Lastly, another continuation method that has been used in the context of explicit wrinkling modelling is the Asymptotic Numerical Method (ANM) [103–105, 132, 400], which computes a Taylor series expansion around a known solution and approximating the solution in the next point and which is also applicable to bifurcation problems [21, 63, 103, 105, 575, 647]. This method is applied to the wrinkling of a membrane on a substrate by Xu *et al.* [647] and to other wrinkling studies [191, 264, 315, 564, 609].

Implicit Modelling

Contrary to explicitly modelling wrinkling patterns in membranes, implicit wrinkling modelling entails the incorporation of a wrinkling condition into the system of equations to solve. A key aspect of implicit membrane modelling is the use of a tension field. The concept of a tension field was coined by Wagner [592, 593, 593] in a series of publications on thin sheets in aerospace applications, and later work was continued by Reissner [465], Kondo, Kazou, Iai, Takeshi, Moriguti, Sigeiti, Marasaki [325] and Mansfield [374, 375]. Tension fields describe the state of the membrane as being either taut, slack, or wrinkled, using the principal stresses or strains. In particular, the tension field $\phi(\mathbf{C})$ can be defined using the deformation tensor $\mathbf{C}$ in different ways, as given in the works of Roddeman *et al.* [478] and Kang & Im [300, 301] among others:

$$\begin{aligned} &\phi^E = \begin{cases} \text{Taut} & \text{if } E_{p,1} > 0 \\ \text{Slack} & \text{if } E_{p,2} \leq 0 \\ \text{Wrinkled} & \text{otherwise} \end{cases}, \qquad \phi^S = \begin{cases} \text{Taut} & \text{if } S_{p,1} > 0 \\ \text{Slack} & \text{if } S_{p,2} \leq 0 \\ \text{Wrinkled} & \text{otherwise} \end{cases}, \\ &\phi^M = \begin{cases} \text{Taut} & \text{if } S_{p,1} > 0 \\ \text{Slack} & \text{if } E_{p,2} \leq 0 \\ \text{Wrinkled} & \text{otherwise} \end{cases}, \end{aligned} \tag{2.109}$$

Where $\phi^E(\mathbf{C})$ denotes the strain-based tension field, $\phi^S(\mathbf{C})$ denotes the stress-based tension field, and $\phi^M(\mathbf{C})$ denotes the mixed tension field. Furthermore, $S_{p,1}$ and $S_{p,2}$ are the principal stresses such that $S_{p,1} \leq S_{p,2}$, and $E_{p,1}$ and $E_{p,2}$ are the principal strains such that $E_{p,1} \leq E_{p,2}$. Instead of the principal strains, the principal stretches λ_i can also be used, yielding equivalent criteria with $\lambda_\alpha = 1$ as a reference condition. A discussion on the three different models is given by Kang & Im [300], where it is concluded that the mixed criterion is the best choice for identifying tension fields.

Given the tension field, implicit wrinkling models assume zero bending stiffness in membranes and, therefore, the elimination of stress in the slack state. Since the wrinkled state corresponds to compression on one axis and tension on another axis, special modifications are required. These modifications can be performed by modifying the constitutive law, i.e., the relation between the deformation and stress, or by modifying the kinematic equations, i.e., modification of the deformation gradient. A general good overview of literature is give by Le Meitour *et al.* [332], and similarities between different methods are shown by Miyazaki [390].

The first implicit models for membrane wrinkling were derived using modifications of the constitutive relations. By modifying the strain energy density function [438], Pipkin and Steigmann & Green present stable modification schemes for incorporating the wrinkling conditions in membrane models [439, 440, 521], which have been extended to hyperelastic models [380] and for which variational [395–397] and interior-point [146] models have been derived. In the work of Haseganu & Steigmann [232], the dynamic relaxation method is used to apply the model presented by [521] for numerical modelling of wrinkled membranes, extended later for anisotropic non-linear materials [18, 19]. Modification schemes for the strain energy density function have been developed for anisotropic hyperelastic materials [176]. In addition, modification schemes for the material matrix were presented based on correction factors [300, 301] or projections [3], for isotropic materials and in [278, 279] for orthotropic materials. Furthermore, Liu *et al.* [356] proposes a non-intrusive implementation for existing FEA codes. Applications of the tension-field models using modified constitutive laws include the development of novel cable arrangements for bridges [34], wrinkle-free topology optimisation [368] or the modelling of biological tissues with cut-outs [6].

Instead of modifying the constitutive relation of the membrane, the works of Roddeman *et al.* [476–478] proposed to modify the deformation gradient in the wrinkling condition by assuming i) plane-stress theory , ii) no bending stiffness, and iii) no support for compressive stresses. Then, Roddeman *et al.* derive a set of inequality conditions that should be satisfied. Furthermore, they impose that a wrinkled membrane implies an elongation of the mean surface of the membrane, therefore changing the deformation gradient. Using this deformation gradient, wrinkling strain and stress tensors are derived, and finite element simulations are performed for isotropic and orthotropic materials [399, 457, 476, 477]. In the work of Lu *et al.* [365], explicit formulations for the tangential stiffness operator given the model of Roddeman *et al.* were derived and later simplified for implementation in [401]. The latter model was applied to isogeometric membrane elements recently [402].

Although implicit membrane wrinkling models based on tension fields do not provide information about wrinkling wave lengths and amplitudes, they gain increasing attention in the graphics community for the design of inflatable structures [515], of skintight clothing [392], and in cloth simulations [99, 288], for example by using tension-field theory on a coarse mesh to graphically reconstruct fine wrinkles [95]. These works in general show that tension-field models provide a good compromise between fast and realistic modelling for animation.

Reduced-Order Modelling

A relatively new approach in the numerical modelling of membrane wrinkling is the multi-scale modelling technique introduced by Damil *et al.* [133–135]. In this approach, the Föppl-Von Kármán model is discretised using Fourier series expansions, serving as a multi-scale model where the large Fourier modes capture the macroscopic deformations and the high-frequency functions capture the wrinkling patterns. Detailed reviews of this approach are given by Potier-Ferry *et al.* [443] and Huang *et al.* [266]. This so-called *Fourier reduced model* is efficient as it can predict wrinkling patterns with few degrees of freedom, but since it is inaccurate near domain boundaries, it can be combined with full shell models [266]. In the work of Khalil *et al.* [314], the Fourier reduced model was extended to cases with non-uniform wrinkling orientations, and in the work of Tian *et al.* [564], the method was combined with the ANM path-following method.

2.5.4 Wrinkling Minimisation and Elimination

Given the physics behind wrinkling (section 2.5.1), the experimental investigations on the wrinkling mechanics (section 2.5.2), and the numerical techniques to solve wrinkling (section 2.5.3), this section elaborates on the minimisation and elimination of wrinkling. Besides cosmetic aspects, wrinkling elimination is relevant because it deteriorates the surface reflectivity of light-weight space structures [486], or the properties of graphene [10, 148, 349, 600, 604, 606]. In this section, wrinkling minimisation and elimination measures are reviewed based on stiffness modifications and shape and topology modifications.

Stiffness Modifications

As discussed in section 2.5.1, wrinkling is a consequence of an energy minimisation between energy stored in the bending of the membrane, in the indentation of the foundation, or in membrane forces. To eliminate wrinkling, modifications to the thickness or material of the membrane can be made to shift the energy balance in a more favourable way, e.g., to wrinkles with lower amplitudes or wave lengths. Following the discussion in section 2.5.1, the most straightforward modifications are changes in the thickness of the membrane, increasing the bending stiffness and therefore increasing the wrinkling wave length. Similarly, the foundation stiffness can be increased, penalising large amplitudes more, hence decreasing the wrinkling wave lengths in cases of constant bending stiffness.

In the work of Li *et al.* [348], local stiffness modification is achieved by *spatio selective coating* driven by topology optimisation of the applied coating. By using a coating, the effect studied by Concha *et al.* [113] and Takei *et al.* [535] on wrinkling in bi-layer materials is employed. Furthermore, Taylor & Shirani [553] use wavy fibres in anisotropic hyperelastic materials to tune the wrinkling pattern in thin membranes, and Wang *et al.* [605] use micro-fibre networks to generate materials with high effective Poisson's ratios to eliminate wrinkling.

Shape and Topology Modifications

By changing the shape or topology of a membrane, the wrinkling pattern can be significantly influenced. Examples of shape modifications include changes in the shape of the boundaries or corners, whereas topology changes typically entail the creation of holes in the membrane. Here, a brief review of both approaches is provided.

In the work of Bonin & Seffen [57], the so-called *de-wrinkling* of pre-tensioned membranes is studied. In their work, they consider membranes of different shapes, typically loaded by tensional loads in the corners. Analysing information from principal stress fields, i.e., tension fields, they showed that trimming the membrane boundaries into a slightly trimmed shape, eliminating negative principal stresses, hence wrinkling. A similar technique was used by Li *et al.* [349] and referred to as a pre-necking strategy. Their technique was applied to a tension wrinkling case and combined with topology optimisation to establish the optimal boundary shapes. Later, Li *et al.* [349] showed that their methodology also applies to membranes with a rigid zone inside. Here, a pattern of elliptic holes around the rigid area was found through topology optimisation. Studies of rigid elements and the effect of wrinkling are also presented by Sun *et al.* [525] and Yan *et al.* [649].

Driven by space applications, light-weight membrane structures with rigid frame elements have been studied under the umbrella term *Gossamer structures* [98]. To increase the performance of deployable solar sails and large reflectors, wrinkling reduction in cable-suspended Gossamer structures was studied by Sakamoto *et al.* [484–486] and Mikulas & Adler [388], among others. In the work of Mikulas & Adler, the wrinkled area of a square solar sail concept proposed by Greschik & Mikulas [210] was improved by applying elliptic edges (as later studied by Bonin & Seffen [57] and Li *et al.* [349]). By adding a web of cables around the solar sail, Sakamoto *et al.* [484–486] managed to reduce the weight [484, 486] and wrinkles induced by dynamics [485]. An alternative to a cable arrangement to eliminate wrinkling in solar sails is to use shear-compliant borders [337–339]. Here, a series of interrogations is added close to the boundary to release local compressive stresses.

Instead of modifying the shape of the boundaries of a membrane, holes can also be added to modify the wrinkling pattern in the membrane. In the work of Yan *et al.* [650], the influence of the location of two small holes on the final wrinkling pattern in a membrane subject to tension was studied. As found in this work, the location of the holes has a significant influence on the wrinkling pattern, potentially eliminating all wrinkles. Similar observations are made by Nazzal *et al.* [404]. Furthermore, Yan *et al.* [650] also added patterns of holes, showing that these patterns can tailor the global wrinkling pattern. In another work, Yan *et al.* [649] did a similar study on the effect of adding stiffness instead of removing it by adding stiff squares on the membrane. Similar to their previous study on the influence of holes, they address the fact that the global wrinkling pattern can be highly influenced by adding stiffness. Finally, inspired by wrinkling in offshore solar platforms, the work of Lavaerts [331] shows that a combination of holes and boundary modification can be used to eliminate wrinkling in a membrane with an embedded solar panel.

Since wrinkling is induced by local compressive stresses in the membrane, Bonfanti & Bhaskar [56] investigated the wrinkling behaviour of a membrane under tension with *auxetic microstructures*. These micro-structures have the property that their Poisson ratio is negative, causing outward displacement of the edges in cases of tension wrinkling.Inspired by the concept of offshore solar platforms, de Bode [51] proposed an auxetic layout of solar panels to reduce global wrinkling. The performance of the proposed concept was evaluated using tension fields.

Inspired by wrinkling minimisation and elimination through shape and topology modifications of membranes, various optimisation studies have been performed. For example, Luo *et al.* [366, 368] performed topology optimisation of membranes in different configurations to generate wrinkling-free designs, maximising the in-plane membrane stiffness. Among the constraints of the optimisation problem is to enforce a taut state of a tension field (see equation (2.109)), hence eliminating wrinkled and slackened regions. Later, Luo *et al.* [368] expanded their topology optimisation model to multi-material geometries, creating topologies resembling Gossamer structures. Another optimisation study to eliminate wrinkling was performed by Xing *et al.* [642]. Here, the shape of corner fixtures in membranes is optimised to reduce wrinkling in different membranes subject to corner loads. Lastly, Punurai *et al.* [450] used genetic optimisation to find optimal cuts for the edges of membranes to make them wrinkle-free.

2.6 Closure

The aim of this chapter is to provide mathematical and physical preliminaries for the remainder of this dissertation. Section 2.1 briefly provides a set of mathematical notations, which will be used in all other chapters. Section 2.2 introduces the basic concepts behind Isogeometric Analysis (IGA), and derives the isogeometric Kirchhoff–Love shell. The derivation of this shell model is relevant when reading chapters 3 to 5, and the shell model is used without derivation in chapters 6 to 8. In section 2.4, backgrounds on numerical structural analysis are given. In particular, the background on the Arc-Length Method (ALM) is useful when reading chapters 3, 5 and 6, the background on static solvers is useful for chapter 4 and the background on modal and buckling analysis is useful in chapters 5 and 7. The structural analysis routines are briefly summarised in chapter 8 for the sake of reference within that chapter. Lastly, section 2.5 provides a thorough review of the literature on wrinkling. On the one hand, this section presents the basic principles of the energy balance behind wrinkling (see section 2.5.1), together with previously published experimental investigations to illustrate those (see section 2.5.2). On the other hand, section 2.5.3 gives an overview of approaches for the numerical analysis of wrinkling, as an alternative to the ones presented in this dissertation.

2.A Result Reproduction

Some results in this chapter are obtained using parts of the Geometry + Simulation Modules (G+Smo) [294]. More details about the modules developed along with this thesis can be found in chapter 8, as well as instructions for installation. Table 2.1 provides a list of commands for reproducing the results along with the respective figures in this chapter.

Table 2.1: File name and run arguments required for the reproducibility of the figures in this chapter. Arguments with a single dash (-) require an argument. See chapter 8 for more detail about the software and installation instructions. All executables in this table are from the gsStructuralAnalysis module, and the path to the XML files mentioned is gsStructuralAnalysis/filedata/pde/.

Figure	**Run File**		
	Arg.	**Description**	**Values**
Figure 2.10	static_shell_multipatch_XML		
	-i	Input file name	TUDFlame_5p_horizontal.xml
	-r	Number of uniform refinements	1
	-m	Unstructured spline construction to use	1: Almost-C^1
Figure 2.11	modal_shell_multipatch_XML		
	-N	Number of modes to compute	4
	-i, -r, -m	As above	
Figure 2.12	buckling_shell_multipatch_XML		
	-i, -r, -m, -N	As above	

3

Stretch-Based Hyperelastic Material Modelling for Isogeometric Kirchhoff–Love Shells

The first chapter of this dissertation elaborates on the modelling of hyperelastic materials under large strains within the isogeometric Kirchhoff–Love shell. In previous works, formulations for hyperelastic material modelling with Kirchhoff–Love shells have been presented. However, these formulations primarily focus on material definitions using so-called invariants. *To enable isogeometric thin shell modelling of rubber-like materials, typically defined using stretch-based material formulations, this chapter presents stretch-based material formulations for hyperelastic isogeometric Kirchhoff–Love shells. Starting from a strain energy density function defined in terms of stretches, all derivatives with respect to the stretches are computed analytically and transformed to local curvilinear coordinates, based on the fact that principal stretches and their directions are a result of a spectral decomposition of the deformation tensor. Using several numerical benchmarks, the formulations presented in this chapter are verified with analytical solutions and benchmark results from previous works. In addition, the model is applied to simulate the collapsing behaviour of a truncated conical shell and the tension wrinkling of a thin sheet. If necessary, the reader is referred to section 2.3.2 for a full derivation of the isogeometric Kirchhoff–Love shell model, to section 2.4.4 for more information about the arc-length method and to figure 2.18 in section 2.5 for more backgrounds related to the tension wrinkling case.*

This chapter is published as:
[587] **H. M. Verhelst**, M. Möller, J. H. Den Besten, A. Mantzaflaris & M.L. Kaminski, “Stretch-Based Hyperelastic Material Formulations for Isogeometric Kirchhoff–Love Shells with Application to Wrinkling”, *Computer-Aided Design* **139**, 103075 (2021)

3.1 Introduction

To model phenomena like wrinkling in membranes [91, 92, 191, 427, 601] or the deformation of biological tissues [253, 480, 644], thin shell formulations with non-linear hyperelastic material models are typically used. These material models are defined using a *strain energy (density) function*, which measures the strain energy stored in the material when deformed [418]. Material models with strain energy density functions based on the invariants (i.e., *invariant-based* models) of the deformation tensor, such as the Neo-Hookean or the Mooney-Rivlin formulations, have been widely used to study the wrinkling or deformation of biological tissues. However, for rubber materials or living organs such as the liver, spine, skin, rectum, bladder, or aorta, material models defined by the eigenvalues and eigenvectors of the deformation tensor (i.e., *stretch-based* models) such as the Ogden, Sharriff, or exponential and logarithmic models [138, 254, 522] provide better accuracy of material characteristics with respect to experimental tests [93, 377, 500].

To include hyperelastic material models in shell formulations, derivatives of the strain energy density function with respect to the components of the deformation tensor are required to define the stress and material tensors. For invariant-based models, this is generally a straight-forward exercise since the invariants of the deformation tensor are defined in terms of the components of the deformation tensor. However, for stretch-based models, these derivatives result in stress and material tensors defined in the spectral basis (i.e., in terms of the eigenvectors), making the incorporation of these models non-trivial. The first incorporation of stretch-based material models in the Finite Element Method (FEM) was obtained for axisymmetric problems [171, 513], and later the extension to generally shaped shells was made [23, 215, 253]. In these works, either closed-form expressions of the tangents of the principal stretches [23, 513] were obtained or explicit computation of principal directions and values [171, 462, 526] was performed. In the former case, an unknown stretching parameter is used, which can be eliminated for incompressible models [22] and, in fact, imposes numerical difficulties when applied to compressible shells [22, 75]. In the latter case, principal directions and values need to be solved using an eigenvalue problem, and a tensor transformation is required. However, for compressible materials, no additional parameters are required.

With the advent of isogeometric analysis (IGA) [268], new spline-based shell formulations have been presented [44, 256, 319]. The advantage of these formulations is that the geometry is exactly preserved after discretisation and that arbitrary continuity of the basis functions across element boundaries provides high convergence rates and allows for achieving necessary continuities in variational formulations, for instance, leading to rotation-free Kirchhoff-Love shell formulations [318, 319]. These formulations have been used to advance the development of refinement splines [88] and to optimise shell structures [28], amongst other developments. A general hyperelastic isogeometric shell formulation has been developed for general compressible and incompressible material models [320], and specific formulations for biological membranes have been obtained [555]. Roohbakashan and Sauer [480] developed formulations to eliminate numerical through-thickness integration for hyperelastic Kirchhoff-Love shells. Isogeometric Kirchhoff-Love shell formulations are successfully used for biomedical applications to model aortic valve closure [394] and bioprosthetic heart valve dynamics [258, 635] as well as for industrial applications to perform buckling, vibration, and nonlinear deformation analyses of composite wind turbine blades [37, 238]. However, all advances in [238, 320, 394, 480, 555, 635] employ the derivatives of the strain energy density function with respect to the components of the deformation tensor; thus, application of these works is possible for invariant-based models. On the other hand, stretch-based models such as the Ogden model require specific treatment of the spectral deformation tensor for the existing generalised formulations. Contrary to the aforementioned developments in the FEM context, stretch-based material models have not yet been considered in isogeometric Kirchhoff-Love shell formulations.

This chapter presents mathematical formulations for the incorporation of stretch-based material models in the isogeometric Kirchhoff-Love shell model for (in)compressible material models. This enables the use of stretch-based material models such as the Ogden model together with the efficient Kirchhoff-Love shell formulation in isogeometric analysis for applications in wrinkling analysis or biomechanical simulations. The formulations hold for material models defined for 3D continua that are integrated over the shell thickness. Explicit determination of the principal directions and values is employed, applicable to compressible and incompressible materials. The tensor transformation from the spectral to the curvilinear basis – which is needed for compatibility with existing codes – implies additional computational costs compared to a component-based formulation. These costs are minimised by using the minor and major symmetry of the hyperelastic material tensor. Besides comparison with analytical solutions, the model is applied to simulate structural instabilities: the collapse of a truncated cone [23] and the wrinkling phenomenon in a stretched sheet. These instabilities are captured with (extended) arc-length methods [125, 632], combined with IGA [583]. The former simulation reveals the complex collapse behaviour of the truncated cone when using the arc-length method, something that was not reported in the literature before. For the latter simulation, this chapter reports the first IGA results for this case, compared to results from commercial FEM packages.

Following the introduction of notations, preliminary identities, and the isogeometric Kirchhoff-Love shell formulation backgrounds (section 3.2), stretch-based formulations including numerical procedures are derived (section 3.3), and the isogeometric Kirchhoff-Love shell implementation aspects are discussed (section 3.4). The model is benchmarked with analytical or reference solutions, and it is applied to model the collapse behaviour of a truncated cone and the wrinkling formation in a stretched thin sheet in section 3.5. Concluding remarks follow in section 3.6.

3.2 The Kirchhoff–Love Shell Model

The Kirchhoff-Love shell element formulation is based on the Kirchhoff Hypothesis, that is, the cross-section does not shear and orthogonal vectors in the undeformed configuration remain orthogonal after deformation. As a consequence, any point in the shell can be represented by a point on the mid-surface and a contribution in the normal direction:

$$\boldsymbol{x} = \boldsymbol{r} + \theta^3 \hat{\boldsymbol{a}}_3, \tag{3.1}$$

with the shell mid-surface $\boldsymbol{r}(\theta^1,\theta^2)$ and the unit normal direction $\hat{\boldsymbol{a}}_3(\theta^3)$ for the deformed configuration $\boldsymbol{x}(\theta^1,\theta^2,\theta^3)$. For the undeformed configuration $\mathring{\boldsymbol{x}}$, the same relation holds with all quantities decorated with a $\mathring{\cdot}$. The parametrisation utilises surface coordinates θ^α and the through-thickness coordinate θ^3. Derivatives with respect to these coordinates are denoted by $(\cdot)_{,i} = \partial(\cdot)/\partial\theta^i$.

The covariant basis of the mid-surface is represented by $\boldsymbol{a}_i$:

$$\boldsymbol{a}_\alpha = \frac{\partial \boldsymbol{r}}{\partial \theta^\alpha}, \qquad \hat{\boldsymbol{a}}_3 = \frac{\boldsymbol{a}_1 \times \boldsymbol{a}_2}{|\boldsymbol{a}_1 \times \boldsymbol{a}_2|}, \tag{3.2}$$

and the first fundamental form is $a_{\alpha\beta} = \boldsymbol{a}_\alpha \cdot \boldsymbol{a}_\beta$. The curvature tensor $\boldsymbol{b} = b_{\alpha\beta}\, \boldsymbol{a}^\alpha \otimes \boldsymbol{a}^\beta$ is represented by the second fundamental form of surfaces, which can be obtained using the Hessian of the surface $\boldsymbol{a}_{\alpha,\beta}$ or the derivative of the normal vector $\hat{\boldsymbol{a}}_{3,\alpha}$

$$b_{\alpha\beta} = \hat{\boldsymbol{a}}_3 \cdot \boldsymbol{a}_{\alpha,\beta} = -\boldsymbol{a}_{3,\beta} \cdot \boldsymbol{a}_\alpha. \tag{3.3}$$

The derivative of the normal vector is obtained by Weingarten's formula $\hat{\boldsymbol{a}}_{3,\alpha} = -b^\beta_\alpha \boldsymbol{a}_\beta$ with $b^\beta_\alpha = a^{\alpha\gamma} b_{\gamma\beta}$ as the mixed curvature tensor [490]. Taking the derivative of equation (3.1), the covariant basis of the shell coordinate system $\boldsymbol{x}$ can be formulated as follows:

$$\boldsymbol{g}_\alpha = \boldsymbol{x}_{,\alpha} = \boldsymbol{a}_\alpha + \theta^3 \hat{\boldsymbol{a}}_{3,\alpha}, \qquad \boldsymbol{g}_3 = \boldsymbol{x}_{,3} = \hat{\boldsymbol{a}}_3. \tag{3.4}$$

The metric coefficients are constructed by taking the inner product of these basis vectors, i.e.

$$g_{\alpha\beta} = \boldsymbol{g}_\alpha \cdot \boldsymbol{g}_\beta = a_{\alpha\beta} - 2\theta^3 b_{\alpha\beta} + \left(\theta^3\right)^2 \hat{\boldsymbol{a}}_{3,\alpha} \cdot \hat{\boldsymbol{a}}_{3,\beta}, \tag{3.5}$$

where in the second equality, equation (3.3) is used. Moreover, $g_{\alpha 3} = 0$ and $g_{33} = 1$ [319]. Using the definition of the covariant metric g_{ij}, the contravariant metric g^{ij} and basis vectors $\boldsymbol{g}^i$ can be found:

$$g^{\alpha\beta} = [g_{\alpha\beta}]^{-1}, \qquad \boldsymbol{g}^\alpha = g^{\alpha\beta} \boldsymbol{g}_\beta. \tag{3.6}$$

The third contravariant basis vector $\boldsymbol{g}_3$ is again the normal vector $\hat{\boldsymbol{a}}_3$ since it has unit-length by construction (see equation (3.2)).

Remark 3.2.1. *In the isogeometric Kirchhoff-Love shell formulations [319, 320], the last term in equation* (3.5) *is neglected because of the thin shell assumption, meaning* $(\theta^3)^2$ *takes small values. However, the co- and contravariant basis vectors (*$\boldsymbol{g}_\alpha$ *and* $\boldsymbol{g}^\alpha$*, respectively) are used in the mapping of the stretch-based material matrix onto the contravariant undeformed basis (section 3.4.3). To enable an accurate comparison of the invariant-based and stretch-based formulations, the* $\mathcal{O}((\theta^3)^2)$ *term is not neglected, contrary to previous works [320, 480].*

3.2.1 Shell Kinematics

The deformed and undeformed configurations ($\boldsymbol{x}$ and $\mathring{\boldsymbol{x}}$, respectively) are related to each other by the mid-plane deformation vector $\boldsymbol{u}$ by $\boldsymbol{r} = \mathring{\boldsymbol{r}} + \boldsymbol{u}$ and $\hat{\boldsymbol{a}}_3 = \hat{\boldsymbol{a}}_3(\mathring{\boldsymbol{r}} + \boldsymbol{u})$. However, in both the invariant-based and stretch-based forms that are described in this chapter, the deformations are defined using the undeformed and deformed geometries. In continuum mechanics, the deformation gradient $\mathbf{F}$ and the deformation tensor $\mathbf{C}$ are defined as [24, 320]:

$$\mathbf{F} = \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\mathring{\boldsymbol{x}}} = \boldsymbol{g}_i \otimes \mathring{\boldsymbol{g}}^i, \quad \mathbf{C} = \mathbf{F}^\top \mathbf{F} = \boldsymbol{g}_i \cdot \boldsymbol{g}_j \, \mathring{\boldsymbol{g}}^i \otimes \mathring{\boldsymbol{g}}^j = g_{ij} \, \mathring{\boldsymbol{g}}^i \otimes \mathring{\boldsymbol{g}}^j. \tag{3.7}$$

Note that the deformation tensor is defined in the contravariant undeformed basis $\mathring{\boldsymbol{g}}^i \otimes \mathring{\boldsymbol{g}}^j$. For Kirchhoff-Love shells, it is known that $g_{\alpha 3} = g_{3\alpha} = 0$, hence, this implies $C_{\alpha 3} = C_{3\alpha} = 0$. Since $g_{33} = 1$, this implies C_{33} to be unity, meaning that the thickness remains constant under deformation. In hyperelastic Kirchhoff-Love shell formulations, the contribution of C_{33} is therefore incorporated by *static condensation*, where the correction of C_{33} is performed analytically for incompressible materials and iteratively for compressible materials. Therefore, the deformation tensor $\mathbf{C}$ and its inverse $\bar{\mathbf{C}}$ are denoted as:

$$\mathbf{C} = g_{\alpha\beta} \, \mathring{\boldsymbol{g}}^\alpha \otimes \mathring{\boldsymbol{g}}^\beta + C_{33} \, \mathring{\boldsymbol{a}}_3 \otimes \mathring{\boldsymbol{a}}_3, \tag{3.8}$$

$$\bar{\mathbf{C}} = g^{\alpha\beta} \, \mathring{\boldsymbol{g}}_\alpha \otimes \mathring{\boldsymbol{g}}_\beta + \frac{1}{C_{33}} \mathring{\boldsymbol{a}}_3 \otimes \mathring{\boldsymbol{a}}_3. \tag{3.9}$$

From equations (3.8) and (3.9), it can be observed that the thickness contribution (index 3) is decoupled from the in-plane contributions (Greek indices α, β). This is a consequence of the Kirchhoff Hypothesis and, therefore is only valid for Kirchhoff-Love shells. Consequently, using the definition $\tilde{\mathbf{C}} = g_{\alpha\beta} \, \mathring{\boldsymbol{g}}^\alpha \otimes \mathring{\boldsymbol{g}}^\beta$, the trace and determinant of $\mathbf{C}$ can be simplified accordingly [252, 631]:

$$\operatorname{tr}\mathbf{C} = \operatorname{tr}\tilde{\mathbf{C}} + C_{33} = g_{\alpha\beta} g^{\alpha\beta} + C_{33}, \tag{3.10}$$

$$\det\{\mathbf{C}\} = \det\{\mathbf{F}\}^2 = J^2 = \frac{\left|g_{\alpha\beta}\right|}{\left|\mathring{g}_{\alpha\beta}\right|} C_{33} = J_0^2 C_{33} = \lambda_1^2 \lambda_2^2 \lambda_3^2, \tag{3.11}$$

where J denotes the *Jacobian determinant* and J_0 is its in-plane counterpart. Furthermore, the tensor invariants of $\mathbf{C}$ simplify to:

$$I_1 := \operatorname{tr}\{\mathbf{C}\} = g_{\alpha\beta} \mathring{g}^{\alpha\beta} + C_{33} = \lambda_1^2 + \lambda_2^2 + \lambda_3^2, \tag{3.12}$$

$$\begin{aligned} I_2 &:= \frac{1}{2}\left(\operatorname{tr}\{\mathbf{C}\}^2 - \operatorname{tr}\{\mathbf{C}^2\}\right) = C_{33} g_{\alpha\beta} \mathring{g}^{\alpha\beta} + J_0^2 \\ &= \lambda_1^2 \lambda_2^2 + \lambda_2^2 \lambda_3^2 + \lambda_1^2 \lambda_3^2, \end{aligned} \tag{3.13}$$

$$I_3 := \det\{\mathbf{C}\} = \lambda_1^2 \lambda_2^2 \lambda_3^2, \tag{3.14}$$

where λ_i are the *principal stretches* of the shell and λ_i^2 are the eigenvalues of the deformation tensor $\mathbf{C}$. The squares of the eigenvalues are the roots of the characteristic polynomial:

$$(\lambda_i^2)^3 - I_1 (\lambda_i^2)^2 + I_2 \lambda_i^2 - I_3 = 0. \tag{3.15}$$

Corresponding eigenvectors are denoted by $\boldsymbol{v}_i$, which are normalised to have unit length. The eigenvalue decomposition (or *spectral decomposition*) of the deformation tensor $\mathbf{C}$ can be written as [252, 631]:

$$\mathbf{C} = C_{ij} \mathring{\boldsymbol{g}}^i \otimes \mathring{\boldsymbol{g}}^j = \lambda_i^2 \boldsymbol{v}_i \otimes \boldsymbol{v}_i. \tag{3.16}$$

Where the Einstein summation convention is used. Since C_{33} is decoupled by construction, one can immediately see from equations (3.8) and (3.16) that $\lambda_3 = \sqrt{C_{33}}$ and $\boldsymbol{v}_3 = \mathring{\boldsymbol{a}}_3$.

For the sake of completeness, the definition of the Green-Lagrange strain tensor $\mathbf{E} = E_{\alpha\beta}\,\mathring{\mathbf{g}}^{\alpha} \otimes \mathring{\mathbf{g}}^{\beta}$ from [319, 320] and its decomposition to membrane and bending contributions (ε and κ, respectively), is recalled:

$$\begin{aligned} E_{\alpha\beta} &= \frac{1}{2}\left(g_{\alpha\beta} - \mathring{g}_{\alpha\beta}\right) = \frac{1}{2}\left(\left(a_{\alpha\beta} - \mathring{a}_{\alpha\beta}\right) - 2\theta^3\left(b_{\alpha\beta} - \mathring{b}_{\alpha\beta}\right)\right) \\ &= \varepsilon_{\alpha\beta} + \theta_3\kappa_{\alpha\beta}. \end{aligned} \tag{3.17}$$

Remark 3.2.2. *Following up on remark 3.2.1; the contribution of the $\mathcal{O}((\theta^3)^2)$ term in equation* (3.5) *is neglected in the strain tensor and its derivatives. The $\mathcal{O}((\theta^3)^2)$ term is only included in equation* (3.5) *to ensure equivalence in comparison of the stretch- and invariant-based formulations.*

3.2.2 Variational Formulation

The shell internal and external equilibrium equations in variational form are derived by the principle of virtual work [319, 320]. The variations of internal and external work are defined as:

$$\begin{aligned} \delta W(\boldsymbol{u}, \delta\boldsymbol{u}) &= \delta W^{\text{int}} - \delta W^{\text{ext}} \\ &= \int_{\Omega} \mathbf{N} : \delta\boldsymbol{\varepsilon} + \mathbf{M} : \delta\boldsymbol{\kappa}\,\mathrm{d}\Omega - \int_{\Omega} \boldsymbol{f} \cdot \delta\boldsymbol{u}\,\mathrm{d}\Omega, \end{aligned} \tag{3.18}$$

with $\delta\boldsymbol{u}$ being the virtual displacement, $\delta\boldsymbol{\varepsilon}$ and $\delta\boldsymbol{\kappa}$ the virtual strain components, Ω the mid-surface, and $\mathrm{d}\Omega = \sqrt{\left|\mathring{a}_{\alpha\beta}\right|}\,\mathrm{d}\theta^1\,\mathrm{d}\theta^2$ the differential area in the undeformed configuration, mapped to the integration domain $\Omega^* = [0,1]^2$ using the undeformed mid-plane measure. Furthermore, with slight abuse of notation, the tensors $\mathbf{N} = N^{\alpha\beta}\,\mathring{\mathbf{g}}_{\alpha} \otimes \mathring{\mathbf{g}}_{\beta}$ and $\mathbf{M} = M^{\alpha\beta}\,\mathring{\mathbf{g}}_{\alpha} \otimes \mathring{\mathbf{g}}_{\beta}$ denote the shell normal force and bending moment tensors, respectively, with

$$N^{\alpha\beta} = \int_{[-t/2,t/2]} S^{\alpha\beta}\,\mathrm{d}\theta^3, \quad M^{\alpha\beta} = \int_{[-t/2,t/2]} \theta^3 S^{\alpha\beta}\,\mathrm{d}\theta^3. \tag{3.19}$$

Here, $S^{\alpha\beta}$ denotes the coefficients of the stress tensor following from the constitutive relations that will be derived in section 3.3, and t stands for the shell thickness. The total differentials of the stress resultants are:

$$\begin{aligned} \delta N^{\alpha\beta} &= \int_{[-t/2,t/2]} \mathbb{C}^{\alpha\beta\gamma\delta}\,\mathrm{d}\theta^3\,\delta\varepsilon_{\gamma\delta} + \int_{[-t/2,t/2]} \mathbb{C}^{\alpha\beta\gamma\delta}\theta^3\,\mathrm{d}\theta^3\,\delta\kappa_{\gamma\delta}, \\ \delta M^{\alpha\beta} &= \int_{[-t/2,t/2]} \mathbb{C}^{\alpha\beta\gamma\delta}\theta^3\,\mathrm{d}\theta^3\,\delta\varepsilon_{\gamma\delta} + \int_{[-t/2,t/2]} \mathbb{C}^{\alpha\beta\gamma\delta}\left(\theta^3\right)^2\mathrm{d}\theta^3\,\delta\kappa_{\gamma\delta}. \end{aligned} \tag{3.20}$$

Discretizing the equations using known formulations from previous publications [319, 320, 490], the solution $\boldsymbol{u}$ is represented by a finite sum of weighted basis functions, and the tensors $\mathbf{N}$, $\mathbf{M}$, $\boldsymbol{\varepsilon}$, and $\boldsymbol{\kappa}$ are linearised around the weights using Gateaux derivatives. The linearised tensors are denoted by $(\cdot)' = \frac{\partial(\cdot)}{\partial u_r}$ in the following, where u_r are individual weights of the solution vector. Note that $\boldsymbol{u}'$ denotes the basis functions [320]. Using the discretised system, the residual vector is defined by:

$$R_r = F_r^{\text{int}} - F_r^{\text{ext}} = \int_\Omega \mathrm{N} : \frac{\partial \boldsymbol{\varepsilon}}{\partial u_r} + \mathrm{M} : \frac{\partial \boldsymbol{\kappa}}{\partial u_r}\,\mathrm{d}\Omega - \int_\Omega \boldsymbol{f} \cdot \frac{\partial \boldsymbol{u}}{\partial u_r}\,\mathrm{d}\Omega, \tag{3.21}$$

and must be equal to the zero vector for the weights $\boldsymbol{u}$ corresponding to the exact solution. To solve the residual equation $\boldsymbol{R} = 0$, another linearisation is performed, yielding the Jacobian matrix or *tangential stiffness matrix* K:

$$\begin{aligned} K_{rs} &= K_{rs}^{\text{int}} - K_{rs}^{\text{ext}} \\ &= \int_\Omega \frac{\partial \mathrm{N}}{\partial u_s} : \frac{\partial \boldsymbol{\varepsilon}}{\partial u_r} + \mathrm{N} : \frac{\partial^2 \boldsymbol{\varepsilon}}{\partial u_r \partial u_s} + \frac{\partial \mathrm{M}}{\partial u_s} : \frac{\partial \boldsymbol{\kappa}}{\partial u_r} + \mathrm{M} : \frac{\partial^2 \boldsymbol{\kappa}}{\partial u_r \partial u_s}\,\mathrm{d}\Omega \\ &\quad - \int_\Omega \frac{\partial \boldsymbol{f}}{\partial u_s} \cdot \frac{\partial \boldsymbol{u}}{\partial u_r}\,\mathrm{d}\Omega. \end{aligned} \tag{3.22}$$

Note that the matrix contains a contribution for the external load depending on the solution vector ($\boldsymbol{f}(\boldsymbol{u})$). For instance, follower pressures are defined by $\boldsymbol{f}(\boldsymbol{u}) = p\hat{\boldsymbol{a}}_3(\boldsymbol{u})$, where N is the surface normal. In order to solve a nonlinear equation, Newton iterations are performed for solution $\boldsymbol{u}$ and increment $\Delta\boldsymbol{u}$ by solving

$$K\Delta\boldsymbol{u} = -\boldsymbol{R}. \tag{3.23}$$

3.3 Stretch-Based Constitutive Relations

Invariant-based (in)compressible material model formulations have been obtained for the strain energy density functions $\Psi(\mathrm{C})$ in component form based on [320]. However, when experimental material data fitting is involved, a formulation in terms of stretches (i.e., in terms of the eigenvalues of C, $\Psi(\boldsymbol{\lambda})$ with $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \lambda_3)$ might be preferred, meaning that a transformation to spectral form is required. Therefore, this section provides the main contribution of this chapter: the generalised formulations for the implementation of stretch-based material models in the isogeometric Kirchhoff-Love shell model. Throughout this section, reference is made to the equations of [320] for comparison purposes.

The section is structured as follows: section 3.3.1 provides the basics for the derivation of the stretch-based constitutive relations. Thereafter, section 3.3.2 and section 3.3.3 provide the derivations for incompressible and compressible material models, respectively, in the stretch-based formulations. These formulations are the novelty of the present chapter.

3.3.1 General Relations

Assuming $\Psi(\boldsymbol{\lambda})$, relations for the stress and material tensor are derived in terms of the (normalised) eigenvector bases (equation (3.16)):

$$\mathrm{S} = \sum_{i,j=1}^{3} S^{ij}\,\boldsymbol{v}_i \otimes \boldsymbol{v}_j, \quad \boldsymbol{\mathcal{C}} = \sum_{i,j,k,l=1}^{3} \mathcal{C}^{ijkl}\,\boldsymbol{v}_i \otimes \boldsymbol{v}_j \otimes \boldsymbol{v}_k \otimes \boldsymbol{v}_l. \tag{3.24}$$

These equations are valid for 3D continua and hence need to be modified to incorporate the through-thickness stress components. Reading equation (3.8), $C_{\alpha\beta} = g_{\alpha\beta}$ but $C_{33} \neq g_{33} = 1$ to avoid violation of the plane stress condition. To correctly incorporate the plane-stress condition ($S_{33} = 0$), the material tensor $\mathcal{C}$ is modified using static condensation, which implies that the material tensor $\hat{\mathcal{C}}$ corrected for plane-stress is defined by [320]:

$$\hat{\mathcal{C}}^{\alpha\beta\gamma\delta} = \mathcal{C}^{\alpha\beta\delta\gamma} - \frac{\mathcal{C}^{\alpha\beta 33}\mathcal{C}^{33\delta\gamma}}{\mathcal{C}^{3333}}. \tag{3.25}$$

For incompressible materials, this term is derived analytically using the incompressibility condition ($J = 1$), whereas for compressible materials, it is corrected iteratively.

When $\mathbf{S}$ and $\mathcal{C}$ are known, these tensors are transformed to the bases $\mathring{\mathbf{g}}_i \otimes \mathring{\mathbf{g}}_j$ and $\mathring{\mathbf{g}}_i \otimes \mathring{\mathbf{g}}_j \otimes \mathring{\mathbf{g}}_k \otimes \mathring{\mathbf{g}}_l$, respectively. This will be discussed in section 3.4.3.

The derivative of any scalar function with respect to the deformation tensor $\mathbf{C}$ can be written as a derivative with respect to the stretch by applying the chain rule [252]:

$$\frac{\partial(\cdot)}{\partial \mathbf{C}} = \sum_{i=1}^{3} \frac{\partial(\cdot)}{\partial\lambda_i^2}\frac{\partial\lambda_i^2}{\partial\mathbf{C}} = \sum_{i=1}^{3} \frac{\partial(\cdot)}{\partial\lambda_i^2}\mathbf{v}_i \otimes \mathbf{v}_i = \sum_{i=1}^{3} \frac{1}{2\lambda_i}\frac{\partial(\cdot)}{\partial\lambda_i}\mathbf{v}_i \otimes \mathbf{v}_i. \tag{3.26}$$

From this, it follows that:

$$S^{ij} = \begin{cases} \dfrac{1}{\lambda_i}\dfrac{\partial\Psi}{\partial\lambda_i}, & i = j \\ 0, & i \neq j \end{cases} \tag{3.27}$$

which shows that the coefficients of the stress tensor are purely diagonal, and thus S^{ii}, $i = 1, \ldots, 3$ refers to the non-sero components of $\mathbf{S}$.

Remark 3.3.1. *From equation* (3.16) *and equation* (3.26), *it follows that*

$$\frac{\partial\mathbf{C}}{\partial(\lambda_i^2)} = \mathbf{v}_i \otimes \mathbf{v}_i = \frac{\partial\lambda_i^2}{\partial\mathbf{C}}. \tag{3.28}$$

Due to the fact that the eigenvector basis with $\mathbf{v}_i$ is orthogonal and normalised (i.e., orthonormal), the product of the basis vectors $\mathbf{v}_i$ spans the identity tensor: $\mathbf{I} = \mathbf{v}_i \otimes \mathbf{v}_i$.

Furthermore, it can also be shown that for the material tensor, the following holds [23, 215, 252, 462, 513]:

$$\mathcal{C}^{ijkl} = \frac{1}{\lambda_k}\frac{\partial S^{ii}}{\partial\lambda_k}\delta_i^j\delta_k^l + \frac{S^{jj} - S^{ii}}{\lambda_j^2 - \lambda_i^2}(\delta_i^k\delta_j^l + \delta_i^l\delta_j^k)(1 - \delta_i^j). \tag{3.29}$$

where the indices (i, j, k, l) refer to specific components of the fourth-order material tensor, thus no summation over the indices is applied. The first part of $\mathcal{C}^{ijkl}$ represents the normal components (diagonal elements), and the second part denotes the shear components (off-diagonal elements). In the formulation of the component-based counterpart of this equation ([320, eq. 36]), these parts are not explicitly visible since the spectral form by definition uses the principal directions of the deformation tensor, whereas shear and normal contributions are mixed in the curvilinear form of the material tensor. Note that for the second part of this equation, the case $\lambda_i = \lambda_j$ results in an undefined result. Hence, using L'Hopital's rule, this limit case can be identified:

$$\lim_{\lambda_j \to \lambda_i} \frac{S^{jj} - S^{ii}}{\lambda_j^2 - \lambda_i^2} = \lim_{\lambda_j \to \lambda_i} \frac{\dfrac{\partial S^{jj}}{\partial \lambda_j} - \dfrac{\partial S^{ii}}{\partial \lambda_j}}{2\lambda_j} = \frac{1}{2\lambda_i}\left(\frac{\partial S^{jj}}{\partial \lambda_j} - \frac{\partial S^{ii}}{\partial \lambda_j}\right). \tag{3.30}$$

Since $J = \lambda_1 \lambda_2 \lambda_3$, the derivatives of J are:

$$\frac{\partial J}{\partial \lambda_i} = \frac{J}{\lambda_i}, \qquad \frac{\partial^2 J}{\partial \lambda_j \partial \lambda_j} = (1 - \delta_i^j)\frac{J}{\lambda_i \lambda_j}. \tag{3.31}$$

3.3.2 Incompressible Material Models

For incompressible materials, the incompressibility condition ($J = 1$) is enforced using a Lagrange multiplier p in the strain energy density function [252, 320]:

$$\Psi(\lambda_i) = \Psi_{el}(\lambda_i) - p(J - 1). \tag{3.32}$$

where Ψ_{el} is the original strain energy density function. Using equation (3.27), the stress tensor becomes:

$$S^{ii} = \frac{1}{\lambda_i}\left(\frac{\partial \Psi_{el}}{\partial \lambda_i} - \frac{\partial p}{\partial \lambda_i} - p\frac{\partial J}{\partial \lambda_i}\right). \tag{3.33}$$

Again, summation over repeated indices is not performed. Comparing S^{ii} with the component-based result in [320, eq. 41] shows that all components can easily be obtained using substitution in equation (3.26). To comply with the plane-stress condition ($S^{33} = 0$), the equation to be solved for the Langrange multiplier p using the incompressibility condition ($J = 1$) denotes:

$$\frac{1}{\lambda_3}\left(\frac{\partial \Psi_{el}}{\partial \lambda_3} - p\frac{\partial J}{\partial \lambda_3}\right) = 0, \tag{3.34}$$

which implies, using the derivative of J from equation (3.31):

$$p = \left(\frac{\partial J}{\partial \lambda_3}\right)^{-1} \frac{\partial \Psi_{el}}{\partial \lambda_3} = \lambda_3 \frac{\partial \Psi_{el}}{\partial \lambda_3}. \tag{3.35}$$

It can easily be shown that equation (3.35) is similar to the expression of p in the component-based form [320, eq. 46] using $\lambda_3^2 = C_{33}$. The derivative of the stress tensor with respect to the stretch is required to find the material tensor, as observed in equation (3.29). From equation (3.33) it follows that:

$$\begin{aligned}\frac{\partial S^{ii}}{\partial\lambda_j} &= \frac{\partial}{\partial\lambda_j}\left(\frac{1}{\lambda_i}\frac{\partial\Psi}{\partial\lambda_i}\right) = \frac{1}{\lambda_i}\frac{\partial^2\Psi}{\partial\lambda_i\partial\lambda_j} - \delta_i^j\frac{1}{\lambda_i^2}\frac{\partial\Psi}{\partial\lambda_i}\\ &= \frac{1}{\lambda_i}\left(\frac{\partial^2\Psi_{el}}{\partial\lambda_i\partial\lambda_j} - \frac{\partial p}{\partial\lambda_i}\frac{\partial J}{\partial\lambda_j} - \frac{\partial p}{\partial\lambda_j}\frac{\partial J}{\partial\lambda_i}\right.\\ &\qquad\left. - p\frac{\partial^2 J}{\partial\lambda_i\partial\lambda_j} - \delta_i^j\frac{1}{\lambda_i}\left(\frac{\partial\Psi_{el}}{\partial\lambda_i} - p\frac{\partial J}{\partial\lambda_i}\right)\right),\end{aligned} \tag{3.36}$$

where the incompressibility condition ($J = 1$) is used again and where no summation over repeated indices is applied. Note that the Kronecker delta δ_i^j covers the case when $i = j$. The derivative of p follows from equation (3.35) and reads:

$$\frac{\partial p}{\partial\lambda_i} = \lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3\partial\lambda_i} + \delta_i^3\frac{\partial\Psi_{el}}{\partial\lambda_3}. \tag{3.37}$$

Again, this result can be compared to its component-based counterpart in [320, eq. 47], and using equation (3.26), it can be observed that these equations are similar. Substituting equations (3.31), (3.35) and (3.37) and $J = 1$ into equations (3.33) and (3.36) then yields:

$$S^{\alpha\alpha} = \frac{1}{\lambda_\alpha}\left(\frac{\partial\Psi_{el}}{\partial\lambda_\alpha} - \frac{\lambda_3}{\lambda_\alpha}\frac{\partial\Psi_{el}}{\partial\lambda_3}\right), \tag{3.38}$$

$$\begin{aligned}\frac{\partial S^{\alpha\alpha}}{\partial\lambda_\beta} = \frac{1}{\lambda_\alpha}\Bigg[&\frac{\partial^2\Psi_{el}}{\partial\lambda_\alpha\partial\lambda_\beta} - \frac{1}{\lambda_\beta}\left(\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3\partial\lambda_\alpha} + \delta_\alpha^3\frac{\partial\Psi_{el}}{\partial\lambda_3}\right)\\ &- \frac{1}{\lambda_\alpha}\left(\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3\partial\lambda_\beta} + \delta_\beta^3\frac{\partial\Psi_{el}}{\partial\lambda_3}\right) - \lambda_3\frac{\partial\Psi_{el}}{\partial\lambda_3}\frac{(1-\delta_\alpha^\beta)}{\lambda_\alpha\lambda_\beta}\\ &- \delta_\alpha^\beta\frac{1}{\lambda_\alpha}\left(\frac{\partial\Psi_{el}}{\partial\lambda_\alpha} - \frac{1}{\lambda_\alpha}\lambda_3\frac{\partial\Psi_{el}}{\partial\lambda_3}\right)\Bigg].\end{aligned} \tag{3.39}$$

Here, summation over repeated indices is not performed. Comparison with the invariant-based formulation shows that λ_i^{-1} in front of the second term in equation (3.38) translates to $\bar{C}^{ij}$ in [320, eq. 49]. Using these identities, the material tensor can be derived from equation (3.29). For the static condensation term, reference is made to equation (3.25), hence the components $\mathscr{C}^{\alpha\beta33}$, $\mathscr{C}^{33\alpha\beta}$, and $\mathscr{C}^{3333}$ need to be evaluated. From equation (3.29) it follows that:

$$\mathscr{C}^{\alpha\beta33} = \frac{1}{\lambda_3}\frac{\partial S^{\alpha\alpha}}{\partial\lambda_3}\delta_\alpha^\beta = -\frac{1}{\lambda_3\lambda_\alpha^2}\left[\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3^2} + 2\frac{\partial\Psi_{el}}{\partial\lambda_3}\right]\delta_\alpha^\beta, \tag{3.40}$$

$$\mathscr{C}^{33\gamma\delta} = \frac{1}{\lambda_\gamma}\frac{\partial S^3}{\partial\lambda_\gamma}\delta_\gamma^\delta = -\frac{1}{\lambda_3\lambda_\gamma^2}\left[\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3^2} + 2\frac{\partial\Psi_{el}}{\partial\lambda_3}\right]\delta_\gamma^\delta, \tag{3.41}$$

$$\mathscr{C}^{3333} = \frac{1}{\lambda_3}\frac{\partial S^3}{\partial\lambda_3} = -\frac{1}{\lambda_3^3}\left[\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3^2} + 2\frac{\partial\Psi_{el}}{\partial\lambda_3}\right], \tag{3.42}$$

such that the static condensation term becomes:

$$\frac{\mathscr{C}^{\alpha\beta 33}\mathscr{C}^{33\gamma\delta}}{\mathscr{C}^{3333}} = -\frac{\frac{1}{\lambda_3^2\lambda_\alpha^2\lambda_\gamma^2}\left[\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3^2}+2\frac{\partial\Psi_{el}}{\partial\lambda_3}\right]^2}{\frac{1}{\lambda_3^3}\left[\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3^2}+2\frac{\partial\Psi_{el}}{\partial\lambda_3}\right]}\delta_\alpha^\beta\delta_\gamma^\delta \tag{3.43}$$

$$= -\frac{1}{\lambda_\alpha^2\lambda_\gamma^2}\left[\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3^2}+2\frac{\partial\Psi_{el}}{\partial\lambda_3}\right]\delta_\alpha^\beta\delta_\gamma^\delta. \tag{3.44}$$

Using this result, the in-plane incompressible material tensor can be evaluated as:

$$\mathscr{C}^{\alpha\beta\gamma\delta} = \frac{1}{\lambda_\gamma}\frac{\partial S^{\alpha\alpha}}{\partial\lambda_\gamma}\delta_\alpha^\beta\delta_\gamma^\delta + \frac{S^{\beta\beta}-S^{\alpha\alpha}}{\lambda_\beta^2-\lambda_\alpha^2}(\delta_\alpha^\gamma\delta_\beta^\delta+\delta_\alpha^\delta\delta_\beta^\gamma)(1-\delta_\alpha^\beta)$$

$$-\frac{1}{\lambda_\alpha^2\lambda_\gamma^2}\left[\lambda_3\frac{\partial^2\Psi_{el}}{\partial\lambda_3^2}+2\frac{\partial\Psi_{el}}{\partial\lambda_3}\right]\delta_\alpha^\beta\delta_\gamma^\delta, \tag{3.45}$$

where the second term should be replaced by equation (3.30) if $\lambda_\alpha = \lambda_\beta$.

3.3.3 Compressible Material Models

For compressible models, the Jacobian determinant J is not necessarily equal to 1. As a consequence, the deformation gradient $\mathbf{F}$ and deformation tensor $\mathbf{C}$ are modified such that $\mathbf{F}$ and $\mathbf{C}$ are a multiplicative decomposition of a volume-changing (*dilational*) part depending on J and a volume-preserving (*distortional*) part depending on the modified deformation gradient and deformation tensors, $\dot{\mathbf{C}}$ and $\dot{\mathbf{F}}$, respectively [188]:

$$\dot{\mathbf{F}} = J^{-\frac{1}{3}}\mathbf{F}, \quad \dot{\mathbf{C}} = J^{-\frac{2}{3}}\mathbf{C}. \tag{3.46}$$

The modified deformation gradient and deformation tensor have determinants that are equal to 1 (corresponding to volume preservation), meaning:

$$\det\{\dot{\mathbf{F}}\} = \dot{\lambda}_1\dot{\lambda}_2\dot{\lambda}_3 = 1, \quad \det\{\dot{\mathbf{C}}\} = 1, \tag{3.47}$$

where the modified principal stretches $\dot{\lambda}_i$ are defined as:

$$\dot{\lambda}_i = J^{-\frac{1}{3}}\lambda_i. \tag{3.48}$$

Consequently, the invariants of the modified deformation tensor $\dot{\mathbf{C}}$ become:

$$\dot{I}_1 = J^{-2/3}I_1, \quad \dot{I}_2 = J^{-4/3}I_2, \quad \dot{I}_3 = 1, \tag{3.49}$$

with I_i the invariants of the deformation tensor $\mathbf{C}$. With $\dot{\mathbf{F}}, \dot{\mathbf{C}}$ and $\dot{I}_k$ as defined above, the strain energy density function $\Psi(\mathbf{C})$ for a compressible material can be described in a decoupled form, separating the response in an isochoric (or distortional) elastic part $\Psi_{\text{iso}}(\dot{\boldsymbol{\lambda}})$ and an volumetric (or dilational) elastic part $\Psi_{\text{vol}}(J)$ [188, 252, 631]:

$$\Psi(\boldsymbol{\lambda}) = \Psi_{\text{iso}}(\dot{\boldsymbol{\lambda}}) + \Psi_{\text{vol}}(J). \tag{3.50}$$

The volumetric elastic part Ψ_{vol} is required to be strictly convex and equal to zero if and only if $J = 1$ and $\dot{\mathbf{C}} = \mathbf{I}$ [252].

For compressible materials, the plane stress condition is incorporated by solving $S_{33} = 0$ for C_{33} using Newton linearisations [320, 324]:

$$S^{33} + \frac{1}{2}\mathscr{C}^{3333}\Delta C_{33} = 0, \tag{3.51}$$

where C_{33} is incrementally updated by $C_{33}^{(n+1)} = C_{33}^{(n)} + \Delta C_{33}^{(n)}$ with the increment on iteration n:

$$\Delta C_{33}^{(n)} = -2\frac{S_{(n)}^{33}}{\mathscr{C}_{(n)}^{3333}}. \tag{3.52}$$

In each iteration, the updated stress tensor $\mathbf{S}$ and material tensor $\mathscr{C}$ can be computed, and iterations are continued until the plane stress condition is satisfied within a certain tolerance, i.e., $\left|S^{33}\right| < \text{tol}$. When converged, static condensation can be performed for the material tensor using equation (3.25). Rather than using $C_{33}^{(0)} = 1$, [320], $C_{33}^{(0)} = J_0^{-2}$ is used for incompressible materials, although the difference between the two approaches is negligible.

Using equation (3.48), any volumetric strain energy density function for incompressible materials can be transformed to its compressible material equivalent by substituting equation (3.48) into equation (3.50) and by selecting a volumetric component Ψ_{vol}. Static condensation (equation (3.25)) is performed before transforming the material tensor.

3.4 Implementation Aspects

In this section, recall the assembly of the nonlinear system for isogeometric Kirchhoff-Love shells (section 3.4.1) as well as the computation of the eigenvalues and eigenvectors of the deformation tensor $\mathbf{C}$ (section 3.4.2) is recalled. Then, details about the transformation of the stress and material tensors $\mathbf{S}$ and $\mathscr{C}$ from spectral to curvilinear bases (section 3.4.3) are provided.

3.4.1 System Assembly

For the implementation of Kirchhoff-Love shells, recall that the vector of internal forces and the tangential stiffness matrix read [319, 320]:

$$F_r^{\text{int}} = \int_\Omega \left(\bar{\mathbf{n}}^\top \frac{\partial \bar{\boldsymbol{\varepsilon}}}{\partial u_r} + \bar{\mathbf{m}}^\top \frac{\partial \bar{\boldsymbol{\kappa}}}{\partial u_r}\right) \mathrm{d}\Omega, \tag{3.53}$$

$$K_{rs} = \int_\Omega \left(\left(\bar{D}^0 \frac{\partial \bar{\boldsymbol{\varepsilon}}}{\partial u_s} + \bar{D}^1 \frac{\partial \bar{\boldsymbol{\kappa}}}{\partial u_s}\right) \frac{\partial \bar{\boldsymbol{\varepsilon}}}{\partial u_r} + \bar{\mathbf{n}}^\top \frac{\partial^2 \bar{\boldsymbol{\varepsilon}}}{\partial u_r \partial u_s}\right. \tag{3.54}$$

$$\left. + \left(\bar{D}^1 \frac{\partial \bar{\boldsymbol{\varepsilon}}}{\partial u_s} + \bar{D}^2 \frac{\partial \bar{\boldsymbol{\kappa}}}{\partial u_s}\right) \frac{\partial \bar{\boldsymbol{\varepsilon}}}{\partial u_r} + \bar{\mathbf{m}}^\top \frac{\partial^2 \bar{\boldsymbol{\kappa}}}{\partial u_r \partial u_s}\right) \mathrm{d}\Omega. \tag{3.55}$$

Here, it should be noted that the matrices $\bar{\mathscr{D}}^k$, $k = 0, 1, 2$, are the k^{th} thickness moments of the material tensor represented as a 3×3 matrix, and $\bar{\mathbf{n}}$ and $\bar{\mathbf{m}}$ are the zero-th and first thickness moments of the stress tensor, see [320]. The thickness integrals are, in the present chapter and in [320], computed using numerical through-thickness integration with four Gaussian points. As discussed in [480], the matrices $\bar{\mathscr{D}}^1$ can differ in the variations of the normal force tensor $\bar{\mathbf{n}}$ and the moment tensor $\bar{\mathbf{m}}$ depending on the analytic projected or directly decoupled alternatives for thickness integration.

3.4.2 Eigenvalue Computation

The eigenvalues of tensor quantity can be computed by solving equation (3.15) or, alternatively, by computing the eigenvalues of the matrix that results from the computation of $\mathrm{C} = C_{ij}\,\mathring{\mathbf{g}}^i \otimes \mathring{\mathbf{g}}^j$, including the outer product. Since $\lambda_3^2 = \sqrt{C_{33}}$ is decoupled by construction, it suffices to compute λ_1^2 and λ_2^2 by computing the eigenvectors and eigenvalues of the 3×3 matrix following from the computation of $\mathrm{C} = C_{\alpha\beta}\,\mathring{\mathbf{g}}_\alpha \otimes \mathring{\mathbf{g}}_\beta$. This computation results in three eigenpairs (eigenvalues and eigenvectors), of which one eigenpair contains the zero vector due to the decoupled construction. The other two eigenpairs ($\lambda_\alpha \in \mathbb{R}, \boldsymbol{v}_\alpha \in \mathbb{R}^3$) are the in-plane principal stretches and their directions.

3.4.3 Tensor Transformation

Since the stretch-based stress and material tensor are derived in spectral form (i.e., in the eigenvector space), a transformation towards the curvilinear basis is required in order to use these entities in further computations. Recall that the spectral forms of S and $\boldsymbol{\mathscr{C}}$ are:

$$\mathrm{S} = \sum_{i=1}^{3} S^{ii}\,\boldsymbol{v}_i \otimes \boldsymbol{v}_i, \qquad \boldsymbol{\mathscr{C}} = \sum_{i,j,k,l=1}^{3} \mathscr{C}^{ijkl}\,\boldsymbol{v}_i \otimes \boldsymbol{v}_j \otimes \boldsymbol{v}_k \otimes \boldsymbol{v}_l. \tag{3.56}$$

The invariant-based stress and material tensors are defined on a curvilinear basis as follows:

$$\mathrm{S} = \sum_{i,j=1}^{3} S^{ij}\,\mathring{\mathbf{g}}_i \otimes \mathring{\mathbf{g}}_j \qquad \boldsymbol{\mathscr{C}} = \sum_{i,j,k,l=1}^{3} \mathscr{C}^{ijkl}\,\mathring{\mathbf{g}}_i \otimes \mathring{\mathbf{g}}_j \otimes \mathring{\mathbf{g}}_k \otimes \mathring{\mathbf{g}}_l. \tag{3.57}$$

Since the strain tensors (c.f. equation (3.17)) are defined in the curvilinear basis, it is convenient to define the quantities in the variational form (c.f. equation (3.18)) defined in the curvilinear basis. Hence, the stretch-based stress and material tensors are transformed to the undeformed covariant curvilinear basis by:

$$\begin{aligned} \tilde{S}^{ij} &= \sum_{p,q=1}^{3} S^{pq}(\boldsymbol{v}_p \cdot \mathring{\mathbf{g}}^i)(\boldsymbol{v}_q \cdot \mathring{\mathbf{g}}^j), \\ \tilde{\mathscr{C}}^{ijkl} &= \sum_{p,q,r,s=1}^{3} \mathscr{C}^{pqrs}(\boldsymbol{v}_p \cdot \mathring{\mathbf{g}}^i)(\boldsymbol{v}_q \cdot \mathring{\mathbf{g}}^j)(\boldsymbol{v}_r \cdot \mathring{\mathbf{g}}^k)(\boldsymbol{v}_s \cdot \mathring{\mathbf{g}}^l), \end{aligned} \tag{3.58}$$

where $\tilde{S}^{ij}$ and $\tilde{\mathscr{C}}^{ijkl}$ are the coefficients of the stress and material tensors in the curvilinear basis.

Obviously, the tensor transformation only needs to be computed for non-zero components of $\mathscr{C}^{pqrs}$. For incompressible material models, the plane-stress correction for C_{33} is applied analytically, which implies that the transformations only need to be applied for indices ranging from $\alpha, \beta, \gamma, \delta = 1, 2$, thus the transformation consists of mapping $2^4 = 16$ entries. However, it is known that for hyperelastic materials, the contravariant components of the material tensor, $\mathscr{C}^{ijkl}$, posses minor and major symmetry [252, 631], i.e.

$$\mathscr{C}^{abcd} = \mathscr{C}^{bacd} = \mathscr{C}^{abdc} \qquad \text{minor symmetry,} \tag{3.59}$$

$$= \mathscr{C}^{cdab} \qquad \text{major symmetry,} \tag{3.60}$$

so that only six unique components exist for the 2 × 2 × 2 × 2 tensor. Furthermore, equation (3.29) implies that the non-zero components of $\mathscr{C}^{ijkl}$ are of the form $\mathscr{C}^{iiii}$, $\mathscr{C}^{iijj}$, $\mathscr{C}^{ijij}$, and $\mathscr{C}^{ijji}$, of which the last two are equal by virtue of the minor symmetry property. This implies that the 2 × 2 × 2 × 2 tensor has only four uniquely defined components, namely $\mathscr{C}^{1111}$, $\mathscr{C}^{1122}$, $\mathscr{C}^{2222}$, and $\mathscr{C}^{1212}$.

For compressible material models, the static condensation term is computed on the spectral basis, i.e., on the tensor $\boldsymbol{\mathscr{C}}$, before it is transformed to the covariant undeformed tensor basis. From equation (3.52), it can be seen that the iterative procedure to find C_{33} requires the computation of $\mathscr{C}^{3333}$, $\mathscr{C}^{\alpha\beta33}$, and $\mathscr{C}^{33\alpha\beta}$, where the last two are equal by virtue of the major symmetry property. Reusing the minor and major symmetries, the computation is reduced to four distinct components, namely $\mathscr{C}^{1133}$, $\mathscr{C}^{2233}$, $\mathscr{C}^{1233}$, and $\mathscr{C}^{3333}$.

Accordingly, it can be concluded that for incompressible materials, four and for compressible materials, eight unique components of the spectral material tensor need to be computed when exploiting minor and major symmetry, as well as the nature of equation (3.29). In summary, the transformations give rise to certain additional costs, which can be limited, however, by exploiting symmetry properties efficiently.

3.5 Numerical Experiments

For benchmarking purposes, the results of four numerical experiments have been used for verification and validation of the presented formulations for incompressible and compressible material models. For the uniaxial tension and pressurised balloon benchmarks (sections 3.5.1 and 3.5.2, respectively), analytical solutions are available, therefore they will serve as verification of the stretch-based material model formulations. Combining the present method with (extended) arc-length methods, the collapsing behaviour of a truncated conical shell [23] (section 3.5.3) and the wrinkling of a stretched thin sheet (section 3.5.4) are simulated.

In order to verify the presented isogemetric Kirchhoff–Love formulation for a stretch-based Ogden material with its FEM couterpart, the conical shell collapse (section 3.5.3) is incorporated. Finally, the approach of this chapter is applied to the modelling of the wrinkling of a thin sheet subject to tension. The models have been implemented in the open-source library `G+Smo` (Geometry + Simulation Modules) [294, 376]. More information on the implementation herein is provided in chapter 8 of this dissertation. Download instructions for the code related to this chapter can be found in the supplementary material of [587].

In the numerical experiments, compressible and incompressible formulations of the Neo-Hookean (NH), Mooney-Rivlin (MR), and Ogden (OG) material models have been used. The Neo-Hookean models are given by (compressible and incompressible, respectively):

$$\Psi(\mathbf{C}) = \frac{\mu}{2}\left(J^{-\frac{2}{3}}I_1 - 3\right) + \Psi_{\text{vol}}(J), \tag{3.61}$$

$$\Psi(\mathbf{C}) = \frac{\mu}{2}(I_1 - 3). \tag{3.62}$$

The Mooney-Rivlin models are given by [393, 475] (compressible and incompressible, respectively):

$$\Psi(\mathbf{C}) = \frac{c_1}{2}\left(J^{-\frac{2}{3}}I_1 - 3\right) + \frac{c_2}{2}\left(J^{-\frac{4}{3}}I_2 - 3\right) + \Psi_{\text{vol}}(J), \tag{3.63}$$

$$\Psi(\mathbf{C}) = \frac{c_1}{2}(I_1 - 3) + \frac{c_2}{2}(I_2 - 3). \tag{3.64}$$

For Ogden models, the following formulations are used (compressible and incompressible, respectively):

$$\Psi(\boldsymbol{\lambda}) = \sum_{p=1}^{N} \frac{\mu_p}{\alpha_p} J^{-\frac{1}{3}}\left(\lambda_1^{\alpha_p} + \lambda_2^{\alpha_p} + \lambda_3^{\alpha_p} - 3\right) + \Psi_{\text{vol}}(J), \tag{3.65}$$

$$\Psi(\boldsymbol{\lambda}) = \sum_{q=1}^{3}\left(\sum_{p=1}^{N} \frac{\mu_p}{\alpha_p}(\lambda_q^{\alpha_p} - 1)\right). \tag{3.66}$$

For all models, the following volumetric part of the strain energy density function is adopted:

$$\Psi_{\text{vol}} = K\mathcal{G}(J) = K\beta^{-2}\left(\beta\log(J) + J^{-\beta} - 1\right). \tag{3.67}$$

To check the consistency of invariant-based models (i.e., the NH and MR models), the invariants can be replaced by equations (3.12) to (3.14) to obtain stretch-based forms, which are thus equivalent to the component-based form from [320]. Unless stated otherwise, for the compressible models, $\beta = -2$ and for the Mooney-Rivlin model, $c_1/c_2 = 7$ [475] is used. For the Ogden model, the coefficients from [570] are re-scaled to the value of μ:

$$\begin{aligned}
\mu_1 &= \frac{6.300}{\mu_0}\mu, & \alpha_1 &= 1.3,\\
\mu_2 &= \frac{0.012}{\mu_0}\mu, & \alpha_2 &= 5.0,\\
\mu_3 &= -\frac{0.100}{\mu_0}\mu, & \alpha_3 &= -2.0,
\end{aligned} \tag{3.68}$$

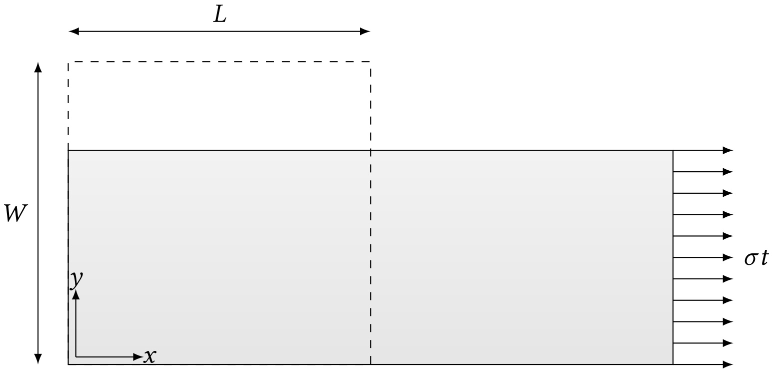


Figure 3.1: Geometry for the uniaxial tension case. The filled geometry represents the deformed configuration, and the dashed line indicates the undeformed geometry. The bottom side of the undeformed sheet is fixed in y-direction and the left side of the sheet is fixed in x-direction. The applied load is σt, where σ is the actual Cauchy stress and t is the thickness of the sheet.

where $\mu_0 = 4.225$.

3.5.1 Uniaxial Tension

The first benchmark case is the uniaxial tension of a material block. A block with dimensions $L \times W \times t = 1 \times 1 \times 0.001\,[\mathrm{m}^3]$ is considered. The shear modulus is $\mu = E/(2(1+\nu))$, where E and ν are the Young's modulus and Poisson ratio, respectively, such that $\mu = 1.5 \cdot 10^6\,[\mathrm{N/m}^2]$. The block is modelled by shell elements, i.e., the $L \times W$ plane is considered, and all edges are restrained in a vertical direction ($z = 0$). The left edge ($x = 0$) is restrained in the x-direction, and on the right edge ($x = L$), a distributed load σt is applied. The bottom edge ($y = 0$) is restrained in the y-direction, and the top edge ($y = B$) is free to move (see figure 3.1).

In figure 3.2, the results for uniaxial tension are depicted. For both compressible and incompressible materials, the analytical solution for the Cauchy stress is obtained from [252, ex. 1]. The numerical and analytical solutions for incompressible and compressible materials show a perfect match for all quantities studied (thickness decrease λ_3, axial Cauchy stress σ, and Jacobian determinant J). Note that the Jacobian determinant for incompressible materials is equal to 1 and hence not shown. The residual norms of the non-linear iteration convergence for the invariant-based and stretch-based Neo-Hookean and Mooney-Rivlin models as well as the stretch-based Ogden model are equal in all iterations (see table 3.1), showing that the present formulation provides exactly the same rates of convergence as the invariant-based method. Last but not least, Newton iterations converge with optimal speed (second-order convergence rate).

3.5.2 Pressurised Balloon

The response of a pressurised spherical balloon is used for benchmarking purposes as well. The analytical pressure formulation is obtained from [252, eq. 6.132]. The numerical model

Table 3.1: Residual norms per iteration for the 10th load-step for uniaxial tension for all material models in compressible and incompressible forms. For the Neo-Hookean and Mooney-Rivlin models, the iteration residuals are provided for the stretch-based and invariant-based approaches. For the Ogden model, only the results for the stretch-based formulations are given, since no invariant-based formulation exists. For the Neo-Hookean and Mooney-Rivlin models, results are only observed in the last iteration due to the machine precision of the arithmetic.

It.	Neo-Hookean		Mooney-Rivlin		Ogden
	Stretch	Invariant	Stretch	Invariant	Stretch
Incompressible					
1	$2.033 \cdot 10^{-4}$	$2.033 \cdot 10^{-4}$	$4.021 \cdot 10^{-3}$	$3.999 \cdot 10^{-3}$	$4.442 \cdot 10^{-2}$
2	$1.129 \cdot 10^{-6}$	$1.129 \cdot 10^{-6}$	$2.248 \cdot 10^{-5}$	$2.253 \cdot 10^{-5}$	$1.313 \cdot 10^{-6}$
3	$3.575 \cdot 10^{-11}$	$3.575 \cdot 10^{-11}$	$7.106 \cdot 10^{-10}$	$7.229 \cdot 10^{-10}$	$4.149 \cdot 10^{-11}$
4	$2.554 \cdot 10^{-16}$	$6.929 \cdot 10^{-16}$	$5.088 \cdot 10^{-16}$	$1.776 \cdot 10^{-15}$	$1.602 \cdot 10^{-16}$
Compressible					
1	$1.617 \cdot 10^{-3}$	$1.617 \cdot 10^{-3}$	$2.100 \cdot 10^{-3}$	$2.100 \cdot 10^{-3}$	$5.215 \cdot 10^{-3}$
2	$2.296 \cdot 10^{-7}$	$2.296 \cdot 10^{-7}$	$2.890 \cdot 10^{-6}$	$2.890 \cdot 10^{-6}$	$1.759 \cdot 10^{-7}$
3	$9.443 \cdot 10^{-13}$	$9.440 \cdot 10^{-13}$	$1.344 \cdot 10^{-11}$	$1.344 \cdot 10^{-11}$	$2.584 \cdot 10^{-13}$
4	$1.153 \cdot 10^{-15}$	$1.252 \cdot 10^{-16}$	$1.115 \cdot 10^{-15}$	$1.988 \cdot 10^{-16}$	$1.625 \cdot 10^{-15}$

results are based on follower pressures, i.e., $\boldsymbol{f} = p_0 \boldsymbol{a}_3$, where $\boldsymbol{a}_3$ is the unit normal in the current configuration. The balloon is modelled as a quarter of a hemisphere, of which the bottom point is fixed in all directions, and on the sides a symmetry condition is applied by clamping the sides in normal direction and restriction deflections orthogonal to the symmetry boundary (see figure 3.3). The geometry is modelled by 2 elements over the height and 2 elements over the quarter-circumference, both of quadratic order.

For $R = 10\,[\mathrm{m}]$, $t = 0.1\,[\mathrm{m}]$ and $\mu = 4.2255 \cdot 10^5\,[\mathrm{N/m^2}]$, a perfect agreement is obtained for all presented material models in comparison to the analytical solutions figure 3.4.

In table 3.2, the total CPU times are presented, related to system assembly for different material models for different mesh refinement levels and quadratic order for $p_0 = 10^4$. The assembly times for both the invariant-based formulations and for the stretch-based formulations are given for the Neo-Hookean and Mooney-Rivlin material models, whereas the stretch-based formulation is only available for the Ogden model. The total number of nonlinear iterations is the same in all cases, and so is the number of assembly operations. The table shows that the stretch-based formulations are slower than the invariant-based formulations, which is expected given the requirement for the transformation of the basis of the deformation tensor. It can also be seen that the Ogden model requires significantly more CPU time than the other models, which is due to the large number of terms in the strain energy density function.

3.5.3 Conical Shell Collapse

A collapsing conical shell (or *frustrum*) is presented as a benchmark for modelling strong non-linearities [23]. A conical shell with height $H = 1\,[\mathrm{m}]$ and top radius $r = 1\,[\mathrm{m}]$, bottom radius $R = 2\,[\mathrm{m}]$ and thickness $t = 0.1\,[\mathrm{m}]$ as depicted in figure 3.5 is considered. Since

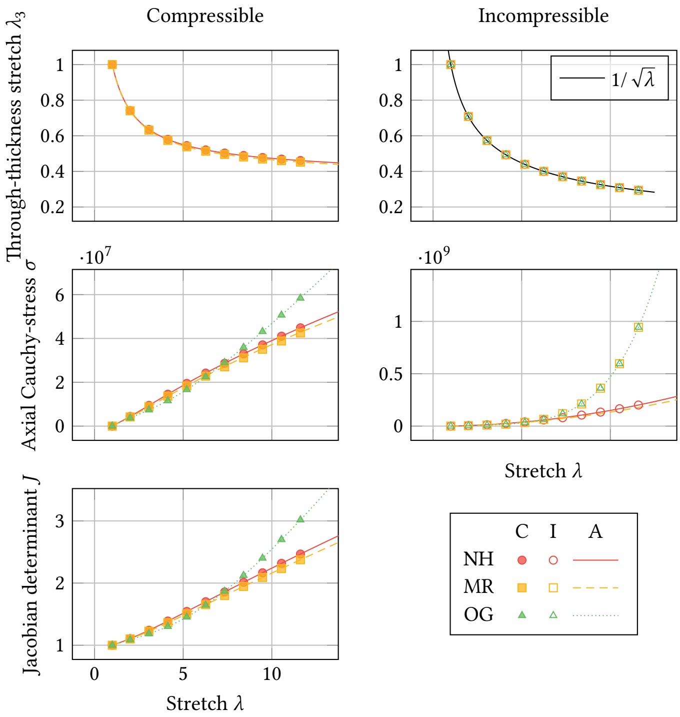


Figure 3.2: Results for uniaxial tension for compressible (C, left column) and incompressible materials (I, right column), where the first row presents the thickness decrease λ_3, the second row the axial Cauchy stress or true axial stress σ, and the last row the Jacobian determinant J for compressible materials, all against the stretch λ. The material models that are used are the Neo-Hookean (NH), the Mooney-Rivlin (MR), and the Ogden (OG) material models, and comparisons are made to analytical (A) solutions from [252, ex. 1].

the reference solution models the frustrum axisymmetrically, a quarter of the geometry is modelled with 32 quadratic elements over the height and one quadratic element over the quarter-circumference to represent axial symmetry. The corresponding material model is of the Ogden type and has the following parameters:

$$\begin{aligned} \mu_1 &= 6.300\,[\mathrm{N/m^2}], & \alpha_1 &= 1.3, \\ \mu_2 &= 0.012\,[\mathrm{N/m^2}], & \alpha_2 &= 5.0, \\ \mu_3 &= -0.100\,[\mathrm{N/m^2}], & \alpha_3 &= -2.0, \end{aligned}$$

implying that $\mu = 4.225\,[\mathrm{N/m^2}]$. Two sets of boundary conditions are considered for this geometry. In both sets, the bottom of the shell (Γ_2) is hinged, hence the displacements are restricted in all directions. The top shell edge (Γ_1) is either kept rigid (no x and y displacements) or free, referred to as *constant* or *variable* radius, respectively [23]. On the top edge, a uniform load p is applied, providing a uniform displacement Δ. Due to symmetry, only one quarter of the geometry is modelled, which means that symmetry boundary conditions are applied on the $x = 0$ and $y = 0$ planes (Γ_3, Γ_4, see figure 3.5), restricting in-plane deformations normal to the boundaries and restricting rotations on the boundary

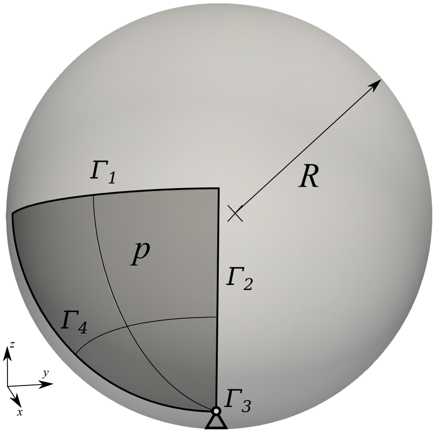


Figure 3.3: Geometry of the inflated balloon with 4 quadratic elements. Symmetry conditions are applied to the boundaries Γ_1, Γ_2, and Γ_4, which means that rotations around these boundaries and displacements in the plane normal to the boundaries are fixed. The bottom of the balloon (Γ_3) is an edge with a radius of 0.01 and is fixed in all directions.

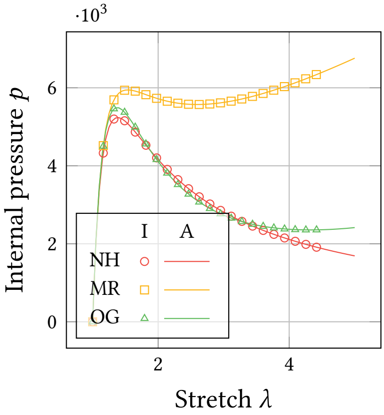


Figure 3.4: Inflation of a balloon. The vertical axis depicts the internally applied pressure, and the horizontal axis depicts the stretch $\lambda_1 = \lambda_2 = \lambda$. The different lines and markers represent different material models, including Neo-Hookean (NH), Mooney-Rivlin (MR), and Ogden (OG). The radius of the sphere is $R = 10\,[\text{m}]$, and the thickness of the sphere is $t = 0.1\,[\text{m}]$.

Table 3.2: Total CPU assembly times (seconds) for the different material models (invariant-based where applicable) for different mesh sizes (#El.) for the inflated balloon benchmark. All results are obtained for the incompressible material models.

#El.	Neo-Hookean		Mooney-Rivlin		Ogden
	Invariant	Stretch	Invariant	Stretch	
1	0.18	0.13	0.18	0.13	0.41
4	0.42	0.28	0.43	0.29	1.07
16	1.42	0.93	1.45	0.94	3.95
64	6.19	4.55	6.69	4.35	18.49
256	40.67	26.77	44.10	28.60	119.65

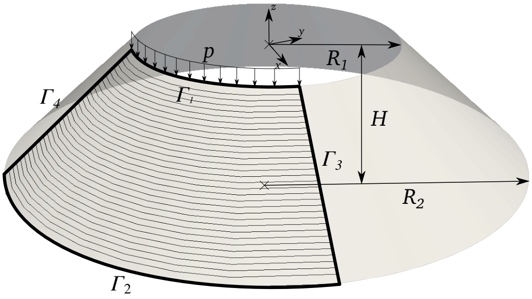


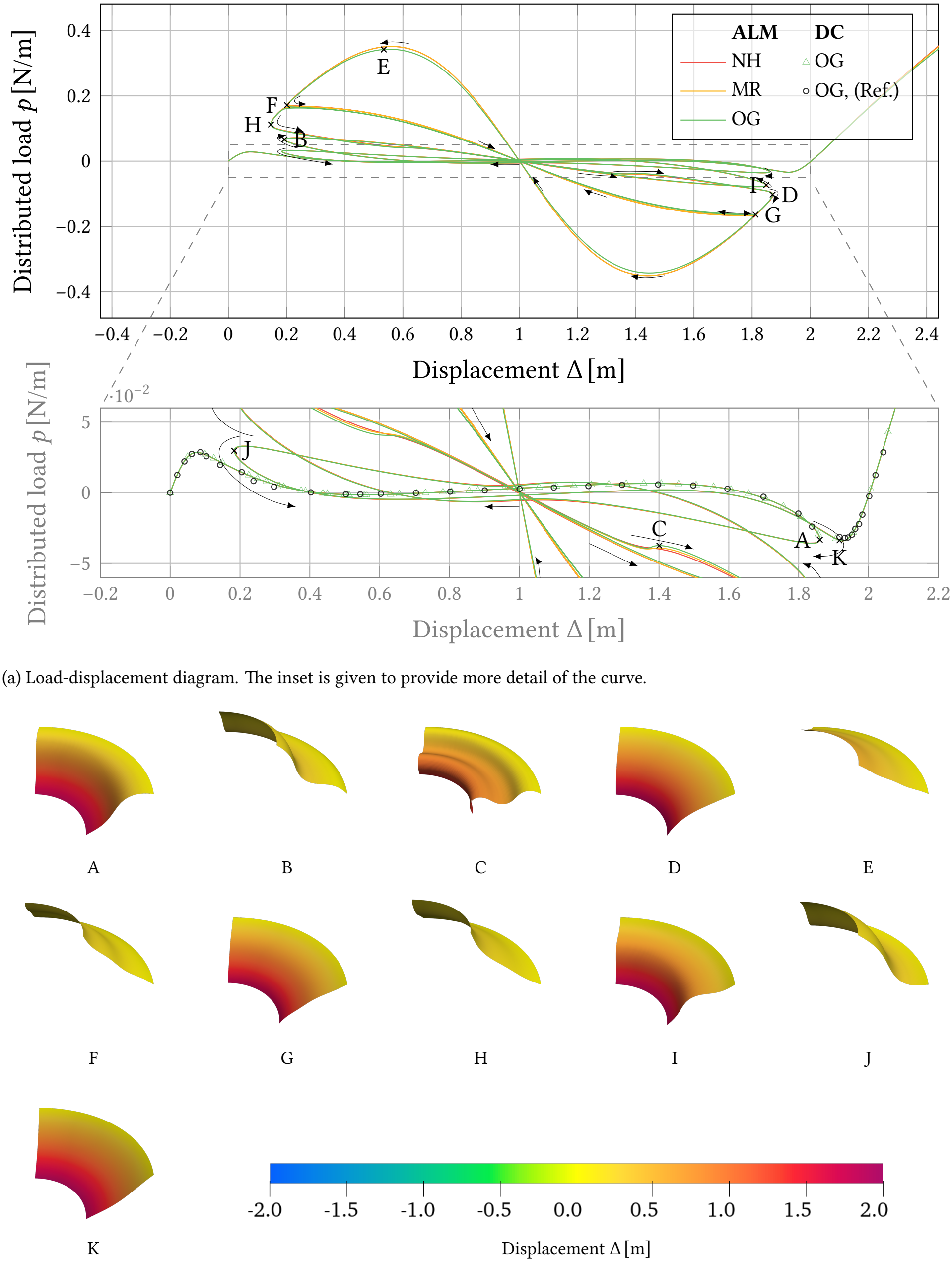
Figure 3.5: Geometry of the collapsing truncated conical shell with top and bottom radii R_1 and R_2, height H and a line load p on Γ_1. On boundaries Γ_3 and Γ_4, symmetry conditions are applied. Boundary Γ_1 is either constrained in x and y directions (fixed radius) or free (variable radius). Boundary Γ_2 is constrained. The geometry is modelled using 32 quadratic elements over the height.

by applying clamped boundary conditions as described in [319]. The quarter-conical shell is modelled with 32 quartic shell elements over the width.

Loads are applied using displacement-control (DC) or arc-length control. In the former case, displacements are applied on the top side of the cone, and the deformation of the cone as well as the corresponding load on the top boundary are computed. In the latter case, Crisfield's spherical arc-length procedure [125] is used with extensions for resolving complex roots [329, 660]. If this method does not converge to an equilibrium point, the step size is bisected until a converged step is found. After this step, the step size is reset to its original value [583].

Figures 3.6 and 3.7 present the result of the collapsing conical shell (constant and variable radius, respectively) of the present study and the reference results from [23]. The results for the displacement-controlled (DC) solution procedure show that the difference between the used material models is negligible since the actual strains are relatively small. The results also agree with the displacement-controlled reference results of [23], and minor differences between the results might be a result of FE shear locking as involved in the reference results. Since more steps have been used for the displacement-controlled calculations, sharp corners in the curve can be observed for $\Delta \sim 1.9$ for constant radius and $\Delta \sim 1.8$ for variable radius.

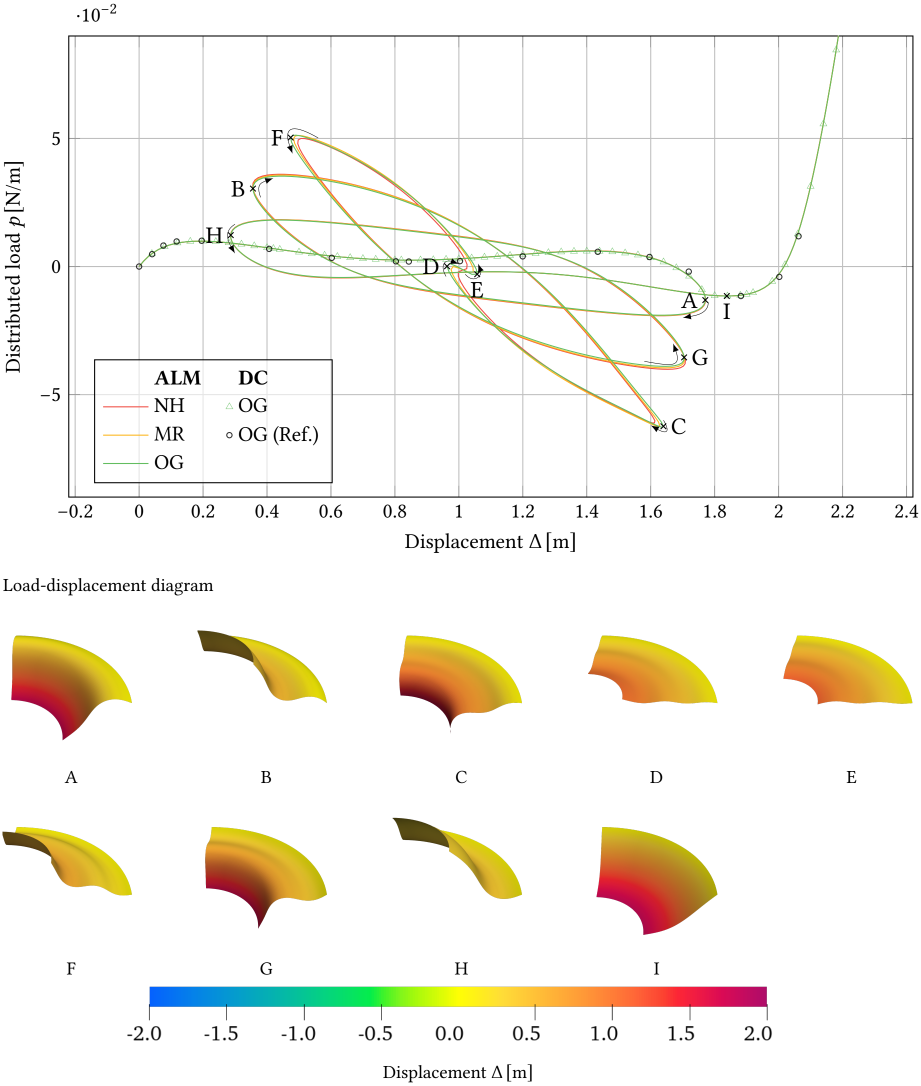
An arc-length-based calculation was used as well. From the results, one can observe the revelation of the collapsing mechanism of the conical shell. For both cases (constant and variable radius), an almost anti-symmetric pattern in the load-deflection space can be observed, which initiates and finishes at the kinks in the curve that was found with the displacement-control procedure. For the constant-radius shell, figure 3.6a shows two loops of large magnitude. In figures 3.6a and 3.7a, it can be seen that the collapsing behaviour of the conical shell consists of states in which multiple *waves* in radial direction occur. For both cases, it can be seen that after the loops with the highest force amplitude,



(a) Load-displacement diagram. The inset is given to provide more detail of the curve.

(b) Intermediate states of the frustrum

Figure 3.6: Result of the collapsing conical shell with *constant* radius; (a) load-displacement diagram; (b) undeformed geometries matching with the points indicated with capital letters in the diagram. The lines represent solutions obtained using the Arc-Length Method (ALM), and the markers represent solutions obtained by displacement control (DC). Note that the solutions for the NH and MR models overlap on most parts of the path. The material models are Neo-Hookean (NH), Mooney-Rivlin (MR), and Ogden (OG). Since the variation between the material models is rather small for the DC solutions, only the results for the OG material model are given. The reference results (Ref.) are obtained from [23].



(a) Load-displacement diagram

(b) Intermediate states of the frustrum

Figure 3.7: Result of the collapsing conical shell with *variable* radius; (a) load-displacement diagram; (b) undeformed geometries matching with the points indicated with capital letters in the diagram. The lines represent solutions obtained using the Arc-Length Method (ALM), and the markers represent solutions obtained by displacement control (DC). The material models are Neo-Hookean (NH), Mooney-Rivlin (MR), and Ogden (OG). Since the variation between the material models is rather small for the DC solutions, only the results for the OG material model are given. The reference results (Ref.) are obtained from [23].

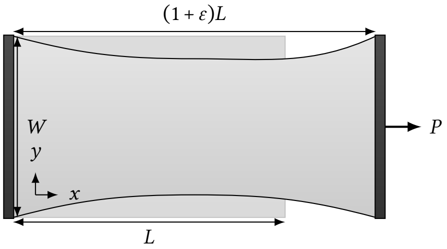


Figure 3.8: Modelling geometry for the uniaxially loaded restrained sheet. The sheet has length L and width W. The strain of the sheet due to load P is denoted by ε.

the shell and its collapse path invert and continue on the path that can be obtained with displacement control.

To the best of the authors' knowledge, the collapsing of a conical shell was not investigated before. Complex load-displacement paths from figures 3.6a and 3.7a show that displacement-controlled simulations in this case ignore the collapsing behaviour of the shell with multiple limit points. The authors highly encourage further investigations into this benchmark for verification and validation.

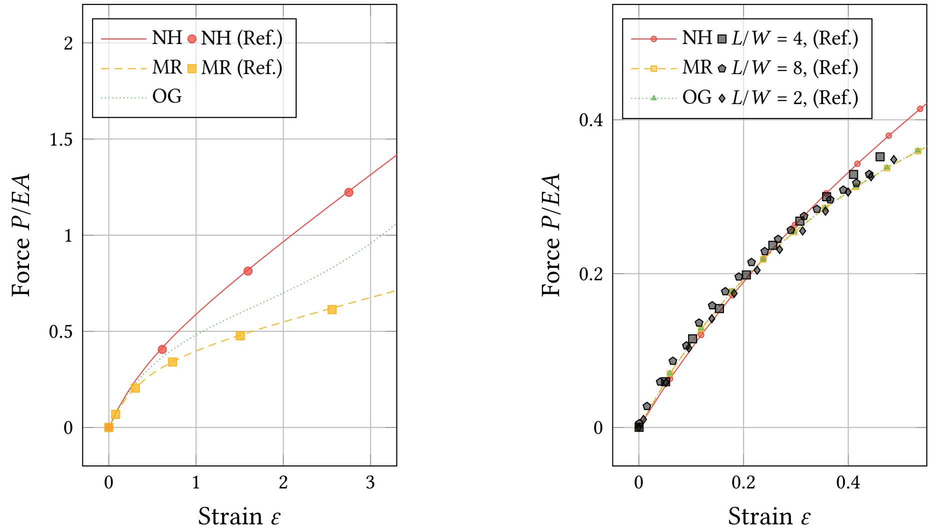
3.5.4 Wrinkling of a Stretched Sheet

As an application of the model, the wrinkling phenomenon of a stretched, thin membrane (see figure 3.11) is considered. Scaling laws based on experiments were first published in [91, 92], and analytical formulations related to this problem were established in [449]. Numerical results to this problem have been established for sheets with different aspect ratios β and different dimensionless thicknesses α [191, 233, 346, 403, 427, 551, 601, 658]. In most numerical studies, Neo-Hookean or Mooney-Rivlin models were used to model the wrinkling phenomenon since strains usually reach high values (typically $\varepsilon \sim 10-50\%$). In this chapter, tension wrinkling is modelled for the sake of benchmarking using incompressible Neo-Hookean, Mooney-Rivlin, and Ogden models and isogeometric Kirchhoff–Love shells, which is a novelty to the best of the authors' knowledge. In the first part of this section, the model is benchmarked on a restrained sheet without wrinkling formation, and material parameter determination is performed. Thereafter, the results of wrinkling simulations are presented.

Material test

Related to the first benchmark in the work of [480] and on the experiments of [427], a tensile load is applied on a strip of which the short edges are fixed and the long edges are free (see figure 3.8). The focus is on the non-domensional load versus end-point displacement in the longitudinal (load and displacement) direction.

Firstly, for the geometric parameters, $L = 9\,[\mathrm{mm}]$, $W = 3\,[\mathrm{mm}]$, and $t = 0.3\,[\mathrm{mm}]$ are used, leading to $L/W = 3$ and $t/W = 0.1$. The material has Poisson's ratio $\nu = 0.5$, and for the NH material model, a Young's modulus of $E = 30\,[\text{kPa}]$ is involved, and for the MR material model, one of $E = 90\,[\text{kPa}]$ leads to $\mu = 10\,[\text{kPa}]$ and $\mu = 30\,[\text{kPa}]$, respectively. For the MR model, $c_1/c_2 = 1/2$, such that $c_1 = 1/9$ and $c_2 = 2/9$. Scaling according to equation (3.68) is applied for the Ogden material model, and 8 × 8 quadratic elements are used. A good match with the results of the directly decoupled method of [480] for the incompressible Neo-Hookean and Mooney-Rivlin models can be observed in figure 3.9a. Note that the forces in the reference paper are normalised by $E = 3c_1$ for both the Neo-Hookean and Mooney-Rivlin models, whereas in the present simulations, the forces are normalised by $E = 3\mu$ (since $\nu = 0.5$ in the comparison with [480]).

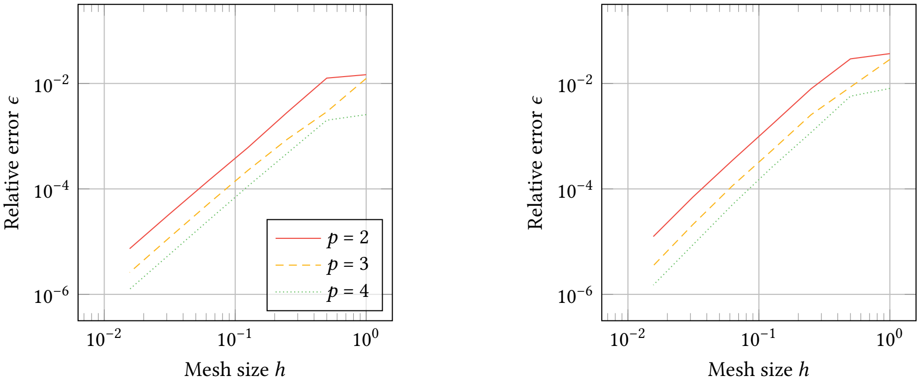


(a) Benchmarking results comparing to the reference results (Ref.) from [480], which are obtained numerically.

(b) Benchmarking results for comparison with [427] (Ref.). Experimental results are depicted by markers for different aspect ratios L/W of the sheet. Numerical results from [427] are not included since they are indistinguishable from the present MR results.

Figure 3.9: Uniaxial tension of a restrained sheet using incompressible material models. The dimensionless force is obtained by normalising the applied force P by the Young's modulus E and the cross-sectional area A.

In figure 3.10, convergence plots of the present model (NH and OG stretch-based models) with respect to the relative error in the strains given a nondimensional load of $P/EA = 0.5$ are provided. The errors are plotted with respect to the Richardson extrapolation from the three finest meshes since analytical solutions to the problem are not available. The results obtained for the NH model obtained from the invariant-based form are exactly the same and hence not provided here. The figures show that the convergence of the method is around second-order, independent of the order of the spline basis. Reference papers [320, 480] do not provide estimates of the order of convergence for the invariant-based material models or convergence plots for similar simulations. Hence, further comparison and investigations on the order of convergence for such membrane-dominated responses for shells with nonlinear material models are recommended.



(a) Neo-Hookean (NH): the orders of convergence following from Richardson extrapolation are 2.11 ($p = 2$), 2.15 ($p = 3$) and 2.17 ($p = 4$).

(b) Ogden (OG): the orders of convergence following from Richardson extrapolation are 2.27 ($p = 2$), 2.40 ($p = 3$) and 2.52 ($p = 4$).

Figure 3.10: Convergence rate of the restrained sheet under uniaxial tension with values from [480] for different material models (a-b). The error is relative error $\epsilon = |\varepsilon_{\text{num}} - \varepsilon_R|/\varepsilon_R$ where ε_{num} is the numerical value of the strain and ε_R is the Richardson-extrapolated value of the strain related to the last three meshes, all for a dimensionless force $P/EA = 0.5$. The orders of convergence following from the Richardson extrapolation are provided in the captions below the subfigures.

Secondly, the numerical model is compared to the experimental results from a similar setup as depicted in figure 3.8 [427]. The corresponding geometric parameters are $L = 280\,[\text{mm}]$, $W = 140\,[\text{mm}]$ and $t = 0.14\,[\text{mm}]$, leading to $L/W = 2$ and $t/W = 10^3$. The material models are incompressible, and for the NH material model, a parameter $\mu = 1.91 \cdot 10^5\,[\text{Pa}]$ is used, while for the MR model, the parameters $c_1 = 3.16 \cdot 10^5\,[\text{Pa}]$ and $c_2 = 1.24 \cdot 10^5\,[\text{Pa}]$ are used. The results are depicted in figure 3.9b, from which it can be seen that there is excellent agreement between the numerical results from [427] (obtained using the ABAQUS S4R element) and the experimental results. In addition, the depicted fit for the Ogden material model was found using parameters $\alpha_1 = 1.1\,[\text{-}]$, $\mu_1 = 1.0\mu_0\,[\text{Pa}]$, $\alpha_2 = -7\,[\text{-}]$, $\mu_2 = -0.003\mu_0\,[\text{Pa}]$, $\alpha_3 = -3\,[\text{-}]$ and $\mu_3 = -0.4\mu_0\,[\text{Pa}]$ with $\mu_0 = 1.91 \cdot 10^5$ Pa.

Wrinkling simulations

For the wrinkling simulations, the work of [427] is followed, with the same parameters for the Mooney-Rivlin and Ogden models as in figure 3.9b. The model setup for the wrinkling simulations is depicted in figure 3.11. The modeling domain is depicted in the shaded area and surrounded by boundaries Γ_k, $k = 1, .., 4$. Firstly, the boundary Γ_1 is free, meaning that no displacement constraints are involved. Furthermore, the boundary at Γ_2 is clamped (matching the adjacent control points parallel to the symmetry axes), and displacements in y-direction and out-of-plane displacements are restricted. The displacements in the x-direction are all equal over Γ_2. Symmetry is imposed over Γ_4 by *clamping* the edges and by restricting deformations orthogonal to the axes ($u_x = 0$). Lastly, anti-symmetry is imposed over Γ_3 by restricting displacements in vertical direction and orthogonal to the boundary ($u_y = 0$). Similar to [427], an anti-symmetry condition is applied over Γ_3 since

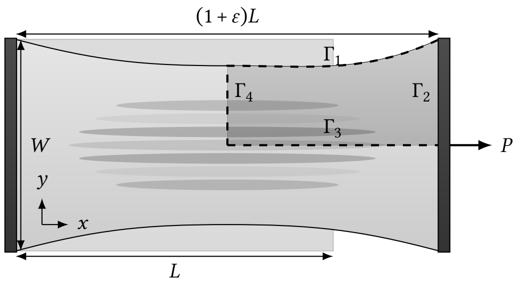
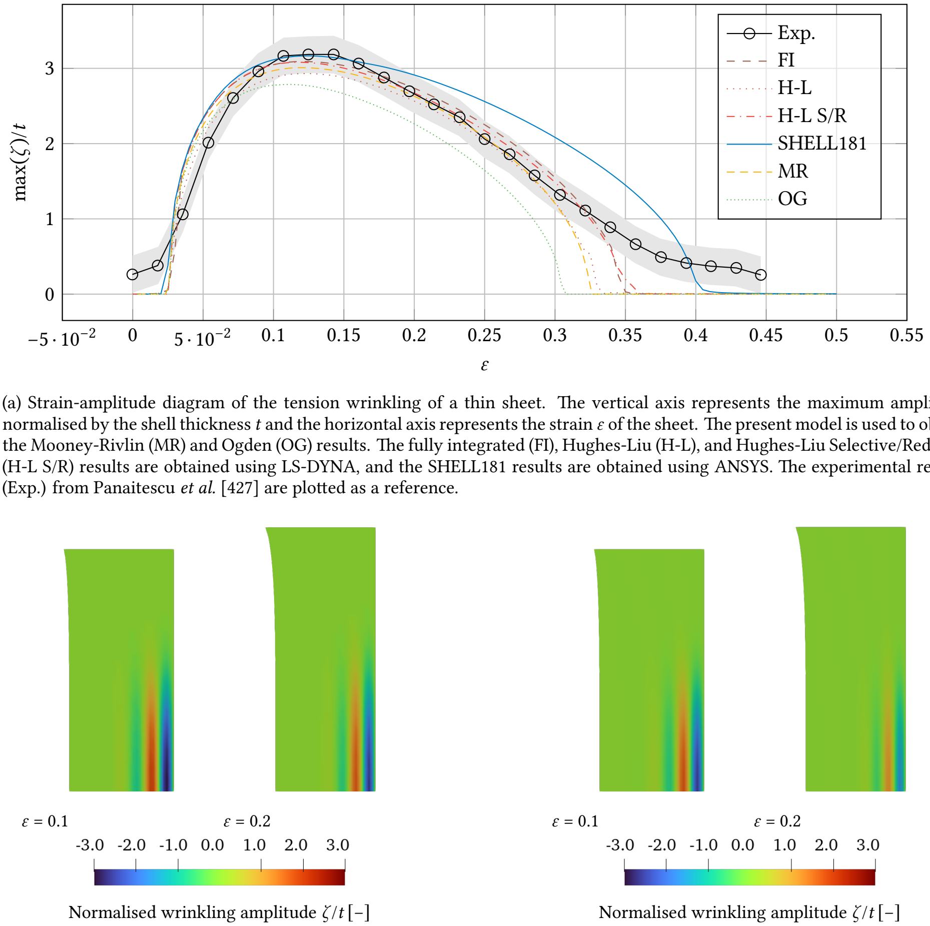


Figure 3.11: Modelling geometry for the wrinkled sheet. The sheet is modelled as only a quarter using symmetry conditions on Γ_3 and Γ_4. The side Γ_1 is free and the side Γ_2 is clamped and fixed in all direction except the x-direction. The sheet has length L and width W. The strain of the sheet due to load P is denoted by ε.

the symmetric and anti-symmetric wrinkling patterns can appear at the same critical load [191, 233]. For continuation, Crisfield's spherical arc-length method [125] is used with an extension for approaching bifurcation points [632], branch switching [193], and complex-root resolving [329, 660], all summarised and applied to IGA in [583].

Furthermore, for comparison, results from LS-DYNA (R11.0) and ANSYS (R19.1) simulations are presented for the same geometry and a Mooney-Rivlin model with the same parameters; however, $\nu = 0.499$ in the LS-DYNA simulations since incompressible materials ($\nu = 0.5$) are not implemented. A displacement control approach is employed with an initial perturbation based on the first buckling mode corresponding to a tension load situation, perturbed with a factor of 10^{-4}. In LS-DYNA, the Hughes-Liu, the Hughes-Liu selective/reduced, and the fully integrated shell elements are used, all with 4 quadrature points through-thickness and a shear correction factor equal to zero [364]. The results for the ANSYS SHELL181 element [11] are obtained using default options, which include reduced integration and hour-glassing control. For both the LS-DYNA and ANSYS simulations, mesh refinements were applied until convergence.

From figure 3.12a, a large difference between the different solvers and between the material models can be observed. Firstly, it can be concluded that the MR results from the isogeometric Kirchhoff–Love shell correspond most with the results obtained with LS-DYNA. Additionally, these results show good correspondence with the experimental results both in the low strain regime (until $\varepsilon \sim 0.08$) as well as towards restabilisation of the wrinkles (between $\varepsilon \sim 0.2$ and $\varepsilon \sim 0.3$), where only the maximum amplitude is slightly underestimated and the restabilisation point (i.e., the point where the wrinkles vanish again) is predicted too early. Secondly, it can be observed that there is a large difference between the results from IGA, LS-DYNA, and ANSYS. Although different shell options in the FEA libraries have been varied (e.g., reduced or full integration, shear correction factors), the origin of these differences is yet unknown to the authors and requires further investigation. Lastly, significant differences between the Ogden and Mooney-Rivlin results can be observed, although there are similarities in the material behaviour in figure 3.9b. From this, it can be concluded that material fitting possibly needs to be done using experimental



(a) Strain-amplitude diagram of the tension wrinkling of a thin sheet. The vertical axis represents the maximum amplitude normalised by the shell thickness t and the horizontal axis represents the strain ε of the sheet. The present model is used to obtain the Mooney-Rivlin (MR) and Ogden (OG) results. The fully integrated (FI), Hughes-Liu (H-L), and Hughes-Liu Selective/Reduced (H-L S/R) results are obtained using LS-DYNA, and the SHELL181 results are obtained using ANSYS. The experimental results (Exp.) from Panaitescu *et al.* [427] are plotted as a reference.

(b) Contour plot of out-of-plane displacements w for different strains ε for the MR model.

(c) Contour plot of out-of-plane displacements w for different strains ε for the OG model.

Figure 3.12: Wrinkling formation in a thin sheet subject to tension.

tests of different loading configurations, e.g., testing the bending response of the material.

3.6 Conclusions and Recommendations

This chapter provides mathematical formulations to accurately and efficiently model thin rubbers and several biological tissues by combining stretch-based material formulations such as the Ogden material model and smooth spline formulations of the isogeometric Kirchhoff-Love shell. The formulations apply to compressible and incompressible material models and are based on an eigenvalue computation to obtain the principal stretches and their direction (i.e., the spectral basis). The spectral stress and material tensors are transformed to the curvilinear basis accordingly, with limited computational costs due to tensor symmetries.

The results from numerical experiments with Neo-Hookean and Mooney-Rivlin material models, which can be represented in terms of invariants as well as in terms of stretches, show that identical iteration residuals and correct Newton-convergence rates have been obtained. This confirms that the stretch-based and invariant-based shell formulations are equivalent. For these models, it is also shown that the present formulation leads to higher CPU times due to the projection of the stress and material tensor; therefore, the advantage of the present formulation is mainly related to stretch-based material models (e.g., the Ogden model) and not to models that can be expressed explicitly in terms of the curvilinear tensor components of the deformation tensor (e.g., the invariant-based Neo-Hookean and Mooney Rivlin models). The analytical benchmarks have shown very good agreement, confirming that the formulations and implementation are correct.

Employing (extended) arc-length methods in combination with the present model, the collapsing behaviour of a truncated conical shell and the wrinkling behaviour of a stretched thin sheet are investigated. In the case of the collapsing truncated conical shell, the Ogden model was used in combination with either displacement-controlled or arc-length-controlled loads on the top boundary. The displacement-controlled results show good agreement with reference results from the literature. Using the arc-length method, the previously unnoticed response of the cone during collapse was obtained while overlapping with the displacement-controlled results on the stable part of the equilibrium path.

The present formulations are also used to model the phenomenon of wrinkling of a stretched thin sheet. To the best of the authors' knowledge, such simulations have only been published for finite element methods and not with Ogden material models. Hence, an Ogden material model is fitted, based on previously published experimental data and from the Mooney-Rivlin material relation, and applied isogeometric Kirchoff-Love shells to this case.

The result of the wrinkling case, which was also modelled using commercial finite element codes, shows that large deviations between commercial finite element codes are observed. The results of our model are in good agreement with the Hughes-Liu shells (reduced and full integration) in LS-DYNA. Furthermore, it was found that the Mooney-Rivlin model provides more accurate results than the Ogden material model, although their fits in the restrained tension test are similar. Based on the variation between the results from the Ogden and Mooney-Rivlin material models and the results obtained from LS-DYNA and ANSYS, it is concluded that the results for this benchmark are sensitive to differences in element assumptions. This motivates the future use of this case as a challenging benchmark problem.

As a topic for future research, it is suggested to develop analytical projection and direct decoupling [480] methods of the constitutive relations in order to prevent numerical through-thickness integration (i.e., eigenvalue computations for all through-thickness Gaussian points). These improvements could lead to a significant reduction in computational times.

3.A Result Reproduction

For the sake of reproducibility of the results in this chapter, this appendix provides brief instructions on the use of the software developed along with this thesis. The full software is available as part of the Geometry + Simulation Modules. For more detail on the contributions to this software library, and its installation, the reader is referred to chapter 8.

Table 3.3 provides per figure in this chapter the name of the file to run along with the arguments to be passed to obtain these figures.

Table 3.3: File name and run arguments required for the reproducibility of the figures in this chapter. Arguments with a single dash (-) require an argument. See chapter 8 for more detail about the software and installation instructions.

Figure			Run File
	Arg.	**Description**	**Values**
Figure 3.2	`benchmark_UniaxialTension`		
	`-M`	Material model	`1`: NH, `2`: MR, `4`: OG (with `I 3`)
	`-I`	Implementation	`1`: Analytical, `2`: Generalised, `3`: Spectral
	`-c`	Compressibility	`0`: Incompressible, `1`: Compressible
Table 3.1	Iteration numbers corresponding to load step 9 from `benchmark_UniaxialTension`		
Figure 3.4	`benchmark_Balloon`		
	`-M`	Material model	`1`: NH, `2`: MR, `4`: OG (with `I 3`)
	`-I`	Implementation	`1`: Analytical, `2`: Generalised, `3`: Spectral
Figure 3.6a	`benchmark_FrustrumALM`		(for the ALM results)
Figure 3.7a	`benchmark_FrustrumDC`		(for the DC results)
	`-t`	Test case	`0`: figure 3.6a, `1`: figure 3.7a
	`-M`	Material model	`1`: NH, `2`: MR, `4`: OG (with `I 3`)
	`-I`	Implementation	`1`: Analytical, `2`: Generalised, `3`: Spectral
	`-N`	Number of load steps	
Figure 3.9a	`benchmark_MaterialTest`		
Figure 3.9b	`-t`	Test case	`0`: Figure 3.9b, `1`: Figure 3.9a
	`-M`	Material model	`1`: NH, `2`: MR, `4`: OG (with `I 3`)
	`-I`	Implementation	`1`: Analytical, `2`: Generalised, `3`: Spectral
Figure 3.10	`benchmark_MaterialTestConv`		
	`-M`	Material model	`1`: NH, `2`: MR, `4`: OG (with `I 3`)
‘	`-I`	Implementation	`1`: Analytical, `2`: Generalised, `3`: Spectral
	`-r`	Number of uniform refinements	`6`
	`-e`	Number of degree elevations	`1`: $p = 2$, `2`: $p = 3$, `3`: $p = 4$
Figure 3.12a	`benchmark_TensionWrinkling`		
	`-M`	Material model	`1`: NH, `2`: MR, `4`: OG (with `I 3`)
	`-I`	Implementation	`1`: Analytical, `2`: Generalised, `3`: Spectral

4

Isogeometric Membrane Modelling using Hyperelastic Tension Field Theory

The previous chapter, chapter 3, presents hyperelastic material models isogeometric Kirchhoff–Love shells and applies these models to simulate wrinkles in a thin membrane subject to tension. However, as the mesh resolution depends on the wrinkling wave length, shell models become expensive for fine wrinkles as the required mesh size decreases. As a remedy, wrinkles can be solved implicitly by modifying the kinematic or constitutive relationship locally based on the taut, slack, or wrinkled state derived from a so-called tension field. This chapter presents a model for implicit wrinkling analysis for general hyperelastic materials, applicable to wrinkling simulations under large strains. The model is an extension of a previously published model for linear elastic materials, and it is applied to isogeometric membrane elements, but it is applicable to other discretisations. Using four benchmark problems – comparing our results to literature and isogeometric Kirchhoff–Love shell simulations – it is demonstrated that the presented model is able to converge with the expected order of convergence in the case of Newton–Raphson iterations when the tension field is fixed in space. For other problems, the model accurately approximates the mean surface of a wrinkled membrane with a reduced number of degrees of freedom compared to the shell simulation. In conclusion, the model presented in this chapter provides an alternative to explicit shell modelling for hyperelastic membrane wrinkling simulations, whenever global structural response instead of local wrinkling amplitudes are of one's interest. If necessary, the reader is referred to section 2.3.2 for a full derivation of the isogeometric Kirchhoff–Love shell model, to section 2.4.1 for more information about static solution methods and to section 2.5 for more literature related to wrinkling.

This chapter is in preparation to be submitted as a journal paper.

4.1 Introduction

Wrinkling is a phenomenon that is omnipresent around us and appears at different scales: it influences the thermo-conductivity of graphene on the nanoscale [148] and the reflectivity of solar sails on the metre scale [486]. Wrinkling shapes floating leaves [643] and plays an important role in cosmetics, for example in wound healing processes [527]. In engineering, the wrinkling simulation is relevant in the design of airbags [209, 383], sails [466], parachutes [548], and floating solar platforms [583]. Membrane wrinkling is influenced by balancing potential energy stored in bending deformations, membrane deformations, or in fluid or solid foundations, as discussed in the seminal works of Cerda *et al.* [92] and Pocivavsek *et al.* [442]. For further reference on the study of wrinkling from a physics perspective, the reader is referred to the review papers by Wang *et al.* [611] on tension-induced wrinkles, by Tan *et al.* [550] on wrinkles in curved surfaces, by Ma *et al.* [370] on wrinkles in membranes with elasticity gradients, by Paulsen [434] on wrinkles in membranes with low bending stiffness and high membrane stiffness, and by Li *et al.* [345] for a complete but less recent review on wrinkling.

For engineering applications where wrinkling is an relevant factor, accurate and efficient numerical modelling of wrinkling patterns becomes of great importance. In general, the numerical modelling can be done in different ways: by explicitly modelling the wrinkling amplitudes and wave lengths using shell models, by using reduced-order models based on the Föppl-Von Kármán plate equations, or by implicitly embedding the effects of wrinkling in element formulations.

Firstly, the modelling of membrane wrinkling can be done using mathematical models that account for the physics of thin films, including both membrane and bending effects. Plate and shell models are particularly suited to simulate wrinkling patterns, whereas membrane models lacking bending stiffness cannot capture wrinkle formations. Many studies on the physics of wrinkling employed commercial finite element methods to numerically investigate wrinkling under different conditions [179, 349, 367, 382, 403, 404, 427, 524, 531, 578, 628, 650, 651]. In addition, dedicated numerical models for the modelling of wrinkling patterns have been developed. For initially flat geometries, the Föppl-Von Kármán (FvK) model incorporating out-of-plane displacements and linear bending strains has been applied [191, 233, 322, 346, 359, 514, 611, 652]. This FvK model has been extended for hyperelastic materials, orthotropy, and general anisotropic properties [190, 192, 599, 603, 609, 610]. Another model based on Koiter's non-linear plate theory, proposed by Steigmann [520], has been used to compute wrinkling cases involving shear, holes, annuli, graphene, and reinforced plates [451, 551–554, 626]. In order to find wrinkling patterns, different algorithms searching for equilibrium solutions have been used, including static methods like the dynamic relaxation method and Newton–Raphson, as demonstrated by Taylor [551], or continuation methods such as the Arc-Length Method [125, 469] and the Asymptotic Numerical Method (ANM) [103–105, 132, 400]. In general, the advantage of models explicitly modelling wrinkling amplitudes and wave lengths is that they resemble actual physics. However, when wrinkling wave lengths decrease, the mesh size required to find the solutions typically decreases as well, implying increased computational costs.

Secondly, the so-called *Fourier reduced model* is a technique where the Föppl-Von Kármán model is discretised using Fourier series expansions. Consequently, the model provides a multi-scale model where the large Fourier modes capture the macroscopic deformations and the high-frequency content captures the wrinkling patterns. This technique, introduced by Damil *et al.* [133–135], is efficient as it can predict wrinkling patterns with few degrees of freedom, but it is inaccurate along boundaries. A remedy to that is to combine it with full shell models [266]. Detailed reviews of this approach are given by Potier-Ferry *et al.* [443] and Huang *et al.* [266]. And recent works include the extension to cases with non-uniform wrinkling orientations [314] and the combination with the ANM path following method [564]. Although the Fourier reduced model provides a reduction in degrees of freedom, potentially independent of the wrinkling wave length, the applicability of the method has not been fully demonstrated yet.

Thirdly, when modelling wrinkles implicitly, the goal is not necessarily to establish the actual wrinkling pattern, but to estimate wrinkling sensitive parts of the structure instead. Such models are typically driven by a so-called *tension field* [325, 374, 375, 465, 592, 593, 593], describing the state of parts of a membrane by being either taut, slack, or wrinkled. Tension fields can be defined based on principal stresses, principal strains, or a combination of those, as discussed by Kang & Im [300, 301] and Roddeman *et al.* [478], among others. Depending on the tension field, constitutive or kinematic equations can be modified to embed wrinkling effects into numerical methods; see the works of Le Meitour *et al.* [332] and Miyazaki [390] for an overview. For example, Pipkin [438, 439] and Steigmann & Green [521] proposed to modify the strain energy density function based on the tension field, for which variational methods [395–397] and interior point models [146] have been derived, and extensions for anisotropic and hyperelastic models have been proposed [18, 19, 176, 380]. In addition, material matrix modifications instead of strain energy density modifications have been developed [3, 278, 279, 300, 301, 356]. Alternatively, modification schemes based on deformation tensor modifications instead of constitutive relation modifications were proposed by Roddeman *et al.* [476–478], and applied to orthotropic materials [399, 457]. Efficient implementations of this model were presented by Lu *et al.* [365] and Nakashino & Natori [401], the latter authors applying it to isogeometric membrane elements as well [402]. Although the model based on modifications of the deformation tensor provides a more generic approach, it has not been applied to hyperelastic material modelling, to the best of the authors' knowledge. Although implicit models do not provide wrinkling amplitudes, post-processing methods for recovering wrinkling amplitudes have been proposed in the computer graphics community [95, 99, 288, 392, 515].

In this chapter, the wrinkling model of Nakashino & Natori [401, 402] is extended to hyperelastic materials. To this end, the model of Roddeman *et al.* [477] is used, where the deformation gradient is modified for a wrinkled material and, as a consequence, the stress and material tensors are modified based on the tension field in the membrane. The extension to hyperelastic materials introduces extra terms in the modified material tensor due to the dependency of the wrinkling stresses on the strain tensor. Although the formulations derived in this work apply to finite element methods, they are applied in the context of isogeometric analysis [268], as has been done by Nakashino *et al.* [402]. Since finite bending stiffness is essential in the modelling of wrinkling amplitudes, reference solutions are computed using the isogeometric Kirchhoff-Love shell model [319] with extensions to hyperelasticity [320, 587].

The chapter is outlined as follows: In section 4.2, the isogeometric membrane formulation is introduced. This element is equivalent to an isogeometric Kirchhoff-Love shell [319] without bending contributions. Section 4.3 recalls the original model by Nakashino & Natori [401, 402] for linear elastic materials to introduce the original concept. Thereafter, in section 4.4, the model of Nakashino *et al.* [402] is extended for hyperelastic materials. In section 4.5, the numerical implementation of the model is discussed, and in section 4.6, numerical benchmark results are provided. Lastly, section 4.7 provides conclusions and recommendations for future work.

4.2 Isogeometric Membrane Formulation

In this section, the isogeometric membrane formulation is derived. The primary purpose is to introduce the notations for geometric, kinematic, and constitutive quantities together with the variational formulation for a membrane. Since the membrane closely relates to parts of the Kirchhoff-Love shell, the notations are based on the ones used in [320, 587]. That is, basis vectors are denoted by lower-case bold and italic letters, e.g., $\boldsymbol{a}$; second-order tensors are denoted by upper-case bold letters, e.g., $\mathbf{A}$; discrete vectors or second-order tensors in Voight notation are denoted by upper-case bold and italic letters, e.g., $\boldsymbol{A}$; and matrices are denoted by upper-case letters, e.g., A. If needed, the notation $[A]$ is used to emphasise a matrix. Lastly, Greek sub- and superscripts take values of $1,2$, while Latin sub- and superscripts take values of $1,2,3$.

Consider surfaces $\mathring{\boldsymbol{x}}(\theta^1,\theta^2)$ and $\boldsymbol{x}(\theta^1,\theta^2) = \mathring{\boldsymbol{x}}(\theta^1,\theta^2) + \boldsymbol{u}(\theta^1,\theta^2)$ denoting points in the undeformed and deformed configurations of a membrane, respectively, with θ^α, $\alpha = 1,2$ the parametric coordinates of the surface and $\boldsymbol{u}(\theta^1,\theta^2)$ the deformation vector field. Consequently, the covariant basis vectors of the deformed and undeformed configurations are defined by $\mathring{\boldsymbol{a}}_\alpha$ and $\boldsymbol{a}_\alpha$, respectively, given by

$$\mathring{\boldsymbol{a}}_\alpha = \frac{\partial \mathring{\boldsymbol{x}}}{\partial \theta^\alpha}, \tag{4.1}$$

and similar for the deformed configuration $\boldsymbol{x}$. In addition, $\mathring{a}_{\alpha\beta} = \mathring{\boldsymbol{a}}_\alpha \cdot \mathring{\boldsymbol{a}}_\beta$ are the coefficients of the covariant metric tensor. The vectors $\mathring{\boldsymbol{a}}^\alpha$ and $\boldsymbol{a}^\alpha$ denote the contravariant basis vectors of the undeformed and deformed membrane surfaces, with identity $\mathring{\boldsymbol{a}}_\alpha \cdot \mathring{\boldsymbol{a}}^\beta = \delta_\alpha^\beta$ (similar for the basis vectors $\boldsymbol{a}_\alpha$) with $\delta_\alpha^\beta = 1$ if $\alpha = \beta$ and 0 otherwise. The contravariant basis vectors are constructed via the metric tensor:

$$\mathring{\boldsymbol{a}}^\alpha = [\mathring{a}_{\alpha\beta}]^{-1} \mathring{\boldsymbol{a}}_\beta, \tag{4.2}$$

where $[\mathring{a}]^{-1}$ denotes the inverse of the metric tensor coefficient matrix $[\mathring{a}]$. For the deformed configuration, the same relation holds between $\boldsymbol{a}^\alpha$ and $\boldsymbol{a}_\alpha$.

Remark 4.2.1. *Contrary to the coordinate system for Kirchhoff-Love shells [319], the through thickness coordinate θ^3 is omitted for the isogeometric membrane, assuming small thickness. Therefore, it is assumed that deformations are constant through the thickness of the membrane.*

4.2.1 Kinematic Equation

The deformation gradient **F** or the deformation tensor **C** relate $\mathring{\boldsymbol{x}}$ with $\boldsymbol{x}$ as follows:¨

$$\mathbf{F} = \boldsymbol{a}_\alpha \otimes \mathring{\boldsymbol{a}}^\beta, \quad \mathbf{C} = \mathbf{F}^\top \mathbf{F} = a_{\alpha\beta}\, \mathring{\boldsymbol{a}}^\alpha \otimes \mathring{\boldsymbol{a}}^\beta. \tag{4.3}$$

Using these quantities, the Green-Lagrange strain tensor **E** is defined by

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^\top \mathbf{F} - \mathbf{I}\right) = \frac{1}{2}(\mathbf{C} - \mathbf{I}) = \frac{1}{2}\left(a_{\alpha\beta} - \mathring{a}_{\alpha\beta}\right) = E_{\alpha\beta}\, \mathring{\boldsymbol{a}}^\alpha \otimes \mathring{\boldsymbol{a}}^\beta, \tag{4.4}$$

where $\mathbf{I} = \mathring{a}_{\alpha\beta}\, \mathring{\boldsymbol{a}}^\alpha \otimes \mathring{\boldsymbol{a}}^\beta$ is the identity tensor. The second Piola-Kirchhoff stress tensor **S** is defined through the constitutive relation; see section 4.2.2.

4.2.2 Constitutive Relation

The relation between the Green-Lagrange strain tensor **E** to the second Piola-Kirchhoff stress tensor **S**. In the following, the indices $i, j, k, l = 1, \dots, 3$ are used, representing the covariant and contravariant bases related to three parametric directions: parameters θ^1 and θ^2 represent the in-plane surface coordinates, and θ^3 represents the through-thickness coordinate in the direction of the unit normal vector $\hat{\boldsymbol{a}}_3$.

For linear elastic materials, the stress and strain tensors are simply related via the following relation:

$$\mathbf{S} = \boldsymbol{\mathscr{C}} : \mathbf{E} = S^{ij}\, \mathring{\boldsymbol{a}}_i \otimes \mathring{\boldsymbol{a}}_j, \tag{4.5}$$

where the coefficients of the stress tensor, S^{ij}, are given by $S^{ij} = \mathscr{C}^{ijkl} E_{kl}$. For a Saint Venant-Kirchhoff material with Lamé parameters λ and μ, the coefficients of the material tensor are given by:

$$\mathscr{C}^{ijkl} = \frac{2\lambda\mu}{\lambda + 2\mu} \mathring{a}^{ij} \mathring{a}^{kl} + \mu\left(\mathring{a}^{ik} \mathring{a}^{jl} + \mathring{a}^{il} \mathring{a}^{jk}\right). \tag{4.6}$$

For hyperelastic materials, the constitutive relationship is defined through the strain energy density function $\Psi(\mathbf{C})$ or $\Psi(\mathbf{E})$. In particular, the coefficients of the second Piola-Kirchhoff stress tensor are given by

$$S^{ij} = 2\frac{\partial \Psi}{\partial C_{ij}} = \frac{\partial \Psi}{\partial E_{ij}}. \tag{4.7}$$

The material tensor $\mathscr{C}^{ijkl}$ is not required in the derivation of the tensor **S**; however, it plays a role in the definition of the variation of the stress tensor, $\delta\mathbf{S}$, as shown in equation (4.21). In terms of the strain energy density function, the coefficients of the material tensor are defined by

$$C^{ijkl} = \frac{\partial S^{ij}}{\partial C_{kl}} = 4\frac{\partial^2 \Psi}{\partial C_{ij} \partial C_{kl}}. \tag{4.8}$$

As for the isogeometric Kirchhoff-Love shell, through thickness deformation is neglected, meaning that $C_{33} = 1$. This violates the plane stress condition ($S^{33} = 0$) since $S^{33} = \frac{\partial \Psi}{\partial C_{33}} \neq 0$. To satisfy the plane stress condition, the normal deformation C_{33} needs to be modified. As described for hyperelastic shells in [320], this can be done analytically for incompressible materials using the property that the Jacobian determinant, given by:

$$J = \sqrt{\frac{|a_{\alpha\beta}|}{|\mathring{a}_{\alpha\beta}|}}\sqrt{C_{33}}, \tag{4.9}$$

is unity, i.e., $J = 1$. For compressible materials, the plane stress condition is iteratively satisfied, as discussed in [320]. Finally, static condensation of the material tensor results in the in-plane material tensor $\hat{\mathscr{C}}^{\alpha\beta\gamma\delta}$[320]:

$$\hat{\mathscr{C}}^{\alpha\beta\gamma\delta} = \mathscr{C}^{\alpha\beta\gamma\delta} - \frac{\mathscr{C}^{\alpha\beta 33}\mathscr{C}^{33\gamma\delta}}{\mathscr{C}^{3333}}. \tag{4.10}$$

For incompressible materials, it can be found that the coefficients of the statically condensed material tensor are [320]:

$$\begin{aligned}\mathscr{C}^{\alpha\beta\gamma\delta} = {} & 4\frac{\partial^2\Psi}{\partial C_{\alpha\beta}\partial C_{\gamma\delta}} + 4\frac{\partial^2\Psi}{\partial C_{33}^2}J_0^{-4}a^{\alpha\beta}a^{\gamma\delta} - 4\frac{\partial^2\Psi}{\partial C_{33}\partial C_{\alpha\beta}}J_0^{-2}a^{\gamma\delta} - 4\frac{\partial^2\Psi}{\partial C_{33}\partial C_{\gamma\delta}}J_0^{-2}a^{\alpha\beta} \\ & + 2\frac{\partial\Psi}{\partial C_{33}}J_0^{-2}\left(2a^{\alpha\beta}a^{\gamma\delta} + a^{\alpha\gamma}a^{\beta\delta} + a^{\alpha\delta}a^{\beta\gamma}\right)\end{aligned} \tag{4.11}$$

Lastly, the tensor $\boldsymbol{\sigma} = J^{-1}\mathbf{F}^{\top}\mathbf{S}\mathbf{F}$ is the Cauchy stress tensor, which is used for stress recovery.

Example 4.2.1. *For a Neo-Hookean material model with* $\Psi(\mathbf{C}) = \frac{1}{2}\mu(I_1(\mathbf{C}) - 3)$*, where* μ *is Lamé's second parameter, equation* (4.11) *simplifies to*

$$\hat{\mathscr{C}}^{\alpha\beta\gamma\delta} = \mu J_0^{-2}\left(2a^{\alpha\beta}a^{\gamma\delta} + a^{\alpha\gamma}a^{\beta\delta} + a^{\alpha\delta}a^{\beta\gamma}\right). \tag{4.12}$$

4.2.3 Variational Formulation

The variational formulation for membranes follows from a variation of the internal and external energy contributions. For derivation of the variational formulation for isogeometric membranes, the reader is referred to [402] or to works on the Kirchhoff-Love shell [319, 320] omitting the bending stiffness contributions. The elastic energy is given by

$$\mathscr{W}_{\text{int}} = -\frac{1}{2}\int_{\Omega^\star} \mathbf{S} : \mathbf{E}\,\mathrm{d}\Omega^\star\,. \tag{4.13}$$

Here, $\Omega^\star = \Omega \times [-t/2, t/2]$ denotes the integration domain, with t the thickness of the membrane and Ω the surface domain. Taking the Gateaux derivative with respect to the displacements $\boldsymbol{u}$, the variation of the internal elastic energy is:

$$\delta\mathscr{W}_{\text{int}} = -\int_{\Omega^\star} \mathbf{S} : \delta\mathbf{E}\,\mathrm{d}\Omega^\star = -\int_{\Omega^\star} \mathbf{N} : \delta\mathbf{E}\,\mathrm{d}\Omega^\star\,, \tag{4.14}$$

where the tensor N denotes the membrane force tensor, obtained by integrating the stress tensor S through the thickness of the membrane. Since the thickness of the membrane is small (see remark 4.2.1), the thickness integral becomes:

$$\mathrm{N}(\theta^1,\theta^2) = \int_{[-t/2,t/2]} \mathrm{S}(\theta^1,\theta^2,\theta^3)\,\mathrm{d}\theta^3 \approx t\mathrm{S}(\theta^1,\theta^2,0). \tag{4.15}$$

The external virtual work is provided by the following:

$$\delta\mathscr{W}_{\mathrm{ext}} = \int_{\Omega} \delta\boldsymbol{u}\cdot\boldsymbol{f}(\boldsymbol{u})\,\mathrm{d}\Omega + \int_{\partial\Omega} \boldsymbol{g}\cdot\delta\boldsymbol{u}\,\mathrm{d}\Gamma, \tag{4.16}$$

where $\boldsymbol{f}(\boldsymbol{x})$ is a vector representing a follower-load acting on a point on the deformed surface $\boldsymbol{x}(\theta^1,\theta^2)$ and $\boldsymbol{g}$ is a line load acting on the boundary $\partial\Omega$. When the sum of the internal and external virtual work is zero, i.e.

$$\delta\mathscr{W}(\boldsymbol{u},\delta\boldsymbol{u}) = \delta\mathscr{W}_{\mathrm{int}} - \delta\mathscr{W}_{\mathrm{ext}} = 0, \tag{4.17}$$

equilibrium is found. Since $\delta\mathscr{W}(\boldsymbol{u},\delta\boldsymbol{u})$ is non-linear, solving the equation requires linearization. The second variation of the internal energy $\mathscr{W}_{\mathrm{int}}$ in the system is given by:

$$\delta^2\mathscr{W}_{\mathrm{int}}(\boldsymbol{u},\delta\boldsymbol{u},\Delta\boldsymbol{u}) = -\int_{\Omega} \delta\mathrm{N} : \delta\mathrm{E} + \mathrm{N} : \delta^2\mathrm{E}\,\mathrm{d}\Omega. \tag{4.18}$$

Here, $\delta^2\mathbf{E}$ denotes the second variation of the Green-Lagrange strain tensor. Since the external virtual work from equation (4.16) depends on the solution $\boldsymbol{u}$, its second variation is non-zero and given by:

$$\delta^2\mathscr{W}_{\mathrm{ext}}(\boldsymbol{u},\delta\boldsymbol{u},\Delta\boldsymbol{u}) = \int_{\Omega} \delta\boldsymbol{u}\cdot\boldsymbol{f}'(\boldsymbol{u},\Delta\boldsymbol{u})\,\mathrm{d}\Omega. \tag{4.19}$$

For a follower pressure, $\boldsymbol{f} = p\hat{\boldsymbol{a}}_3$, and its variation is $\boldsymbol{f}' = p\hat{\boldsymbol{a}}_3'(\boldsymbol{u},\Delta\boldsymbol{u})$. Here $\hat{\boldsymbol{a}}_3(\boldsymbol{u})$ and $\hat{\boldsymbol{a}}_3'(\boldsymbol{u},\Delta\boldsymbol{u})$ are the unit normal vector and its variation, respectively, which can be found in the derivation of the Kirchhoff–Love shell [319]. Furthermore, the variation of the normal force tensor is:

$$\delta\mathrm{N} = \int_{\Omega} \delta\mathrm{S}\,\mathrm{d}\Omega. \tag{4.20}$$

Here, the variation of the second Piola-Kirchhoff stress tensor is defined as:

$$\delta\mathrm{S} = \mathscr{C} : \delta\mathrm{E}. \tag{4.21}$$

Equations (4.18) and (4.21) show that given the variations δE and δ^2E, the first and second variations of the internal energy δW_{int} and $\delta^2\mathscr{W}_{\mathrm{int}}$ from equations (4.17) and (4.18) can be found. Since the deformed configuration $\boldsymbol{x}$ is unknown, the problem needs to be discretised such that the variations of E can be defined.

4.2.4 Discretisation

Discretisation of the variational problem in equation (4.17) is achieved by discretising the undeformed and deformed configurations $\mathring{\boldsymbol{x}}$ and $\boldsymbol{x}$, respectively. In the context of isogeometric analysis, this is done by choosing splines as a basis for the geometry, i.e., by describing the geometries as a weighted sum of basis functions $\varphi_i(\theta^1, \theta^2)$ and control points $\mathring{\boldsymbol{x}}_i^h$ and $\boldsymbol{x}_i^h$, respectively:

$$\begin{aligned} \mathring{\boldsymbol{x}}^h(\theta^1, \theta^2) &= \sum_k \varphi_k(\theta^1, \theta^2)\mathring{\boldsymbol{x}}_k^h, \\ \boldsymbol{x}^h(\theta^1, \theta^2) &= \sum_k \varphi_k(\theta^1, \theta^2)\boldsymbol{x}_k^h. \end{aligned} \tag{4.22}$$

The superscript h indicates a discretisation of the undeformed and deformed geometries, $\mathring{\boldsymbol{x}}$ and $\boldsymbol{x}$, respectively. Using the same basis for $\mathring{\boldsymbol{x}}^h$ and $\boldsymbol{x}^h$, the discrete displacement vector $\boldsymbol{u}^h$ is given as the difference between the two, i.e., $\boldsymbol{u}^h = \boldsymbol{x}^h - \mathring{\boldsymbol{x}}^h$. Since the variational formulation is expressed in terms of the displacement field $\boldsymbol{u}$, the variations of the control points of the field $\boldsymbol{u}^h$ are the virtual displacements in the discrete system, hence the unknowns. In the sequel, the subscripts r and s denote the indices of the components of conveniently numbered degrees of freedom of $\boldsymbol{u}^h$ incorporating the spatial dimensions of the surface. Furthermore, the notation $(\cdot)_r = \partial(\cdot)\partial u_r$ is used for derivatives, and the superscript h is omitted. Following from equation (4.22), the variation of the deformed geometry is given by

$$\boldsymbol{x}_{,r} = \sum_k \left(\mathring{\boldsymbol{x}}_{k,r} + \boldsymbol{u}_{k,r}\right) = \sum_k \varphi_k \boldsymbol{u}_{k,r} = \boldsymbol{u}_{,r}, \tag{4.23}$$

where the last equality follows from the fact that the undeformed configuration is trivially independent of the deformation field $\boldsymbol{u}$. Similarly, the derivatives of the covariant basis vectors $\boldsymbol{a}_\alpha$ of the discrete deformed configuration $\boldsymbol{x}_h$, see equation (4.23), are:

$$\boldsymbol{a}_{\alpha,r} = \left(\frac{\partial \boldsymbol{x}_k}{\partial \theta^\alpha}\right)_{,r} = \sum_k \frac{\partial \varphi_k}{\partial \theta^\alpha}\boldsymbol{u}_{k,r}. \tag{4.24}$$

As a consequence, the variation of the surface metric tensor of the deformed configuration, $a_{\alpha\beta}$, becomes:

$$a_{\alpha\beta,r} = \left(\boldsymbol{a}_\alpha \cdot \boldsymbol{a}_\beta\right)_r = \boldsymbol{a}_{\alpha,r} \cdot \boldsymbol{a}_\beta + \boldsymbol{a}_\alpha \cdot \boldsymbol{a}_{\beta,r}. \tag{4.25}$$

Since the undeformed configuration is invariant to the deformation field $\boldsymbol{u}$, the first variation of the membrane strain tensor $\boldsymbol{\varepsilon}$ from equation (4.4) becomes

$$E_{\alpha\beta,r} = \frac{1}{2}a_{\alpha\beta,r}. \tag{4.26}$$

Similarly, the second variation of the deformed configuration, the deformed surface metric tensor, and the membrane strain can be derived. Starting with the first variation of the deformed configuration from equation (4.23), the second variation becomes

$$\boldsymbol{x}_{,rs} = \sum_k \varphi_k \boldsymbol{u}_{k,rs} = 0. \tag{4.27}$$

The second variation of $\boldsymbol{u}_k$ is zero since the components of these nodal weights are linear in u_r. Similarly, $\boldsymbol{a}_{\alpha,rs} = 0$. As a consequence, the second variation of the surface metric tensor in the deformed configuration, $a_{\alpha\beta}$, becomes

$$a_{\alpha\beta,rs} = \boldsymbol{a}_{\alpha,rs} \cdot \boldsymbol{a}_{\beta} + \boldsymbol{a}_{\alpha,r} \cdot \boldsymbol{a}_{\beta,s} + \boldsymbol{a}_{\alpha,s} \cdot \boldsymbol{a}_{\beta,s} + \boldsymbol{a}_{\alpha} \cdot \boldsymbol{a}_{\beta,rs}, = \boldsymbol{a}_{\alpha,r} \cdot \boldsymbol{a}_{\beta,s} + \boldsymbol{a}_{\alpha,s} \cdot \boldsymbol{a}_{\beta,s}. \tag{4.28}$$

Again, since the undeformed configuration is invariant to the deformation field $\boldsymbol{u}$, the second variation of the membrane strain tensor becomes

$$E_{\alpha\beta,rs} = \frac{1}{2} a_{\alpha\beta,rs} \tag{4.29}$$

Since $E_{\alpha\beta,r}$ and $E_{\alpha\beta,rs}$ are the discrete variations of $\delta\mathrm{E}$ and $\delta^2\mathrm{E}$, respectively, the discrete residual and Jacobian can be derived. The discrete residual vector follows from equation (4.17) with equations (4.14) to (4.16) and (4.26):

$$R_r(\boldsymbol{u}) = \int_\Omega \mathrm{N}(\boldsymbol{u}) : \mathrm{E}_{,r}(\boldsymbol{u}) \,\mathrm{d}\Omega - \int_\Omega \boldsymbol{f}(\boldsymbol{u}) \cdot \boldsymbol{u}_{,r} \,\mathrm{d}\Omega. \tag{4.30}$$

Furthermore, the discrete Jacobian matrix follows from equation (4.18) with equations (4.14), (4.15), (4.19), (4.20), (4.26) and (4.29):

$$K_{rs} = \int_\Omega \mathrm{N}_{,s}(\boldsymbol{u}) : \mathrm{E}_{,r}(\boldsymbol{u}) + \mathrm{N}(\boldsymbol{u}) : \mathrm{E}_{,rs}(\boldsymbol{u}) \,\mathrm{d}\Omega - \int_\Omega \boldsymbol{f}(\boldsymbol{u})_{,s} \cdot \boldsymbol{u}_{,r} \,\mathrm{d}\Omega. \tag{4.31}$$

Here, the product $\mathbf{A} : \mathbf{B}$ denotes an inner product of two second-order tensors. Furthermore, the contribution of the displacement-dependent load $\boldsymbol{f}(\boldsymbol{u})$ requires a derivative, defined as $\boldsymbol{f}(\boldsymbol{u})_{,s} = p\hat{\boldsymbol{a}}_{3,s}$ for a follower pressure, where $\hat{\boldsymbol{a}}_{3,s}(\boldsymbol{u})$ is the discrete derivative of the surface normal vector $\hat{\boldsymbol{a}}_3(\boldsymbol{u})$, which can be found in the derivations of the Kirchhoff–Love shell [319]. The contribution $\boldsymbol{f}(\boldsymbol{u}) \cdot \boldsymbol{u}_{,rs} = 0$ since $\boldsymbol{u}_{,rs} = 0$ [316]. Given the discretisation of $\boldsymbol{x}$ and $\mathring{\boldsymbol{x}}$ using splines, see equation (4.22), and given the residual and Jacobian from equations (4.30) and (4.31), respectively, only the definition of the stress tensor S and the material tensor $\mathcal{C}$ are remaining undefined. In sections 4.3 and 4.4, definitions for S and $\mathcal{C}$ are provided, incorporating the wrinkling model of Nakashino & Natori [401] for linear elasticity and incorporating the extension for hyperelastic materials, which is the novelty of this chapter.

4.2.5 Implementation

Since the tensors **E** and **S** are symmetric second-order tensors, they can be written in Voight notation:

$$\boldsymbol{S} = \begin{bmatrix} S^{11} & S^{22} & S^{12} \end{bmatrix}^\top, \quad \boldsymbol{E} = \begin{bmatrix} E^{11} & E^{22} & 2E^{12} \end{bmatrix}^\top. \tag{4.32}$$

As a consequence, the material tensor is represented in Voight notation as well, using equation (4.21):

$$\boldsymbol{C} = \begin{bmatrix} \mathcal{C}^{1111} & \mathcal{C}^{1122} & \mathcal{C}^{1112} \\ \mathcal{C}^{2211} & \mathcal{C}^{2222} & \mathcal{C}^{2212} \\ \mathcal{C}^{1211} & \mathcal{C}^{1222} & \mathcal{C}^{1212} \end{bmatrix}. \tag{4.33}$$

Using Voight notation for the tensors **E**, **S**, and $\mathcal{C}$, the second-order tensor inner products $:$ in equations (4.21), (4.30) and (4.31) can simply be evaluated as matrix-vector products and vector inner-products. In sections 4.3.4 and 4.4.3, the elastic and hyperelastic constitutive laws for wrinkling are provided in Voight notations for fast computer implementation.

4.3 Linear Elastic Wrinkling Model

In the linear elastic wrinkling model, the taut, wrinkling, and slack conditions are governed by a modification of the deformation gradient and consequently the strain, stress, and material tensors, following the model originally proposed by Roddeman *et al.* [477, 478]. This model lays the foundation of the wrinkling model proposed by Nakashino & Natori [401], which was later implemented for isogeometric membranes [402]. This model is presented with the assumption that the material behaviour is linear; therefore, it is referred to herein as the linear elastic wrinkling model. In this section, an overview of the linear elastic wrinkling model proposed by Nakashino & Natori [401] is provided. In section 4.4, the linear elastic wrinkling model will be extended to hyperelastic material models.

In general, tension field-based models rely on the definition of a tension field ϕ over the domain. A tension field classifies the stress state in a membrane as either slack, taut, or wrinkled, depending on the deformation tensor $\boldsymbol{C}$. In general, three different definitions of the tension field are used in the literature: based on principal strains, principal stresses, or combinations of those (mixed), in this chapter represented by ϕ^E, ϕ^S, and ϕ^M, respectively. These tension fields are defined as:

$$\phi^E = \begin{cases} \text{Taut} & \text{if } E_{p,1} > 0 \\ \text{Slack} & \text{if } E_{p,2} \leq 0 \\ \text{Wrinkled} & \text{otherwise} \end{cases}, \qquad \phi^S = \begin{cases} \text{Taut} & \text{if } S_{p,1} > 0 \\ \text{Slack} & \text{if } S_{p,2} \leq 0 \\ \text{Wrinkled} & \text{otherwise} \end{cases},$$

$$\phi^M = \begin{cases} \text{Taut} & \text{if } S_{p,1} > 0 \\ \text{Slack} & \text{if } E_{p,2} \leq 0 \\ \text{Wrinkled} & \text{otherwise} \end{cases}, \qquad (4.34)$$

where $S_{p,1}$ and $S_{p,2}$ are the principal stresses such that $S_{p,1} \leq S_{p,2}$ and $E_{p,1}$ and $E_{p,2}$ are the principal strains such that $E_{p,1} \leq E_{p,2}$. Given the tension field, tension field models typically modify the stress tensor S and consequently the material tensor $\mathcal{C}$ based on the tension field:

$$\mathrm{S} = \begin{cases} 0 & \text{if } \phi = \text{ Slack} \\ \mathrm{S} & \text{if } \phi = \text{ Taut} \\ \mathrm{S}' & \text{if } \phi = \text{ Wrinkled} \end{cases}, \qquad \mathcal{C} = \begin{cases} 0 & \text{if } \phi = \text{ Slack} \\ \mathcal{C} & \text{if } \phi = \text{ Taut} \\ \mathcal{C}' & \text{if } \phi = \text{ Wrinkled} \end{cases}, \qquad (4.35)$$

where S′ is a modified stress tensor. This modified stress tensor can be obtained in different ways, either by adjusting the constitutive or kinematic equations provided by the tension field. For example, the work of [6] modifies the Ogden constitutive relation based on the tension field ϕ^S, whereas [401] modifies the kinematic equation based on ϕ^M. In this chapter, the approach of [401] is followed. As discussed in the work of Kang & Im [300], the definition of the tension field using ϕ^M has advantages over ϕ_E and ϕ_S.

4.3.1 Kinematic Equation

Given the deformation gradient $\mathbf{F}$, the modified deformation gradient [477, 478] is given by

$$\mathbf{F}' = (\mathbf{I} + b\hat{\boldsymbol{w}} \otimes \hat{\boldsymbol{w}}) \cdot \mathbf{F}. \tag{4.36}$$

Here, $\mathbf{I}$ is the second-order identity tensor, and b is the measure of the amount of 'wrinkliness' [365, 401, 478], by definition $b > 0$. Furthermore, $\hat{\boldsymbol{w}}$ is the unit vector transverse to the wrinkles. Using the modified deformation tensor, the modified strain tensor can be computed, given by:

$$\mathbf{E}' = E'_{\alpha\beta}\mathring{\boldsymbol{a}}^{\alpha} \otimes \mathring{\boldsymbol{a}}^{\beta} = \frac{1}{2}\left(\mathbf{F}'^{\top} \cdot \mathbf{F}' - \mathbf{I}\right) = \mathbf{E} + \mathbf{E}_W, \tag{4.37}$$

where $\mathbf{E} = \mathbf{F}^{\top}\mathbf{F} - \mathbf{I} = \mathbf{C} - \mathbf{I}$ is the Green-Lagrange strain with

$$\mathbf{C} = C_{\alpha\beta}\, \mathring{\boldsymbol{a}}^{\alpha} \otimes \mathring{\boldsymbol{a}}^{\beta} = a_{\alpha} a_{\beta}\, \mathring{\boldsymbol{a}}^{\alpha} \otimes \mathring{\boldsymbol{a}}^{\beta}, \tag{4.38}$$

being the deformation tensor. $\mathbf{E}_W$ is the wrinkling strain, given by

$$\mathbf{E}_W = \frac{1}{2}b(b+2)\hat{\hat{\boldsymbol{w}}} \otimes \hat{\hat{\boldsymbol{w}}}. \tag{4.39}$$

Here, $\hat{\boldsymbol{w}} = \boldsymbol{w} \cdot \mathbf{F} = w_a \mathring{\boldsymbol{a}}^{\alpha}$ is the projection of $\hat{\boldsymbol{w}}$ onto the undeformed contravariant basis. Introducing the rotation ϑ and magnitude $a \neq 0$ of the projected wrinkling direction $\hat{\boldsymbol{w}}$ using $\hat{w}_1 = an_1$ and $\hat{w}_2 = an_2$ using $n_1 = \cos\vartheta$ and $n_2 = \sin\vartheta$, the coefficients of the wrinkling strain tensor can be written as

$$E'_{\alpha\beta} = E_{\alpha\beta} + \frac{1}{2}\left(\mathbf{F}'^{\top}\mathbf{F}' - \mathbf{I}\right) = E_{\alpha\beta} + \gamma n_{\alpha} n_{\beta}, \tag{4.40}$$

where $\gamma = \frac{1}{2}a^2 b(b+1)$. In this definition of the strain tensor $\mathbf{E}'$, the wrinkling strain amplitude γ and the angle of the wrinkles ϑ are unknown. Through the uniaxial tension condition for wrinkled materials, these unknowns will be determined in the next subsection.

4.3.2 Constitutive Relation

Since the definition of the strain tensor $\mathbf{E}$ changes for the wrinkled state of the membrane, the constitutive relation from equation (4.5) also changes. For linear materials, the wrinkled stress tensor $\mathbf{S}'$ simply becomes:

$$\mathbf{S}' = \mathscr{C} : \mathbf{E}' = \mathscr{C} : (\mathbf{E} + \mathbf{E}_W), \tag{4.41}$$

Where $\mathbf{E}_W$ depends on the unknowns γ and ϑ, as in equation (4.40). This equation can also be written in terms of the component of the stress tensor $\mathbf{S}' = S'^{\alpha\beta}\, \mathring{\boldsymbol{a}}_{\alpha} \otimes \mathring{\boldsymbol{a}}_{\beta}$:

$$S'^{\alpha\beta} = \mathscr{C}^{\alpha\beta\gamma\delta}(E_{\gamma\delta} + \gamma n_{\gamma} n_{\delta}) = S^{\alpha\beta} + \gamma \mathscr{C}^{\alpha\beta\gamma\delta} n_{\gamma} n_{\delta}. \tag{4.42}$$

For a wrinkled membrane, an uniaxial tension state is assumed [356], meaning the stress orthogonal to the wrinkles should vanish,

$$\sigma' \cdot \boldsymbol{w} = 0, \tag{4.43}$$

And the stress parallel to the wrinkles should be positive

$$\boldsymbol{t} \cdot \sigma' \cdot \boldsymbol{t} = 0. \tag{4.44}$$

Here, $\sigma' = \frac{1}{\det \mathbf{F}'} \mathbf{F}'\mathbf{S}'\mathbf{F}'^{\top}$ is the modified Cauchy stress tensor. The uniaxial tension condition in equation (4.43) can be written as

$$\mathbf{S}' \cdot \boldsymbol{w} = 0, \tag{4.45}$$

Using the components n_α and n_β, the conditions equations (4.43) and (4.44) can be written as [365, 401]:

$$\begin{aligned} S'^{\alpha} n_\alpha n_\beta &= 0, \\ S'^{\alpha\beta} m_\alpha n_\beta &= 0, \\ E'^{\alpha\beta} m_\alpha m_\beta &> 0, \end{aligned} \tag{4.46}$$

Here, $m_\alpha = \partial n_\alpha / \partial \vartheta$. From equation (4.42), the uniaxial tension condition equation (4.46) becomes:

$$\begin{aligned} S^{\alpha\beta} n_\alpha n_\beta + \gamma \mathscr{C}^{\alpha\beta\gamma\delta} n_\alpha n_\beta n_\gamma n_\delta &= 0, \\ S^{\alpha\beta} m_\alpha n_\beta + \gamma \mathscr{C}^{\alpha\beta\gamma\delta} m_\alpha n_\beta n_\gamma n_\delta &= 0. \end{aligned} \tag{4.47}$$

From the first line of equation (4.47), the variable γ can be found as:

$$\gamma = -\frac{S^{\alpha\beta} n_\alpha n_\beta}{\mathscr{C}^{\alpha\beta\gamma\delta} n_\gamma n_\delta n_\alpha n_\beta}, \tag{4.48}$$

and substituting γ in the second line of equation (4.47), the following equation is found:

$$f(\vartheta) \equiv S^{\alpha\beta} m_\alpha n_\beta + \gamma \mathscr{C}^{\alpha\beta\gamma\delta} m_\alpha n_\beta n_\gamma n_\delta = 0. \tag{4.49}$$

The only unknown in this equation is the angle ϑ, thus the equation can be solved by root finding. The root ϑ of $f(\vartheta) = 0$ must satisfy the uniaxial tension conditions equations (4.43) and (4.44). Since $f(\vartheta)$ follows from equation (4.43), the first uniaxial tension condition is satisfied when the root is found. The condition for positive stress along the wrinkling direction, i.e., equation (4.44), is satisfied by selecting the feasible root. In the work of [356], the bounds of an interval for the root that satisfies equation (4.44) are derived, such that a bounded root finding algorithm, e.g., Brent's method [71], can be used. If the procedure to find the root ϑ fails on the prescribed interval, the domain $[0, 2\pi]$ can be subdivided into sub-intervals $[\vartheta_A, \vartheta_B) \in [0, 2\pi]$, and Brent's method can be started for each sub-interval $[\vartheta_A, \vartheta_B) \in [0, 2\pi]$ that satisfies $f(\vartheta_A) f(\vartheta_B) < 0$. The root $\vartheta = 0$ is a root if and only if $\lim_{\vartheta \to 0} f(\vartheta) f(2\pi - \vartheta) < 0$. As soon as the angle ϑ is found, the wrinkling strain $\mathbf{E}'$ from equation (4.40) and the wrinkling stress $\mathbf{S}'$ from equation (4.42) can be computed.

4.3.3 Variational Formulation

In the variational equation from equation (4.17), the variation of the strain tensor **E** and stress tensor S, respectively δ**E** and δS, are required. For the taut state, the variations remain unchanged, as seen in equation (4.35). In the slack state, the stress tensor and its variation are equal to zero, hence the variation of the internal energy becomes zero. For the wrinkling state, the variations of **E′** and **S′** need to be found.

Firstly, the variation of **E′** is independent of the constitutive law [401]. Here, it is shown that the contribution of the virtual wrinkling strain, $\delta\mathbf{E}_W$, in the variational formulation is zero, since the product of the wrinkling stress tensors **S′** and $\delta\mathbf{E}_W$ is zero. Physically, this means that the wrinkling strain corresponds to the rigid body movements to stretch the wrinkled membrane, hence not altering the strain energy.

Secondly, the variation of **S′** needs to be found. Taking the variation of equation (4.42), it follows that:

$$\begin{aligned}\delta S'^{\alpha\beta}(\mathbf{E}) &= \frac{\mathrm{d}S'^{\alpha\beta}}{\mathrm{d}E_{\sigma\tau}}\delta E_{\sigma\tau} = \mathscr{C}'^{\alpha\beta\sigma\tau}\delta E_{\sigma\tau}\\ &= \left(\frac{\partial S^{\alpha\beta}}{\partial E_{\sigma\tau}} + \frac{\mathrm{d}\gamma}{\mathrm{d}E_{\sigma\tau}}\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta + \gamma\mathscr{C}^{\alpha\beta\gamma\delta}\frac{\mathrm{d}(n_\gamma n_\delta)}{\mathrm{d}E_{\sigma\tau}}\right)\delta E_{\sigma\tau}.\end{aligned} \tag{4.50}$$

The full derivative of γ with respect to $E_{\sigma\tau}$ can be found using the definition of γ from equation (4.48):

$$\frac{\mathrm{d}\gamma}{\mathrm{d}E_{\sigma\tau}} = \frac{\partial\gamma}{\partial E_{\sigma\tau}} + \frac{\partial\gamma}{\partial\vartheta}\frac{\partial\vartheta}{\partial E_{\sigma\tau}}. \tag{4.51}$$

The derivative of $(n_\gamma n_\delta)$ to $E_{\sigma\tau}$ directly follows from the definitions of n_α and m_α and a chain rule,

$$\frac{\mathrm{d}(n_\gamma n_\delta)}{\mathrm{d}E_{\sigma\tau}} = \frac{\partial(n_\gamma n_\delta)}{\partial\vartheta}\frac{\partial\vartheta}{\partial E_{\sigma\tau}} = \left(m_\gamma n_\delta + n_\gamma m_\delta\right)\frac{\partial\vartheta}{\partial E_{\sigma\tau}}. \tag{4.52}$$

The derivative of γ with respect to ϑ follows from equation (4.48):

$$\begin{aligned}\frac{\partial\gamma}{\partial\vartheta} &= -\frac{\left(\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta n_\alpha n_\beta\right)\left(S^{\alpha\beta}\frac{\partial(n_\alpha n_\beta)}{\partial\vartheta}\right) - \left(S^{\alpha\beta}n_\alpha n_\beta\right)\left(\mathscr{C}^{\alpha\beta\gamma\delta}\left(\frac{\partial(n_\gamma n_\delta)}{\partial\vartheta}n_\alpha n_\beta + n_\gamma n_\delta\frac{\partial(n_\alpha n_\beta)}{\partial\vartheta}\right)\right)}{\left(\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta n_\alpha n_\beta\right)^2}\\ &= -\frac{\gamma\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta\frac{\partial(n_\alpha n_\beta)}{\partial\vartheta}}{\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta n_\alpha n_\beta},\end{aligned} \tag{4.53}$$

where equations (4.47), (4.48) and (4.52) are used in the second equality. Furthermore, the derivative $\frac{\partial\gamma}{\partial E_{\alpha\beta}}$ follows directly from equation (4.48)

$$\frac{\partial\gamma}{\partial E_{\sigma\tau}} = -\frac{\mathscr{C}^{\alpha\beta\sigma\tau}n_\alpha n_\beta}{\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta n_\alpha n_\beta}. \tag{4.54}$$

Lastly, the derivative of the angle ϑ with respect to the strain tensor component $E_{\sigma\tau}$ can be found by using equation (4.49)

$$\frac{\partial f}{\partial E_{\sigma\tau}} + \frac{\partial f}{\partial \vartheta}\frac{\partial \vartheta}{\partial E_{\sigma\tau}} = 0, \tag{4.55}$$

giving:

$$\frac{\partial \vartheta}{\partial E_{\sigma\tau}} = -\frac{\partial f}{\partial E_{\sigma\tau}}\left(\frac{\partial f}{\partial \vartheta}\right)^{-1}. \tag{4.56}$$

The derivative of f with respect to $\mathbf{E}$ directly follows from equation (4.49):

$$\frac{\partial f}{\partial E_{\sigma\tau}} = \mathscr{C}^{\alpha\beta\sigma\tau} m_\alpha n_\beta - \frac{\partial \gamma}{\partial E_{\sigma\tau}} \mathscr{C}^{\alpha\beta\gamma\delta} m_\gamma n_\delta n_\alpha n_\beta, \tag{4.57}$$

and the derivative of f with respect to ϑ follows from equation (4.49) as well:

$$\frac{\partial f}{\partial \vartheta} = S^{\alpha\beta}\frac{\partial (m_\alpha n_\beta)}{\partial \vartheta} + \frac{\partial \gamma}{\partial \vartheta}\mathscr{C}^{\alpha\beta\gamma\delta} m_\alpha n_\beta n_\gamma n_\delta + \gamma \mathscr{C}^{\alpha\beta\gamma\delta}\left(\frac{\partial (m_\alpha n_\beta)}{\partial \vartheta} n_\gamma n_\delta + m_\alpha n_\beta \frac{\partial (n_\gamma n_\delta)}{\partial \vartheta}\right), \tag{4.58}$$

Where the derivative of $m_\alpha n_\beta$ with respect to ϑ can be easily obtained from the definitions of m_α and n_α:

$$\frac{\partial (m_\alpha n_\beta)}{\partial \vartheta} = -n_\alpha n_\beta + m_\alpha m_\beta. \tag{4.59}$$

Using equations (4.53) to (4.58), the variation of S from equation (4.50) is found, as well as the definition of the modified material tensor $\mathscr{C}'$.

4.3.4 Implementation

As discussed in the work of Nakashino & Natori [401], the modifications of the stress and material tensor for the wrinkling model, see equations (4.42) and (4.50), can be expressed in terms of matrix-vector multiplications when employing Voight notation for the stress and strain tensors, S and E, respectively, see equations (4.32) and (4.33). Following the notation of Nakashino & Natori [401], the terms n_α and m_α are collected in the following vectors:

$$\begin{aligned}
\boldsymbol{n}_1 &= \begin{bmatrix} n_1 n_1 & n_2 n_2 & 2 n_1 n_2 \end{bmatrix}^\top, \\
\boldsymbol{n}_2 &= \begin{bmatrix} n_1 m_1 & n_2 m_2 & n_1 m_2 + m_1 n_2 \end{bmatrix}^\top = \frac{1}{2}\frac{\partial \boldsymbol{n}_1}{\partial \vartheta} \\
\boldsymbol{n}_3 &= \begin{bmatrix} m_1 m_1 - n_1 n_1 & m_2 m_2 - n_2 n_2 & 2(m_1 m_2 - n_1 n_2) \end{bmatrix}^\top = \frac{\partial \boldsymbol{n}_2}{\partial \vartheta}, \\
\boldsymbol{n}_4 &= \begin{bmatrix} m_1 m_1 & m_2 m_2 & 2 m_1 m_2 \end{bmatrix}^\top = \boldsymbol{n}_3 + \boldsymbol{n}_1.
\end{aligned} \tag{4.60}$$

Using equation (4.60), the wrinkled stress and strain tensors from equations (4.40) and (4.42) are written in Voight notation as

$$\boldsymbol{S}' = \boldsymbol{S} + \gamma \boldsymbol{C} \cdot \boldsymbol{n}_1, \tag{4.61}$$

$$\boldsymbol{E}' = \boldsymbol{E} + \gamma \boldsymbol{n}_1. \tag{4.62}$$

Furthermore, using the Voight notation of the stress and strain tensor coefficients (equation (4.32)) together with equation (4.60), the formulations in equation (4.46) become:

$$\begin{aligned} \boldsymbol{S}' \cdot \boldsymbol{n}_1 &= 0, \\ \boldsymbol{S}' \cdot \boldsymbol{n}_2 &= 0, \\ \boldsymbol{E}' \cdot \boldsymbol{n}_4 &> 0. \end{aligned} \tag{4.63}$$

Following from these relations, the equations equation (4.48) and equation (4.49) are written in Voight notation as

$$\gamma = -\frac{\boldsymbol{S} \cdot \boldsymbol{n}_1}{\boldsymbol{n}_1^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1}, \tag{4.64}$$

$$f(\vartheta) = \boldsymbol{S} \cdot \boldsymbol{n}_1 - \gamma \boldsymbol{n}_2^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1. \tag{4.65}$$

To compute the wrinkling material tensor $\boldsymbol{C}'$ in Voight notation, the derivatives from equations (4.53) to (4.55) need to be expressed in terms of $\boldsymbol{E}$, $\boldsymbol{S}$, $\boldsymbol{C}$, and $\boldsymbol{n}_k$, $k = 1, \ldots, 4$; see equations (4.32), (4.33) and (4.60). The derivative of γ with respect to ϑ from equation (4.53) becomes:

$$\frac{\partial \gamma}{\partial \vartheta} = -2\gamma \frac{\boldsymbol{n}_2^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1}{\boldsymbol{n}_1^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1}. \tag{4.66}$$

Furthermore, the derivative of γ with respect to $\boldsymbol{E}$ becomes (see equation (4.54)):

$$\left[\frac{\partial \gamma}{\partial \boldsymbol{E}}\right] = -\frac{\boldsymbol{C} \cdot \boldsymbol{n}_1}{\boldsymbol{n}_1^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1}, \tag{4.67}$$

where the bracket $[\cdot]$ is used to stress that $\frac{\partial \gamma}{\partial \boldsymbol{E}}$ is a vector with the derivatives of γ with respect to $E_{\alpha\beta}$ ordered in Voight notation. To obtain the derivative of ϑ with respect to $\boldsymbol{E}$, the derivatives of f with respect to ϑ and $\boldsymbol{E}$ are used, as in equation (4.56). The derivative of f with respect to $\boldsymbol{E}$ is a vector as well. Following from equation (4.55), it is given by:

$$\left[\frac{\partial f}{\partial \boldsymbol{E}}\right] = \boldsymbol{C} \cdot \boldsymbol{n}_2 + \left(\boldsymbol{n}_2^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1\right)\left[\frac{\partial \gamma}{\partial \boldsymbol{E}}\right], \tag{4.68}$$

and the derivative of f with respect to ϑ is obtained from equation (4.58):

$$\frac{\partial f}{\partial \vartheta} = (\boldsymbol{n}_4^{\top} \cdot \mathrm{S}) + \frac{\partial \gamma}{\partial \vartheta}\left(\boldsymbol{n}_2^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1\right) + \gamma\left(\boldsymbol{n}_4^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_1 + 2\boldsymbol{n}_2^{\top} \cdot \boldsymbol{C} \cdot \boldsymbol{n}_2\right). \tag{4.69}$$

Using equations (4.68) and (4.69), the derivative of ϑ with respect to $\boldsymbol{E}$ in Voight notation is simply obtained by scalar division, using equation (4.56):

$$\left[\frac{\partial \vartheta}{\partial \boldsymbol{E}}\right] = \left[\frac{\partial f}{\partial \boldsymbol{E}}\right] \Big/ \frac{\partial f}{\partial \vartheta}. \tag{4.70}$$

Using the definition of $\mathscr{C}'$ from equation (4.50), the matrix $\boldsymbol{C}'$ can be expressed in linear algebra operations using the scalars and vectors defined in equations (4.66), (4.67) and (4.70):

$$\boldsymbol{C}' = \boldsymbol{C}\left(\mathbf{I} + \boldsymbol{n}_1 \otimes \left[\frac{\partial \gamma}{\partial \boldsymbol{E}}\right] + \frac{\partial \gamma}{\partial \vartheta} \boldsymbol{n}_1 \otimes \left[\frac{\partial \vartheta}{\partial \boldsymbol{E}}\right] + 2\gamma \boldsymbol{n}_2 \otimes \left[\frac{\partial \vartheta}{\partial \boldsymbol{E}}\right]\right). \tag{4.71}$$

By using the definition of the tension field ϕ from equation (4.34), the stress and material tensors are defined using equation (4.35). Therefore, the assembly of the residual and Jacobian from equations (4.30) and (4.31), respectively, involves computing the tension field and inserting the corresponding option from equation (4.35) on each integration point.

Remark 4.3.1. *The modification scheme presented in this section, originally proposed by Nakashino & Natori [401], provides an analytical derivation of the derivative* $\boldsymbol{\mathscr{C}}'$ *of the wrinkling stress tensor* $\mathbf{S}'$. *Consequently, the definitions in equation* (4.35) *are consistent and should provide optimal convergence in Newton–Raphson iterations. However, the definition of*$\mathbf{S}'$ *in equation* (4.35) *depends on the tension field* ϕ *from equation* (4.34). *The dependency of* $\mathbf{S}'$ *on* ϕ *is not included in its derivative, and therefore the convergence behaviour of Newton–Raphson iterations can be suboptimal or diverging.*

4.4 Hyperelastic Wrinkling Model

In this section, the novelty of this chapter is presented. The theory from Nakashino & Natori [401], recalled in the previous section section 4.3, is extended to hyperelastic materials. The outline of this section is similar to the outline of section 4.3, but since the assumption for hyperelastic materials only affects the constitutive relation, the kinematic equations are as in section 4.3, hence not included in this section. The present section primarily presents the differences with the elastic theory from section 4.3, first for the constitutive relation for the wrinkled membrane and then for the variational formulation.

4.4.1 Constitutive Relation

The derivation in section 4.3 assumes a linear elastic constitutive model in equation (4.42). In the case of non-linear hyperelastic material models, the constitutive relation is defined by a strain energy density function $\Psi(\mathbf{C})$ (see equation (4.7)), where $\mathbf{C} = \mathbf{F}^\top\mathbf{F} = a_{\alpha\beta}\, \mathring{\boldsymbol{a}}^\alpha \otimes \mathring{\boldsymbol{a}}^\beta$ is the deformation tensor (see equation (4.38)). In order to derive the wrinkling stress for hyperelastic materials, denoted by $\mathbf{S}' = \mathbf{S}(\mathbf{E}')$, the stress tensor is linearized around the Green-Lagrange strain $\mathbf{E}$, i.e.

$$\begin{aligned}\mathbf{S}' = \mathbf{S}(\mathbf{E}') = \mathbf{S}(\mathbf{E}+\mathbf{E}_W) = \mathbf{S}(\mathbf{E}) + \frac{\partial \mathbf{S}}{\partial \mathbf{E}} : \mathbf{E}_W + \mathcal{O}(\mathbf{E}_W^2) &= \mathbf{S}(\mathbf{E}) + \boldsymbol{\mathscr{C}}(\mathbf{E}) : \mathbf{E}_W + \mathcal{O}(\mathbf{E}_W^2)\\ &\approx \mathbf{S}(\mathbf{E}) + \boldsymbol{\mathscr{C}}(\mathbf{E}) : \mathbf{E}_W,\end{aligned} \tag{4.72}$$

In the second last equality, the definition of the material tensor $\boldsymbol{\mathscr{C}}(\mathbf{E})$ from equation (4.8) is used. The definition in equation (4.72) is similar to equation (4.42) when the Taylor expansion is truncated and the contribution of $\mathcal{O}(\mathbf{E}_W^2)$ is neglected under the assumption that $\mathbf{E}_W^2$ is small . Therefore, equivalent to equation (4.42), the coefficients of the wrinkling stress tensor $\mathbf{S}'$ are given by:

$$S'^{\alpha\beta}(\mathbf{E}) = S^{\alpha\beta}(\mathbf{E}) + \gamma \mathscr{C}(\mathbf{E})^{\alpha\beta\gamma\delta} n_\gamma n_\delta. \tag{4.73}$$

Following from the hyperelastic modified wrinkling strain tensor from equation (4.72), the hyperelastic counterpart of equation (4.47) is derived:

$$\begin{aligned} S^{\alpha\beta}(\mathbf{E})n_\alpha n_\beta + \gamma\boldsymbol{\mathscr{C}}(\mathbf{E})^{\alpha\beta\gamma\delta}n_\alpha n_\beta n_\gamma n_\delta &= 0, \\ S^{\alpha\beta}(\mathbf{E})m_\alpha n_\beta + \gamma\boldsymbol{\mathscr{C}}(\mathbf{E})^{\alpha\beta\gamma\delta}m_\alpha n_\beta n_\gamma n_\delta &= 0. \end{aligned} \tag{4.74}$$

From these equations, the hyperelastic counterpart of equation (4.48) is straightforward:

$$\gamma(\mathbf{E}) = -\frac{S^{\alpha\beta}(\mathbf{E})n_\alpha n_\beta}{\mathscr{C}^{\alpha\beta\gamma\delta}(\mathbf{E})n_\gamma n_\delta n_\alpha n_\beta}, \tag{4.75}$$

as well as the equation for $f(\vartheta)$:

$$f(\vartheta, \mathbf{E}) \equiv S^{\alpha\beta}(\mathbf{E})m_\alpha n_\beta + \gamma(\mathbf{E})\mathscr{C}^{\alpha\beta\gamma\delta}(\mathbf{E})m_\alpha n_\beta n_\gamma n_\delta = 0. \tag{4.76}$$

Like in the linear expression in equation (4.49), the result in equation (4.76) is also dependent on E, which is fixed while finding the root. In this way, the same root-finding procedure can be applied as discussed for the linear elastic model.

4.4.2 Variational Formulation

For the hyperelastic model, the variation of the wrinkling stress tensor S′ is derived from equation (4.73):

$$\begin{aligned} \delta S'^{\alpha\beta}(\mathbf{E}) &= \frac{\partial S'^{\alpha\beta}}{\partial E_{\sigma\tau}}\delta E_{\sigma\tau} = \mathscr{C}'^{\alpha\beta\sigma\tau}\delta E_{\sigma\tau} \\ &= \left(\frac{\partial S^{\alpha\beta}}{\partial E_{\sigma\tau}} + \frac{\partial\gamma}{\partial E_{\sigma\tau}}\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta + \gamma\frac{\partial\mathscr{C}^{\alpha\beta\gamma\delta}}{\partial E_{\sigma\tau}}n_\gamma n_\delta + \gamma\mathscr{C}^{\alpha\beta\gamma\delta}\frac{\partial(n_\gamma n_\delta)}{\partial E_{\sigma\tau}}\right)\delta E_{\sigma\tau}. \end{aligned} \tag{4.77}$$

Compared to equation (4.50), the expression in equation (4.77) contains an extra term with the derivative of the material tensor $\boldsymbol{\mathscr{C}}$ with respect to the strains **E**. This derivative can be written in terms of the deformation tensor **C** following equation (4.4)

$$\frac{\partial\mathscr{C}^{\alpha\beta\gamma\delta}}{\partial E_{\sigma\tau}} = 2\frac{\partial\mathscr{C}^{\alpha\beta\gamma\delta}}{\partial C_{\sigma\tau}}. \tag{4.78}$$

Considering equations (4.52) and (4.54), the derivatives of γ and f with respect to **E**, see equations (4.53) to (4.56) and (4.58), need to be re-defined due to the dependency of **S** and $\boldsymbol{\mathscr{C}}$ on **E**. Starting with γ, the derivative with respect to the strain tensor **E** becomes:

$$\begin{aligned} \frac{\partial\gamma}{\partial E^{\sigma\tau}} &= -\frac{(\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta n_\alpha n_\beta)(\mathscr{C}^{\alpha\beta\sigma\tau}n_\alpha n_\beta) - \left(S^{\alpha\beta}n_\alpha n_\beta\right)\left(\frac{\partial\mathscr{C}^{\alpha\beta\gamma\delta}}{\partial E_{\sigma\tau}}n_\gamma n_\delta n_\alpha n_\beta\right)}{\left(\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta n_\alpha n_\beta\right)^2} \\ &= -\frac{\mathscr{C}^{\alpha\beta\sigma\tau}n_\alpha n_\beta + \gamma\frac{\partial\mathscr{C}^{\alpha\beta\gamma\delta}}{\partial E_{\sigma\tau}}n_\gamma n_\delta n_\alpha n_\beta}{\mathscr{C}^{\alpha\beta\gamma\delta}n_\gamma n_\delta n_\alpha n_\beta}. \end{aligned} \tag{4.79}$$

Furthermore, the derivative of the root equation (4.76) with respect to the strain becomes:

$$\frac{\partial f}{\partial E_{\sigma\tau}} = \left(\mathscr{C}^{\alpha\beta\sigma\tau} m_\alpha n_\beta + \frac{\partial \gamma}{\partial E_{\sigma\tau}} \mathscr{C}^{\alpha\beta\gamma\delta} m_\gamma n_\delta n_\alpha n_\beta + \gamma \frac{\partial \mathscr{C}^{\alpha\beta\sigma\tau}}{\partial E_{\sigma\tau}} m_\gamma n_\delta n_\alpha n_\beta \right). \tag{4.80}$$

Since the other derivatives with respect to ϑ (see equations (4.53) and (4.58)) do not change for hyperelastic materials, equations (4.52) and (4.56) can be evaluated using equations (4.53), (4.58), (4.79) and (4.80), given the derivative of $\mathscr{C}$ with respect to $\mathbf{E}$. Since $\mathscr{C}$ depends on the strain energy density function $\Psi(\mathbf{C})$ (see equation (4.8)), its derivative can be computed analytically. However, when static condensation to satisfy the plane-stress criterion is performed numerically, which could be the case for compressible materials [320, 587], it is not straightforward to compute the derivative of $\mathscr{C}$. Alternatively, the analytical expression for the statically condensed material tensor (see equation (4.10)) can become lengthy, making the analytical derivation of its derivative a tedious exercise. As an alternative to analytical derivation of the material tensor, it can therefore be opted to use finite differences or automatic differentiation to obtain the derivative of $\mathscr{C}$ with respect to the strain. In example 4.4.1, the analytical derivative of $\mathscr{C}$ with respect to $\mathbf{E}$ for a statically condensed incompressible Neo-Hookean material model is provided.

Example 4.4.1. *From example 4.2.1, it follows that for an incompressible Neo-Hookean material model,*

$$\begin{aligned}\frac{\partial \mathscr{C}^{\alpha\beta\gamma\delta}}{\partial E_{\sigma\tau}} &= 2\frac{\partial \mathscr{C}^{\alpha\beta\gamma\delta}}{\partial C_{\sigma\tau}} \\ &= \mu \frac{\partial (J_0^{-2})}{\partial C_{\sigma\tau}} \left(2a^{\alpha\beta} a^{\gamma\delta} + a^{\alpha\xi} a^{\beta\eta} + a^{\alpha\eta} a^{\beta\xi} \right) + \mu J_0^{-2} \frac{\partial}{\partial C_{\sigma\tau}} \left(2a^{\alpha\beta} a^{\gamma\delta} + a^{\alpha\xi} a^{\beta\eta} + a^{\alpha\eta} a^{\beta\xi} \right).\end{aligned} \tag{4.81}$$

Since $J_0^2 = |a_{\alpha\beta}|/|\mathring{a}_{\alpha\beta}|$ *and since* $\frac{\partial J_0}{\partial \sigma\tau} = \frac{J}{2} a^{\sigma\tau}$ *[252], the derivative of* J_0^{-2} *is*

$$\frac{\partial (J_0^{-2})}{\partial C_{\sigma\tau}} = -J_0^{-2} a^{\sigma\tau}. \tag{4.82}$$

Furthermore, the derivative of the contravariant metric tensor $a^{\alpha\beta}$ *is given by*

$$\frac{\partial a^{\alpha\beta}}{\partial C_{\sigma\tau}} = -\frac{1}{2} \left(a^{\alpha\sigma} a^{\beta\tau} + a^{\alpha\tau} a^{\beta\sigma} \right). \tag{4.83}$$

Using equations (4.82) *and* (4.83)*, the derivatives in equation* (4.81) *can be evaluated, and the analytical expression for* $\partial \mathscr{C}^{\alpha\beta\gamma\delta}/\partial E_{\sigma\tau}$ *for the incompressible Neo-Hookean material is found.*

4.4.3 Implementation

As for the linear elastic model, the modified wrinkling model for hyperelastic materials can be expressed in terms of linear algebra operations using the Voight notation of the strain, stress, and material tensor as a basis (see equations (4.32) and (4.33)). First of all, the derivative of γ with respect to the strain tensor E for hyperelastic materials becomes, following from equation (4.79),

$$\left[\frac{\partial\gamma}{\partial \mathrm{E}}\right] = -\frac{\boldsymbol{C}\cdot\boldsymbol{n}_1 + \gamma\boldsymbol{n}_1^\top\cdot\left[\frac{\partial\mathscr{C}}{\partial \mathrm{E}}\right]\cdot\boldsymbol{n}_1}{\boldsymbol{n}_1^\top\cdot\boldsymbol{C}\cdot\boldsymbol{n}_1}. \tag{4.84}$$

Moreover, using equation (4.80), the derivative of f with respect to E for hyperelastic materials becomes:

$$\left[\frac{\partial f}{\partial \mathrm{E}}\right] = \boldsymbol{C}\cdot\boldsymbol{n}_2 + \left(\boldsymbol{n}_2^\top\cdot\boldsymbol{C}\cdot\boldsymbol{n}_1\right)\left[\frac{\partial\gamma}{\partial \mathrm{E}}\right] + \gamma\boldsymbol{n}_2^\top\cdot\left[\frac{\partial\mathscr{C}}{\partial \mathrm{E}}\right]\cdot\boldsymbol{n}_1. \tag{4.85}$$

Both expressions in equations (4.84) and (4.85), as well as the extra contribution of $[\partial\mathscr{C}/\partial\mathrm{E}]$ in equation (4.77), contain the inner-product of $\partial\mathscr{C}/\partial\mathrm{E}$ with $\boldsymbol{n}_1$. Instead of storing $[\partial\mathscr{C}/\partial\mathrm{E}]$ and multiplying it in both expressions with $\boldsymbol{n}_1$, the product $[\partial\mathscr{C}/\partial\mathrm{E}]\cdot\boldsymbol{n}_1$ can also be stored as a matrix:

$$\left[\frac{\partial\mathscr{C}}{\partial \mathrm{E}}\cdot\boldsymbol{n}_1\right] = \left[\begin{matrix}\frac{\partial\mathscr{C}}{\partial E_{11}}\cdot\boldsymbol{n}_1 & \frac{\partial\mathscr{C}}{\partial E_{22}}\cdot\boldsymbol{n}_1 & \frac{\partial\mathscr{C}}{\partial E_{12}}\cdot\boldsymbol{n}_1\end{matrix}\right]. \tag{4.86}$$

Using equations (4.84) to (4.86) together with equations (4.66), (4.69) and (4.70), the modified material tensor for hyperelastic wrinkled materials employing Voight notations, $\boldsymbol{C}'$, can be computed, based on equation (4.77):

$$\boldsymbol{C}' = \boldsymbol{C}\left(\mathrm{I} + \boldsymbol{n}_1\otimes\left[\frac{\partial\gamma}{\partial \mathrm{E}}\right] + \frac{\partial\gamma}{\partial\vartheta}\boldsymbol{n}_1\otimes\left[\frac{\partial\vartheta}{\partial \mathrm{E}}\right] + 2\gamma\boldsymbol{n}_2\otimes\left[\frac{\partial\vartheta}{\partial \mathrm{E}}\right]\right) + \gamma\left[\frac{\partial\mathscr{C}}{\partial \mathrm{E}}\cdot\boldsymbol{n}_1\right]. \tag{4.87}$$

Similar to the linear elastic wrinkling modification scheme from section 4.3, the modified tensors $\boldsymbol{S}'$ and $\boldsymbol{C}'$ can be used in the definition in equation (4.35) depending on the tension field, evaluated per quadrature point. In addition, the remark in remark 4.3.1 regarding the variation of the stress tensor with respect to the tension field also applies for the hyperelastic model.

4.5 Numerical Solution Strategies

To obtain the solution $\boldsymbol{u}$ to the variational formulation equation (4.17), discretised by the residual from equation (4.30), different solution strategies can be adopted. In this section, a brief overview of the Newton–Raphson and the Dynamic Relaxation methods employed in the numerical examples is given.

Firstly, the Newton–Raphson method solves the system of equations

$$K(\boldsymbol{u}_k)\Delta\boldsymbol{u}_{k+1} = -\boldsymbol{R}(\boldsymbol{u}_k), \tag{4.88}$$

for $\Delta\boldsymbol{u}_{k+1}$ given $\boldsymbol{u}_k$ and updating $\boldsymbol{u}_{k+1} = \boldsymbol{u}_k + \Delta\boldsymbol{u}_{k+1}$ in iteration k. Here, K is the Jacobian of the system from equation (4.31) and $\boldsymbol{R}$ is the residual vector from equation (4.30). Typically, the iterations are terminated if the update norm is $\|\Delta\boldsymbol{u}_k\|/\|\Delta\boldsymbol{u}_0\| < \epsilon_{\Delta\boldsymbol{u}}$ or if the relative residual norm is $\|\boldsymbol{R}(\boldsymbol{u}_k)\|/\|\boldsymbol{R}(\boldsymbol{u}_0)\| < \epsilon_{\boldsymbol{R}}$. Although the Newton–Raphson iterations provide second-order convergence towards the solution $\boldsymbol{u}$, the convergence region is bounded, meaning that the method is guaranteed to converge only for an initial guess $\boldsymbol{u}_0$ sufficiently close to the final solution $\boldsymbol{u}$. Furthermore, if the Jacobian matrix K is not exact, the speed of convergence can be decreased and the convergence region can shrink. In the case of the methods presented by Nakashino & Natori [401] and in this chapter, the variation of the stress tensor with respect to the tension field is not included in the Jacobian (see remark 4.3.1), possibly deteriorating the convergence behaviour.

A commonly used alternative for Newton–Raphson iterations for solving problems involving wrinkling stabilities is the Dynamic Relaxation (DR) method [421]. In this method, a dynamic system with artificial stiffness and damping is solved. The advantages of the dynamic relaxation method are that it is robust given a sufficiently small step size and that it only requires the residual vector $\boldsymbol{R}$. However, its convergence is very slow. For the mass contribution, the dynamic relaxation method is often used with lumped mass, to avoid solving a linear system [32]. The mass is typically scaled with a factor that can be used to tune the method [429]. Furthermore, damping contributions can be added using an appropriate scaling technique, summarised by Rodriguez *et al.* [479] and Rezaiee-Pajand & Estiri [467]. Alternatively, one can use the so-called *kinetic damping* approach, eliminating the velocities in the system when the kinetic energy reaches a peak [32, 33, 508]. In this chapter, the DR method with kinetic damping is used with a row-lumped mass matrix and a scaling parameter α tuning the speed of convergence of the DR iterations. Typically, the DR iterations are terminated if the relative residual norm is below a tolerance $\|\boldsymbol{R}(\boldsymbol{u}_k)\|/\|\boldsymbol{R}(\boldsymbol{u}_0)\| < \epsilon_{\boldsymbol{R}}$, or if the relative kinetic energy is below a certain tolerance $E^K_k/E^K_0 < \epsilon_{E^K}$.

4.6 Numerical Experiments

In this section, four benchmarks are presented for verification of the model presented in this chapter. The benchmarks are selected from previous works on hyperelastic wrinkling simulations. Firstly, an uniaxial tension test is performed to verify the implementation of the model, based on the examples in the works of Kiendl *et al.* [320] and Verhelst *et al.* [587]. In this case, the full domain is in wrinkling condition, hence the tension field does not change during the iterations. Secondly, the inflation of a square membrane is modelled in section 4.6.2, inspired by the work of Diaby *et al.* [152]. Thirdly, section 4.6.3 provides an example of a planar annular sheet in which the inner boundary is pulled out of the plane and twisted. This benchmark is inspired by the example given by Taylor *et al.* [551] for linear elastic materials modified using a hyperelastic material model. Lastly, section 4.6.4 models a cylindrical surface subject to large axial strain and a radial twist to demonstrate the capabilities of the present model on conic surfaces under large strains. For all benchmarks, a wrinkling simulation resolving wrinkling amplitudes is provided as a reference, along with results from the literature if available. The former are generated using isogeometric Kirchhoff-Love shells with hyperelastic constitutive models [319, 320, 587].

In the sequel, different hyperelastic material models are used. The compressible and incompressible Neo-Hookean (NH) and Mooney-Rivlin (MR) material models are given by the following strain energy density functions:

$$\Psi(\mathrm{C}) = \frac{\mu}{2}\left(J^{-\frac{2}{3}}I_1 - 3\right) + \Psi_{\mathrm{vol}}(J) \qquad \text{NH Compressible,} \tag{4.89}$$

$$\Psi(\mathrm{C}) = \frac{\mu}{2}(I_1 - 3), \qquad \text{NH Incompressible,} \tag{4.90}$$

$$\Psi(\mathrm{C}) = \frac{c_1}{2}\left(J^{-\frac{2}{3}}I_1 - 3\right) + \frac{c_2}{2}\left(J^{-\frac{4}{3}}I_2 - 3\right) + \Psi_{\mathrm{vol}}(J), \qquad \text{MR Compressible,} \tag{4.91}$$

$$\Psi(\mathrm{C}) = \frac{c_1}{2}(I_1 - 3) + \frac{c_2}{2}(I_2 - 3), \qquad \text{MR Incompressible.} \tag{4.92}$$

Here, μ is the second Lamé parameter, defined as $\mu = E/(2(1+\nu))$ and c_1 and c_2 are the coefficients controlling the Mooney-Rivlin model via $\mu = c_1 + c_2$. Furthermore, Ψ_{vol} the volumetric strain energy density function using bulk modulus K and parameter $\beta = -2$:

$$\Psi_{\mathrm{vol}} = K\mathcal{G}(J) = K\beta^{-2}\left(\beta\log(J) + J^{-\beta} - 1\right). \tag{4.93}$$

For the incompressible Neo-Hookean material model, the analytical derivative of the material tensor is implemented, whereas for the other material models a finite-difference technique is used to obtain this term.

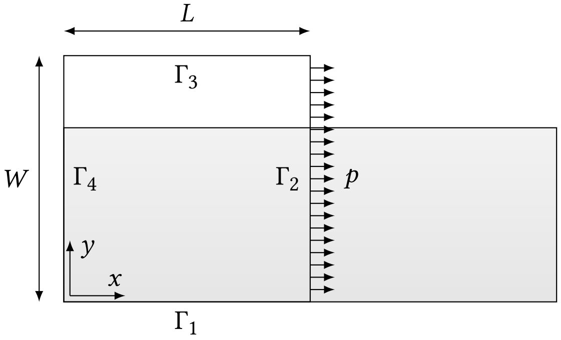
4.6.1 Square subject to Tension

As a first example, a uniaxial tension test is performed on a square membrane, see figure 4.1 for the model parameters inspired by Kiendl *et al.* [320] and Verhelst *et al.* [587]. The uniaxial tension test is performed both with an increasing load and with the value given in figure 4.1. The load stepping simulation provides a load displacement curve, which will be used against the implementation in our previous work [587], replicating analytical solutions. The simulation with the fixed value of the line load p is used to assess the convergence in Newton–Raphson iterations. This is used to confirm the correct derivation and implementation of the modified stress and material tensors $\boldsymbol{S}'$ and $\boldsymbol{\mathcal{C}}'$. The tests are performed for compressible and incompressible Neo-Hookean (NH) and Mooney-Rivlin (MR) material models; see equations (4.89) to (4.92).

Figure 4.2 presents the results of the uniaxial tension test. In the top figures, the stretch λ is plotted against the applied line load p for the compressible and incompressible Neo-Hookean and Mooney-Rivlin material models. In the bottom figures, the convergence is presented using the current and previous relative residual norms, $\|R_{i+1}\|/\|R_0\|$ and $\|R_i\|/\|R_0\|$, respectively. The load-displacement curves (top) show that the tension field theory modification scheme accurately predicts the constitutive behaviour of the membrane compared with the original model. Furthermore, the convergence plots (bottom) show that for compressible and incompressible Neo-Hookean and Mooney-Rivlin models, the convergence rate is optimal, i.e., quadratic, for Newton–Raphson iterations, but flattens out due to machine precision.

4.6.2 Square subject to Inflation

In the next example, the inflation of a square membrane is modelled (see figure 4.3). This example is inspired by the works of Diaby *et al.* [152], among others. In most previous



Geometry		
L, W	1.0	[m]
t	1.0	[mm]
Material *Neo-Hookean or Mooney-Rivlin*		
μ	1.5	[MPa]
ν	0.45 (comp.)	[–]
	0.50 (incomp.)	[–]
E	$2\mu(1+\nu)$	[MPa]
c_1/c_2	7.0	[–]
Boundary Conditions		
Γ_1	$u_y = 0$	
Γ_2,Γ_3	Free	
Γ_4	$u_x = 0$	
Loads		
p	1.0	[MPa]

Figure 4.1: Set-up for the uniaxial tension benchmark problem. In the figure on the left, the filled geometry represents the deformed configuration, and the dashed line indicates the undeformed geometry. The load p indicates a line load acting on the undeformed geometry. The table on the right provides the parameter values for the specific benchmark problem for Neo-Hookean (NH) and Mooney-Rivlin (MR) materials.

works, the inflated square membrane was modelled using linear elastic models [114, 279, 301, 402, 662], but in the work of Diaby *et al.* [152] the case was used with a hyperelastic Neo-Hookean material model, which is adopted in this chapter as well. Furthermore, the inflation is modelled using a surface loading $\boldsymbol{f}(\boldsymbol{u}) = p\hat{\boldsymbol{n}}$, see equation (4.16), with the pressure $p = 5000\,[\text{Pa}]$. The simulation is solved in two stages, inspired by Diaby *et al.* [152], to ensure bounded equilibrium iterations for the membrane. First, an in-plane boundary load of $p = 5000\,[\text{Pa}]$ orthogonal to the boundary is applied to Γ_1 and Γ_4 to pre-stretch the membrane. Thereafter, the pressure is applied to the pre-stretched domain, and the boundary load is removed. The equilibrium iterations are performed using a two-stage procedure with the Dynamic Relaxation method followed by Newton–Raphson iterations (see section 4.5). The threshold for the dynamic relaxation method is $\epsilon_R = 10^{-2}$ and the threshold for the Newton–Raphson iterations is $\epsilon_R = 10^{-4}$.

Figure 4.4 depicts the results of the inflated square membrane. As can be seen in figure 4.4a, the membrane inflates into a pillow-shape with wrinkles along the free boundaries. Indeed, figure 4.4b shows that the tension field in the boundary regions indicates wrinkling, whereas the other parts are in a taut state, which is in line with similar observations as in [402]. The contour lines on the $z = 0$-plane, see figure 4.4c, show large differences in the wrinkling pattern, varying the number of elements and the degree of the basis for the Kirchhoff–Love shell model. However, the membrane model using the tension field theory modification for hyperelastic materials as presented accurately captures the wrinkling mid-plane and is consistent across mesh refinement and degree elevation. The numerical results in table 4.1 show that the mid-point and corner-point displacements

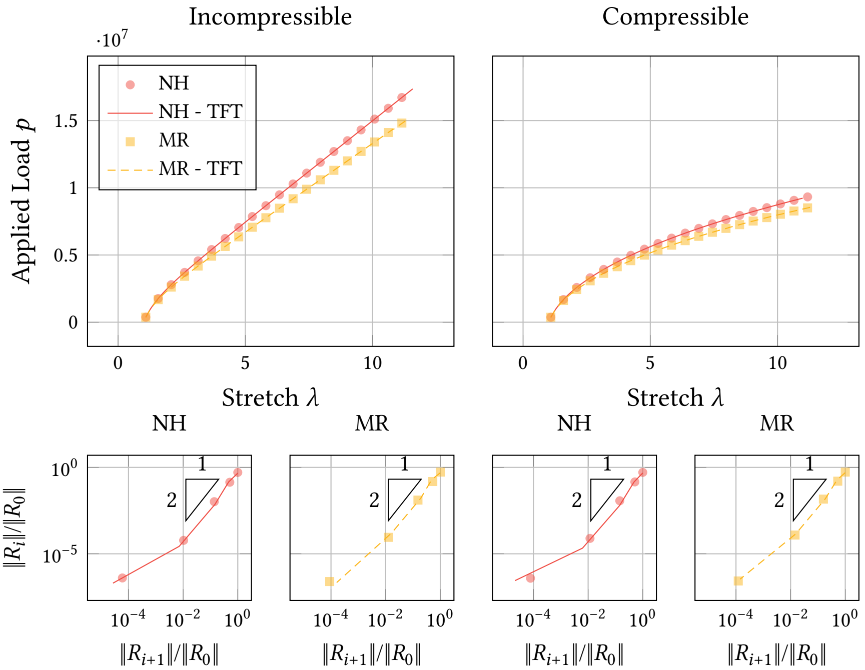
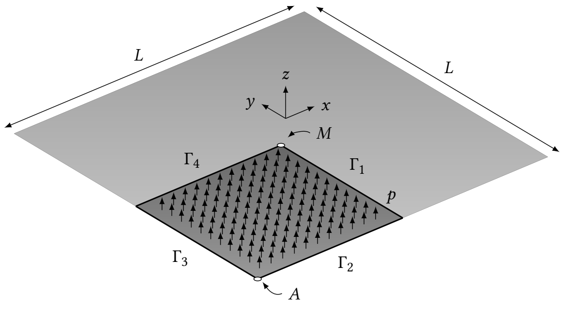


Figure 4.2: Results for the uniaxial tension benchmark problem from figure 4.1. The top figures depict the stretch λ versus the applied load p. The bottom figures present the convergence of the relative residual norm $\|\boldsymbol{R}_i\|/\|\boldsymbol{R}_0\|$ for a load step with $p = 1.0\,[\mathrm{MPa}]$, with the triangle indicating second-order convergence. The results are depicted for Neo-Hookean (NH) and Mooney-Rivlin (MR) materials with the parameters from figure 4.1. The lines indicate the results without a tension field theory (TFT) modification, and the markers indicate the results including the modification proposed in this chapter.



Geometry		
L	$\sqrt{1.2^2/2}$	[m]
t	0.1	[mm]
Material *Compressible Neo-Hookean*		
E	588	[MPa]
ν	0.4	[-]
Boundary Conditions		
Γ_1	$u_x = 0$	
Γ_2,Γ_3	$u_z = 0$	
Γ_4	$u_y = 0$	
Loads		
p	5000	[Pa]

Figure 4.3: Problem definition for a square membrane with diagonal length 1.2 [m] subject to pressure p. The membrane is modelled using in-plane symmetry boundary conditions on Γ_1 and Γ_4. Furthermore, the sides Γ_2 and Γ_3 have restricted z-displacement. The square membrane has a Neo-Hookean material model with the parameters provided in the table on the right.

Table 4.1: Results for the inflated square membrane; see figure 4.3 for the problem parameters; for the Kirchhoff–Love (KL) shell resolving actual wrinkles; and for the membrane model using the tension field theory (TFT) modifications proposed in this chapter. The reference results are provided as the vertical displacement in the mid-point M, $u_{z,M}$, and as the horizontal displacement along the x-axis in the point A, $u_{x,A}$. The results are compared with the work of Diaby *et al.* [152].

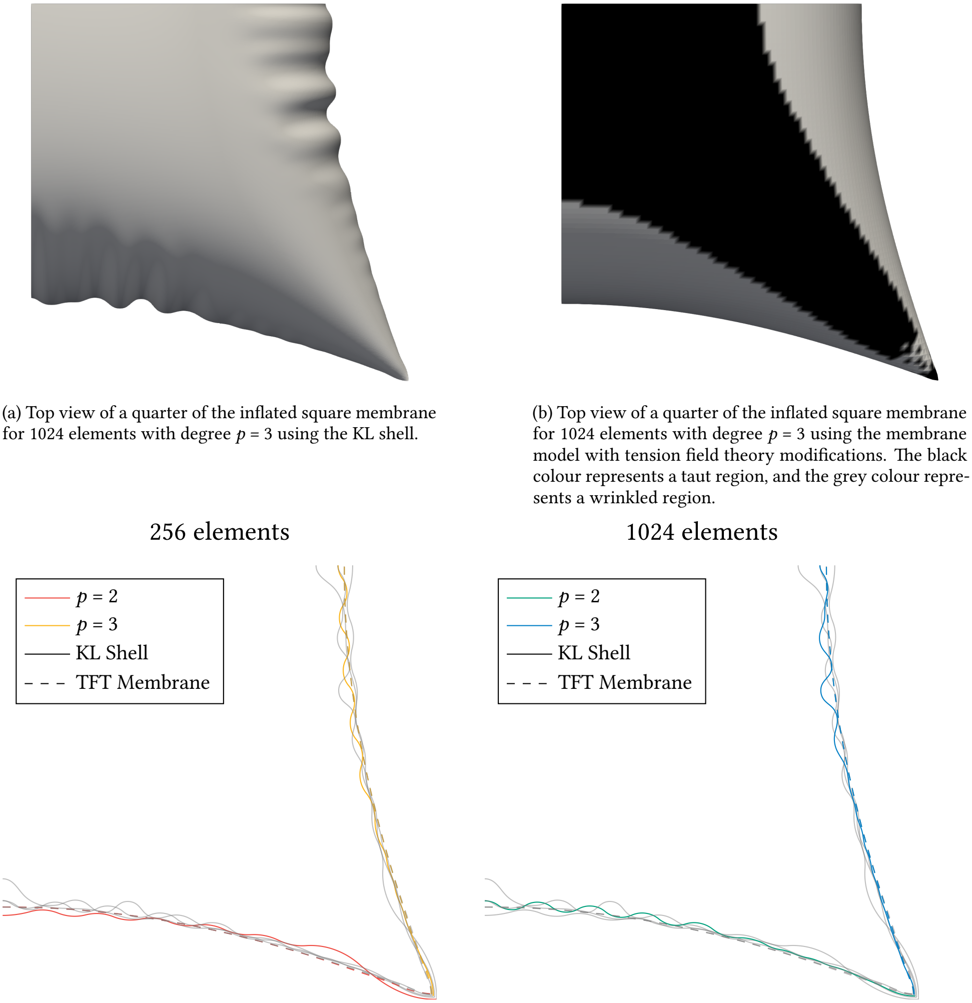
		KLShell			Membrane (TFT)			Diaby *et al.* [152]	
# elements		64	256	1024	64	256	1024	25	625
$u_{z,M}$	$p = 2$	0.2093	0.2197	0.2231	0.2167	0.2181	0.2187	0.2144	0.2245
	$p = 3$	0.2133	0.2173	0.2205	0.2138	0.2190	0.2190		
$u_{x,A}$	$p = 2$	0.0236	0.0252	0.0282	0.0312	0.0302	0.0298	0.0265	0.0307
	$p = 3$	0.0251	0.0270	0.0293	0.0278	0.0299	0.0296		

of the membrane rapidly converge for the tension field membrane simulation, compared to the shell simulation. The final values of both simulations show small differences between the results obtained by the KL shell model and the membrane TFT model and slightly bigger differences with the results obtained by the shell model of Diaby *et al.* [152], possibly because of the lower degree finite element method and the lower number of elements used there.

4.6.3 Annulus Subject to Tension and Twist

As a next example, an annular planar surface subject to an out-of-plane translation and a twist of 90° of the inner-boundary Γ_i is modelled, see figure 4.5. The example is inspired by the work of Taylor *et al.* [551] using the same geometric dimensions but different material parameters. An incompressible Neo-Hookean material model with the strain energy density function from equation (4.90) is used as constitutive model, with the parameters from figure 4.5. Furthermore, the example is solved using a Dynamic Relaxation method with tolerance $\epsilon = 10^{-1}$ followed by a Newton method with tolerance $\epsilon = 10^{-6}$. The geometry is modelled using 4 patches representing a quarter annulus of degree $p = 2$ with 16 elements in both directions per patch for the Kirchhoff–Love shell simulation and 8 elements in both directions for the tension field theory membrane simulation. The interfaces of the patches are smoothened with C^1 continuity using basic interface smoothing.

Figure 4.6 shows the results for the hyperelastic annulus subject to a vertical displacement and a rotation at the inner boundary. Firstly, figure 4.6a and figure 4.6b show a qualitatively good comparison between the wrinkled region given by the Kirchhoff–Love shell model and the tension field from the membrane model. Indeed, as shown by the contours in figure 4.6d, the tension field theory membrane model accurately predicts the average of the wrinkles given by the wrinkling simulation using the shell model, which is as expected from the construction of the model. Furthermore, it should be noted that the wrinkling deformations depicted in figure 4.6 have four times more elements than the tension field theory model.

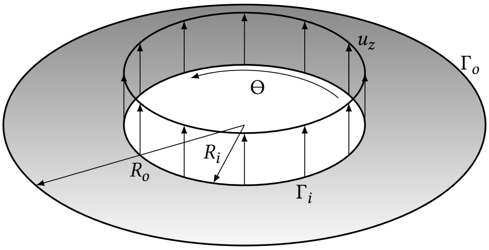


(a) Top view of a quarter of the inflated square membrane for 1024 elements with degree $p = 3$ using the KL shell.

(b) Top view of a quarter of the inflated square membrane for 1024 elements with degree $p = 3$ using the membrane model with tension field theory modifications. The black colour represents a taut region, and the grey colour represents a wrinkled region.

(c) Contour lines at $z = 0$ for the inflated square membrane for a mesh with 256 elements (left) and with 1024 elements (right) for degrees $p = 2$ and $p = 3$. The results of the shell simulation are depicted as solid lines, and the membrane tension field theory results are represented by dashed lines. The coloured lines highlight a certain number of elements and degrees, using symmetry for the horizontal and vertical directions, for the purpose of compactness. The grey lines indicate the other non-highlighted lines.

Figure 4.4: Results for the square membrane subject to a pressure load from figure 4.3. (a) represents the deformed shape from a Kirchhoff–Love shell simulation, providing wrinkles. (b) provides the deformed shape from a membrane simulation with the proposed tension field theory modification scheme, together with the tension field. (c) provides the contours of the deformation for both models for different numbers of elements and degrees.



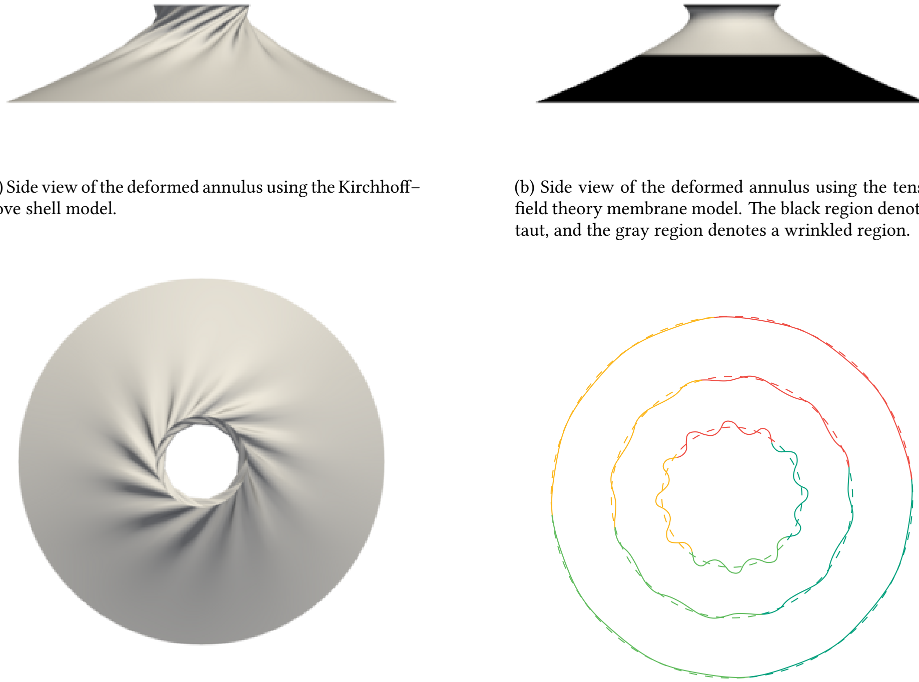
Geometry		
R_i	62.5	[mm]
R_o	250	[mm]
t	0.05	[mm]
Material *Incompressible Neo-Hookean*		
E	1.0	[GPa]
ν	0.5	[–]
Boundary Conditions		
Θ	$\pi/2$	[rad]
u_z	125	[mm]

Figure 4.5: Problem definition for an annulus with inner radius R_i and outer radius R_o subject to a vertical translation u_z and a rotation Θ on the inner boundary Γ_i while being fixed on the outer boundary Γ_o. The annulus has a Neo-Hookean material model with the parameters provided in the table on the right.

4.6.4 Cylinder Subject to Tension and Twist

Similar to the previous example regarding the pulled hyperelastic annulus (see section 4.6.3), the next benchmark models a cylinder subject to a translation along its length and a rotation around its centre axis (see figure 4.7). As for the annulus in section 4.6.3, an incompressible Neo-Hookean material model is used. Furthermore, the problem is solved using a Dynamic Relaxation method with tolerance $\epsilon = 10^{-1}$ followed by a Newton method with tolerance $\epsilon = 10^{-6}$, and the geometry is modelled with 4 patches representing a quarter of a cylinder with degree $p = 2$ and 16 elements per patch in both directions for the Kirchhoff–Love shell simulation and 8 elements per patch in both directions for the tension field theory membrane simulation. The interfaces of the patches are smoothed with C^1 continuity using basic interface smoothing. Although there are similarities in the problem set-up between the annulus from section 4.6.3 and the cylinder benchmark, it should be noted that the case of the cylinder involves larger strains, hence this example is more suitable for hyperelastic material models.

Figure 4.8 shows the results for the hyperelastic cylinder subject to a combined extension and rotation, as described in figure 4.7. The wrinkling results from the Kirchhoff–Love shell model are presented in figures 4.8a and 4.8c, respectively, as a side and top view. Furthermore, the tension field from the modification scheme proposed in this chapter is provided in figure 4.8b. Comparing figure 4.8a with figure 4.8b shows that the present model captures the envelope of the wrinkles qualitatively well. In addition, figure 4.8d shows that the dashed contour lines approximate the mean of the wrinkles, as expected based on the model of Roddeman *et al.* [477, 478]. As for the hyperelastic annulus, the results of the cylinder also have four times more elements for the shell model compared to the tension field theory model.



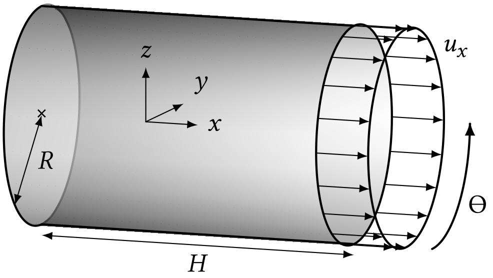
(a) Side view of the deformed annulus using the Kirchhoff–Love shell model.

(b) Side view of the deformed annulus using the tension field theory membrane model. The black region denotes a taut, and the gray region denotes a wrinkled region.

(c) Top view of the deformed annulus using the Kirchhoff–Love shell model.

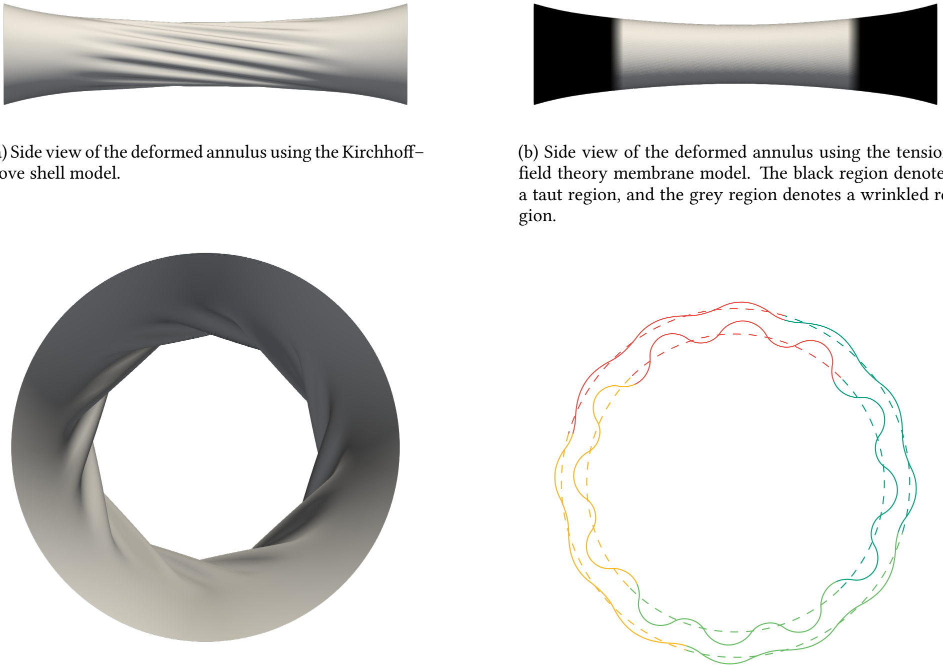
(d) Contour lines for parametric coordinates $\xi_r \in \{0.5, 0.7, 0.9\}$ (outside to inside) such that $\xi_r = 0$ corresponds to R_o and $\xi_r = 1$ with R_i. The solid lines represent the Kirchhoff-Love shell results, and the dashed lines represent the tension field theory membrane results. The colours are used to indicate the different patches.

Figure 4.6: Results of the example with an annulus with fixed outer boundary and with an inner boundary subject to a translation and a rotation, see figure 4.5. A side view (a) and a top view (c) of the wrinkled membrane using the Kirchhoff–Love shell simulation are provided, as well as the deformed geometry from the tension field theory membrane simulation with the tension field for colouring (b). Furthermore, contour lines of the deformation for different parametric coordinates are provided; see (d).



	Geometry	
R	250	[mm]
L	1.0	[m]
t	0.05	[mm]
	Material *Incompressible Neo-Hookean*	
E	1.0	[GPa]
ν	0.5	[–]
	Boundary Conditions	
Θ	$\pi/2$	[rad]
u_x	1.0	[m]

Figure 4.7: Problem definition for an cylinder with inner radius R and length L subject to an elongation u_x and a rotation Θ on the right boundary Γ_r while being fixed on the left boundary Γ_l. The cylinder has a Neo-Hookean material model with the parameters provided in the table on the right.



(a) Side view of the deformed annulus using the Kirchhoff–Love shell model.

(b) Side view of the deformed annulus using the tension field theory membrane model. The black region denotes a taut region, and the grey region denotes a wrinkled region.

(c) Top view of the deformed annulus using the Kirchhoff–Love shell model.

(d) Contour lines for parametric coordinates $\xi_x \in \{0.50, 0.75\}$ (left to right) such that $\xi_x = 0$ and $\xi_x = 1$ correspond to the left and right of the cylinder, respectively. The solid lines represent the Kirchhoff-Love shell results, and the dashed lines represent the tension field theory membrane results. The colours are used to indicate the different patches.

Figure 4.8: Results of the example with an annulus with fixed outer boundary and with an inner boundary subject to a translation and a rotation, see figure 4.5. A side view (a) and a top view (c) of the wrinkled membrane using the Kirchhoff–Love shell simulation are provided, as well as the deformed geometry from the tension field theory membrane simulation with the tension field for colouring (b). Furthermore, contour lines of the deformation for different parametric coordinates are provided; see (d).

4.7 Conclusions

Aiming for efficient modelling of membrane wrinkling, this chapter presents a modification strategy for wrinkled hyperelastic membranes through implicit modelling. The model extends the work of Nakashino & Natori [401, 402], who presented the modification scheme for linear elastic membranes based on tension fields. The modification scheme for hyperelastic membranes presented in this chapter uses the assumptions from Roddeman *et al.* [477, 478], implying a modification of the deformation tensor rather than the constitutive model. As a consequence, the kinematic equation changes. Assuming a non-linear constitutive relation, the introduction of the wrinkling strain tensor adds contributions to the stress and material tensors, including dependency on the derivative of the material tensor with respect to the strain tensor. Since the latter term can be difficult to obtain due to condensation of the through-thickness strains for incompressible and compressible material models, it is computed through finite differences in the present work. Since the modification scheme is defined through derivatives of the strain energy density function, it can be used for general hyperelastic materials.

The present model is verified using a series of benchmarks. The isogeometric Kirchhoff–Love shell formulation is used as a reference for wrinkling computations. The proposed tension field theory-based (TFT) membrane is implemented in isogeometric finite elements as well, although the formulations do not necessarily depend on high geometric continuity. The first benchmark problem involves an uniaxial tension test, verifying the speed of convergence for a fixed tension field. The second benchmark demonstrates application on an inflatable hyperelastic membrane, showing good mesh convergence in terms of displacements compared to the shell model. Third, the TFT model applied to an annular geometry subject to displacement and rotation of the inner boundary shows a good comparison with the shell model, similar to the linear model from Taylor *et al.* [551]. Similar to the annular membrane, a new benchmark problem is provided using a cylinder that is elongated and rotated along its central axis. In this case, the TFT membrane model provides great similarity with the shell model. Concluding, the benchmarks demonstrate the validity and applicability of the proposed model. It should be noted that the use of the Dynamic Relaxation method in combination with the Newton–Raphson method is necessary in some situations due to evolving tension fields during the iterations, making the Newton–Raphson methods diverge with a poor initial guess.

As for the linear elastic model from Nakashino & Natori [401], the present model relies on a tension field evaluated given a certain deformation. If the tension field is subject to large changes during iterations, the convergence behaviour of the model deteriorates since the changing tension field is not included in the variation of the wrinkling stress. In future work, it is recommended to incorporate the tension field into the derivative of the wrinkling stress. Furthermore, an equivalent for the Dynamic-Relaxation method combined with Newton–Raphson methods for quasi-static simulations could be investigated as well, starting with the explicit arc-length method from Lee *et al.* [333].

4.A Result Reproduction

For the sake of reproducibility of the results in this chapter, this appendix provides brief instructions on the use of the software developed along with this thesis. The full software is available as part of the Geometry + Simulation Modules. For more detail on the contributions to this software library, and its installation, the reader is referred to chapter 8.

Table 4.2 provides per figure in this chapter the name of the file to run along with the arguments to be passed to obtain these figures. The benchmark problems in this chapter make use of a single executable to run most of the examples. The executable takes an `XML` input file defining the whole problem.

Table 4.2: File name and run arguments required for the reproducibility of the figures in this chapter. Arguments with a single dash (-) require an argument while double-dashed arguments (--) are switches. See chapter 8 for more detail about the software and installation instructions. The `XML` files are located in `gsStructuralAnalysis/filedata/pde/`.

Figure		**Run File**	
	Arg.	**Description**	**Values**
Figure 4.2	`static_shell_XML`		
	`-i`	Input file	`uniaxial_tension_1p.xml`†
		†*Different material models can be (un)commented in this file*	
	`--NR`	Use Newton–Raphson solver	
Table 4.1	`benchmark_Pillow`		
Figure 4.4	`-e`	Number of degree elevation steps	1: $p = 2$, 2: $p = 3$
	`-r`	Number of uniform refinement steps	4: 256 elements, 5: 1024 elements
	`-a`	Mass tuning factor of Dynamic–Relaxation solver	`1e14`
	`--TFT`	Use Tension-Field Theory	
	`--NR`	Use Newton–Raphson solver	
	`--DR`	Use Dynamic–Relaxation solver	
Figure 4.6	`static_shell_multipatch_XML`		
	`-i`	Input file	`annulus_4p.xml`
	`-e`	Number of degree elevation steps	1: $p = 2$, 2: $p = 3$
	`-r`	Number of uniform refinement steps	4: 256 elements, 5: 1024 elements
	`--NR`	Use Newton–Raphson solver	
	`--DR`	Use Dynamic–Relaxation solver	
Figure 4.8	`static_shell_multipatch_XML`		
	`-i`	Input file	`cylinder_4p.xml` `cylinder_4p_TFT.xml`
	`-e`	Number of degree elevation steps	1: $p = 2$, 2: $p = 3$
	`-r`	Number of uniform refinement steps	4: 256 elements, 5: 1024 elements
	`--NR`	Use Newton–Raphson solver	
	`--DR`	Use Dynamic–Relaxation solver	

5

Goal-Adaptive Meshing of Isogeometric Kirchhoff–Love Shells

In this chapter, an adaptive isogeometric method for Kirchhoff–Love shell modelling is developed. This adaptive isogeometric method is a combination of an error estimator and a mesh refinement algorithm, with the goal of optimally allocating degrees of freedom. The error estimator used in this chapter is based on the so-called Dual-Weighted Residual (DWR) method. This uses a pre-defined goal functional, e.g., the displacement at a point or an eigenfrequency, and provides an estimate for the error in terms of this goal functional with respect to a converged solution. The DWR error contributions can be computed as a field over the computational domain, making them suitable as an error estimator for the adaptive meshing method. Using the Dörfler marking and suitably graded Truncated Hierarchical B-splines (THB-splines), a marking and meshing strategy is developed for combined refinement and coarsening of the computational domain. The developed goal-adaptive isogeometric method for Kirchhoff–Love shell models is applied to various structural analysis problems, including modal analysis, buckling analysis, and non-linear snap-through and bifurcation problems, showing high accuracy of the DWR estimator and efficient allocation of degrees of freedom for advanced shell computations. In conclusion, the developed framework demonstrates the potential of the DWR to serve as an error estimator in several different structural analyses, with the freedom of choosing goal functionals closely related to engineering practice. If necessary, the reader is referred to section 2.3.2 for a full derivation of the isogeometric Kirchoff–Love shell formulation, to section 2.4 for the mathematics behind several numerical structural analysis routines and to section 2.2.1 for a brief background on refinement splines.

This chapter is under revision for the journal *Engineering with Computers.* A pre-print is available:
[586] **H. M. Verhelst**, M. Möller, A. Mantzaflaris & J. H. Den Besten, “Goal-Adaptive Meshing of Isogeometric Kirchhoff–Love Shells”, arXiv:2307.08356 (2023)

5.1 Introduction

The idea behind isogeometric analysis (IGA) [268] is to bridge the gap between computer-aided design (CAD) and finite element analysis (FEA). By employing B-splines or non-uniform rational B-splines (NURBS) as the basis for FEA, IGA not only provides geometrically exact analysis, but the high smoothness of the spline bases also provides high accuracy per degree of freedom [488]. The close link with conventional engineering fields such as automotive, offshore, aircraft, or civil engineering makes structural analysis with isogeometric analysis a particular field of interest. Besides the performance of the different isogeometric element formulations for thin Kirchhoff–Love shells [5, 319, 320, 587], moderately thick Reissner-Mindlin shells [43, 45, 263, 317, 516] or thicker solid-like shells [256, 341] in static and dynamic simulations, conventional engineering disciplines also rely on accurate modal and (post-)buckling simulations. In addition, the ability to handle complex (multipatch) CAD geometries via trimming [116, 219, 335] or patch coupling methods [62, 118, 222, 238, 342] improves the applicability of IGA in structural engineering. For problems with a large number of degrees of freedom or problems with a large number of load or time steps, mesh adaptivity can play a key role in providing efficient simulations for industrial applications.

A loop in an adaptive isogeometric method (AIGM) consists of the steps *solve* the Partial Differential Equation (PDE) at hand, *estimate* element-wise error contributions, *mark* regions for refinement, *refine (coarsen)* marked regions [79], see figure 5.1. Here, localised regions can be defined element-wise or function-wise. The AIGM process can be repeated in an iterative manner (e.g., for static, buckling, or modal analysis) until satisfactory accuracy is achieved, or it can be applied iteratively within a time- or load-stepping procedure. A broad overview of the mathematical foundations of AIGMs is given in [78]. In previous works, AIGMs were developed for different applications (*solve*), using different *estimation* strategies, *marking* strategies, and often for mesh *refinement*, with a few also providing *coarsening* strategies [84, 137, 199, 234, 363].

Solve The *solve* block contains the partial differential equation (PDE) at hand. It can be a physics-based problem, e.g., to solve shell [13, 116, 117], linear elasticity [590] or free-surface flow [328] problems. Alternatively, the solve step can involve a non-physics PDE, e.g., for mesh generation [244].

Estimate Determination of localised errors is done in the *estimate* block. In the works [13, 116, 117], an error estimator based on a residual-like variational problem in the so-called *bubble-space* was presented for Kirchhoff plates, Kirchhoff–Love shells, and trimmed domains. This method has proven a large decrease in CPU time compared to a residual-based error estimator in its strong form, due to its easy parallelisation and the small block structure of the linear system to solve. As an alternative to this method, error estimation can also be performed in a goal-oriented fashion, e.g., by the Dual-Weighted Residual (DWR) method. This method has been applied in the FEA context in various works [102, 200, 228, 229, 229, 230, 391, 460] and was used in the works [328, 572] for Poisson and free-boundary problems, in [147] for a geometrically non-linear rod, in [244] for PDE-based domain parametrisations,

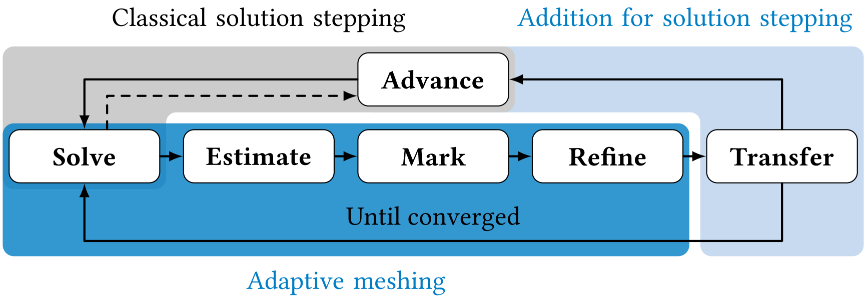


Figure 5.1: A typical flowchart for an adaptive meshing routine. The *classical solution stepping* depicts a process without adaptive meshing. Here, a solution is obtained by the *solve*, and the solution is advanced (e.g., in time or load step) and recomputed. The *adaptive meshing step* denotes the additional operations for mesh adaptivity, and the *Addition for solution stepping* includes an additional *transfer* step in case the adaptive meshing method is applied to a solution. The *Estimate* block provides an error estimation with local contributions per element or per degree of freedom (DoF). The *Mark* block contains a marking rule that marks regions for refinement based on a specific rule. The *Refine* block transforms the current mesh to a new mesh, where regions are refined and coarsened based on the marking rule. The block *Transfer* transfers the previous solution to the new mesh so that it can be used to recompute the present interval on a modified mesh. This recomputation is performed again in the *Solve* block and follows through the subsequent blocks until an adaptivity criterion is reached. For example, a criterion that requires the total error in the mesh to be within certain bounds.

and in [590] for micromechanical modelling of trabecular bone. Goal-oriented refinement in general provides localised error estimates by solving a linear adjoint problem on the current space and an enriched space.

Mark As soon as localised error contributions are known, regions can be marked for refinement. This marking is mostly done using the Dörfler marking strategy [160], as in [79, 84, 200], which involves marking the regions with the largest error contributions until their sum exceeds a certain percentage of the total error. An alternative is to mark the regions with an error higher than a threshold (an absolute threshold based on the maximum error) [13, 202] or based on a relative threshold, taking a fixed percentage of the total number of cells for refinement. In [202], the latter two strategies are discussed.

Refine Local refinement for adaptive meshing in isogeometric analysis is enabled by the use of Hierarchical B-splines (HB-splines) [591], Truncated Hierarchical B-splines (THB-splines) [202, 203], or T-splines [36], amongst other spline constructions, which are reviewed in [236]. HB-splines provides a nested, linearly dependent space that violates the partition of unity property. To preserve the latter, THB-splines have been introduced in [203]. For (T)HB-splines, *suitable grading* to generate admissible meshes should be taken into account in order to guarantee a bounded error [79], for which algorithms have been presented in [67]. In the work of [235], a distinction is made between *greedy* and *safe* refinement, the former being a refinement of cells with a 1-level difference with adjacent cells and the latter being a refinement complying with the refinement neighbourhoods defined in [79]. Besides adaptive meshing for solving PDEs [84, 137, 199, 234, 244, 363], THB-splines have also been successfully applied in the context of fitting [66, 323].

Transfer The *transfer* from previous time/load steps onto a new mesh can be done using different methodologies. In the work of [234], different least-squares approaches are provided. Furthermore, quasi-interpolation [205, 519] is a technique that can be used to transfer solutions between hierarchical meshes.

In this chapter, goal-oriented adaptive refinement for isogeometric thin shell analysis is employed, to facilitate THB-adaptive meshing for a variety of structural analysis problems. The developed framework is versatile in terms of the goal functional being used and provides an adaptive meshing strategy for linear an non-linear static, modal, buckling and post-buckling problems. In brief, the contribution of the chapter is threefold. Firstly, the Dual-Weighted Residual (DWR) method is used to derive novel error estimators for structural shell analysis, given goal functionals based on displacements, (principal) stresses and strains, forces, and moments. Secondly, the eigenvalue DWR method from [330, 460] is used for error estimations for modal and buckling analyses. Lastly, the goal functionals are used to drive an adaptive meshing strategy with suitable grading and efficient transfer of solutions by quasi-interpolation method on hierarchical spline spaces [205, 519]. This adaptive meshing strategy is applied to non-linear shell analysis with focus on buckling problems with snap-through and bifurcation instabilities - being new applications in the realm of adaptive meshing research for nonlinear shell problems. It should be noted that the present framework is developed for isogeometric Kirchhoff–Love shells - since it provides a natural separation of bending and membrane terms - but it is easily adapted for other shell formulations. By defining a frame work for 2-dimensional parametric domains and by presenting a wide range of mechanics-inspired goal functionals, the present work extends an earlier work by [147] for geometrically non-linear rods.

The chapter is structured as follows: In section 5.2, the isogeometric Kirchhoff–Love shell analysis proposed by [319] is briefly revised, and some basic concepts for structural analysis computations are given. In section 5.3, the Dual-Weighted Residual (DWR) method is provided for the isogeometric Kirchhoff–Love shell using the membrane and flexural strain split. However, it can be used for general elasticity problems. Moreover, the section provides the DWR method for eigenvalue problems to compute error estimators for modal and buckling analyses. Thereafter, section 5.4 provides the details for adaptivity for isogeometric analysis. This includes the concept of Truncated-Hierarchical B-splines (THB-splines) and admissible refinement. Furthermore, the *mark* and *transfer* operations are described. In section 5.5, a summary of the preceding sections is provided by means of a global algorithm for the AIGM for structural analysis computations with load-stepping. In section 5.6, the present work is evaluated on numerical benchmark problems, ranging from linear problems with analytical solutions to non-linear shell problems. Finally, section 5.7 provides conclusions and an outlook based on this work.

5.2 Isogeometric Kirchhoff–Love Shell Analysis

In this section, provide a brief background on the Kirchhoff–Love shell formulation is provided. For more details on this formulation, the reader is referred to [319, 320, 480, 490, 587] and to chapter 2 of this dissertation.

5.2.1 Shell Kinematics

Since Kirchhoff–Love shells satisfy the Kirchhoff Hypothesis [461], the coordinates $\boldsymbol{x}$ of any parametric point $\boldsymbol{\theta} = (\theta^1, \theta^2, \theta^3)$ in the shell surface can be represented by the surface position $\boldsymbol{r}(\theta^1, \theta^2)$ and contribution in normal direction $\theta^3 \hat{\boldsymbol{a}}_3$ as

$$\boldsymbol{x}(\boldsymbol{\theta}) = \boldsymbol{r}(\theta^1, \theta^2) + \theta^3 \hat{\boldsymbol{a}}_3, \tag{5.1}$$

Given the covariant basis of the surface $\boldsymbol{r}$, defined by $\boldsymbol{a}_\alpha, \alpha = 1, 2$ and the orthogonal unit normal $\hat{\boldsymbol{a}}_3$, the covariant basis of $\boldsymbol{x}$ is defined as follows:

$$\mathbf{g}_\alpha = \boldsymbol{x}_{,\alpha} = \boldsymbol{a}_\alpha + \theta^3 \hat{\boldsymbol{a}}_{3,\alpha}, \quad \mathbf{g}_3 = \boldsymbol{x}_{,3} = \hat{\boldsymbol{a}}_3. \tag{5.2}$$

Given the second fundamental form $b_{\alpha\beta} = \hat{\boldsymbol{a}}_3 \cdot \boldsymbol{a}_{\alpha,\beta} = -\boldsymbol{a}_{3,\beta} \cdot \boldsymbol{a}_\alpha$ and the metric coefficients defined as

$$g_{\alpha\beta} = \mathbf{g}_\alpha \cdot \mathbf{g}_\beta = a_{\alpha\beta} - 2\theta^3 b_{\alpha\beta}, \tag{5.3}$$

the contravariant basis vectors $\mathbf{g}^\alpha$ can simply be obtained by $\mathbf{g}^\alpha = g^{\alpha\beta} \mathbf{g}_\beta$. The undeformed configuration $\boldsymbol{r}$ and the deformed configuration $\mathring{\boldsymbol{r}}$ of the surface are related by $\boldsymbol{r} = \mathring{\boldsymbol{r}} + \boldsymbol{u}$. From the defintion of the deformation gradient $\mathbf{F} = \mathbf{g}_i \otimes \mathring{\mathbf{g}}^i$, the deformation tensor $\mathbf{C}$ can be obtained:

$$\mathbf{C} = \mathbf{F}^\top \mathbf{F} = \mathbf{g}_i \cdot \mathbf{g}_j \, \mathring{\mathbf{g}}^i \otimes \mathring{\mathbf{g}}^j = g_{ij} \, \mathring{\mathbf{g}}^i \otimes \mathring{\mathbf{g}}^j. \tag{5.4}$$

Note that the deformation tensor is defined in the contravariant undeformed basis $\mathring{\mathbf{g}}^i \otimes \mathring{\mathbf{g}}^j$. For Kirchhoff–Love shells, it is known that $g_{\alpha 3} = g_{3\alpha} = 0$, hence this implies $C_{\alpha 3} = C_{3\alpha} = 0$, since $g_{33} = 1$, which implies C_{33} to be one and meaning that the thickness remains constant under deformation. As a result, the Green-Lagrange strain tensor $\mathbf{E} = E_{\alpha\beta} \, \mathring{\mathbf{g}}^\alpha \otimes \mathring{\mathbf{g}}^\beta$ and its decomposition to membrane and bending contributions (ε and κ, respectively) is [319, 320]:

$$\begin{aligned} E_{\alpha\beta} &= \frac{1}{2}\left(g_{\alpha\beta} - \mathring{g}_{\alpha\beta}\right) = \frac{1}{2}\left((a_{\alpha\beta} - \mathring{a}_{\alpha\beta}) - 2\theta^3\left(b_{\alpha\beta} - \mathring{b}_{\alpha\beta}\right)\right) \\ &= \varepsilon_{\alpha\beta} + \theta_3 \kappa_{\alpha\beta}. \end{aligned} \tag{5.5}$$

5.2.2 Constitutive Relation

The constitutive relations for the Kirchhoff–Love shell relate the Green-Lagrange strain tensor $\mathbf{E}$ to the second Piola-Kirchhoff stress tensor $\mathbf{S}$. For linear elastic materials, this is achieved by:

$$S^{\alpha\beta} = \mathscr{C}^{\alpha\beta\gamma\delta} E_{\gamma\delta} \tag{5.6}$$

where $\boldsymbol{\mathscr{C}} = \mathscr{C}^{\alpha\beta\gamma\delta} \, \mathring{\mathbf{g}}_i \otimes \mathring{\mathbf{g}}_j \otimes \mathring{\mathbf{g}}_k \otimes \mathring{\mathbf{g}}_l$ is the material tensor, which takes for linear materials the form $\mathscr{C}^{\alpha\beta\gamma\delta} = 4\frac{\lambda\mu}{\lambda+2\mu} \mathring{g}^{\alpha\beta} \mathring{g}^{\gamma\delta} + 2\mu\left(\mathring{g}^{\alpha\delta} \mathring{g}^{\beta\gamma} + \mathring{g}^{\alpha\gamma} \mathring{g}^{\beta\delta}\right)$ [208]. For non-linear hyperelastic constitutive relations, the stress and material tensors are derived from the 3D constitutive relations for (in)compressible materials and due to through-thickness integration, the shell normal force and bending moment tensors $\mathbf{N} = N^{\alpha\beta} \, \mathring{\mathbf{g}}_\alpha \otimes \mathring{\mathbf{g}}_\beta$ and $\mathbf{M} = M^{\alpha\beta} \, \mathring{\mathbf{g}}_\alpha \otimes \mathring{\mathbf{g}}_\beta$, respectively, are defined as

$$N^{\alpha\beta}(\boldsymbol{u}) = \int_T S^{\alpha\beta}(\boldsymbol{u}) \, \mathrm{d}\theta^3, \quad M^{\alpha\beta}(\boldsymbol{u}) = \int_T \theta^3 S^{\alpha\beta}(\boldsymbol{u}) \, \mathrm{d}\theta^3, \tag{5.7}$$

where $T = [-t/2, t/2]$ is the through-thickness domain. For more details on hyperelastic material models, the reader is referred to [320, 480] and specifically for stretch-based ones to [587].

5.2.3 Variational Formulation

The shell internal and external equilibrium equations in variational form are derived by the principle of virtual work [319, 320]. The weak formulation follows from the principle of virtual work with virtual displacements $\boldsymbol{\phi}$:

$$\begin{aligned}\text{Find } \boldsymbol{u}\in\mathbb{S} \text{ s.t.} \mathscr{W}(\boldsymbol{u},\boldsymbol{\phi}) &:= \delta W^{\text{int}} - \delta W^{\text{ext}}\\ &= \int_\Omega \mathrm{N}(\boldsymbol{u}) : \boldsymbol{\varepsilon}'(\boldsymbol{u},\boldsymbol{\phi}) + \mathrm{M}(\boldsymbol{u}) : \boldsymbol{\kappa}'(\boldsymbol{u},\boldsymbol{\phi})\,\mathrm{d}\Omega - \int_\Omega \boldsymbol{f}(\boldsymbol{u})\cdot\boldsymbol{\phi}\,\mathrm{d}\Omega, \\ \forall\boldsymbol{\phi}\in\mathbb{S}\end{aligned} \tag{5.8}$$

With $\boldsymbol{f}(\boldsymbol{u})$ the surface load acting on the mid-surface, for the sake of generality defined as a function of the displacements $\boldsymbol{u}$ (e.g. a follower pressure p gives $\boldsymbol{f}(\boldsymbol{u}) = p\hat{\boldsymbol{a}}_3(\boldsymbol{u})$). Furthermore, $\boldsymbol{\varepsilon}'(\boldsymbol{u},\boldsymbol{\phi})$ and $\boldsymbol{\kappa}'(\boldsymbol{u},\boldsymbol{\phi})$ are the virtual strain components given displacements $\boldsymbol{u}$ and being linear with respect to variation $\boldsymbol{\phi}$, hence $\mathscr{W}(\boldsymbol{u},\boldsymbol{\phi})$ is also linear in its second argument. The coefficients of the variations of the membrane force and bending moment tensors are

$$\begin{aligned}(n')^{\alpha\beta}(\boldsymbol{u},\boldsymbol{\phi}) &= \int_T \mathscr{C}^{\alpha\beta\gamma\delta}(\boldsymbol{u})\,\mathrm{d}\theta^3\,\varepsilon'_{\gamma\delta}(\boldsymbol{u},\boldsymbol{\phi}) + \int_T \theta^3\mathscr{C}^{\alpha\beta\gamma\delta}(\boldsymbol{u})\,\mathrm{d}\theta^3\,\kappa'_{\gamma\delta}(\boldsymbol{u},\boldsymbol{\phi}),\\ (m')^{\alpha\beta}(\boldsymbol{u},\boldsymbol{\phi}) &= \int_T \theta^3\mathscr{C}^{\alpha\beta\gamma\delta}(\boldsymbol{u})\,\mathrm{d}\theta^3\,\varepsilon'_{\gamma\delta}(\boldsymbol{u},\boldsymbol{\phi}) + \int_T \left(\theta^3\right)^2\mathscr{C}^{\alpha\beta\gamma\delta}(\boldsymbol{u})\,\mathrm{d}\theta^3\,\kappa'_{\gamma\delta}(\boldsymbol{u},\boldsymbol{\phi}).\end{aligned} \tag{5.9}$$

Linearizing the virtual work from equation (5.8) provides the continuous equivalent of the Jacobian or *tangential stiffness matrix* for Newton iterations which will be performed to solve the non-linear weak formulation equation (5.8) in a discrete setting [490]:

$$\begin{aligned}\mathscr{W}'(\boldsymbol{u},\boldsymbol{\phi},\boldsymbol{\psi}) := \int_\Omega &\mathrm{N}'(\boldsymbol{u},\boldsymbol{\psi}) : \boldsymbol{\varepsilon}'(\boldsymbol{u},\boldsymbol{\phi}) + \mathrm{N}(\boldsymbol{u}) : \boldsymbol{\varepsilon}''(\boldsymbol{u},\boldsymbol{\phi},\boldsymbol{\psi})\\ &+ \mathrm{M}'(\boldsymbol{u},\boldsymbol{\psi}) : \boldsymbol{\kappa}'(\boldsymbol{u},\boldsymbol{\phi}) + \mathrm{M} : \boldsymbol{\kappa}''(\boldsymbol{u},\boldsymbol{\phi},\boldsymbol{\psi})\,\mathrm{d}\Omega\\ -&\int_\Omega \boldsymbol{f}'(\boldsymbol{u},\boldsymbol{\psi})\cdot\boldsymbol{\phi}\,\mathrm{d}\Omega,\end{aligned} \tag{5.10}$$

where $\boldsymbol{\varepsilon}''(\boldsymbol{u},\boldsymbol{\phi},\boldsymbol{\psi})$ and $\boldsymbol{\kappa}(\boldsymbol{u},\boldsymbol{\phi},\boldsymbol{\psi})$ are the second variations of the membrane and bending strains and $\boldsymbol{f}'$ is the first variation of the applied force, being nonzero when the force is depending on the displacements $\boldsymbol{u}$. For the details on these formulations, the reader is referred to previous publications [208, 319, 490]. It should be noted that in the undeformed case, $\boldsymbol{u} = 0$, the internal membrane forces and bending forces, $\mathrm{N}(\boldsymbol{u})$ and $\mathrm{M}(\boldsymbol{u})$, respectively, vanish. As a result, the continuous equivalent for the linear stiffness matrix is:

$$\mathring{\mathscr{W}}'(\boldsymbol{\phi},\boldsymbol{\psi}) = \int_\Omega \mathring{\mathrm{N}}'(\boldsymbol{\psi}) : \mathring{\boldsymbol{\varepsilon}}'(\boldsymbol{\phi}) + \mathring{\mathrm{M}}'(\boldsymbol{\psi}) : \mathring{\boldsymbol{\kappa}}'(\boldsymbol{\phi})\,\mathrm{d}\Omega - \int_\Omega \boldsymbol{f}'(\boldsymbol{\psi})\cdot\boldsymbol{\phi}\,\mathrm{d}\Omega, \tag{5.11}$$

where the $\mathring{\cdot}$ denotes tensors and functions on the undeformed geometry, i.e. with $\boldsymbol{u} = 0$.

In our implementation, the tangent stiffness matrix is computed using appropriate Gauss-Lengendre quadrature for each element in the hierarchical mesh. It should be noted that more efficient numerical integration approaches exist [204, 424, 425] for hierarchical splines that might further reduce the computational cost.

5.2.4 Structural Analysis

In the present chapter, different goal functionals will be provided for different structural analysis applications. Therefore, the different structural analysis types are briefly recalled. Firstly, in case of *static analysis*, the problem as in equation (5.8) is solved. In case of *quasi-static analysis*, load and/or displacement steps are performed successively and in each step, a static solve is performed. Typically, one writes equation (5.8) for load-control as

$$\begin{aligned} &\text{Find } \boldsymbol{u} \in \mathbb{S} \text{ s.t.} \mathscr{W}(\boldsymbol{u},\boldsymbol{\phi},\lambda) := \delta W^{\text{int}} - \lambda \delta W^{\text{ext}} \\ &\qquad = \int_{\Omega} \mathrm{N}(\boldsymbol{u}) : \boldsymbol{\varepsilon}'(\boldsymbol{u},\boldsymbol{\phi}) + \mathrm{M}(\boldsymbol{u}) : \boldsymbol{\kappa}'(\boldsymbol{u},\boldsymbol{\phi})\,\mathrm{d}\Omega - \int_{\Omega} \lambda \boldsymbol{f}_0 \cdot \boldsymbol{\phi}\,\mathrm{d}\Omega, \\ &\quad \forall \boldsymbol{\phi} \in \mathbb{S}, \lambda \in \mathbb{R}, \end{aligned} \tag{5.12}$$

where λ is the load factor scaling the reference load $\boldsymbol{f}_0$. Quasi-static simulations can be solved using simple load or displacement controlled schemes, using arc-length continuation such as Riks' method or Crisfield's method [124, 469]. When quasi-static analysis is performed for post-buckling analysis, one or multiple bifurcation points are passed by definition. On a bifurcation point, the determinant of the tangential stiffness matrix K is equal to zero, hence this matrix is singular. To cope with instabilities, *a priori* perturbations can be applied to the geometry, or a procedure for approximating singular points [633] can be used. In our previous work, more details are provided on arc-length continuation for post-buckling analysis without the use of *a priori* perturbations [583].

In the case of *modal analysis* and *buckling analysis*, a generalised eigenvalue problem needs to be solved. These eigenvalue problems have the general form

$$\text{find}\,(\mu,\boldsymbol{v}) \in \mathbb{R} \times \mathbb{S}\,\text{s.t.}\,\mathscr{A}(\boldsymbol{v},\boldsymbol{\phi}) = \mu\mathscr{B}(\boldsymbol{v},\boldsymbol{\phi})\,\forall\boldsymbol{\phi} \in \mathbb{S} \tag{5.13}$$

Where μ provides the eigenfrequency in modal analysis and the critical load factor in buckling analysis and where $\boldsymbol{v}$ denotes the vibration or buckling mode shape. The operators $\mathscr{A}$ and $\mathscr{B}$ are bi-linear. For buckling analysis, $\mathscr{A}(\boldsymbol{v},\boldsymbol{\phi}) = \mathscr{W}'(\boldsymbol{u}_L,\boldsymbol{v},\boldsymbol{\phi})$ and $\mathscr{B}(\boldsymbol{v},\boldsymbol{\phi}) = \mathscr{W}'(0,\boldsymbol{v},\boldsymbol{\phi})$ with $\boldsymbol{u}_L$ the pre-buckling solution given load λ_L. For modal analysis, $\mathscr{A}(\boldsymbol{v},\boldsymbol{\phi}) = \mathscr{W}'(0,\boldsymbol{v},\boldsymbol{\phi})$ and $\mathscr{B}(\boldsymbol{v},\boldsymbol{\phi}) = \mathscr{M}(\boldsymbol{v},\boldsymbol{\phi})$ with $\mathscr{M}$ the mass operator:

$$\mathscr{M}(\boldsymbol{v},\boldsymbol{\phi}) = \int_{\Omega} \rho\boldsymbol{v}\boldsymbol{\phi}\,\mathrm{d}\Omega \tag{5.14}$$

Where ρ is the mass density over the surface.

5.3 Dual-Weighted Residual method

This section elaborates on the Dual-Weighted Residual (DWR) method [29, 39], which is used in the *Estimate* step of figure 5.1. The DWR method is a method to compute the *a posteriori* error of a solution in terms of a given goal functional of interest by solving a linear dual problem. The DWR method provides a global estimate of the error, but given a partition of unity of the spline space, it can be used to provide an error contribution per basis function.

5.3.1 General Framework

The general framework of the dual weighted residual (DWR) method is presented by [29, 39, 229]. For the sake of completeness, a brief overview of the DWR method is provided here. Consider the following non-linear problem to solve:

$$\text{find } \boldsymbol{u} \in \mathbb{S} \text{ s.t. } \mathscr{W}(\boldsymbol{u}, \boldsymbol{\phi}) = 0 \, \forall \boldsymbol{\phi} \in \mathbb{S}, \tag{5.15}$$

where $\mathscr{W}(\boldsymbol{u}, \boldsymbol{\phi})$ is a semi-linear operator (linear in the second argument), $\boldsymbol{u}$ is the solution, $\boldsymbol{\phi}$ is a test function, and $\mathbb{S}$ is a suitably chosen vector space, including $\boldsymbol{u} \in \mathbb{S}$. The approximation of $\boldsymbol{u}$, denoted by $\boldsymbol{u}_h$, can be found by solving the discrete counterpart of equation (5.15):

$$\text{find } \boldsymbol{u}_h \in \mathbb{S}_h^p \text{ s.t. } \mathscr{W}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = 0 \, \forall \boldsymbol{\phi}_h \in \mathbb{S}_h^p, \tag{5.16}$$

where $\boldsymbol{u}_h$ and $\boldsymbol{\phi}_h$ are the discrete counterparts of $\boldsymbol{u}$ and $\boldsymbol{\phi}$, respectively, and the space $\mathbb{S}_h^p \subset \mathbb{S}$ is a function space on the (isogeometric) mesh $\mathscr{T}_h^p(\Omega)$ with mesh size h and order p covering the computational domain Ω. The solution to this problem is typically obtained by iteratively solving

$$\text{find } \boldsymbol{\phi}_h \in \mathbb{S}_h^p \text{ s.t. } \mathscr{W}'(\boldsymbol{u}_h, \boldsymbol{\phi}_h, \boldsymbol{\psi}_h) = \mathscr{R}(\boldsymbol{u}_h, \boldsymbol{\psi}_h) \, \forall \boldsymbol{\psi}_h \in \mathbb{S}_h^p, \tag{5.17}$$

while updating the discrete solution. Here, the residual is defined as

$$\boldsymbol{\mathscr{R}}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = -\boldsymbol{\mathscr{W}}(\boldsymbol{u}_h, \boldsymbol{\phi}_h). \tag{5.18}$$

Let us now define a non-linear and differentiable goal functional $\boldsymbol{\mathscr{L}}(\boldsymbol{u})$ or quantity of interest, such that

$$\Delta\boldsymbol{\mathscr{L}}(\boldsymbol{u}_h) = \boldsymbol{\mathscr{L}}(\boldsymbol{u}) - \boldsymbol{\mathscr{L}}(\boldsymbol{u}_h). \tag{5.19}$$

Then, from Proposition 4.1 of [231], it follows that:

$$\Delta\boldsymbol{\mathscr{L}}(\boldsymbol{u}_h) = \boldsymbol{\mathscr{R}}(\boldsymbol{u}_h, \xi - \xi_h) \approx \boldsymbol{\mathscr{R}}(\boldsymbol{u}_h, \tilde{\xi} - \xi_h). \tag{5.20}$$

Here, the solutions $\xi \in \boldsymbol{\mathscr{S}}$ and $\xi_h \in \boldsymbol{\mathscr{S}}_h^p$ are the exact and discrete solutions to the adjoint problem defined using the mean value linearizations of $\boldsymbol{\mathscr{W}}$ and $\boldsymbol{\mathscr{L}}$, see Equations 10 to 12 of [231]. Since the exact dual solution ξ is not available, it is approximated by $\tilde{\xi} \in \tilde{\boldsymbol{\mathscr{S}}}$. The discrete dual solution $\xi_h \in \boldsymbol{\mathscr{S}}_h^p$ is obtained by solving the following discrete adjoint problem:

$$\text{find } \xi_h \in \boldsymbol{\mathscr{S}}_h^p \text{ s.t. } \boldsymbol{\mathscr{W}}'(\boldsymbol{u}_h, \zeta_h, \xi_h) = \boldsymbol{\mathscr{L}}'(\boldsymbol{u}_h, \zeta_h) \, \forall \zeta_h \in \boldsymbol{\mathscr{S}}_h^p. \tag{5.21}$$

The approximation $\tilde{\xi} \in \tilde{\mathcal{S}}$ is obtained by solving the adjoint problem in an enriched space, i.e.

$$\text{find}\; \tilde{\xi}_h \in \tilde{\mathcal{S}}^p_h \;\text{s.t.}\; \boldsymbol{\mathscr{W}}'(\boldsymbol{u}_h, \tilde{\zeta}_h, \tilde{\xi}_h) = \boldsymbol{\mathscr{L}}'(\boldsymbol{u}_h, \tilde{\zeta}_h) \,\forall \tilde{\zeta}_h \in \tilde{\mathcal{S}}^p_h, \tag{5.22}$$

with $\tilde{\xi}_h$ and $\tilde{\zeta}_h$ the dual solution and test functions on the enriched space $\tilde{\mathcal{S}}^p_h$, respectively. A choice for $\tilde{\mathcal{S}}^p_h$ is to use the same mesh as for $\mathcal{S}^p_h$, with the same regularity but with a higher degree, i.e. $\tilde{\mathcal{S}}^p_h = \mathcal{S}^{p+1}_h$. This is easily achieved using spline bases. When using B-Splines, one can repeat all knots of the knot vector an extra time compared to the original basis, such that $\mathcal{S}^p_h \subset \mathcal{S}^{p+1}_h \subset \mathcal{S}$ is a nested space.

Finally, using equation (5.20) together with the dual solution $\xi_h \in \mathbb{S}^p_h$ and the enriched dual solution $\tilde{\xi}_h \in \tilde{\mathbb{S}}^p_h$, an estimate for the global error with respect to the goal functional $\mathscr{L}$ can be obtained. To obtain the local element-wise error estimations r_i for element $\omega_i \in \mathscr{T}^p_h(\Omega)$, such that

$$\Delta\mathscr{L}(\boldsymbol{u}_h) = \mathscr{R}(\boldsymbol{u}_h, \tilde{\xi}_h - \xi_h) = \sum_{\omega_i \in \mathscr{T}^p_h(\Omega)} r_i. \tag{5.23}$$

Element-wise integration of equation (5.20) is simply performed to obtain r_i. However, as discussed in section 5.4.3 it can be beneficial to have strictly positive element error contributions for element labelling. One can either take the absolute values of r_i or integrate the squared norm of the integrand in equation (5.20) to ensure the positivity of element error contributions. Obviously, the sum of the element errors would not be equal to $\Delta\mathscr{L}$.

For Kirchhoff–Love shells specifically, the operator $\mathscr{W}(\boldsymbol{u}, \boldsymbol{\phi})$ and its linearisation $\mathscr{W}'(\boldsymbol{u}, \boldsymbol{\phi}, \boldsymbol{\psi})$ are used to perform the DWR analysis.

5.3.2 Eigenvalue Problems

When the problem of interest is an eigenvalue problem, the DWR routine is slightly different. Here, the works [29, 39, 102, 200, 201, 330] are followed to give a brief overview of the DWR method for eigenvalue problems. Let us consider the following eigenvalue problem:

$$\text{find}\,(\mu, \boldsymbol{V}) \in \mathbb{R} \times \mathbb{S} \;\text{s.t.}\; \mathscr{A}(\boldsymbol{V}, \boldsymbol{\phi}) = \mu\mathscr{B}(\boldsymbol{V}, \boldsymbol{\phi}) \,\forall \boldsymbol{\phi} \in \mathbb{S}. \tag{5.24}$$

Here, $\mathscr{A}$ and $\mathscr{B}$ are bi-linear operators. For uniqueness of the problem, the discrete eigenvectors $\boldsymbol{v}_h$ are normalised by the condition [330]

$$\boldsymbol{\mathscr{B}}(\boldsymbol{v}, \boldsymbol{v}) = 1. \tag{5.25}$$

Typically, discretizing the system gives the following:

$$\text{find}\,(\mu_h, \boldsymbol{v}_h) \in \mathbb{R} \times \mathcal{S}^p_h \;\text{s.t.}\; \boldsymbol{\mathscr{A}}(\boldsymbol{v}_h, \boldsymbol{\phi}_h) = \mu\boldsymbol{\mathscr{B}}(\boldsymbol{v}_h, \boldsymbol{\phi}_h) \,\forall \boldsymbol{\phi}_h \in \mathcal{S}^p_h, \tag{5.26}$$

where the eigenpairs $\hat{\boldsymbol{v}}_h = (\mu_h, \boldsymbol{v}_h)$ are the solutions of the eigenvalue problem. In addition, the adjoint eigenvalue problem is defined by the eigenvalue problem [29]:

$$\text{find}\,(\eta, \boldsymbol{\psi}) \in \mathbb{R} \times \mathcal{S} \;\text{s.t.}\; \boldsymbol{\mathscr{A}}(\boldsymbol{\psi}, \boldsymbol{\phi}) = \eta\boldsymbol{\mathscr{B}}(\boldsymbol{\psi}, \boldsymbol{\phi}) \,\forall (\boldsymbol{\phi}) \in \mathcal{S}, \tag{5.27}$$

for which the normalization similar to equation (5.25) is used for the dual eigenvectors $\boldsymbol{\psi}$

$$\boldsymbol{\mathcal{B}}(\boldsymbol{v},\boldsymbol{\psi}) = 1. \tag{5.28}$$

To derive the DWR method for the eigenvalue problem in equation (5.24), the functional $\mathcal{V}(\cdot,\cdot)$ is defined, such that the following problem should be solved:

$$\begin{aligned}
&\text{Find } \hat{\boldsymbol{V}} = (\mu, \boldsymbol{V}) \in \mathbb{R}\times\mathbb{S} \text{ s.t.}\\
&\quad \mathcal{V}(\hat{\boldsymbol{V}},\hat{\boldsymbol{\phi}}) = \mu\mathcal{B}(\boldsymbol{V},\boldsymbol{\phi}) - \mathcal{A}(\boldsymbol{V},\boldsymbol{\phi}) + \tau(\mathcal{B}(\boldsymbol{V},\boldsymbol{V})-1) = 0,\\
&\forall\hat{\boldsymbol{\phi}} = (\tau,\boldsymbol{\phi}) \in \mathbb{R}\times\mathbb{S},
\end{aligned} \tag{5.29}$$

where the normalisation condition from equation (5.25) is enforced weakly. The discrete counterpart of this equation reads:

$$\begin{aligned}
&\text{Find } \hat{\boldsymbol{V}}_h = (\mu_h, \boldsymbol{V}_h) \in \mathbb{R}\times\mathbb{S}_h^p \text{ s.t.}\\
&\quad \mathcal{V}(\hat{\boldsymbol{V}}_h,\hat{\boldsymbol{\phi}}_h) = \mu\mathcal{B}(\boldsymbol{V}_h,\boldsymbol{\phi}_h) - \mathcal{A}(\boldsymbol{V}_h,\boldsymbol{\phi}_h) + \tau_h\big(\mathcal{B}(\boldsymbol{V}_h,\boldsymbol{V}_h)-1\big) = 0,\\
&\forall\hat{\boldsymbol{\phi}}_h = (\tau_h,\boldsymbol{\phi}_h) \in \mathbb{R}\times\mathbb{S}_h^p.
\end{aligned} \tag{5.30}$$

Furthermore, a goal function for the eigenvalues is defined as follows:

$$\mathcal{L}(\hat{\boldsymbol{V}}) = \mu = \mu\mathcal{B}(\boldsymbol{V},\boldsymbol{V}), \tag{5.31}$$

giving

$$\Delta\mathcal{L}(\hat{\boldsymbol{\phi}}_h) = \mu - \mu_h. \tag{5.32}$$

Using the non-linear functional $\mathcal{V}$ and the goal functional $\mathcal{L}$, the same derivations as in section 5.3.1 can be followed to find a system of equations to solve the DWR eigenvalue problem. The Gateaux derivative of $\mathcal{V}$, denoted by $\mathcal{V}'$, is given by:

$$\mathcal{V}'(\hat{\boldsymbol{V}},\hat{\boldsymbol{\phi}},\hat{\boldsymbol{\psi}}) = \eta\mathcal{B}(\boldsymbol{V},\boldsymbol{\psi}) + \mu\mathcal{B}(\boldsymbol{\psi},\boldsymbol{\phi}) - \mathcal{A}(\boldsymbol{\psi},\boldsymbol{\phi}) + \tau(\mathcal{B}(\boldsymbol{v},\boldsymbol{\psi}) + \mathcal{B}(\boldsymbol{\psi},\boldsymbol{v})), \tag{5.33}$$

where the derivatives $\mathcal{A}'(\boldsymbol{\psi},\boldsymbol{\phi})$ and $\mathcal{B}'(\boldsymbol{\psi},\boldsymbol{\phi})$ are equal to the bi-linear operators $\mathcal{A}(\boldsymbol{u},\boldsymbol{\phi})$ and $\mathcal{B}(\boldsymbol{u},\boldsymbol{\phi})$ themselves. Furthermore, the solution around which the linerisation is performed is denoted by $\hat{\boldsymbol{V}} = (\mu,\boldsymbol{V})$, the test functions are denoted by $\hat{\boldsymbol{\phi}} = (\tau,\boldsymbol{\phi})$, and the trial functions are denoted by $\hat{\boldsymbol{\psi}} = (\eta,\boldsymbol{\psi})$. Furthermore, the linearisation of the goal functional equation (5.31) is

$$\mathcal{L}'(\hat{\boldsymbol{V}},\hat{\boldsymbol{\psi}}) = \eta\mathcal{B}(\boldsymbol{V},\boldsymbol{V}) + \mu[\mathcal{B}(\boldsymbol{V},\boldsymbol{\psi}) + \mathcal{B}(\boldsymbol{\psi},\boldsymbol{V})], \tag{5.34}$$

such that the adjoint eigenvalue problem, analoguously to equation (5.21), given by

$$\text{Find } \hat{\boldsymbol{\phi}} = (\tau,\boldsymbol{\phi}) \in \mathbb{R}\times\boldsymbol{\mathcal{S}} \text{ s.t. } \boldsymbol{\mathcal{V}}'(\hat{\boldsymbol{v}},\hat{\boldsymbol{\phi}},\hat{\boldsymbol{\psi}}) = \boldsymbol{\mathcal{L}}'(\hat{\boldsymbol{v}},\hat{\boldsymbol{\psi}})\forall\hat{\boldsymbol{\psi}} = (\eta,\boldsymbol{\psi}) \in \mathbb{R}\times\boldsymbol{\mathcal{S}}, \tag{5.35}$$

becomes [29, 330]:

$$\begin{aligned}
&\text{Find } \hat{\boldsymbol{\phi}} = (\tau,\boldsymbol{\phi}) \in \mathbb{R}\times\boldsymbol{\mathcal{S}} \text{ s.t.}\\
&\quad \eta\boldsymbol{\mathcal{B}}(\boldsymbol{v},\boldsymbol{\phi}) + \mu\boldsymbol{\mathcal{B}}(\boldsymbol{\psi},\boldsymbol{\phi}) - \boldsymbol{\mathcal{A}}(\boldsymbol{\psi},\boldsymbol{\phi}) + \tau(\boldsymbol{\mathcal{B}}(\boldsymbol{v},\boldsymbol{\psi}) + \boldsymbol{\mathcal{B}}(\boldsymbol{\psi},\boldsymbol{v})) = \quad \eta\boldsymbol{\mathcal{B}}(\boldsymbol{v},\boldsymbol{v}) +\\
&\mu[\boldsymbol{\mathcal{B}}(\boldsymbol{v},\boldsymbol{\psi}) + \boldsymbol{\mathcal{B}}(\boldsymbol{\psi},\boldsymbol{v})],\\
&\forall\hat{\boldsymbol{\psi}} = (\eta,\boldsymbol{\psi}) \in \mathbb{R}\times\boldsymbol{\mathcal{S}}.
\end{aligned} \tag{5.36}$$

This equation can be simplified to obtain the following [29, 330]:

$$
\begin{aligned}
&\text{Find } \hat{\boldsymbol{\phi}} = (\tau, \boldsymbol{\phi}) \in \mathbb{R} \times \boldsymbol{\mathscr{S}} \text{ s.t.} \\
&\quad \mu \boldsymbol{\mathscr{B}}(\boldsymbol{\psi}, \boldsymbol{\phi}) - \boldsymbol{\mathscr{A}}(\boldsymbol{\psi}, \boldsymbol{\phi}) + \eta[\boldsymbol{\mathscr{B}}(\boldsymbol{v}, \boldsymbol{\phi}) - \boldsymbol{\mathscr{B}}(\boldsymbol{v}, \boldsymbol{v})] + (\tau - \mu)[\boldsymbol{\mathscr{B}}(\boldsymbol{v}, \boldsymbol{\psi}) + \boldsymbol{\mathscr{B}}(\boldsymbol{\psi}, \boldsymbol{v})] = 0, \\
&\forall \hat{\boldsymbol{\psi}} = (\eta, \boldsymbol{\psi}) \in \mathbb{R} \times \boldsymbol{\mathscr{S}}.
\end{aligned}
\tag{5.37}
$$

Using the normalizations from equations (5.25) and (5.28) and the fact that equation (5.27) solves the same equation as equation (5.24), it follows that equation (5.37) is solved by equation (5.27) [29].

Using equations (5.15), (5.20) and (5.32) with $\boldsymbol{\mathscr{W}} = \boldsymbol{\mathscr{V}}$ according to equation (5.29) and with $\boldsymbol{\psi}$ denoting the dual eigenvector and η the dual eigenvalue, the error estimation according to the DWR method for an eigenvalue problem is

$$
\Delta \boldsymbol{\mathscr{L}}(\hat{\boldsymbol{v}}_h) = \boldsymbol{\mathscr{A}}(\boldsymbol{v}_h, \boldsymbol{\psi} - \boldsymbol{\psi}_h) - \mu_h \boldsymbol{\mathscr{B}}(\boldsymbol{v}_h, \boldsymbol{\psi} - \boldsymbol{\psi}_h) + (\eta - \eta_h)(\boldsymbol{\mathscr{B}}(\boldsymbol{v}_h, \boldsymbol{v}_h) - 1), \tag{5.38}
$$

for $\hat{\boldsymbol{V}}_h = (\mu_h, \boldsymbol{V}_h) \in \mathbb{R} \times \mathbb{S}_h^p$, $\hat{\boldsymbol{\psi}}_h = (\eta_h, \boldsymbol{\psi}_h) \in \mathbb{R} \times \mathbb{S}_h^p$ and $\hat{\boldsymbol{\psi}} = (\eta, \boldsymbol{\psi}) \in \mathbb{R} \times \mathbb{S}$. The exact adjoint solution $\hat{\boldsymbol{\psi}}_h$ is again approximated by solving equation (5.27) on an *enriched* space $\tilde{\mathbb{S}}_h^p \subset \mathbb{S}$, $\tilde{\mathbb{S}}_h^p \supset \mathbb{S}_h^p$, providing $(\tilde{\eta}_h, \tilde{\boldsymbol{\psi}}_h) \in \mathbb{R} \times \tilde{\mathbb{S}}_h^p$. In [230], different choices for constructing $\tilde{\mathbb{S}}_h^p$ are given, including an h-refinement and a p-refinement. As in the work of [244], the second approach is used in the present chapter, with the same mesh as for $\mathbb{S}_h^p$, but with a higher order and with the same regularity, i.e., $\tilde{\mathbb{S}}_h^p = \mathbb{S}_h^{p+1}$ as it introduces less degrees of freedom compared to an h-refinement.

As specified at the end of section 5.2.4, the DWR method for modal analysis requires $\mathscr{A}(\boldsymbol{V}, \boldsymbol{\phi}) = \mathscr{W}'(0, \boldsymbol{V}, \boldsymbol{\phi})$ and $\mathscr{B}(\boldsymbol{V}, \boldsymbol{\phi}) = \mathscr{M}(\boldsymbol{V}, \boldsymbol{\phi})$. For buckling analysis, $\mathscr{A}(\boldsymbol{V}, \boldsymbol{\phi}) = \mathscr{W}'(\boldsymbol{u}_L, \boldsymbol{V}, \boldsymbol{\phi})$ and $\mathscr{B}(\boldsymbol{V}, \boldsymbol{\phi}) = \mathscr{W}'(0, \boldsymbol{V}, \boldsymbol{\phi})$ with the first operator defined as a pre-buckling solution $\boldsymbol{u}_L$.

5.3.3 Goal Functionals for Isogeometric Kirchhoff–Love Shells

The remainder of this section focuses on defining the goal functional $\mathscr{L}(\boldsymbol{u})$, see equation (5.19), together with its variation $\mathscr{L}'(\boldsymbol{u})$ such that the dual problem (equation (5.21)) can be solved and the error estimate (equation (5.20)) can be computed.

In general, the goal functional can be defined at a point, on a boundary, or over the domain:

$$
\mathscr{L}(\cdot) = \int_{\Omega} l(\cdot, \boldsymbol{x})\,\mathrm{d}\Omega, \qquad \text{Domain-wise,} \tag{5.39}
$$

$$
\mathscr{L}(\cdot) = \int_{\partial\Omega} l(\cdot, \boldsymbol{x})\,\mathrm{d}\Gamma, \qquad \text{Boundary-wise,} \tag{5.40}
$$

$$
\mathscr{L}(\cdot) = \sum_{i \in \mathscr{I}} l(\cdot, \boldsymbol{x}_i), \qquad \text{Point-wise,} \tag{5.41}
$$

where Ω denotes the integration domain, $\partial\Omega$ a side of Ω, and $\mathscr{I}$ a set of indices corresponding to points $\boldsymbol{x}_i \in \Omega$. Furthermore, $l(\cdot, \boldsymbol{x}_i)$ denotes a goal functional summant or integrand,

Table 5.1: Overview of the goal functionals. Here $\boldsymbol{u}_h = \boldsymbol{u}(\boldsymbol{x})$ is the discrete deformation tensor depending on position coordinate $\boldsymbol{x}$, $\mathrm{C}_h = \mathrm{C}(\boldsymbol{u}_h)$ is the deformation tensor based on $\boldsymbol{u}_h$, and $\mathrm{C}(\boldsymbol{\phi}_h') = \mathrm{C}'(\boldsymbol{u}_h, \boldsymbol{\phi})$ is its variation, $\mathscr{T}(\mathrm{A})$ is the transformation of a second-order tensor $\mathbf{A}$ from the undeformed contravariant basis to the basis spanned by the principal directions. Note that the variation $[\mathscr{T}(\mathrm{A})]'$ of $\mathscr{T}(\mathrm{A})$ is $\mathscr{T}(\mathrm{A}')$, since the spectral decomposition of the deformation tensor itself is just a linear change of tensor basis.

Displacement norm	$l_{\lVert\boldsymbol{u}\rVert}(\boldsymbol{u}_h) = \lVert\boldsymbol{u}_h(\boldsymbol{x})\rVert^2$	$l'_{\lVert\boldsymbol{u}\rVert}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = 2\boldsymbol{u}_h \cdot \boldsymbol{\phi}_h$
Displacement-component	$l_{\boldsymbol{u}_i}(\boldsymbol{u}_h) = \boldsymbol{u}_h(\boldsymbol{x}) \cdot \boldsymbol{e}_i$	$l'_{\lVert\boldsymbol{u}\rVert}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = \boldsymbol{e}_i \cdot \boldsymbol{\phi}_h$
Stretch-norm	$l_{\lVert\boldsymbol{\lambda}\rVert}(\boldsymbol{u}_h) = \lVert\mathscr{T}(\mathrm{C}_h)\rVert^2$	$l'_{\lVert\boldsymbol{\lambda}\rVert}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = 2\mathscr{T}(\mathrm{C}_h) \cdot \boldsymbol{\phi}_h$
Stretch-component	$l_{\boldsymbol{\lambda}_i}(\boldsymbol{u}_h) = \mathscr{T}(\mathrm{C}_h) \cdot \boldsymbol{e}_i$	$l'_{\lVert\boldsymbol{\lambda}\rVert}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = \boldsymbol{e}_i \cdot \mathscr{T}(\mathrm{C}_h')$
Tensor-norm	$l_{\lVert\mathrm{A}\rVert}(\boldsymbol{u}_h) = \lVert\mathrm{A}\rVert^2$	$l'_{\lVert\mathrm{A}\rVert}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = 2\mathrm{A} \cdot \boldsymbol{\phi}_h$
Tensor-component	$l_{\mathrm{A}_i}(\boldsymbol{u}_h) = \mathrm{A} \cdot \boldsymbol{e}_i$	$l'_{\lVert\mathrm{A}\rVert}(\boldsymbol{u}_h, \boldsymbol{\phi}_h) = \boldsymbol{e}_i \cdot \mathrm{A}$

which has a variation denoted by $l'(\cdot, \boldsymbol{\phi}\boldsymbol{x}_i)$. The variation of $\mathscr{L}$, denoted by $\mathscr{L}'(\cdot, \boldsymbol{\phi}, \boldsymbol{x}_i)$, directly follows from $l'(\cdot, \boldsymbol{\phi}\boldsymbol{x}_i)$ due to the linearity of integrals and summation. In addition, two different types of goal functional integrands are classified, resulting in norm-based and component-based goal functionals. In the former case, l is of the form $l = \lVert\mathrm{A}\rVert^2$ with variation $l' = 2\mathrm{A} \cdot \mathrm{A}'$. For component-based goal functionals, $l = \mathrm{A} \cdot \boldsymbol{e}_i$ is defined, with variation $l' = \mathrm{A}' \cdot \boldsymbol{e}_i$. Here, $\boldsymbol{e}_i$ is a unit vector in direction i. In table 5.1 some goal functional integrands or summants $l(\cdot, \boldsymbol{x}_i)$ are provided. Together with their variations $l'(\cdot, \boldsymbol{\phi}\boldsymbol{x}_i)$, these provide $\mathscr{L}'(\cdot, \boldsymbol{\phi}, \boldsymbol{x}_i)$ due to the linearity of integrals and summation. The *tensor-based* goal functionals refer to goal functionals that could be used for any second-order tensor, e.g., the membrane strain tensor $\boldsymbol{\varepsilon}(\boldsymbol{u})$ or the flexural moment tensor $\mathbf{M}(\boldsymbol{u})$.

5.4 Coarsening and Refinement using THB Splines

This section elaborates on the coarsening and refinement of isogeometric meshes using THB-splines. In particular, this section elaborates on the *Mark*, *Refine*, and *Transfer* blocks of figure 5.1. Firstly, section 5.4.1 will provide a brief background on THB-splines, which enable the *Refine* step of the adaptive meshing flowchart. Then, section 5.4.2 elaborates on methods for suitable grading for refinement meshes, which is required to provide admissible refinement with (Truncated) Hierarchical B-spline ((T)HB) bases, given labelled elements. Section 5.4.3 elaborates on the labelling method for the *Mark* step, given an element-wise error distribution, taking admissibility into account. Lastly, section 5.4.4 elaborates on the quasi-interpolation method that is used to *Transfer* the solution of one solution step to the next. The notations in this section will be closely related to those used in [67, 84].

5.4.1 (T)HB-Splines

Refinement of B-spline meshes can be done using (Truncated) Hierarchical B-splines ((T)HB-splines), of which the details can be found in [203, 591]. The conceptual idea behind (T)HB-splines is that they are constructed from a sequence of N nested tensor B-spline spaces in different levels $l = 0, ..., N-1$, denoted by $V^0 \subset V^1 \subset, ..., V^{N-1}$ with associated bases $\mathbb{B}^\ell$ of de-

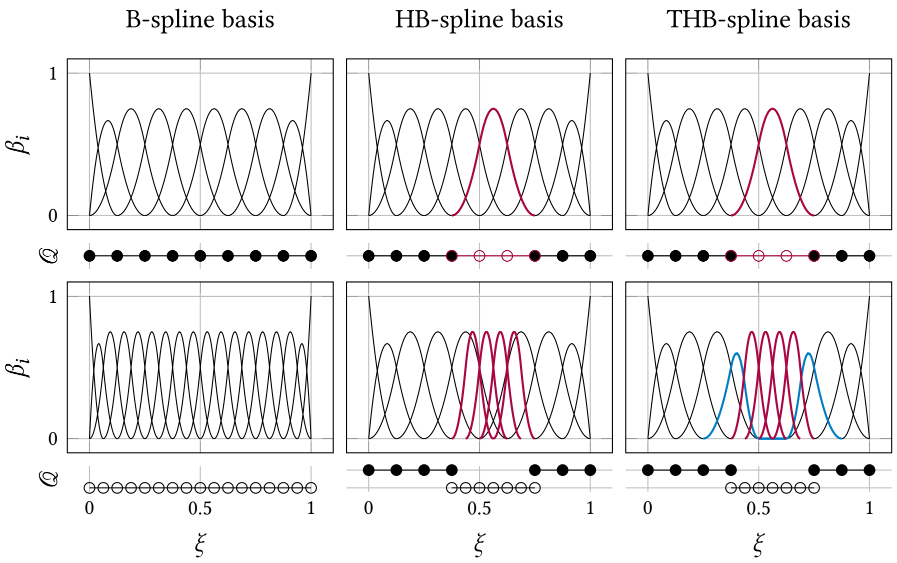


Figure 5.2: Principles of refinement for different spline bases. The top plots represent the basis on level 0, optionally with (to be) refined functions given in red color. The bottom plots illustrate the refined bases: uniform refinement (left), hence level 1; HB-refinement (middle); and THB-refinement (right) with truncated basis functions in blue color. The line $\mathcal{Q}$ represents the elements of the basis. The unrefined unique knot vector in all cases is $\Xi = \{0, 1/8, 2/8, \dots, 7/8, 1\}$, and the degree of the basis is 2. All bases are generated with the open-source IGA library G+Smo [294].

gree p on a grid G^ℓ with elements Q. The parametric domains are defined as $\Omega = \Omega^0 \supseteq \Omega^1 \supseteq \dots \supseteq \Omega^{N-1} = \emptyset$. By defining the set of active cells by $\mathcal{G}^\ell := \{Q \in G^\ell : Q \subset \Omega^\ell \wedge Q \not\subset \Omega^{\ell+1}\}$, the hierarchical mesh is defined as $\mathcal{Q} = \{Q \in \mathcal{G}^\ell : \ell = 0, \dots, N-1\}$. In figure 5.2, an illustration is given for a refined B-spline basis (left), a refined HB-spline basis (middle), and a refined THB-spline basis (right). For the (T)HB-spline basis, this picture depicts the refinement of a single basis function, corresponding to the elements in its support. The (T)HB-spline bases show that for THB-splines, a truncation is performed to ensure partition of unity, which is discussed in more detail in [203].

5.4.2 Admissible Meshing

The concept of admissible meshing was discussed in [67, 78, 79]. An admissible mesh of class m is a mesh in which the truncated basis functions belong to at most m successive levels, and mesh admissibility ensures that the number of basis functions acting on mesh elements does not depend on the number of levels in the hierarchy but on the parameter m. In order to guarantee mesh admissibility for refinement and coarsening operations, refinement and coarsening neighbourhoods are defined such that admissible meshes can be constructed recursively, which is discussed in more detail in [67, 78, 79]. Figure 5.3, illustrates a simple mesh together with the refinement neighbourhood of some selected elements. The T-refinement neighbourhood $\mathcal{N}_r(\mathcal{Q}, Q, m)$ of element Q is defined as

$$\mathcal{N}_r^{\text{THB}}(\mathcal{Q}, Q, m) = \left\{Q' \in \mathcal{G}^{\ell-m+1} : \exists Q'' \in S(Q, \ell-m+2), Q'' \subseteq Q'\right\}, \quad (5.42)$$

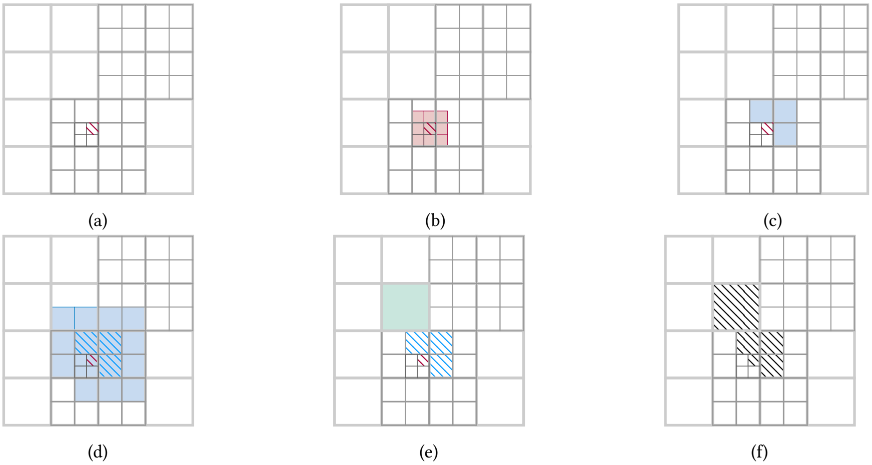


Figure 5.3: Recursive marking strategy for the marked element of level ℓ on the initial mesh represented in (a). As a first step, the support extension of the marked element is obtained (b), from which the parents that are active on level $\ell-1$ define the T-neighborhood of the marked cell (c). Starting the same procedure on the marked cells of level $\ell-1$, the support extension can again be obtained (d) with their corresponding parents on level $\ell-2$, marking the T-neighborhood of the marked elements of level $\ell-1$ (e). The complete recursive marking of the marked element in (a) is depicted in (f).

where $S(Q,k)$ is the *multi-level* support extension with respect to level k.

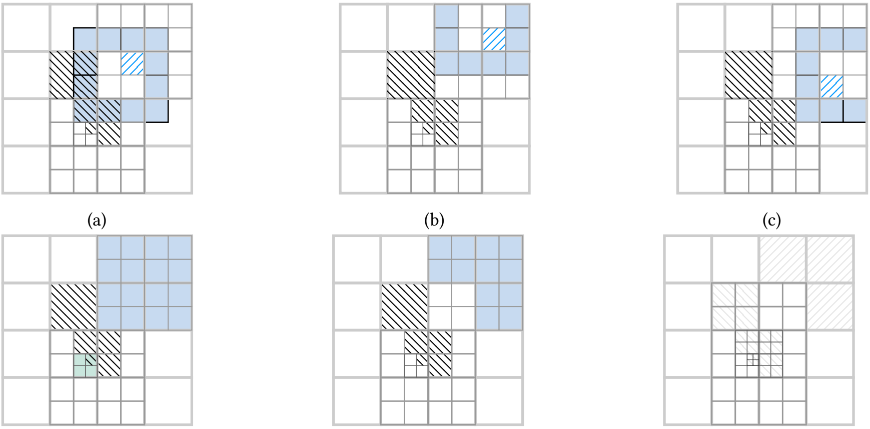
The coarsening neighbourhood $\mathcal{N}_c(Q)$ of element $Q \in G^\ell$ is defined by [84]. When coarsening element Q^ℓ, the coarsening neighbourhood ensures that the newly activated basis functions are not active on the surrounding basis functions of level $\ell+m$. In other words, if element Q of level ℓ is the element to be coarsened, then the coarsening neighbourhood is defined by

$$\mathcal{N}_c(\mathcal{Q},Q,m) := \left\{Q' \in \mathcal{G}^{\ell+m-1} \,:\, \exists Q'' \in \mathcal{G}^{\ell} \text{ and } Q'' \subset P(Q), \text{ with } Q' \in S(Q'',\ell)\right\}, \tag{5.43}$$

must be empty. Here, $P(Q)$ denotes the parent of Q, i.e., the unique cell $Q' \in G^{\ell-1}$ such that $Q \subset Q'$. Note the small difference with respect to the definition given in [84] since the coarsening neighbourhood in their work is defined for the element $\hat{Q}$ of level ℓ which will be activated, i.e., $\hat{Q}$ is the parent of Q for which the coarsening neighbourhood is defined here. Given the definition in equation (5.43) and given a set of elements marked for refinement $\mathcal{M}_r$, a coarsening neighbourhood checking elements marked for refinement, can be defined:

$$\mathcal{N}_c^r(\mathcal{Q},Q,m,\mathcal{M}_r) := \Big\{Q' \in \mathcal{G}^{\ell+m-2} \,:\, Q' \in \mathcal{M}_r, \exists Q'' \in \mathcal{G}^{\ell-1} \text{ and } Q'' \subset P(Q), \\ \text{with } Q' \in S(Q'',\ell-1)\Big\}. \tag{5.44}$$

In other words, this is the coarsening neighbourhood that checks whether for element Q of level ℓ to be coarsened, there are elements in the marked set $\mathcal{M}_r$ that will be part of the coarsening neighbourhood as soon as they are refined; thus, it uses equation (5.43)


(a) (b) (c)

(d) (e) (f)

Figure 5.4: Given the mesh from figure 5.3-(f), the coarsening neighbourhoods are given in (a)-(c) of this figure. The cell ▨ marks the cell of level ℓ that is marked for coarsening to its parent, and the cells ▧ mark cells that are marked for refinement. The ring around the cell marked for coarsening depicts the region that should be checked for the coarsening neighborhood. That is, it defines the region that should not contain cells of level $\ell+1$ (for $\mathscr{N}_c$) or cells of level ℓ that are marked for refinement (for $\mathscr{N}_c'$). The cells for which $\mathscr{N}_c = \emptyset$ are marked in (d), and the cells with $\mathscr{N}_c' = \emptyset$ are marked in (e). The final mesh after refinement and coarsening is depicted in (f). The coarsened elements that satisfy $\mathscr{N}_c \cup \mathscr{N}_c' = \emptyset$ are marked as coarsened elements.

with $\ell-1$. This neighbourhood ensures that coarse labelling can be performed in accordance with the Dörfler marking strategy without refining first. This avoids computing element-error contributions on an in-between mesh that has been refined first. Obviously, if another element with the same parent as Q is marked for refinement, no coarsening should take place. An element can be coarsened if $\mathscr{N}_c^r(\mathcal{Q}, Q, m, \mathscr{M}_r) = \emptyset$. Combining both neighbourhoods, an element Q of level ℓ can be coarsened if and only if $\hat{\mathscr{N}}_c(\mathcal{Q}, Q, m, \mathscr{M}_r) = \mathscr{N}_c(\mathcal{Q}, Q, m) \cup \mathscr{N}_c^r(\mathcal{Q}, Q, m, \mathscr{M}_r) = \emptyset$. In figure 5.4, the coarsening neighbourhood is illustrated for a simple mesh.

5.4.3 Labeling Methods

Let $\tilde{\mathcal{Q}}$ be the ordered set of $\mathcal{Q}$ such that $e_k \geq e_{k+1}\ \forall k \in \tilde{\mathcal{Q}}$, where e_k denotes the error of element k. Then, the Dörfler marking strategy [160] is defined as the elements $Q_i \in \tilde{\mathcal{Q}}$, $i = 0, \ldots, k$ such that the sum of their respective errors is smaller than a fraction ρ_r of the total element error $e = \sum_i e_i$:

$$\mathscr{M}_r^{\text{Dörfler}} = \left\{ Q_i \in \mathcal{Q} \ :\ \sum_{k=0}^{i} e_k < \rho_r e \right\}. \tag{5.45}$$

This marking strategy, however, does not take into account the contributions of the elements that are marked because they are part of a refinement neighbourhood of a marked element $Q_i \in \mathscr{M}_r$. Therefore, the index set $\mathscr{I}_r^K$ is defined as the set of element indices whose span contains elements $Q_k \in \tilde{\mathcal{Q}}$ and their refinement neighbourhoods:

$$\mathcal{I}_r^K := \left\{k \in 1,\dots,K \,:\, \mathcal{N}_r(Q_k,\mathcal{Q},m) \cup Q_k,\, Q_k \in \tilde{\mathcal{Q}}\right\}, \tag{5.46}$$

and define κ_r as the maximum index for which the sum of all elements with indices i in $\mathcal{I}_r^{\kappa_r}$ is smaller than the error tolerance $\rho_r e$:

$$\kappa_r := \arg\max \sum_{i \in \mathcal{I}_r^K} e_i < \rho_r e, \tag{5.47}$$

such that the Dörfler marking, including refinement neighbourhoods,

$$\mathcal{M}_r = \left\{Q_k \in \mathcal{Q} \,:\, k \in \mathcal{I}_r^{\kappa_r}\right\}. \tag{5.48}$$

For marking a set of coarsening elements, $\mathcal{M}_c$, the Dörfler marking procedure can be followed again. The original Dörfler marking strategy would be coarsening the elements $Q_i \in \mathcal{Q}$ such that their total element error is smaller than a fraction of the total element error $\rho_c e$, with coarsening parameter ρ_c:

$$\mathcal{M}_c^{\text{Dörfler}} = \left\{Q_i \in \mathcal{Q} \,:\, \sum_{k=i}^{N} e_k < \rho_c e\right\}. \tag{5.49}$$

Similar to marking for refinement, the marking rule for coarsening can be specified more precisely by including admissible coarsening. In this case, the elements for which $\hat{\mathcal{N}}_c(\mathcal{Q},Q,m,\mathcal{M}_r) = \emptyset$ holds are added to the sum of marked elements. Therefore, let us define the index set $\mathcal{J}_K$ that contains all elements $Q_k \in \mathcal{Q}$ for which the admissible coarsening condition holds, starting from the element with the smallest error, i.e., Q_N.

$$\mathcal{I}_c^K := \left\{k \in 1,\dots,K \,:\, \hat{\mathcal{N}}_c(\mathcal{Q},Q,m,\mathcal{M}_r) = \emptyset,\, Q_{N-k-1} \in \tilde{\mathcal{Q}}\right\}. \tag{5.50}$$

Similar to κ_r, κ_c is defined as the maximum index for which the sum of all elements with indices i in $\mathcal{I}_c^{\kappa_c}$ is smaller than the error tolerance $\rho_c e$:

$$\kappa_c := \arg\max \sum_{i \in \mathcal{I}_r^K} e_i < \rho_c e, \tag{5.51}$$

such that the Dörfler marking strategy, taking into account coarsening admissibility, is defined as

$$\mathcal{M}_c = \left\{Q_k \in \mathcal{Q} \,:\, k \in \mathcal{I}_c^{\kappa_c}\right\}. \tag{5.52}$$

An alternative to the Dörfler marking strategy is a strategy where a given fraction of the total number of elements is marked. In that case, the formulations from equations (5.48) and (5.52) would still hold, but in equations (5.46) and (5.50) the indices κ_r and κ_c are defined by the sum of the marked elements in respectively $\mathcal{I}_r^K$ and $\mathcal{I}_c^K$.

Whether to mark a set for refinement or coarsening, i.e., to construct $\mathcal{M}_r$ and $\mathcal{M}_c$, depends on the global error $\Delta\mathcal{L}$ following from the DWR method and user-defined tolerances for refinement and coarsening. Let tol_r be the tolerance for refinement and tol_c

the tolerance for coarsening, such that $\mathscr{M}_r \neq \emptyset$ if and only if $\Delta\mathscr{L} > \mathrm{tol}_r$ and $\mathscr{M}_c \neq \emptyset$ if and only if $\Delta\mathscr{L} < \mathrm{tol}_r$. As a consequence, if $\mathrm{tol}_r \geq \mathrm{tol}_c$, refinement and coarsening are never performed simultaneously. If $\mathrm{tol}_r < \mathrm{tol}_c$ a band with bandwidth $\mathrm{tol}_c - \mathrm{tol}_r$ is defined, in which refinement and coarsening are performed simultaneously, In the present work, tolerances are defined such that the latter condition is satisfied, and the adaptivity iterations are terminated when $\Delta\mathscr{L} \in [\mathrm{tol}_r, \mathrm{tol}_c]$, i.e.:

$$\begin{cases} \mathscr{M}_r = \emptyset,\ \mathscr{M}_c \neq \emptyset & \text{if}\, \Delta\mathscr{L} < \mathrm{tol}_r, \\ \mathscr{M}_r \neq \emptyset,\ \mathscr{M}_c = \emptyset & \text{if}\, \Delta\mathscr{L} > \mathrm{tol}_c, \\ \mathscr{M}_r \neq \emptyset,\ \mathscr{M}_c \neq \emptyset & \text{if}\, \mathrm{tol}_r \geq \Delta\mathscr{L} \geq \mathrm{tol}_c, \end{cases} \tag{5.53}$$

given $\mathrm{tol}_r \leq \mathrm{tol}_c$.

Note that the total element error e and the total estimated error of the system of equations $\Delta\mathscr{L}$ are not necessarily the same, since the element error measure e_k can be defined in different ways. In the case of the DWR method, a natural choice is to choose e_k as the element-wise integrals of $\Delta\mathscr{L}$ from equation (5.20). However, integrating the squared norm of the integrand from equation (5.20) would yield strictly positive element errors, making the ordering of the set of element errors simple.

5.4.4 Quasi-Interpolation

In the discrete setting, the solution of the problem $\boldsymbol{u}_h$ is represented by the THB-spline basis $\boldsymbol{\phi}_i \in \mathbb{S}_h^p$ together with the solution coefficients $\alpha_i \in \mathbb{R}$. In the case of analyses with multiple solution steps (e.g., dynamic or quasi-static analysis), mesh refinements can be performed after each solution step. As a consequence, the solution at load step $k+1$ is defined on another set of basis functions $\{\bar{\boldsymbol{\phi}}_i\} \in \mathbb{S}_h^p$ with corresponding coefficients $\bar{\alpha}_i^k$ compared to the previous solution at step k. In order to transfer the coefficients α_i^k to the new basis, an interpolation scheme needs to be used.

Interpolation on a spline basis can be a costly part of the simulation. Global interpolation implies that the contributions of all basis functions are taken into account in the interpolation. This requires solving a large, dense system. An efficient way of interpolating spline coefficients on a hierarchical basis is a so-called quasi-interpolation scheme [205, 519]. Here, on each level of the hierarchical basis, a quasi-interpolant is constructed. This quasi-interpolant interpolates a given function f over the support of each basis function individually to find the coefficient related to that basis function. More precisely, given a function $f \in C(\Omega^0)$, the quasi-interpolant for level ℓ is defined as

$$\Lambda^\ell(f) = \sum_{i=1}^{N} \lambda_{i,\ell}(f) B_{i,\ell}, \quad \ell = 0, \dots, n-1, \tag{5.54}$$

where the coefficients $\lambda_{i,\ell}$ are suitable linear functionals on $C(\Omega^0)$. Across all levels $\ell =$

$0,\dots,N-1$, the interpolant for the function becomes:

$$\Lambda(f) = \sum_{\ell=0}^{N-1} \sum_{i\in\mathcal{I}_{\ell,\Omega_n}} \lambda_{i,\ell}(f) B_{i,\ell}(\mathcal{T}(\Omega_n)), \tag{5.55}$$

where $B_{i,\ell}$ is a THB spline of level ℓ constructed on domain Ω_n. For any basis function $B_{i,\ell}$, the coefficient $\lambda_{i,\ell}$ is found by locally interpolating the function f onto all active basis functions $B_{j,\ell}, j \in \mathcal{J}$ in the support of $B_{i,\ell}$. This gives coefficients $\lambda_{j,\ell}, j \in \mathcal{J}$, of which coefficient i gives $\lambda_{i,\ell}$. This quasi-interpolation scheme is used in the present framework to express the solution obtained from the previous load-step in terms of the newly, adaptively refined and coarsened, basis.

5.5 Algorithmic Overview

In figure 5.1, the adaptive isogeometric method for solution-stepping problems has been presented. Based on sections 5.2 to 5.4, a summarised workflow for adaptive isogeometric shell analysis is depicted in figure 5.5 and algorithm 1.

The *Solve* block involves solving the non-linear isogeometric Kirchhoff–Love shell equation from equation (5.16). This variational formulation involves geometric and material non-linearities and can potentially also involve load non-linearities. After solving the Kirchhoff–Love shell problem, the discrete solution vector $\boldsymbol{u}_h$ is passed to the *Estimate* block. Here, the DWR method is solved by computing the adjoint problem in the primal space (equation (5.21)) and in the enriched space (equation (5.22)). Then, the element-wise error estimate can be obtained by integrating equation (5.20) element-wise. The element-wise errors e_k can be passed to the *Mark* block, where elements are marked for refinement (equation (5.52)) if the total error $\Delta\mathcal{L}$ is larger than a lower (refinement) tolerance tol_r and a coarsening marking (equation (5.52)) is performed if the total error is above an upper (coarsening) tolerance tol_c. This implies that if $\text{tol}_c < \Delta\mathcal{L} < \text{tol}_r$, a combined coarsening and refinement step is performed, as described in equation (5.53). In this case, the coarsening marking from equation (5.52) is performed given $\mathcal{M}_r$. Given the elements marked for refinement and coarsening, collected in $\mathcal{M}_r$ and $\mathcal{M}_c$, respectively, the mesh can be *Adapt*ed. In order to start the solution interval again, the start point should be *Transfer*red to the new mesh, and the governing equation can be solved again. If the error is not in the interval $[\text{tol}_r, \text{tol}_c]$ or if the number of refinement iterations i exceeds the maximum number of refinement iterations, $I_{\max}$. If the total error is in the interval $[\text{tol}_r, \text{tol}_c]$ or if the number of refinement iterations i exceeds the maximum number of refinement iterations, $I_{\max}$, the solution can be advanced, e.g., using a load-stepping or an arc-length method. Thereafter, the governing equations can be *Solve*d again. Note that if $I_{\max} = 1$, no inner iterations for adaptive meshing are performed.

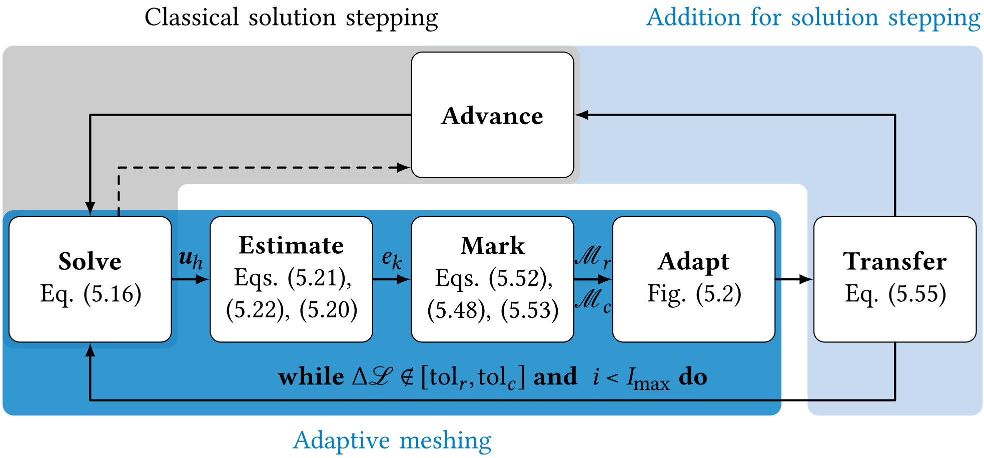

Classical solution stepping
Addition for solution stepping
Advance
Solve
Eq. (5.16)
u_h
Estimate
Eqs. (5.21), (5.22), (5.20)
e_k
Mark
Eqs. (5.52), (5.48), (5.53)
$\mathscr{M}_r$
$\mathscr{M}_c$
Adapt
Fig. (5.2)
Transfer
Eq. (5.55)
while $\Delta\mathscr{L} \notin [\text{tol}_r, \text{tol}_c]$ and $i < I_{\max}$ do
Adaptive meshing

Figure 5.5: A graphical summary of the adaptive meshing flowchart from figure 5.1 used in the present work. The equations that are used in each step are indicated in the blocks. The adaptive meshing iterations are performed within each solution step until the total error $\Delta\mathscr{L}$ is contained in the interval $[\text{tol}_r, \text{tol}_c]$, $\text{tol}_r < \text{tol}_c$, following the tolerances in section 5.4.3. In case of convergence, the solution is advanced, e.g., with an arc-length iteration. Algorithm 1 provides an algorithm corresponding to this flow chart.

Algorithm 1 An algorithmic summary of the goal-adaptive meshing routine employed in the present work. Figure 5.5 provides a graphical summary of this algorithm.

```
                for loadsteps do
                    while $\Delta\mathscr{L} \notin [\text{tol}_l, \text{tol}_u]$ and $i < I_{\max}$ do
Solve         |         Compute the primal solution $\boldsymbol{u}_h$ (Eq. (5.16))
              |         Compute the dual solution, $\xi_h$ given $\boldsymbol{u}_h$ (Eq. (5.21))
Estimate      |         Compute the enriched dual solution, $\tilde{\xi}_h$ given $\boldsymbol{u}_h$ (Eq. (5.22))
              |         Compute the total error estimation $\Delta\mathscr{L}$ according to Eq. (5.20) and find the
                element-wise errors $e_k$ and the total element error $e$.
              |         if $\Delta\mathscr{L} > \text{tol}_r$ then
              |             Mark elements for refinement into $\mathscr{M}_r$ using $e$, see Eq. (5.48).
              |         end if
Mark          |         if $\Delta\mathscr{L} < \text{tol}_c$ then
              |             Mark elements for coarsening into $\mathscr{M}_c$ using $e$, see Eq. (5.52).
              |         end if
Adapt         |         Refine all $Q \in \mathscr{M}_r$ using THB-splines, see Fig. 5.2
              |         Coarsen all $Q \in \mathscr{M}_c$ using THB-splines, see Fig. 5.2
Transfer      |         Transfer the solutions required to start the new solution step to the new mesh
                using Quasi-Interpolation, see Eq. (5.55).
                    end while
Advance       |     Advance the solution to the next solution step
                end for
```

5.6 Numerical Examples

In this section, several numerical examples are presented. The examples represent different applications of the theory presented in this chapter, and, without loss of generality, all employ isogeometric Kirchhoff–Love shells. The first three examples illustrate the performance of the DWR error estimator, and the last three examples demonstrate the use of this error estimator for adaptive meshing. More precisely, the numerical examples performed in this section, as well as their purpose, are:

Linear static analysis of a square plate (section 5.6.1) A simple example of linear Kirchhoff-Love shell theory is presented. In this case, error estimators using the DWR method are computed for different goal functionals and verified using the actual error computed from manufactured solutions. The goal of this benchmark problem is to evaluate the accuracy of the error estimators in linear static analysis.

Modal analysis of a circular plate (section 5.6.2) Since the analytical eigenvalues and eigenmodes are known for this case, the goal of this benchmark problem is to verify the error estimator for a vibration eigenvalue problem, given in equation (5.26) with the stiffness and mass operators $\mathscr{A}(\boldsymbol{v},\boldsymbol{\phi})=\mathscr{W}'(0,\boldsymbol{v},\boldsymbol{\phi})$ and $\mathscr{B}(\boldsymbol{v},\boldsymbol{\phi})=\mathscr{M}(\boldsymbol{v},\boldsymbol{\phi})$.

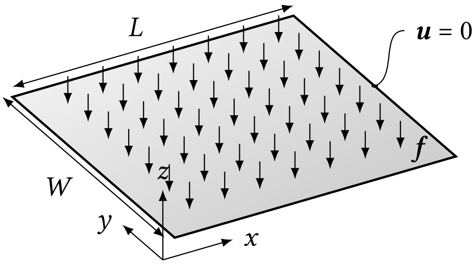
Linear buckling analysis of a square plate (section 5.6.3) Analytical critical buckling loads and mode shapes are also known in this case. Therefore, the goal of this benchmark problem is to provide verification for the buckling error estimator from equation (5.26) with the buckling operators $\mathscr{A}(\boldsymbol{v},\boldsymbol{\phi})=\mathscr{W}'(\boldsymbol{u}_L,\boldsymbol{v},\boldsymbol{\phi})$ and $\mathscr{B}(\boldsymbol{v},\boldsymbol{\phi})=\mathscr{M}(\boldsymbol{v},\boldsymbol{\phi})$.

Non-linear analysis of a pinched thin plate (section 5.6.4) In this example, a thin plate with very low bending stiffness subjected to an out-of-plane load is analysed. The error estimator is used to provide mesh adaptivity to compare to uniform refinement. The goal of this benchmark problem is to evaluate the performance of the DWR method as a driver for adaptive meshing in a static load case with geometric non-linearities.

Snap-through instability of a cylindrical roof (section 5.6.5) The snap-through behaviour of a cylindrical roof is considered in this example. The benchmark problem is a well-known application of arc-length methods and shells. The goal of solving this problem is to test the full adaptive solution stepping procedure from figure 5.5 on a benchmark problem.

Wrinkling analysis (section 5.6.6) In the last example, the procedure from figure 5.5 is applied to the modelling of membrane wrinkling. This problem contains geometric non-linearities and material non-linearities. The results are compared to uniform refinements to evaluate the efficiency of adaptive meshing for such applications. The goal of this example is to demonstrate the use of the adaptive meshing routine from figure 5.5 on a complex load-stepping problem with geometric and material non-linearities.

In the following subsections, the short-hand notations $\mathscr{L}_{\text{an}}=\mathscr{L}(\boldsymbol{u}_{\text{an}})$, $\mathscr{L}_{\text{num}}=\mathscr{L}(\boldsymbol{u}_{\text{num}})$, $\Delta\mathscr{L}_{\text{an}}=\mathscr{L}_{\text{an}}-\mathscr{L}_{\text{num}}$ and $\Delta\mathscr{L}_{\text{num}}=\mathscr{R}(\boldsymbol{u}_{\text{num}},\tilde{\xi}_h-\xi_h)$ (see equation (5.20)) are used, given



	Geometry	
L	1	[m]
W	1	[m]
t	0.01	[m]
	Material *Saint-Venant Kirchhoff*	
E	10^6	[Pa]
ν	0.3	[-]
	Boundary Conditions	
$\boldsymbol{u}$	0	
	Loads	
$\boldsymbol{f}$	See equation (5.56)	[Pa]

Figure 5.6: Geometry and parameters for the example of a unit-square plate with a distributed vertical load $\boldsymbol{f}$ given by equation (5.56). The displacements are fixed on all edges.

the analytical and numerical solutions $\boldsymbol{u}_{\text{an}}$ and $\boldsymbol{u}_{\text{num}}$, respectively. Furthermore, where relevant, the parameters ρ_r, ρ_c, tol_r, and tol_c (see equations (5.48), (5.52) and (5.53)) are fixed per example. A study on finding optimal values for these parameters is out of the scope of this chapter. Lastly, all simulations are performed using the open-source Geometry+Simulation modules [294].

5.6.1 Linear Static Analysis of a Square Plate

For the linear shell example, let us consider a flat plate with unit dimensions $L = W = 1\,[\text{m}]$, a thickness of $t = 10^{-2}\,[\text{m}]$ and with material parameters $E = 10^6\,[\text{Pa}]$, $\nu = 0.3$, which is clamped on all sides; see figure 5.6. A load vector of

$$\boldsymbol{f} = \frac{2EAt^3}{1-\nu^2}\Big(x^4 - 2x^3 + 3x^2 - 2x + y^4 - 2y^3 + 3y^2 - 2y + 12x^2y^2 \\ - 12x^2y - 12xy^2 + 12xy + \frac{1}{3}\Big)\boldsymbol{e}_z \tag{5.56}$$

is applied, based on the manufactured solution given by

$$\boldsymbol{u}_{\text{an}} = Ax^2(x-1)^2y^2(y-1)^2\boldsymbol{e}_z. \tag{5.57}$$

Using this manufactured solution, any goal functional $\mathscr{L}_{\text{an}}$ can be evaluated. Solving the primal problem for this linear shell example gives $\boldsymbol{u}_{\text{num}}$, which can be used to compute the DWR error estimate of $\Delta\mathscr{L}_{\text{num}}$.

In figure 5.7, the results for the linear shell problem are given. The title of each column represents the goal function that is used for error estimation in this column. The top row provides the errors $\Delta\mathscr{L}_{\text{an}}$ and $\Delta\mathscr{L}_{\text{num}}$ with respect to a uniformly refined mesh size. As

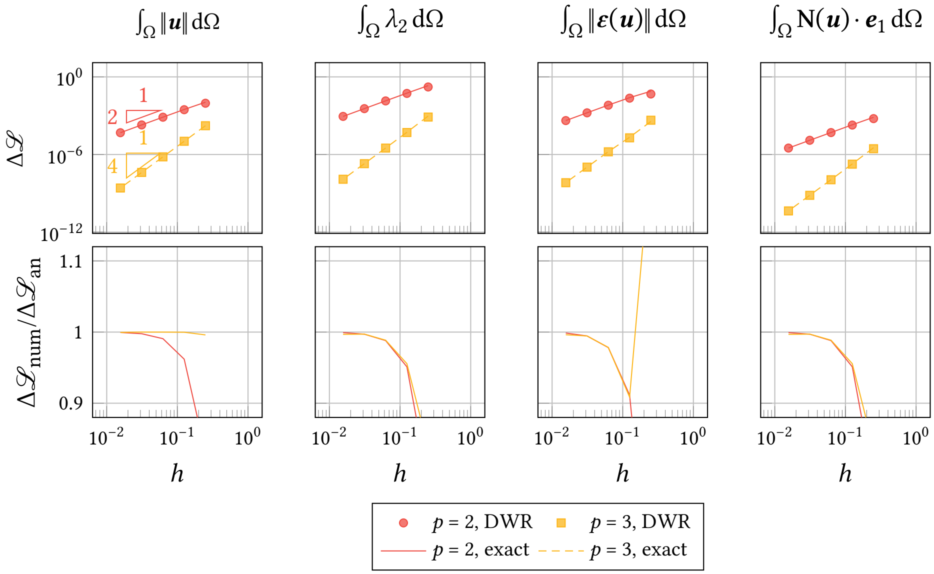


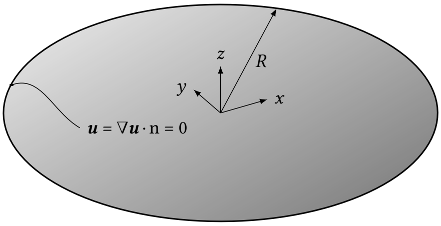
Figure 5.7: Linear static analysis of a clamped plate with a uniformly distributed load according to equation (5.56) according to a manufactured solution from equation (5.57). The top row provides $\Delta\mathscr{L}$ against the uniform mesh size h. The bottom row presents the efficiency of the error estimators against the mesh size h. The markers represent error estimates computed via the DWR method, and the lines represent the exact error, i.e., the error of the numerical solution with respect to the analytical solution. The results are given for goal functionals (from left to right) $\mathscr{L} = \int_\Omega \|\boldsymbol{u}\|\,\mathrm{d}\Omega$ (displacement norm), $\mathscr{L} = \int_\Omega \lambda_2\,\mathrm{d}\Omega$ (second principal stretch), $\mathscr{L} = \int_\Omega \|\boldsymbol{\varepsilon}(\boldsymbol{u})\|\,\mathrm{d}\Omega$ (membrane strain norm) and $\mathscr{L} = \int_\Omega \mathrm{N}(\boldsymbol{u})\cdot\boldsymbol{e}_1\,\mathrm{d}\Omega$ (first component of membrane force).

can be seen in this figure, $\Delta\mathscr{L}_{\mathrm{num}}$ quickly converges to $\Delta\mathscr{L}_{\mathrm{an}}$ for different spline orders p. In addition, the bottom row of figure 5.7 provides the efficiency of the error estimator, given by $\Delta\mathscr{L}_{\mathrm{num}}/\Delta\mathscr{L}_{\mathrm{an}}$. These figures confirm the convergence of the DWR estimates to the analytical goal functional errors for all considered goal functionals. Only for the membrane strain norm goal functional, the error estimate for coarse meshes is inaccurate. This can possibly be explained by the in-plane shear strain that cancels out over the whole domain but does contribute to the norm $\|\boldsymbol{\varepsilon}\|$.

Concluding, the linear shell benchmarks show that for different goal functionals, the DWR method provides an accurate estimation of the error $\Delta\mathscr{L}$ of the goal functional $\mathscr{L}$, starting at relatively small mesh sizes of $h < 10^{-1}$.

5.6.2 Modal Analysis of a Circular Plate

As a next example, the vibration modes of a circular plate with clamped boundary conditions are computed. The geometry with boundary conditions is illustrated in figure 5.8. The circular plate has a unit diameter and a thickness $t = 10^{-2}$ [m], Young's modulus $E = 10^{6}$ [Pa], Poisson's ratio $\nu = 0.3$ and density $\rho = 1$ [kg/m^3]. The analytical solutions


z
R
y
x
$\boldsymbol{u} = \nabla\boldsymbol{u} \cdot \mathrm{n} = 0$

Geometry		
R	0.5	[m]
t	0.01	[m]
Material *Saint-Venant Kirchhoff*		
E	10^6	[Pa]
ν	0.3	[-]
ρ	1	[kg/m^3]
Boundary Conditions		
$\boldsymbol{u} = \nabla\boldsymbol{u} \cdot \hat{\boldsymbol{a}}_3$	0	

Figure 5.8: Geometry and parameters for a vibrating circular plate with a clamped boundary.

for the eigenfrequencies of the circular plate are obtained by

$$\omega_n = \gamma_n^2 \sqrt{D/\rho t}, \tag{5.58}$$

where γ_n is the n^{th} root of the equation $(I_{m-1}(\gamma R) - m/R I_m(\gamma R))J_m(\gamma R) - I_m(\gamma R)(J_{m-1}(\gamma R) - m/R J_m(\gamma R)) = 0$ following from a separation of variables solution [583], R is the radius of the plate, and $D = Et^3/(12(1-\nu^2)) = 9.16 \cdot 10^{-8}$ is the flexural rigidity. As stated in section 5.3.2, the goal functional of eigenvalue problems is given in equation (5.31), and requires the eigenvalue problem in equation (5.24) to be solved with linear operators $\mathscr{A}(\boldsymbol{v}, \boldsymbol{\phi}) = \mathscr{W}'(0, \boldsymbol{v}, \boldsymbol{\phi})$ and $\mathscr{B}(\boldsymbol{v}, \boldsymbol{\phi}) = \mathscr{M}(\boldsymbol{v}, \boldsymbol{\phi})$, see equations (5.10) and (5.14).

Figure 5.9 presents the first four eigenmodes in the top row. Furthermore, $\Delta\mathscr{L}_{\text{an}}$ and $\Delta\mathscr{L}_{\text{num}}$ as a function of the element size for the uniformly refined domain are given in the middle row for the first four eigenmodes. These plots show that the approximation for the error $\Delta\mathscr{L}_{\text{num}}$ approximates $\Delta\mathscr{L}_{\text{an}}$. In the bottom row of figure 5.9, the efficiencies also show that the approximation converges to an efficiency equal to 1. However, for the $p = 4$ line, the efficiency degrades when the 'exact' error obtained by the analytical solution approaches values around 10^{-11}, which is attributed to the approximation of the roots γ_i and the precision of the eigenvalue solver.

Concluding, the modal analysis benchmark shows that the DWR method provides accurate estimation of the eigenfrequency error for different considered mode shapes.

5.6.3 Linear Buckling Analysis of a Square Plate

Similar to modal analysis, DWR error estimation for buckling analysis also relies on the formulations in section 5.3.2. The difference with the modal analysis error estimation is that the buckling analysis error estimation involves the solution of a pre-buckling solution and no mass matrix. As an example for buckling analysis, a square simply supported plate is considered, see figure 5.10, with a Saint-Venant Kirchhoff constitutive law with Young's modulus $E = 10^6$ [Pa] and Poisson's ratio $\nu = 0.3$ [-]. The dimensions of the plate are $L \times W \times t = 1 \times 1 \times 0.01$ [m^3]. The plate is subject to a distributed line load of σt in both

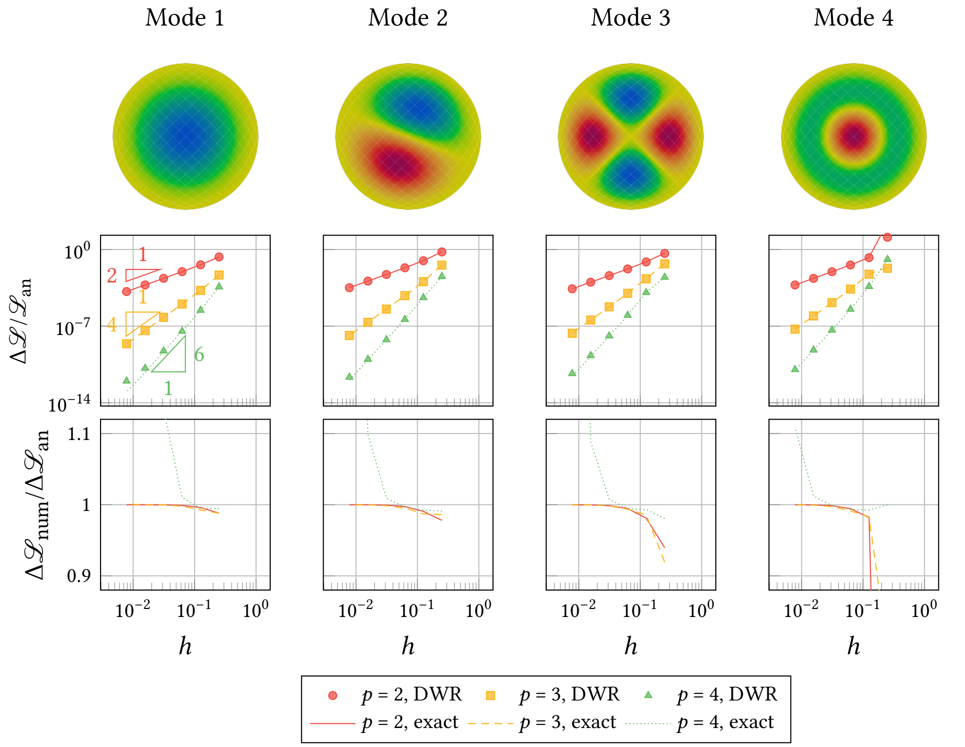
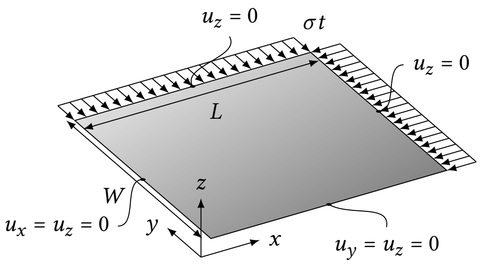


Figure 5.9: Modal analysis of a circular plate. The top row provides the mode shapes from mode 1 up to 4. The mid row provides $\Delta\mathscr{L}$ against the uniform mesh size h, and the bottom row provides the efficiency of the error estimators against the mesh size h. The markers represent error estimates computed via the DWR method, and the lines represent the exact error, i.e. the error of the numerical solution with respect to the analytical solution. The eigenfrequencies for modes 1 up to 4 are, respectively, $\omega_1 = 9.56 \cdot 10^{-4}$ [Hz], $\omega_2 = 4.14 \cdot 10^{-3}$ [Hz], $\omega_3 = 1.11 \cdot 10^{-2}$ [Hz] and $\omega_4 = 1.45 \cdot 10^{-2}$ [Hz].



Geometry		
L	1	[m]
W	1	[m]
t	0.01	[m]
Material *Saint-Venant Kirchhoff*		
E	10^6	[Pa]
ν	0.3	[-]
Boundary Conditions		
$u_x = u_z$	0	$x = 0, y$
u_z	0	$x = L, y$
$u_y = u_z$	0	$x, y = 0$
u_z	0	$x, y = W$
Loads		
σt	Variable	

Figure 5.10: Geometry and parameters for the plate buckling example. A distributed load of σt is acting on two boundaries in two different directions, and the other boundaries are simply supported and fixed in an out-of-plane direction.

directions. The analytical solution for the buckling load with m half waves in x-direction and n half waves in y-direction for a square plate with sides L and with equal loads is given in [293] and reads:

$$\sigma_c^{m,n} t = \frac{D\pi^2}{L^2}\left(m^2 + n^2\right), \tag{5.59}$$

with $D = Et^3/12(1-\nu^2)$ the flexural rigidity of the plate. Using this expression, the first four unique modes are, indexed in ascending order: $\sigma_c^{1,1} t = \sigma_c^1 t = 1.808$ [N/m], $\sigma_c^{2,1} t = \sigma_c^{1,2} t = \sigma_c^2 t = 4.519$ [N/m], $\sigma_c^{2,2} t = \sigma_c^3 t = 7.230$ [N/m], $\sigma_c^{3,1} t = \sigma_c^{1,3} t = \sigma_c^4 t = 9.038$ [N/m].

Figure 5.11 depicts the analytical error $\Delta\mathscr{L}_{\text{an}}$ and the DWR error estimate $\Delta\mathscr{L}_{\text{num}}$ as a function of the mesh size h for uniform refinements. Both errors are normalised with the analytical value of the critical buckling load. As can be seen in this figure, the DWR prediction of the error converges with a rate of convergence of $2(p-1)$ for all degrees p until it reaches values of around 10^{-10}, after which the errors stagnate and increase again (in particular for $p = 4$). This behaviour is similar to the behaviour observed in [358] and can be attributed to the round-off errors as discussed. These errors occur when the number of degrees of freedom is large enough and the machine precision is limited. In the case of a buckling simulation, where the non-linear stiffness operator is constructed on an initial solution of a linear simulation, the influence of round-off errors is expected to occur sooner. In addition, it can also be seen that the error computed using the analytical solution stagnates. This is due to the fact that the numerical approximation of the critical buckling load shows small variations depending on the solution to the linear problem that is solved to obtain the tangential stiffness matrix to compute the generalised eigenvalue problem for buckling. For the results presented in figure 5.11, the load $\sigma t = 10^{-4}$ [N/m]

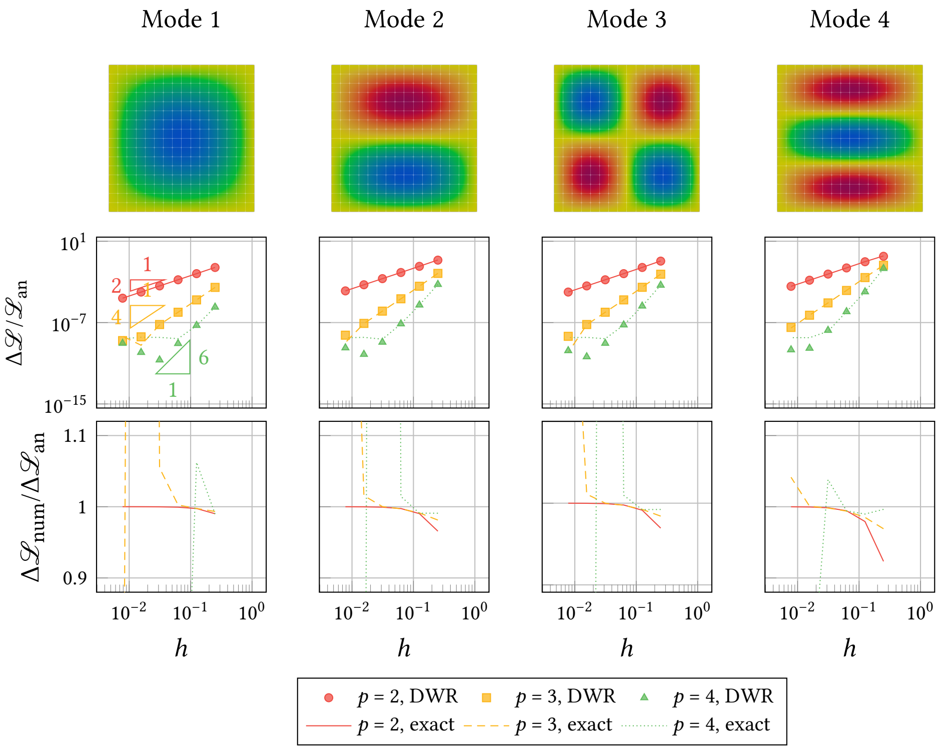
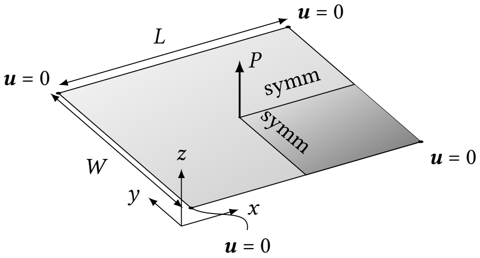


Figure 5.11: Buckling analysis of a square plate with simply supported boundary conditions. The top row provides the mode shapes from mode 1 up to 4. The bottom row provides $\Delta\mathscr{L}$ against the uniform mesh sise h. The markers represent error estimates computed via the DWR method, and the lines represent the exact error, i.e., the error of the numerical solution with respect to the analytical solution. The critical loads for unique modes 1 up to 4 are, respectively, $\sigma_c^{1,1}t = \sigma_c^{1}t = 1.808$ [N/m], $\sigma_c^{2,1}t = \sigma_c^{1,2}t = \sigma_c^{2}t = 4.519$ [N/m], $\sigma_c^{2,2}t = \sigma_c^{3}t = 7.230$ [N/m], $\sigma_c^{3,1}t = \sigma_c^{1,3}t = \sigma_c^{4}t = 9.038$ [N/m].

was used to compute the buckling linearisation.

5.6.4 Non-Linear Analysis of a Pinched Thin Plate

In the next example, a square membrane subject to a point load in the middle and with corners fixed in all directions is considered; see figure 5.12. The membrane is modelled with a Saint-Venant Kirchhoff constitutive law with Young's modulus $E = 1.0\,[MPa]$ and a Poisson ratio $\nu = 0.3$. The thickness of the membrane is $t = 10^{-3}$ [mm], and the length and width are $L \times W = 1 \times 1$ [mm]. The simulation is performed on a quarter of the domain, employing symmetry conditions as depicted in figure 5.12. A load of $P = 4 \cdot 10^{-7}$ [N] is applied in the centre of the sheet. The static load case is solved using an arc-length method to ensure convergence of the solution. Furthermore, an adaptive refinement strategy is employed with admissible refinement. The jump parameter m is set to 2, and the maximum number of refinement levels is 8 or 11, which equals a tensor basis level with $2^8 \times 2^8 =$



	Geometry	
L	1	[m]
W	1	[m]
t	$1 \cdot 10^{-3}$	[m]
	Material *Saint-Venant Kirchhoff*	
E	1.0	[MPa]
ν	0.3	[-]
	Boundary Conditions	
$\boldsymbol{u}$	0	At corners
	Loads	
P	$4 \cdot 10^{-7}$	[N]

Figure 5.12: Geometry and parameters for a square thin plate subject to a point load P in the middle. The plate is fully constrained in every corner. Because the problem is symmetric, only a quarter of the domain is modelled. Hence, symmetry conditions are applied. On the x-aligned symmetry axis, this implies that $u_y = \frac{\partial u_z}{\partial y} = \frac{\partial u_x}{\partial y} = 0$ and on the y-aligned symmetry axis, this implies that $u_x = \frac{\partial u_z}{\partial x} = \frac{\partial u_y}{\partial x} = 0$.

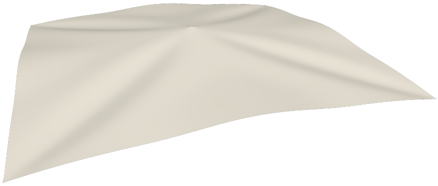


Figure 5.13: Deformed surface from the benchmark presented in figure 5.12. The result is the last solution from the adaptive meshing routine with deformation norm goal-functional, of which the results are presented in figure 5.14.

256 × 256 or $2^{11} \times 2^{11} = 2048 \times 2048$ elements, respectively. The refinement parameter is set to $\rho_r = 0.5$. The goal functionals considered in this case are based on displacements as well as principal stresses:

$$\begin{aligned} \mathscr{L}(\boldsymbol{u}) &= \int_\Omega \|\boldsymbol{u}\| \,\mathrm{d}\Omega, \\ \mathscr{L}(\boldsymbol{u}) &= \int_\Omega \boldsymbol{\sigma}_p \cdot \boldsymbol{e}_y \,\mathrm{d}\Omega. \end{aligned} \tag{5.60}$$

Figure 5.13 presents the deformed membrane for the last step of the adaptive simulation. Furthermore, figure 5.14 presents the estimated error $\Delta\mathscr{L}$ given the goal functionals in equation (5.60) for the uniform refinement as well as for the adaptive refinement simulation with a maximum level of 8 or 11. Moreover, figures 5.15 and 5.16 provide the absolute element-wise errors for the adaptive refinement simulation and for the uniform refinement series for both considered goal functionals. The contour lines in these error fields represent the vertical deflection of the sheet.

From all the provided results, it can be observed that the adaptive mesh provides for both goal functionals an efficient converging mesh, where the accuracy per degree of freedom is higher compared to uniformly refined meshes. However, it can also be seen that the total error $\Delta\mathscr{L}$ is not strictly decreasing for both goal functionals. The bottom plots of figure 5.14 indicate that this point is closely related to the maximum depth that is reached: the percentage of the total element error e that can still be refined rapidly decreases, meaning that the only elements that are still available for refinement are the ones that have an insignificant contribution to the total element error e, deeming refinement of these elements meaningless. It can be seen from figures 5.15 and 5.16 that the maximum element depth is reached in the corner where the sheet is fixed. After the maximum level is reached, the refined elements start distributing over the diagonal of the domain and in the top-left corner, where the load is applied. For both goal functionals, figure 5.14 shows that some further decrease in the total error $\Delta\mathscr{L}$ can be gained after the maximum error is reached, but that it is most effective to increase the maximum refinement depth. Comparing the error fields and meshes for both goal functionals (see figures 5.15 and 5.16), it can be seen that the second-principal stress-based error field shows narrower error bands in the finest depicted uniform refinement error fields than the displacement-based error estimator. As a consequence, the corresponding adaptive meshes show that the elements indeed tend to be broader distributed along the diagonal of the domain in the case of displacement-driven refinement (figure 5.15).

Concluding, the non-linear shell benchmark shows that an adaptive meshing strategy provides accurate solutions on meshes with a small number of degrees of freedom compared to uniform meshes. Furthermore, the benchmarks show the importance of refinement levels, meaning that convergence of the adaptive meshing strategy vanishes as soon as the elements contributing to a large extent to the total error are at the lowest allowed level. This stresses the relevance of spline constructions that allow for deep levels of refinement with moderate computational costs.

5.6.5 Snap-Through Instability of a Cylindrical Roof

In order to present the results of the adaptive isogeometric method developed in this chapter for quasi-static problems, hence completing full cycles in figure 5.1, the well-known benchmark of a collapsing cylindrical roof [530] subject to a point load is considered. The goal of this benchmark problem is to evaluate whether the presented adaptive isogeometric method provides DoF-wise efficient solutions compared to solutions with uniform refinements.

The geometry for the benchmark problem is presented in figure 5.17. Here, the radius of the roof is $R = 2.540$[m], the angle is $\theta = 0.1$ [rad] and the length and thickness are, respectively, $L = 0.508$ [m] and $t = 6.35 \cdot 10^3$ [mm]. Moreover, the material properties are $E = 3102$ [MPa] and $\nu = 0.3$ [-] for a Saint-Venant Kirchhoff material. Only a quarter of the roof is modelled, as depicted in figure 5.17, because of symmetry. The simulation is performed using the Crisfield arc-length method [124] with arc-length $\Delta L = 25$ and zero force-scaling. The goal functional that is used for error evaluation and adaptivity is based

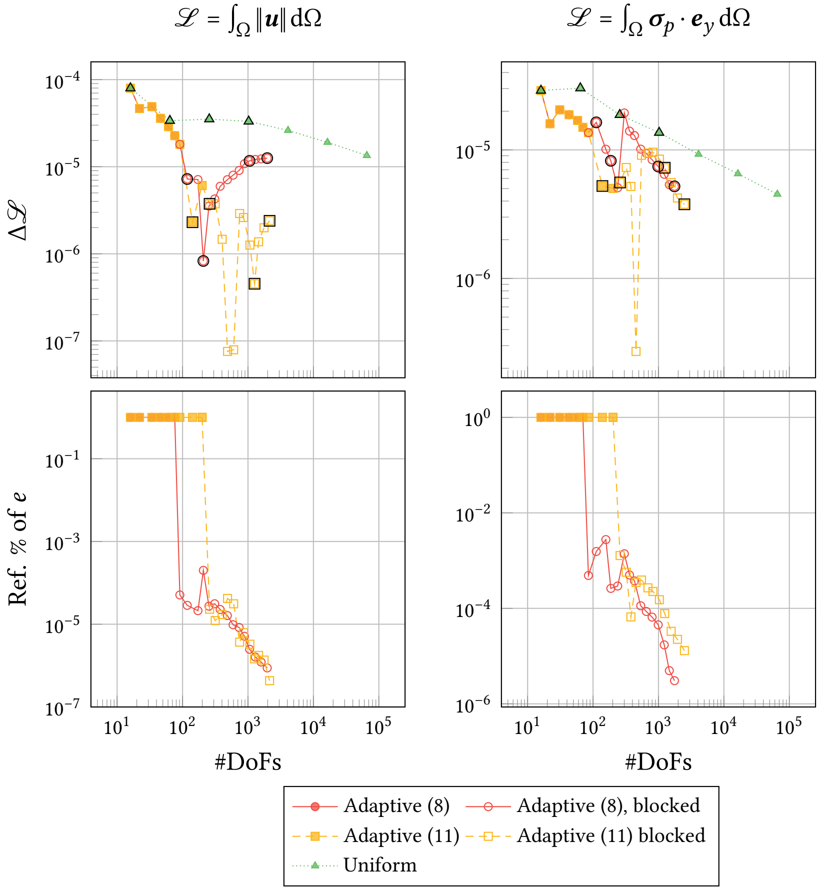
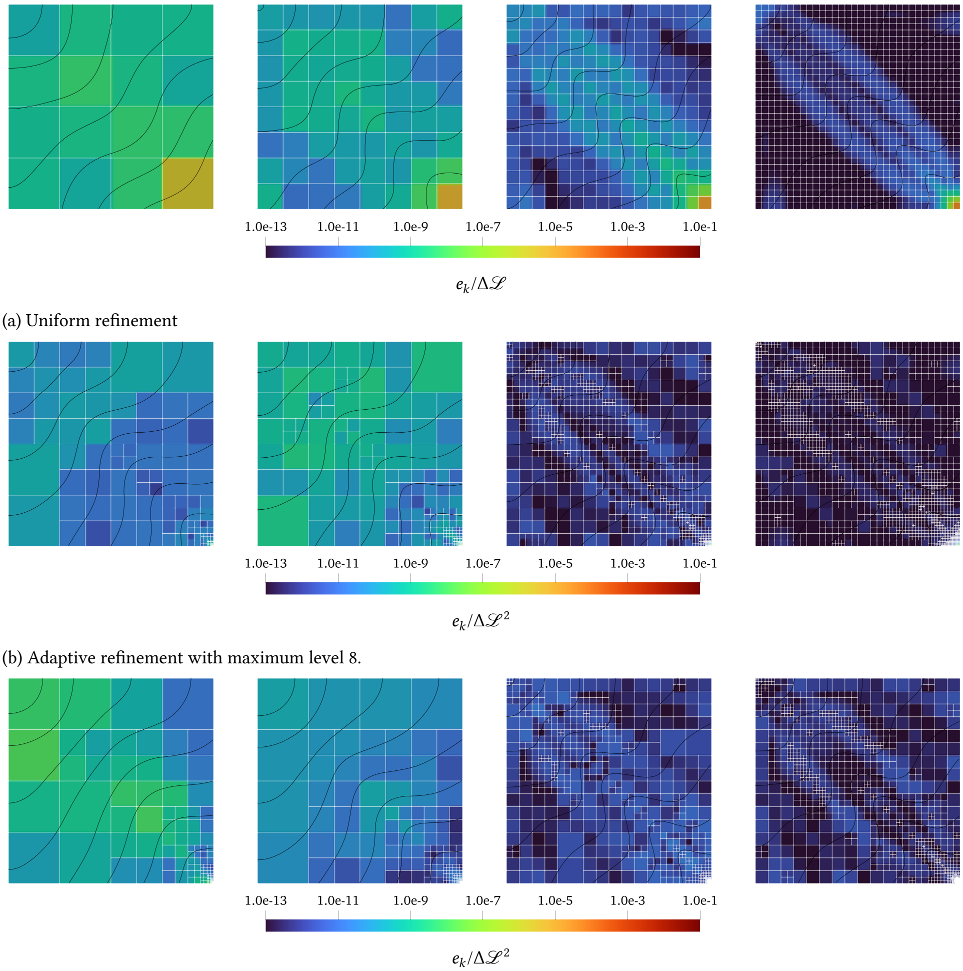


Figure 5.14: Estimated error convergence (top) and the percentage of the total element error e that is available for refinement (bottom) against the number of degrees of freedom (DoFs) for adaptively and uniformly refined meshes with respect to the goal functionals from equation (5.60). The markers labelled with a black border are the markers for which the mesh is plotted in figures 5.15 and 5.16. The filled markers represent points where refinement is not blocked, and the empty markers represent points where refinement is blocked because the maximum refinement depth is reached. Note that the errors are computed before refinement; hence, blocked elements in iteration i have an effect on the error computation in iteration $i + 1$.

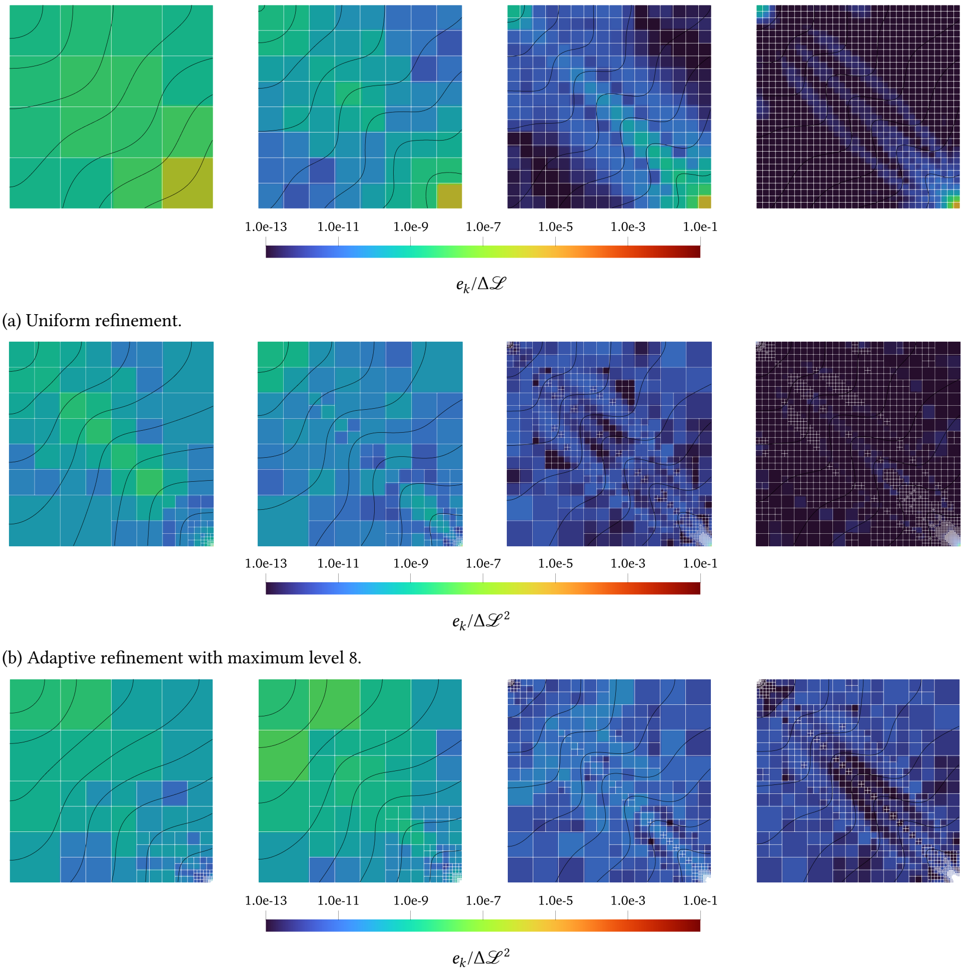


(a) Uniform refinement

(b) Adaptive refinement with maximum level 8.

(c) Adaptive refinement with maximum level 11.

Figure 5.15: Normalised element error values $e_k/\Delta\mathscr{L}$ for uniformly (a) and adaptively (b, c) refined meshes using goal-function $\mathscr{L}(\boldsymbol{u}) = \int_\Omega \|\boldsymbol{u}\|\,\mathrm{d}\Omega$. The meshing steps increase from left to right. The contour lines represent the displacement of the membrane, with intervals of 0.1 [mm]. The bottom right corners of the pictures indicate the fixed corner, and the top left corners are the corners where the load is applied.

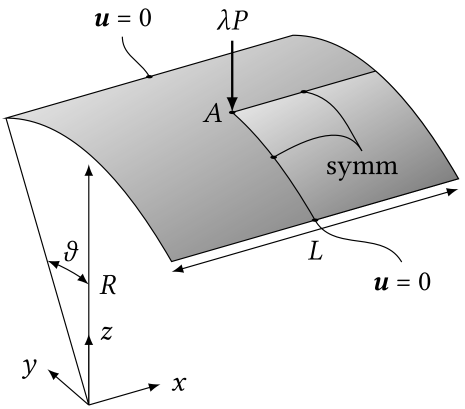


(a) Uniform refinement.

(b) Adaptive refinement with maximum level 8.

(c) Adaptive refinement with maximum level 11.

Figure 5.16: Normalised element error values $e_k/\Delta\mathscr{L}^2$ for uniformly (a) and adaptively (b, c) refined meshes using goal-function $\mathscr{L}(\boldsymbol{u}) = \int_\Omega \boldsymbol{\sigma}_p \cdot \boldsymbol{e}_y \, \mathrm{d}\Omega$. The meshing steps increase from left to right. The contour lines represent the displacement of the membrane, with intervals of 0.1 [mm]. The bottom right corners of the pictures indicate the fixed corner, and the top left corners are the corners where the load is applied.



Geometry		
L	0.508	[m]
R	2.540	[m]
t	$6.35 \cdot 10^{-3}$	[m]
ϑ	0.1	[rad]
Material *Saint-Venant Kirchhoff*		
E	3102	[MPa]
ν	0.3	[-]
Boundary Conditions		
$\boldsymbol{u}$	0	At corners
Loads		
P	$3 \cdot 10^{3}$	[N]

Figure 5.17: Geometry and parameters for a cylindrical roof with a point load P in the middle. The bottom-right corner of each domain corresponds to the point A: the reference point for which the z-displacements are plotted. The roof is free on the curved edges and simply supported ($\boldsymbol{u} = 0$) on the straight edges. As a consequence, the problem is symmetric, and a quarter of the domain is modelled. On the x-aligned symmetry axis, this implies that $u_y = \frac{\partial u_z}{\partial y} = \frac{\partial u_x}{\partial y} = 0$, and on the y-aligned symmetry axis, this implies that $u_x = \frac{\partial u_z}{\partial x} = \frac{\partial u_y}{\partial x} = 0$.

on the norm of the flexural strain tensor over the whole domain $\mathscr{L} = \int_\Omega \|\mathrm{M}(\boldsymbol{u})\| \,\mathrm{d}\Omega$. The jump parameter for admissible meshing is set to $m = 2$. The mesh will be refined when $\Delta\mathscr{L} > \mathrm{tol}_r$ and coarsened when $\Delta\mathscr{L} < \mathrm{tol}_c$. As discussed in section 5.5, $\mathrm{tol}_r < \mathrm{tol}_c$ such that the mesh is refined and coarsened simultaneously when $\Delta\mathscr{L} \in [\mathrm{tol}_r, \mathrm{tol}_c]$, which is also the condition for termination. The maximum number of mesh adaptivity iterations is set to 5 in this case. The tolerances tol_r and tol_c are determined based on the results of uniformly refined simulations with 16 × 16 (918 DoFs) and 32 × 32 elements (3366 DoFs) by taking a wide band around the error envelopes in figure 5.18 excluding peaks. The tolerances are $(\mathrm{tol}_r, \mathrm{tol}_c) = (10^{-10}, 10^{-8})$. It should be noted that these tolerances can also be based on requirements in engineering, or they can be determined during the computations; both are beyond the scope of this chapter. The refinement parameter is set to $\rho_r = 0.5$, the coarsening parameter to $\rho_c = 0.05$, and the maximum refinement level is 11. The adaptive and uniform meshes are modelled with bi-cubic B-spline basis functions (i.e., $p = 3$).

In figure 5.19, the results of a simulation of the collapsing roof for the adaptive mesh are plotted in a $\|\boldsymbol{u}\|$, λP, w_A-space. Reference solutions for the present benchmark problem are typically given in the λP, w_A-space, but since the solution curve is not bijective, an alternative coordinate $\|\boldsymbol{u}$ is used to represent solutions for this benchmark problem. This is motivated by the projection of the solutions λP and w_A projected against $\|\boldsymbol{u}\|$ in figure 5.19. The results of [530] are provided as a reference.

In figure 5.18, the error and the number of degrees of freedom are plotted against $\|\boldsymbol{u}\|$. The error envelopes for the uniform meshes show that a large peak occurs around $\|\boldsymbol{u}\| = 0.2$, relating to the first limit point of the collapse of the roof, as seen by the markers in figure 5.19. Additionally, it can be seen that the present algorithm providing mesh

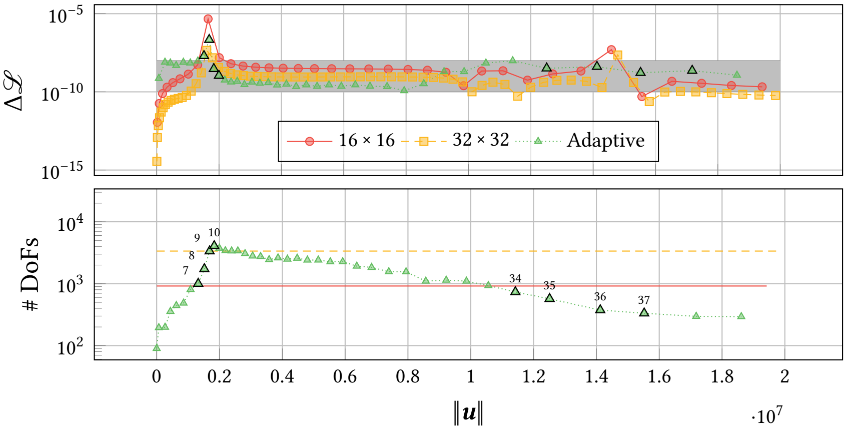


Figure 5.18: Errors (top) and number of degrees of freedom (DoFs, bottom) for the goal functional $\mathscr{L} = \int_\Omega \|\mathbf{M}(\boldsymbol{u})\| \,\mathrm{d}\Omega$ with tolerances $\mathrm{tol}_r = 10^{-10}$ and $\mathrm{tol}_c = 10^{-8}$ for the collapsing roof subject to the displacement norm $\|\boldsymbol{u}\|$. The grey region represents the region $\Delta\mathscr{L} \in [\mathrm{tol}_r, \mathrm{tol}_c]$ where refinement and coarsening are performed and where the adaptivity iterations are terminated. Above the grey region, only refinement is performed, and below the grey region, only coarsening is performed.

adaptivity manages to keep the error within specified bounds (see figure 5.18, top), except on the peak just before $\|\boldsymbol{u}\|$ where the maximum number of adaptivity iterations is insufficient. Furthermore, it has consistently fewer degrees of freedom than the uniform mesh with 32 × 32 elements. In figure 5.20, a selection of meshes is provided. The meshes are provided as a series of four consecutive meshes around the limit points of the solution curve, as indicated in figure 5.19, and are depicted in increasing order from left to right for the first (top) and second (bottom) limit points. The black-bordered markers in figures 5.18 and 5.19 indicate the points of which the meshes are shown. From the first row of meshes in figure 5.20, it can be seen that the first limit point requires relatively fine meshes and that the elements start concentrated around point A and its diagonal opposite and spread out on the bottom symmetry boundary as the snapping takes place. Furthermore, in the bottom row of figure 5.20, it can be seen that the second limit point does not require many elements; hence, the number of elements is slowly decreasing throughout this section of the load-displacement curve. For a complete overview of the mesh in each load step, the reader is referred to Video 1 in the supplementary material of [586].

Concluding, this example shows that the goal-adaptive meshing procedure is capable of keeping the error in terms of a goal function within pre-defined bounds for a solution-stepping simulation with limit-point instabilities. Throughout the simulation, the procedure maintains a relatively high efficiency per degree of freedom compared to uniform meshes. It should be noted, however, that the adaptive refinement iterations require a higher computational demand. Therefore, the next and final example provides a procedure where no adaptivity iterations are performed.

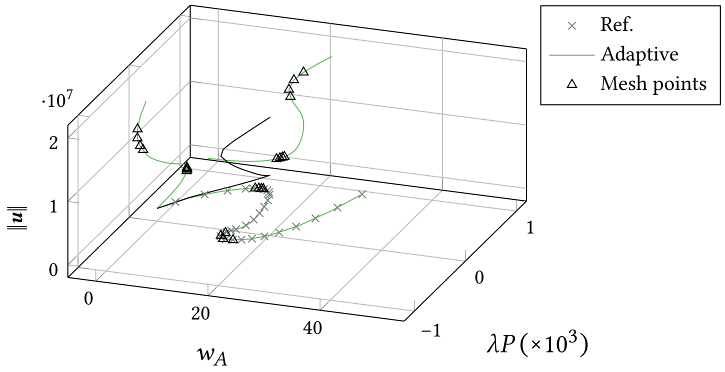


Figure 5.19: Projection the adaptively refined result from the commonly used $\lambda P, w_A$-space [530] onto the displacement norm $\|\boldsymbol{u}\|$. The solid lines correspond to the results obtained by the adaptively refined mesh, the triangular markers correspond to points of which the mesh is provided in figure 5.20, and the cross-markers (Ref.) indicate the reference result from [530].

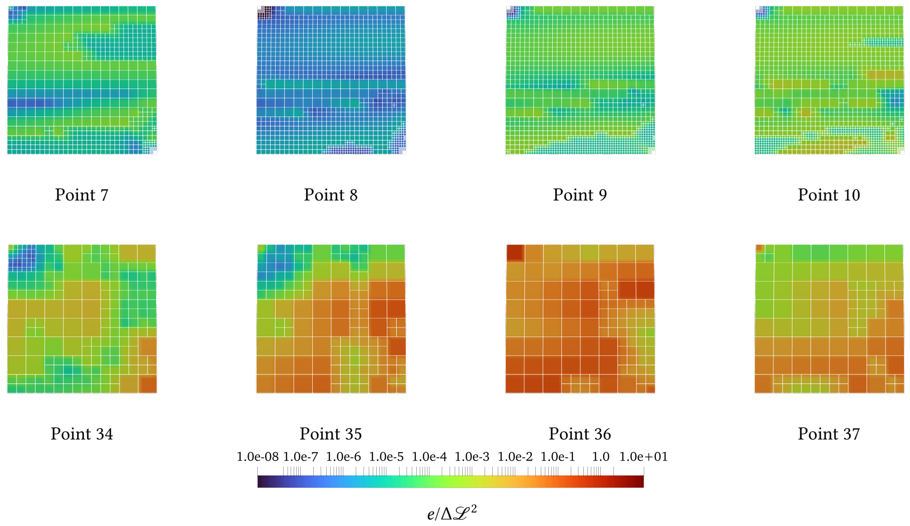


Figure 5.20: Meshes corresponding to the points marked in figures 5.18 and 5.19 by the black-bordered marks. The point A marks the point where the load is applied in figure 5.17. The top row of meshes corresponds to the first limit point in figure 5.19, and the bottom row corresponds to the second limit point. The meshes are ordered from left to right for increasing solution steps. The elements are coloured according to the squared error e_k normalised by the total error $\Delta\mathscr{L}$, i.e., $e_k/\Delta\mathscr{L}^2$.

5.6.6 Wrinkling Analysis

As a last example, the wrinkling analysis of a thin membrane subjected to a tensile load is considered. This problem is inspired by [91, 92, 427] and was previously modelled using isogeometric Kirchhoff–Love shells in [587], to which the reader is referred for a detailed problem set-up. The goal of this benchmark in the present chapter is to demonstrate the use of the goal-adaptive meshing procedure on a bifurcation problem with geometric and material non-linearities.

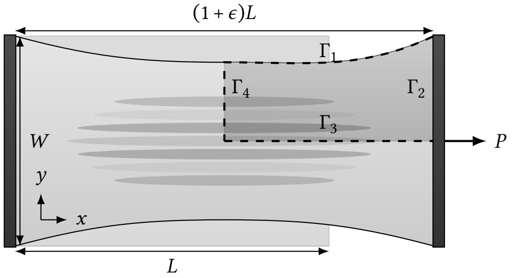
Given the geometry in figure 5.21, a quarter of the domain is considered with a symmetry boundary condition on Γ_4 and an antisymmetry condition on Γ_3. Furthermore, the boundary Γ_1 is free, and on Γ_2 the x-displacement is constant, and a horizontal load is applied on this side. The sheet has dimensions $L = 280\,[\text{mm}]$, $W = 140\,[\text{mm}]$, and $t = 0.14\,[\text{mm}]$ such that $L/W = 2$ and $t/W = 10^3$. Furthermore, the material is modelled using a Mooney-Rivlin material model with a strain energy density function:

$$\Psi(\mathbf{C}) = \frac{c_1}{2}(I_1 - 3) + \frac{c_2}{2}(I_2 - 3) \tag{5.61}$$

and with parameters $c_1 = 3.16 \cdot 10^5$ [Pa] and $c_1 = 1.24 \cdot 10^5$ [Pa]. Reference solutions are given by [587] for isogeometric Kirchhoff–Love shell analysis and using ANSYS and LS-DYNA FEA models, respectively. Furthermore, [427] provides experimental data on the maximum amplitude with respect to the strain of the sheet. In the present chapter, the reference simulations are performed on uniform cubic meshes with 32×32 and 64×64 elements, respectively.

For the adaptive simulation, a THB spline mesh with initially 32×32 elements is used, and mesh adaptivity is activated after wrinkling initiation since the errors in the pre-wrinkling regime are small due to the lack of out-of-plane deformations of the sheet. The goal-functional is a displacement-based functional on the z-component, i.e., $\mathscr{L}(\boldsymbol{u}) = \int_\Omega \boldsymbol{u} \cdot \boldsymbol{e}_z \,\mathrm{d}\Omega$. The tolerances for refinement and coarsening are $\text{tol}_r = 10^{-14}$ and $\text{tol}_c = 10^{-10}$, respectively, and they are chosen based on the error envelope of the uniform refinement. The adaptive meshing parameters are chosen as $(\rho_r, \rho_c) = (0.5, 0.005)$. These parameters are chosen based on the behaviour of the global error in the first load steps after bifurcation. Contrary to the previous example in section 5.6.5, there are no refinement iterations performed within the load step. This means that the refinement and coarsening operations are performed after the load step based on the magnitude of the error compared to the tolerances. Furthermore, when the error is below a tolerance of $\rho_{c,\min}$, the initial mesh is used again. This is done to prevent further coarsening after re-stabilisation of the wrinkles, i.e., the moment when the wrinkles have disappeared. For the refinement algorithm, the maximum depth of the THB grid is fixed to 11 levels, and the jump parameter is set to $m = 2$. The wrinkling simulation is performed using the Crisfield arc-length method [124] with a quadratic procedure to compute the mode shape at the bifurcation [633]. This procedure is further described in [583].

In figure 5.22, the results for an adaptive isogeometric wrinkling simulation are provided. The top figure provides the normalised wrinkling amplitude with respect to the



Geometry		
L	280	[mm]
W	140	[mm]
t	0.14	[mm]
Material *Incompressible Mooney–Rivlin*		
c_1	$3.13 \cdot 10^5$	[Pa]
c_2	$1.24 \cdot 10^5$	[Pa]
Boundary Conditions		
$\boldsymbol{u}$	Free	At Γ_1
$u_y = u_z = \frac{\partial u_z}{\partial x}$	0	At Γ_2
$u_y = \frac{\partial u_x}{\partial y} = \frac{\partial u_z}{\partial y}$	0	At Γ_3
$u_x = \frac{\partial u_x}{\partial x} = \frac{\partial u_z}{\partial x}$	0	At Γ_4
Loads		
P	Variable	

Figure 5.21: Geometry and parameters for the wrinkling problem. The boundary Γ_1 is free, the boundary Γ_2 is fixed in the y and z-directions, and rotations around the y-axis are fixed. The boundaries Γ_3 and Γ_4 have symmetry conditions applied.

strain of the sheet (ϵ), compared to the IGA and ANSYS SHELL181 element reference results from [587] as well as the experimental results from [427]. The figure in the middle provides the error estimate in terms of the goal functional $\mathscr{L}$ with respect to the strain of the sheet, and the bottom figure provides the number of degrees of freedom with respect to the strain of the sheet. Firstly, the results from both uniform meshes show that the error estimate $\Delta\mathscr{L}$ is close to zero when the wrinkles initiate, since the sheet is perfectly flat. As soon as the wrinkles form, the error estimate becomes non-zero, and it peaks at the moment of re-stabilisation (i.e., the moment when the amplitude vanishes). After re-stabilisation, the error estimate is low but slightly higher than before wrinkling, probably because the sheet is not numerically flat.

The adaptive meshing simulations show that even with zero inner iterations for mesh adaptivity, the adaptive mesh provides accurate results for a relatively small number of degrees of freedom. Just after wrinkling initiates, the adaptive meshing error peaks due to the coarsening of a large number of elements (as can be seen by the drop of degrees of freedom in the bottom figure). However, the mesh adaptively refines until it reaches the grey region in the middle figure, where combined refinement and coarsening imply that the error balances around tol_c, i.e., the upper bound of the marked region. Towards the end of the wrinkling phase, the error increases for the uniformly refined mesh, explaining the increase in the number of degrees of freedom for the adaptive mesh. Nevertheless, the adaptive mesh provides more accurate results compared to the 64 × 64 uniform mesh, which has fewer degrees of freedom.

In figure 5.23, a selection of meshes from both adaptive simulations are provided,

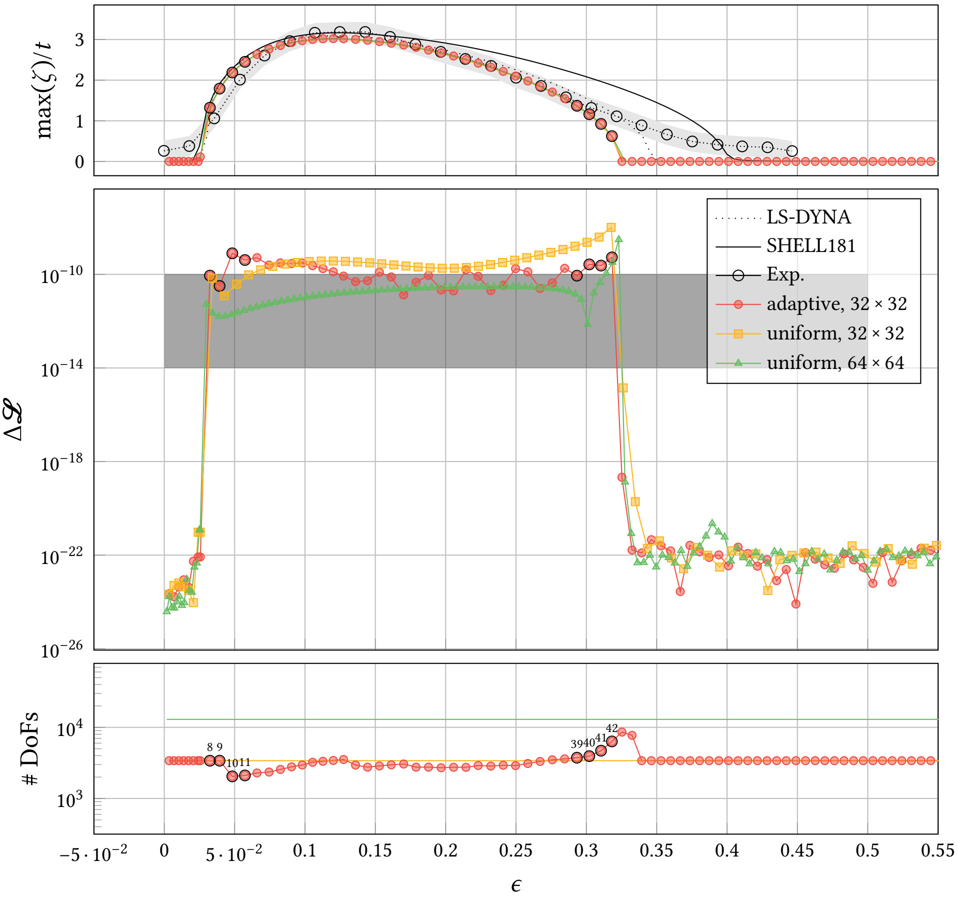


Figure 5.22: The non-dimensional maximal wrinkling amplitude $\max(\zeta)/t$ (top), the goal functional error $\Delta\mathscr{L}$ (mid), and the number of degrees of freedom of the computational mesh (bottom) with respect to the strain of the sheet ϵ. The markers with a black border represent the points at which the meshes are provided in figure 5.23. The coloured lines are the solutions obtained by the present model with a Mooney-Rivlin material model with uniform or adaptive meshes. For the adaptive simulation, the parameters $(\rho_r, \rho_c) = (0.5, 0.005)$ are used. The solid line in the top figure is a SHELL181 result obtained using ANSYS; the dotted line in the top figure is a result obtained using the fully integrated shell in LS-DYNA; and the dashed line with markers represents experimental data obtained by [427] (Ref.).

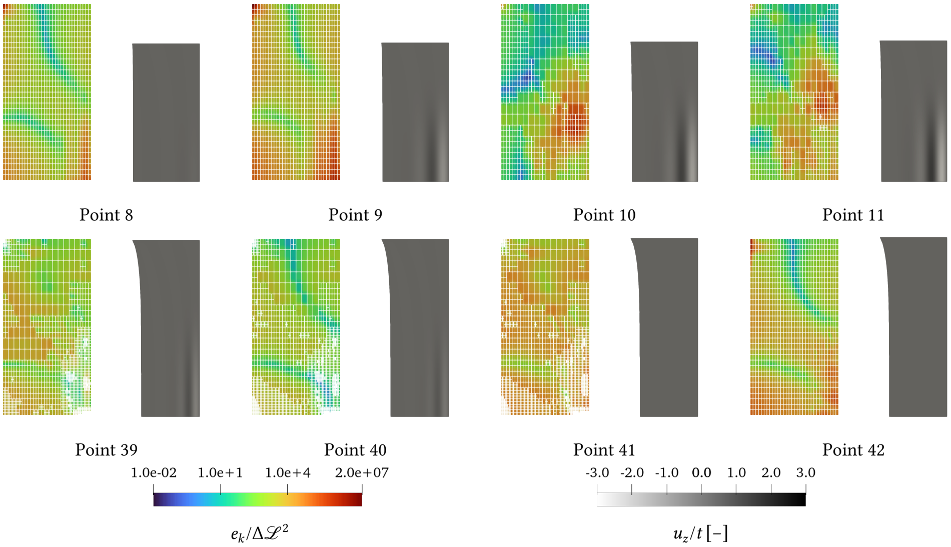


Figure 5.23: Normalised element errors (left) and the normalised wrinkling amplitude (right) for the wrinkling benchmark plotted on the undeformed geometry with the corresponding elements.

specifically for the wrinkling initiation and re-stabilisation points. For a complete overview of the mesh in each load step, the reader is referred to Video 2 in the supplementary material of [586]. The evolution of the meshes shows that the mesh elements concentrate around the wrinkles (bottom-right) and in the top-left corner, which represents the corner between the clamped edge (top) and the free edge. The former is expected since the employed goal functional is based on out-of-plane deformations. However, the fact that the mesh concentrates around the top-left corner is non-intuitive, but it shows that this area is important to reduce the global error in terms of the out-of-plane deformations. Furthermore, it can be seen that around the re-stabilisation of the wrinkles, the mesh concentrates around the bottom-left corner, indicating that this corner is of importance in accurately modelling the wrinkling amplitudes in the whole domain.

Concluding, the wrinkling benchmark shows the potential of mesh adaptivity for such applications. With fewer degrees of freedom, the THB-spline mesh is able to approximate the solution around a pre-defined error, even though the selection of the refinement and coarsening parameters and the tolerances have not been optimised in this study.

5.7 Conclusions

This chapter presents an adaptive method for isogeometric Kirchhoff–Love shells. The main contributions of this chapter are a goal-adaptive error estimator for isogeometric Kirchhoff–Love shells using the Dual-Weighted Residual method and a slightly modified suitably graded refinement scheme taking into account refined elements in the definition of the coarsening neighbourhood.

Using the Dual-Weighted Residual method and given a pre-defined goal functional (e.g., the second-principal stress integrated over the domain), an estimator for the error in terms of this goal functional can be defined. The adjoint problem that needs to be solved on the original mesh and on a nested degree-elevated ('enriched') mesh has been defined for the isogeometric Kirchhoff–Love shell. In addition, the operators for modal and linear buckling analysis have been derived, implying an additional generalised eigenvalue problem to be solved on the enriched mesh. For suitable grading, the works of [67, 78, 79, 84] have been closely followed. In order to be able to refine and coarsen in the same iteration, refined elements have been added to the original definition of the coarsening neighborhood.

To assess the proposed adaptive isogeometric method for Kirchhoff–Love shells, a few numerical benchmark problems have been evaluated. Linear static analysis, provided an analytical solution, has been used to evaluate the DWR error estimators. The eigenvalue problems for modal and buckling analyses have been evaluated on, respectively, the problems of circular plate vibration and square plate buckling. Based on the linear, modal, and buckling analyses with analytical solutions, it can be concluded that the DWR estimator for Kirchhoff–Love shells can be used with several goal functionals and in several applications, as it provides high accuracy with respect to the exact errors.

Using the problem of a pinched membrane, the error estimator has been used to adaptively refine a mesh in a non-linear example. From this example, it can be concluded that, despite the challenging non-linear problem, high accuracy per DoF can be obtained compared to uniformly refined meshes based on different goal functionals. Lastly, the adaptive isogeometric method from the present chapter has been evaluated in solution-stepping problems for structural instabilities. Firstly, the method was applied to the limit-point instability problem of the collapse of a cylindrical roof. Here, inner adaptivity iterations were performed for each load step until the error was located at a desired interval. Again, the present method provides high accuracy per degree of freedom. In addition, it was shown that the method is indeed able to provide adaptive meshes with respect to a pre-defined interval for a given goal functional. The last benchmark problem involves the tension-wrinkling bifurcation instability of a thin membrane. In this benchmark problem, adaptive meshing has been applied in the post-buckling regime based on wrinkling amplitudes. Here, no adaptivity iterations within the load steps were performed, showing that the method is still able to provide good adaptivity with respect to the pre-defined tolerances. Also, the results of the wrinkling error estimators show that the error peaks in the re-stabilisation phase, where the highest deviation with experimental results is observed.

Future developments for the present method include the application on multi-patch domains, both coupled with penalty methods as well as with globally continuous bases as presented in [182] to handle more complex geometries. Furthermore, structural dynamics have been left out of the scope of this chapter, since the DWR method for dynamic problems requires backwards-in-time evaluation of the adjoint problem, which is ideally combined with parallel-in-time methods like ParaReal or MGRIT [180]. Lastly, future work can be done on the (adaptive) determination of the adaptive meshing parameters. On the one hand, one can apply the present method to real-world engineering applications, bringing realistic goal functionals and margins into the adaptivity algorithm. On the other hand, advanced schemes for triggering pure or combined refinement or coarsening, together with their parameters, can be further investigated.

5.A Result Reproduction

For the sake of reproducibility of the results in this chapter, this appendix provides brief instructions on the use of the software developed along with this thesis. The full software is available as part of the Geometry + Simulation Modules. For more detail on the contributions to this software library, and its installation, the reader is referred to chapter 8.

Table 5.2 provides per figure in this chapter the name of the file to run along with the arguments to be passed to obtain these figures.

Table 5.2: File name and run arguments required for the reproducibility of the figures in this chapter. Arguments with a single dash (`-`) require an argument while double-dashed arguments (`--`) are switches. See chapter 8 for more detail about the software and installation instructions. The XML files with the mesher options are in `gsStructuralAnalysis/filedata/options/`.

Figure	**Run File**		
	Arg.	**Description**	**Values**
Figure 5.7	`example_shell3D_DWR`		
	`-e`	Number of degree elevation steps	`1`: $p = 2$, `2`: $p = 3$
	`-r`	Number of uniform refinements	`6`
	`-g`, `-c`	Goal functional (`-g`)	`1,9`: $\int_\Omega \|\boldsymbol{u}\| \,\mathrm{d}\Omega$, `2,1`: $\int_\Omega \lambda_2 \,\mathrm{d}\Omega$
		and its component (`-c`)	`3,9`: $\int_\Omega \|\boldsymbol{\varepsilon}(\boldsymbol{u}) \,\mathrm{d}\Omega$
			`7,0`: $\int_\Omega \|\mathbf{N}\varepsilon(\boldsymbol{u}) \cdot \boldsymbol{e}_1 \,\mathrm{d}\Omega$
Figure 5.9	`example_shell3D_DWR_modal`		
	`-e`	Number of degree elevation steps	`1`: $p = 2$, `2`: $p = 3$, `3`: $p = 4$
	`-r`	Number of uniform refinements	`7`
	`-i`	Mode index	`0`: Mode 1, `1`: Mode 2, `3`: Mode 3, `5`: Mode 4
Figure 5.11	`example_shell3D_DWR_buckling`		
	`-e`,`-r`	See `example_shell3D_DWR_modal`	
	`-i`	Mode index	`0`: Mode 1, `1`: Mode 2, `3`: Mode 3, `4`: Mode 4
Figures 5.13 to 5.16	`example_PinchedMembrane_DWR`		
	`-r`	Number of adaptive refinements	`15`
		Number of uniform refinements	`6`
	`-g`, `-C`	Goal functional (`-g`)	`1,9`: Displacement norm,
		and its component (`-C`)	`6,2`: Second principal stress
	`-A`	Adaptive refinement	`0`: Uniform, `1`: Adaptive
	`-O`	Options file (for adaptive meshing)	
		`shell_mesher_options_Membrane.xml`, `shell_mesher_options_Membrane_max8.xml`	
	`--loop`	Perform refinement loops	
Figure 5.18 Figure 5.20	`benchmark_Roof_DWR`		
	`-T`	Target error	`1e-9`
	`-B`	Error bandwidth	`0.1`
	`-g`, `-C`	Goal functional (`-g`)	`9,9`: Bending moment norm
		and its component (`-C`)	
	`-O`	Adaptive meshing options	`shell_mesher_options_Roof.xml`
	`--adaptMesh`	Perform refinement loops	
Figure 5.22 Figure 5.23	`benchmark_Wrinkling_DWR`		
	`-r`	Number of initial uniform refinements	`5`: 32 × 32, `6`: 64 × 64,
	`-T`	Target error	`1e-12`
	`-B`	Error lower-bound multiplier	`0.1`
	`-D`	Coasening error limit	`1e-19`
	`-g`, `-C`	Goal functional (`-g`)	`1,2`: x-displacement
		and its component (`-C`)	
	`-O`	Adaptive meshing options	`shell_mesher_options_Wrinkling.xml`
	`--adaptMesh`	Perform refinement loops	

6

An Adaptive Parallel Arc-Length Method

Where the previous chapter, chapter 5, presents a novel adaptive method for spatial adaptivity, this chapter focuses on the quasi-static domain. As discussed in the preliminaries of this dissertation, in particular in section 2.4, arc-length methods play an important role in quasi-static post-buckling analysis. However, arc-length methods are serial processes, hence not benefiting from the increasing development in parallelisation of today's scientific computing facilities. In this chapter, a novel parallelisation of arc-length methods is presented, where the parallelisation is independent of the number of solution branches. The method is inspired by parallel time-integration methods, where solution intervals are computed and corrected by a refined interval in parallel afterwards. When an interval is computed with satisfactory accuracy, no more corrections are performed. This method, referred to as the Adaptive Parallel Arc-Length Method (APALM) is adaptive by design to facilitate its parallelism, and the concept is easily extended for bifurcation problems. The performance of the method is demonstrated using isogeometric Kirchhoff–Love shells on problems with snap-through and pitchfork instabilities and applied to analyse snapping instabilities in a snapping meta-material. Computational time is compared with serial implementations as well as an Adaptive Serial-Parallel Arc-Length Method (ASPALM) to show the scaling of the parallelisation. The results show that the APALM is able to provide solutions in the same computational time as an ordinary serial arc-length method without inherent adaptivity, i.e., the APALM provides more detail at the same time. Furthermore, the APALM shows to be robust in challenging snapping and bifurcation problems considered in this chapter. If necessary, the reader is referred to section 2.4.4 for background information on arc length methods.

This chapter under revision for the journal *Computers & Structures*. A pre-print is available:
[584] **H. M. Verhelst**, J. H. Den Besten & M. Möller, “An Adaptive Parallel Arc-Length Method”, arXiv:2303.01075 (2023)

6.1 Introduction

Over the last decades, computational power has increased exponentially. In the last year, most improvements were due to an increasing number of threads per processing unit rather than an increase in single-thread performance [483]. The trend of increasing logical cores with stagnating single-threaded performance calls for parallelisation of existing codes to improve computational efficiency, amongst which numerical algorithms in computational mechanics. In the field of computational mechanics, parallelisation in the spatial domain is common practice by using shared-memory assembly routines or distributed-memory parallelisation using domain decomposition of meshes. Parallelisation can also be achieved in linear solvers or in the temporal domain using parallel-in-time solvers [196] in the case of dynamic analyses or using parallel continuation for quasi-static or continuation problems - the latter two being sequential by nature.

For quasi-static problems, continuation methods can be used when the solution of an equation or a system of equations is desired, given the varying parameters of the system. Such methods, typically referred to as Arc-Length Methods (ALMs), are widely used for (but not limited to) the analysis of the stability of structures. The Riks and Crisfield methods [124, 469] are commonly used and in combination with bifurcation algorithms [633], whereas ALMs provide a valuable tool in the analysis of the collapse and post-buckling behaviour of structures. Recent developments for ALMs include a new displacement-controlled formulation [445], an improved predictor scheme [295], and automatic exploration techniques [561, 630]. Like time-stepping methods, ALMs are sequential by nature, meaning that the solution at a point is obtained from the solution at a previous point obtained previously.

Amongst many parallel time-integration schemes, Parareal is a parallel time-integration method proposed by [354] and works with a two-level parallel correction scheme of time intervals. The method starts with a series of solutions obtained in serial with a large time step, after which each sub-interval is computed with a finer time step such that a new solution is found at the end-point of the time interval. A multi-level extension of Parareal is proposed in [180] and is referred to as Multi-Grid Reduced in Time (MGRIT). This method is similar to Parareal but applies the two-level approach recursively. As a consequence, multi-grid-like cycles can be used to correct previously computed sub-intervals. This method has not only been applied to dynamic problems but also to the training of neural networks [129] and [242]. Alternative methods for parallel time integration are reviewed in the work of [196].

Compared to temporal parallelisation methods, parallelisation of ALMs has received less attention in the academic community. As ALMs are typically used for explorations of solutions across branches, parallel evaluation of branches can be performed as soon as the starting point (and tangent) of each branch is known. The number of branches related to a problem, however, depends typically on the problem that is solved; hence, the parallel scalability of ALMs over branches is not guaranteed. Parallelisation within a branch is enabled by the Parallel Adaptive Method for Pseudo-Arclength Continuation (PAMPAC) [17]. This method works with multiple predictors (with different step sizes) and consequently correctors to select an optimal step size, which can be performed in parallel. The PAMPAC method focuses on selecting a maximal step size for the ALM for which the method does not converge.

In this chapter, a parallelisation of the arc-length method is presented that is independent of the physical nature of the underlying problem. That is, the method is developed such that the parallelisation can be performed within the branches. In addition to parallelisation, the presented arc-length scheme also provides inherent adaptivity; therefore, the method is referred to as the Adaptive Parallel Arc Length Method (APALM). The working principle of the APALM is based on a multi-level approach – inspired by MGRIT methods – where a coarse serial approximation of the solution space is refined in parallel until a measure of convergence is achieved. Contrary to PAMPAC, the present method does not maximise the step size for convergence of the ALM iterations, but instead the parallelisation is based on convergence of the solution sub-intervals. Without loss of generality, the method is developed given a constraint equation for the arc-length method; thus, it is generalised for the Riks and Crisfield methods, amongst other methods available.

The outline of this chapter is as follows: Section 6.2 provides a background on arc-length methods. In section 6.3, the parallelisation of arc-length methods is presented, referred to as the APALM. Thereafter, section 6.4 provides algorithms for non-intrusive implementation of the APALM, given an implementation of an existing ALM. Section 6.5 provides numerical benchmark problems and an application to the analysis of a snapping meta-material, inspired by [455]. Finally, section 6.6 provides conclusions on the presented method.

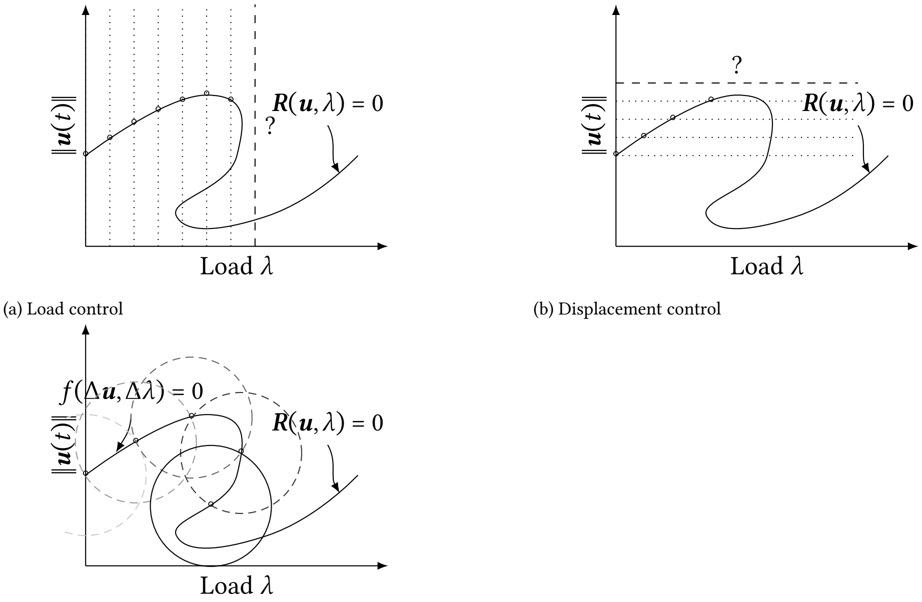
6.2 Arc-Length Methods

In this section, the concept of arc-length methods is presented for the sake of completeness. For a detailed overview, one can consult references [124, 456, 469, 472]. Let $\boldsymbol{R}(\boldsymbol{u},\lambda)=0$ be a non-linear system of equations to be solved, with $\boldsymbol{u}$ the solution to the system of equations given a parameter λ. For structural analyses, $\boldsymbol{u}$ is typically a vector containing discrete displacements of the degrees of freedom, and λ is a factor scaling the magnitude of an applied load P, i.e.

$$\boldsymbol{R}(\boldsymbol{u},\lambda)=\boldsymbol{N}(\boldsymbol{u})-\lambda\boldsymbol{P}, \tag{6.1}$$

where $\boldsymbol{N}(\boldsymbol{u})$ is a vector of internal forces, depending on the deformation $\boldsymbol{u}$. For incremental analyses, i.e., quasi-static analyses, a series of solutions $\boldsymbol{w}_i=(\boldsymbol{u}_i,\lambda_i)$ is obtained by computing increments $\Delta\boldsymbol{w}_i=(\Delta\boldsymbol{u}_i,\Delta\lambda_i)$ such that $\boldsymbol{w}_{i+1}=\boldsymbol{w}_i+\Delta\boldsymbol{w}_i$ and equation (6.1) is satisfied for $\boldsymbol{w}_{i+1}$. These solutions can be obtained by Newton iterations: i) fixing λ and finding $\boldsymbol{u}$ (*load control*); ii) fixing some degrees of freedom in $\boldsymbol{u}$ and finding all $\boldsymbol{u}$ and λ (*displacement control*); or iii) constraining λ and $\boldsymbol{u}$ and solving for both (*arc-length control*); see figure 6.1. In the case of arc-length control, the increment $\Delta\boldsymbol{w}$ is measured by an increment length $d(\Delta\boldsymbol{w})$

$$d(\Delta\boldsymbol{w})=\Delta\boldsymbol{u}^\top\Delta\boldsymbol{u}+\Psi^2\Delta\lambda^2\boldsymbol{P}^\top\boldsymbol{P}, \tag{6.2}$$


$\|\boldsymbol{u}(t)\|$
$\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$
?
Load λ
$\|\boldsymbol{u}(t)\|$
?
$\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$
Load λ
$f(\Delta\boldsymbol{u},\Delta\lambda) = 0$
$\|\boldsymbol{u}(t)\|$
$\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$
Load λ

(a) Load control

(b) Displacement control

(c) Arc-length control

Figure 6.1: Load (top left), displacement (top right), and arc-length control (bottom left) for structural analysis problems. The question mark (?) indicates the iteration where load and displacement control encounter a limit point. In these situations, the next point obtained is typically difficult to find.

where Ψ is a scaling parameter given in [42, 497]. The increment $\Delta\boldsymbol{w}$ is constrained by the arc-length $\Delta\ell$ in the constraint equation

$$f(\Delta\boldsymbol{w}) = d(\Delta\boldsymbol{w}) - \Delta\ell = 0. \tag{6.3}$$

Since $\boldsymbol{G}(\boldsymbol{u},\lambda)$ is non-linear, the increment $\Delta\boldsymbol{w}_i$ is obtained iteratively, i.e., $\Delta\boldsymbol{w}_{i,k+1} = \Delta\boldsymbol{w}_{i,k} + \delta\boldsymbol{w}_i$ with iteration count k. The constraint equation is solved together with equation (6.1) in every iteration, yielding the Riks and Crisfield methods [124, 469]

$$f(\Delta\boldsymbol{w}_{i,k},\Delta l) = \Delta\boldsymbol{u}_{i,0}^{\top}\Delta\boldsymbol{u}_{i,k} + \Psi^2\Delta\lambda_{i,0}\Delta\lambda_{i,0}\boldsymbol{P}^{\top}\boldsymbol{P} - \Delta\ell^2 = 0, \qquad \text{Riks,} \tag{6.4}$$

$$f(\Delta\boldsymbol{w}_{i,k},\Delta l) = \Delta\boldsymbol{u}_{i,k}^{\top}\Delta\boldsymbol{u}_{i,k} + \Psi^2\Delta\lambda_{i,k}\Delta\lambda_{i,k}\boldsymbol{P}^{\top}\boldsymbol{P} - \Delta\ell^2 = 0, \qquad \text{Crisfield,} \tag{6.5}$$

where $\Delta\boldsymbol{w}_0$ is the increment in the first iteration. The Crisfield method generally performs well with sharp snap-backs but has the disadvantage that the constraint equation has two intersections with the path formed by equation (6.1). Hence, a root has to be selected, which is elaborated in the works [124, 472]. When multiple intersections are found, complex roots are found [85], which can be resolved using one of the methods proposed in [329, 660]. It should be noted that any other arc-length method can be used within the scheme proposed in this chapter, as long as the constraint equation is satisfied when the arc-length step is converged.

6.3 Adaptive Parallel Arc-Length Method

In this section, our new approach, the APALM, is presented. Firstly, the method is conceptualised along with some illustrative figures (section 6.3.1). Secondly, details are provided on the curve parametrisation and the measurement of errors (section 6.3.2). Lastly, section 6.3.3 presents (re-)parametrisation methods for the solution curve. These parametrisations will be essential to the data structure of the APALM. It should be noted that the method described in this section is presented only for one continuation parameter, λ.

6.3.1 Concept

Learning from parallel-in-time methods like Parareal or MGRIT, parallelisation in the APALM is achieved from a subdivision of the *curve length domain*. Contrary to MGRIT and Parareal, where the temporal domain $t \in [T_0, T_1]$ is fixed, the APALM will work with a changing *curve length domain* $s \in [S_0, S_1]$ depending on the length of the traversed path, with an underlying fixed parametric domain with parametric coordinate $\xi \in [0,1]$. The APALM is initialised with an initial coarse grid approximation, in which the parametric and the curve length domains are subdivided into sub-domains $\xi \in [\xi_i, \xi_{i+1}]$ and $s \in [s_i, s_{i+1}]$, respectively, as illustrated in figure 6.2.

In the initialisation phase of the APALM, the first subdivision into sub-intervals is made (see figure 6.2a). Here, the sizes of the sub-intervals $s \in [s_i^\ell, s_{i+1}^\ell]$ are determined based on the distance measure that is used by the corresponding ALM; see equation (6.2). Note that the superscript ℓ denotes the ℓ^{th} level. Based on the initial curve-length domain $s \in [0, S]$, where S is the total length of the initial curve, and the corresponding sub-intervals, the curve-length domain can be mapped accordingly onto a parametric domain; see section 6.4 for more details.

With an initialised computational domain, the number of sub-intervals determines the degree of parallelisation. On any sub-interval, $[s_i^\ell, s_{i+1}^\ell]$ data at the start-point and end-point is known, which can be used to initiate an arc-length method to re-compute the sub-interval with N increments, i.e., with an arc-length of $\Delta L_i^{\ell+1} = \Delta L_i^\ell / N$ (see figure 6.2b).

After sub-interval $[s_i^{\ell+1}, s_{i+1}^{\ell+1}]$ has been finished, the distance of the end-point of the sub-interval can be compared to the previously known solution at s_{i+1}^ℓ, which is called *parallel verification of intervals* in figure 6.2c. Since the sub-interval is traversed in N increments with length $\Delta s_i^0 / N$, the triangle inequality with the arc-length measure implies that there must be a distance greater than or equal to zero between the newly found end-point and the reference end-point. The more 'curved' the domain in-between, the larger this distance. Based on an error measure (see section 6.3.2), intervals with a relatively large deviation between the coarse-level arc length and the fine-level arc length are to be marked for 'refinement'.

Lastly, the intervals with a too large deviation in the newly computed curve length need to be reparameterized (see section 6.3.3). This is because the total curve-length parametrisation is elongated exactly by the distance between the newly computed end-point and the previously known point. By this means, the reference interval is subdivided into $N + 1$ sub-intervals, and the data corresponding to the N newly computed point is stored. For sub-intervals that have an error below the tolerance, only $N - 1$ points are stored as references, and no reparametrisation takes place. The process is sketched in figure 6.2d. After reparametrisation, the marked interval can be re-computed, and the process can be repeated from figure 6.2b onwards.

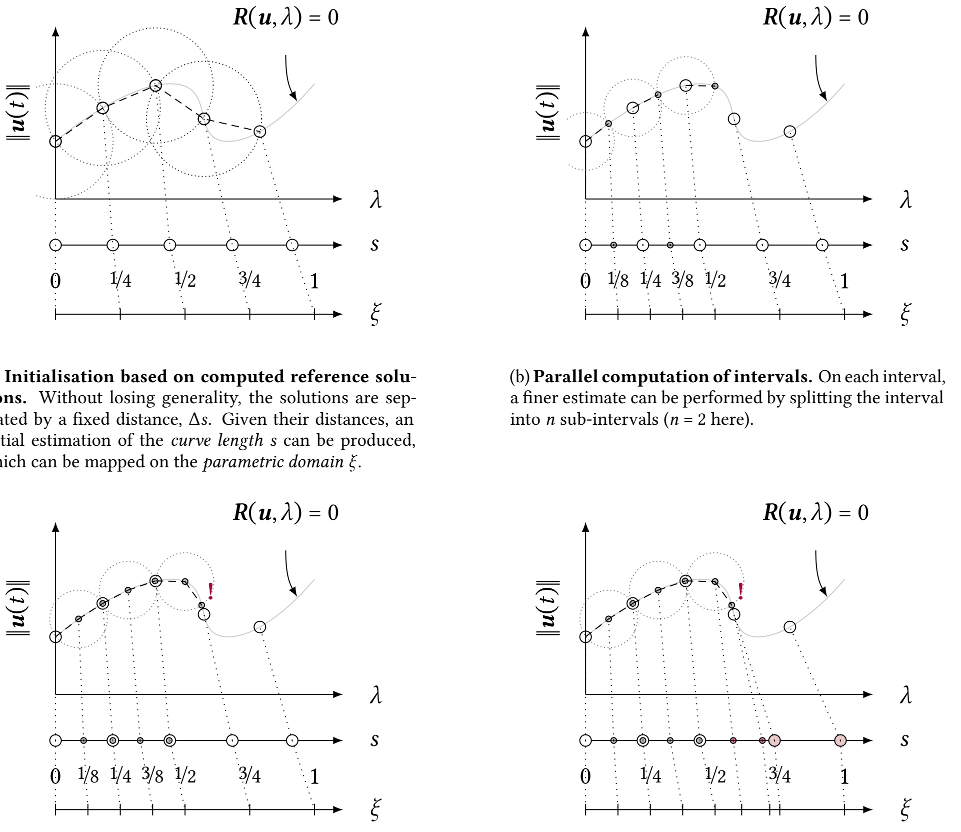
Remark 6.3.1 (Difference with parallel-in-time methods)**.** *As mentioned, the multi-level approach that is employed in this method is derived from the idea of parallel-in-time methods. However, the fundamental difference between time integration and continuation comes from the fact that time integration methods typically compute the solution on the next time step with a certain time integration error $\mathcal{O}(\Delta t^p)$. Parallel-in-time methods rely on this time integration error to mark solution intervals as converged or not, and additionally, updated solutions contain smaller time integration errors, so sub-intervals need to be recomputed as soon as solutions previously in time have been updated.*

For arc-length methods, Newton's method is applied to a system of equations that solves the arc-length constraint equation together with the discretized system $\mathbf{G}(\boldsymbol{u}, \lambda) = 0$. Therefore, the error of the solution that is found after an arc-length increment is independent of the arc-length increment size but solely depends on the convergence tolerance of Newton's method. Therefore, the end-point of an interval does not have to be updated, nor do intervals after the update be recomputed. This implies, in principle, that parallel corrections of the arc-length steps are not needed, since the intervals already capture the structural response at the equilibrium path. However, the parallel corrections are still meaningful to capture the equilibrium path in desired detail, in the case where the initial step size is chosen very coarse. As will follow from the results section, cf. section 6.5, the parallel performance increases for fewer (hence coarser) initial intervals.

Remark 6.3.2 (Path-dependency)**.** *The concept presented in this chapter assumes path-independence of the equation to be solved, in order to assume that from a given starting point on the equilibrium path, the same end-point could be reached irrespective of the computed intervals on the path in-between the points. Path-dependent problems are out of the scope of this paper.*

6.3.2 Error Measures

The refinement of computed sub-intervals depends on the distances between the points in the original (coarse) interval and the newly obtained solutions in this sub-interval. Here, error measures are presented, that can be used to mark an interval $[s_i^\ell, s_{i+1}^\ell]$ based on the obtained solutions $\{s_k^{\ell+1}\}_{k=0,\dots,N}$ at the finer level. Figure 6.3 presents two possible situations: a nearly straight interval that would not be marked for refinement, and a curved interval that would be marked for refinement. Here, the interval is considered 'curved' in the discrete solution space if the hyperdimensional path between two solutions differs from the hyperplane between these solutions. The errors that determine the marking of an interval for refinement are illustrated in figure 6.3b and can be interpreted as follows: ΔL is the original arc length between two coarse solutions, $\Delta L'$ is the newly obtained length between two coarse solutions, the *lower distance* $\overline{\Delta L}$ is the distance between the start of



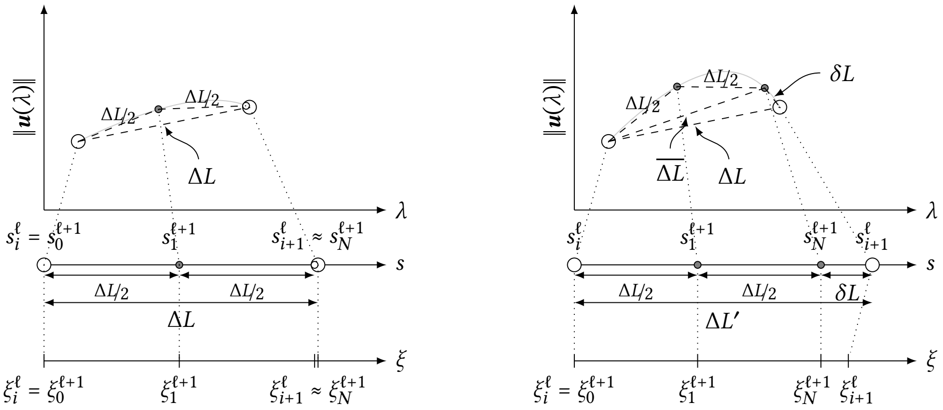
(a) **Initialisation based on computed reference solutions.** Without losing generality, the solutions are separated by a fixed distance, Δs. Given their distances, an initial estimation of the *curve length* s can be produced, which can be mapped on the *parametric domain* ξ.

(b) **Parallel computation of intervals.** On each interval, a finer estimate can be performed by splitting the interval into n sub-intervals ($n = 2$ here).

(c) **Parallel verification of intervals.** When the last sub-interval is computed, the solution is verified with the next known reference solution (here, the solutions following from the initial simulations). When the distance is sufficiently small, the segment can be marked as convergent, and the in-between solutions on the interval can be written. The point where the difference is too large is denoted by !.

(d) **Curve-length reparametrisation.** For sub-intervals where the length deviates too much from the previous length, i.e., where the distance between the last computed point and the reference is too large (denoted by !), all solutions of the sub-interval are added to the parametrisation, and the known data points ahead of the newly computed points are shifted in the curve parametric coordinates.

Figure 6.2: Concept of the APALM. The large open circles represent *reference solutions* from a previously computed level. The small solid circles represent *new data* on the interval between two reference solutions, computed by the arc-length method (here the large dashed circle). The black dashed line indicates the curve estimation for which the sum is equal to the total curve length.



(a) Nearly straight interval where the small white dot is sufficiently close to the coarse reference. In this case, the solution of the second increment is not added to the parametrisation.

(b) Curved interval where the solution of the second increment is ΔL_u away from the coarse reference. In this case, the distance between the coarse reference points δL is smaller than the actually traversed distance $\Delta L'$.

Figure 6.3: Error measures on a nearly straight interval (a) and a curved interval (b). For the nearly straight interval, the distance δL (see b) is sufficiently small, whereas for the curved interval, it is too big. The measures ΔL and $\Delta L'$, $\overline{\Delta L}$, and δL are, respectively, the coarse arc length, the fine arc length, the lower distance, and the absolute error.

the interval and the last solution at the fine level, and δL is the distance between the last obtained solution on the fine level and the final point on the coarse level. Using these distances, the *total error* (ε), the *lower error* (ε_l), and the *upper error* (ε_u) can be defined:

$$\varepsilon = (\Delta L' - \Delta L)/\Delta L, \qquad \text{total error,} \tag{6.6}$$

$$\varepsilon_l = (\Delta L - \overline{\Delta L})/\Delta L, \qquad \text{lower error,} \tag{6.7}$$

$$\varepsilon_u = (\varepsilon - \varepsilon_l)/\Delta L, \qquad \text{upper error.} \tag{6.8}$$

Here, the total error is the total difference between the coarse and fine intervals; the lower error is the contribution of the first N sub-intervals; and the upper error is the contribution of δL to the total error. Depending on these errors and specified tolerances, refinement rules can be set up, in particular:

$$\text{Refine the first } N \text{ intervals} \longleftrightarrow \varepsilon_l > \mathrm{TOL}_l, \tag{6.9}$$

$$\text{Refine the last interval} \longleftrightarrow \varepsilon_u > \mathrm{TOL}_u. \tag{6.10}$$

6.3.3 Curve (Re-)Parametrisation

As indicated in figure 6.2, the concept of the APALM is supported by the parametrisation of the solutions of $\boldsymbol{R}(\boldsymbol{u},\lambda) = 0$ by parameterizing the curve length using the increment length $d(\Delta\boldsymbol{w})$ embedded in the arc-length method. As illustrated in figure 6.2, the APALM maps solutions $\boldsymbol{w}$ to a point on the curve-length domain $[0,S]$, and points on the curve-length domain are mapped on a parametric domain $[0,1]$.

Provided a series of solutions from the initialisation phase $\{\boldsymbol{w}_i^0\}_{i=0,\dots,I}$, with I denoting the total number of initial points, and defining solution intervals by $\Delta\boldsymbol{w}_i^\ell = \boldsymbol{w}_{i+1}^\ell - \boldsymbol{w}_i^\ell$, each solution $\boldsymbol{w}_i^0$ can recursively be assigned to the curve-length and parametric domains by

$$s_{i+1}^0 = s_i^0 + d(\Delta\boldsymbol{w}_i^0), \qquad i = 1,\dots,I-1, \quad s_0 = 0, \tag{6.11}$$

$$\xi_i^0 = \frac{s_i}{s_I}, \qquad i = 0,\dots,I, \tag{6.12}$$

where the superscript 0 represents the 0th level. In addition, equation (6.11) guarantees that $S = s_I$ marks the total length of the curve that has been traversed, measured by the distance between each solution. Given the curve-length coordinates of each point as an increasing sequence, the parametric domain can simply be obtained by scaling the domain back to $[0,1]$; see equation (6.12). In the following, two ways of adding solutions to the parametrisation are defined: i) interior insertion, and ii) full insertion and stretching. The operations are defined given a parent interval $[s_i^\ell, s_{i+1}^\ell)$ in which a set of new solutions $\{s_k^{\ell+1}\}_{k=0,\dots,N}$, where $s_0^{\ell+1} = s_i^\ell$, is computed, with N the total number of points in the interval; see figure 6.4a.

Firstly, the interior insertion operation inserts solutions *within* the sub-interval, see figure 6.4b, and is later used for intervals where the error is small. The idea behind this operation is that the solutions $\{s_k^{\ell+1}\}_{k=1,\dots,N-1}$ between s_i^ℓ and s_{i+1}^ℓ are inserted and that the solution $s_N^{\ell+1}$ is not added to the map. In the case of the interior insertion, the points $s_k^{\ell+1}$ and their parametric coordinates $\xi_k^{\ell+1}$ are added by:

$$s_{k+1}^{\ell+1} = s_k^{\ell+1} + d(\Delta\boldsymbol{w}_k^{\ell+1}), \qquad k = 0,\dots,N-2, \quad s_0^{\ell+1} = s_i^\ell, \quad s_N^{\ell+1} = s_{i+1}^\ell, \tag{6.13}$$

$$\xi_{k+1}^{\ell+1} = \xi_k^{\ell+1} + (\xi_{i+1}^\ell - \xi_i^\ell)\frac{s_{k+1}^{\ell+1} - s_k^{\ell+1}}{s_{i+1}^\ell - s_i^\ell}, \qquad j = 0,\dots,N-2. \tag{6.14}$$

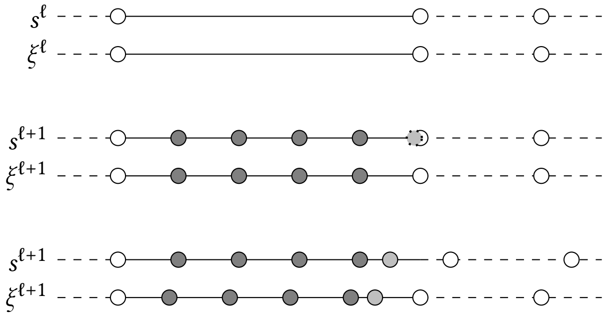
Note that $\boldsymbol{w}_k^{\ell+1}$ denotes the kth solution on level $\ell+1$ on the computed sub-interval, here $[s_i^\ell, s_{i+1}^\ell]$.

The full insertion and stretching operation inserts the solutions of the sub-interval, including its end point, and also stretches the curve parametrisation (see figure 6.4c), which is later used for intervals where the error is large, hence intervals that need refinement. The idea behind this operation is that the solutions $\{s_k^{\ell+1}\}_{k=1,\dots,N-1}$ between s_i^ℓ and s_{i+1}^ℓ are inserted and that the point s_{i+1}^ℓ is shifted such that $s_{i+1}^\ell = s_N^{\ell+1}$ and such that all points further than s_{i+1}^ℓ are updated to $\tilde{s}_j^\ell$ by the a shift using the distance between the last computed solution and the reference solution, i.e. $d(\Delta\boldsymbol{w}_N^{\ell+1})$, $\Delta\boldsymbol{w}_N^{\ell+1} = \boldsymbol{w}_{i+1}^\ell - \boldsymbol{w}_N^{\ell+1}$:

$$s_{k+1}^{\ell+1} = s_k^{\ell+1} + d(\Delta\boldsymbol{w}_k^{\ell+1}), \qquad k = 0,\dots,N-1, \quad s_0^{\ell+1} = s_i^\ell, \quad s_{i+1}^\ell = s_N^{\ell+1}, \tag{6.15}$$

$$\xi_{k+1}^{\ell+1} = \xi_k^{\ell+1} + (\xi_{i+1}^\ell - \xi_i^\ell)\frac{s_{k+1}^{\ell+1} - s_k^{\ell+1}}{s_{i+1}^\ell - s_i^\ell}, \qquad k = 0,\dots,N-1, \tag{6.16}$$

$$\tilde{s}_j^\ell = s_j^\ell + d(\boldsymbol{w}_N^{\ell+1}, \boldsymbol{w}_{i+1}^\ell), \qquad j = i+1,\dots,I. \tag{6.17}$$



(a) Original domain

(b) Interior insertion

(c) Full insertion and stretching

Figure 6.4: Domain parametrisations on the curve-length domain s and the parameter domain ξ with levels ℓ and $\ell+1$. Figure 6.4a illustrates the original domain, figure 6.4b illustrates the insertion of interior points in the case of a sufficiently close approximation of the end-point of the domain and figure 6.4c illustrates the full insertion of all sub-domain solutions combined with the stretching of the curve length domain.

As can be noticed in equation (6.16), the re-scaling of ξ is done using the parametric length of the original interval at level ℓ, $(\xi^\ell_{i+1} - \xi^\ell_i)$ and the curve coordinate s_{i+1} relative to the beging point of the interval s_i with respect to the (updated) total curve length of the interval $s_{i+1} - s_i$, which is similar to the well-known *chord-length parametrisation* in splines [437].

6.4 Implementation

In this section, data structures and algorithms for the implementation of the APALM are presented. In section 6.4.1, a data structure is provided for the implementation of the APALM. Thereafter, section 6.4.2 provides algorithms for the implementation of the APALM, and section 6.4.3 elaborates on the extension of these methods to multiple branches, hence enabling arc-length exploration.

6.4.1 Data Structure

Since the APALM is based on a sub-interval approach where the start and end points of each sub-interval are known, a data structure referencing the sub-intervals is essential. Since the curve-length coordinate is subject to change after reparametrisation of the curve and since the curve parameter is fixed, the logical choice is to connect the data to parametric coordinates. That is, a series of discrete maps is constructed such that solutions, levels, and curve-length coordinates can be obtained via a parametric point ξ^ℓ_k.

Figure 6.5 shows the data structure behind the APALM. Firstly, the data structure contains the map $\mathcal{S}(\xi) : [0,1] \longrightarrow [0,S]$, which is the map that maps the parametric coordinate to the curve-length domain. Secondly, the maps $\mathcal{U}(\xi) : [0,1] \longrightarrow \mathbb{R}^{n+1}$ and $\mathcal{U}'(\xi) : [0,1] \longrightarrow \mathbb{R}^{n+1}$ map the solution and the previous solution from the parametric domain to the solution and previous solution domain, respectively. The mapper $\mathcal{U}'(\xi) : [0,1] \longrightarrow \mathbb{R}^{n+1}$ is constructed in order to construct the predictor of the ALM. Lastly, the map $\mathscr{L}(\xi) : [0,1] \longrightarrow \mathbb{N}$

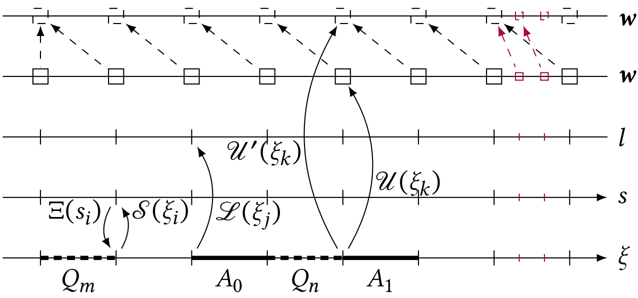


Figure 6.5: The data structure behind the APALM. The axes represent data sets, which are monotonically increasing when the axis has an arrow. Solid arrows represent mappers from one axis to another, and dashed arrows represent data references. The mappers $\Xi(s_i)$ and $\mathscr{S}(\xi_i)$ map between the curve parametrisation and the curve length axes. The former takes a curve length s_i and returns the curve parameter ξ_i, and the latter maps the inverse. The mappers $\mathscr{U}(\xi_k)$ and $\mathscr{U}'(\xi_k)$ return the solution $\boldsymbol{w}_j$ and the previous solution $(\boldsymbol{w}')_j$, respectively, given a parametric coordinate ξ_j, and the mapper $\mathscr{L}(\xi_j)$ returns the level on which the coordinate ξ_j was computed. The guess is a data reference to the previous solution. The thick solid intervals represent running jobs assigned with an ID, and the thick dashed intervals represent queued intervals. Each interval is represented by a start-point and an end-point tuple (ξ_l, ξ_{l+1}). The red lines, squares, and arrows represent the submit operation when solutions are added to the data structure.

is a map that can be used to obtain the level of a parametric point, which is optional but can be useful to limit the method to a certain depth.

6.4.2 Algorithms

Given the underlying data structure of the APALM (see section 6.4.1), algorithms are defined for its implementation. Firstly, it is assumed that the APALM is based on an ALM with possibly a black-box implementation, striving for the non-intrusiveness of the method. The required routines for the underlying ALM are:

- $\boldsymbol{w}^\ell_{i+1} \leftarrow$ `step`$(\boldsymbol{w}^\ell_i, \Delta\boldsymbol{w}^\ell_i \Delta L)$: Performs a step with length ΔL starting at point $\boldsymbol{w}^\ell_i$ and returns the new solution $\boldsymbol{w}^\ell_{i+1}$. Given the current solution and the previous solution, a predictor for the initial iteration that employs $\Delta\boldsymbol{w}^\ell_i = \boldsymbol{w}^\ell_i - \boldsymbol{w}^\ell_{i-1}$ could be available, as well as one for a cold start, i.e., $\Delta\boldsymbol{w}^\ell_i = 0$.
- $\Delta s \leftarrow$`distance`$(\boldsymbol{w}^\ell_i, \boldsymbol{w}^\ell_j)$: gets the distance between two points $\boldsymbol{w}^\ell_i$ and $\boldsymbol{w}^\ell_j$, using equation (6.2) with $\Delta\boldsymbol{w}^\ell_i = \boldsymbol{w}^\ell_i - \boldsymbol{w}^\ell_j$.

In the following, three implementations are presented. The first implementation is a *serial implementation* without communication but with queueing, referred to as the Adaptive Serial Arc-Length Method (ASALM). The serial implementation provides the building blocks for the *hybrid implementation* and the *parallel implementation*. The *hybrid implementation*, referred to as the Adaptive Serial-Parallel Arc-Length Method (ASPALM), is a hybrid version of the APALM where parallel corrections are performed after a serial solve has finished. The *parallel implementation*, on the other hand, starts parallel corrections as soon as the first interval has been initialised; hence, there is no separation between a serial phase and a parallel phase. The parallel implementation is the final APALM.

Serial implementation

The global workflow for a serial APALM, i.e., the ASALM, is illustrated in algorithm 2. As seen in this algorithm, the initialisation is performed using a `serialSolve` routine, which defines the initial solution sequence $\{\boldsymbol{w}_i^0\}_{i=0}^I$ with I steps on level $\ell = 0$. In addition, this routine also provides a sequence of curve-length coordinates, $\{s_i^0\}_{i=0}^I$. Based on these sequences, the mappers from figure 6.5 and a queue Q are initialised in the `initializeMap` routine. Given the queue Q, the `correctQueueSerial` routine provides a sequence of solutions $\{\boldsymbol{w}_i\}$ and of curve parameters $\{s_i\}$ spanning multiple levels, hence the superscript ℓ is omitted.

Algorithm 2 Global ASALM routine (`ASALM`). The ASALM first consists of a serial solve of the whole curve length domain, followed by an evaluation of subintervals.

```
Input: $\Delta L, I$
$\{\boldsymbol{w}_i^0\}_{i=0}^I, \{s_i^0\}_{i=0}^I \longleftarrow$ serialSolve($\Delta L, I$)
$Q \longleftarrow$ initializeMap($\{\boldsymbol{w}_i^0\}_{i=0}^I, \{s_i^0\}_{i=0}^I$)
$\{\boldsymbol{w}_i\}, \{s_i\} \longleftarrow$ correctQueueSerial($Q$)
Output: $\{\boldsymbol{w}_i\}, \{s_i\}, \{\xi_i\}$
```

Using basic ALM routines, algorithm 3 defines an algorithm to obtain in serial a coarse approximation to initialise the ASALM. Optionally, a stability computation can be performed after the arc-length step, which could lead to a specialised solution towards a bifurcation point. This allows for automatic exploration of bifurcation diagrams [561, 630] and is discussed more in detail in section 6.4.3.

Algorithm 3 Serial solve (`serialSolve`). This routine provides the initial step for the APALM/ASALM, i.e., the solution data $\{\boldsymbol{w}_i^\ell\}_{i=0,\dots,I}$ and the corresponding curve-length parameters $\{s_i^\ell\}_{i=0,\dots,I}$ on level $\ell = 0$.

```
Input: $\Delta L, I$
Initialise $\boldsymbol{w}_0^\ell$, $s_0^\ell = 0$
$\boldsymbol{w}_1^\ell \longleftarrow$step($\boldsymbol{w}_0^\ell, 0, \Delta L$)                ▷ Compute first solution
$s_1 \longleftarrow \Delta L$
for $k = 1, \dots, I-1$ do
    $\boldsymbol{w}_{k+1}^\ell, \Delta s_{k+1}^\ell \longleftarrow$initiate($\boldsymbol{w}_k^\ell, \Delta\boldsymbol{w}_{k-1}^\ell, \Delta L$)      ▷ Compute new solution
    $s_{k+1}^\ell = s_k^\ell + \Delta s_{k+1}^\ell$                ▷ Compute curve coordinate
end for
Output: $\{\boldsymbol{w}_i^\ell\}_{i=0}^I, \{s_i^\ell\}_{i=0}^I$
```

In algorithm 3, the `initiate` routine (see algorithm 4) computes an arc-length interval and returns a new point $\boldsymbol{w}_k^\ell$ and the traversed distance Δs_k^ℓ, provided the previous point $\boldsymbol{w}_{k-1}^\ell$, the previous solution interval $\Delta\boldsymbol{w}_k^\ell$ and the intended arc-length step size ΔL and using the `step` and `distance` functions. Typically, Δs_k^ℓ is equal to ΔL unless the arc-length step does not converge and needs to be bisected.

As soon as a set of solution data and curve parameters, $\{\boldsymbol{w}_i^\ell\}_{i=0,\dots,I}$ and $\{s_i^\ell\}_{i=0,\dots,I}$, respectively, are known, the initialisation of the parallel computations can take place. In this initialisation, the maps from figure 6.5 are constructed and a queue Q of jobs is created, c.f. algorithm 5.

Algorithm 4 Initiation of an interval (`initiate`). Given a previous solution $\boldsymbol{w}^\ell_k$, the previous step size $\boldsymbol{w}^\ell_{k-1}$, and the desired arc-length ΔL, this routine returns a new solution $\boldsymbol{w}^\ell_{k+1}$ and a distance with respect to the previous solution $\boldsymbol{w}^\ell_k$, denoted by Δs^ℓ_{k+1}.

```
Input: $\boldsymbol{w}^\ell_k$ , $\boldsymbol{w}^\ell_{k-1}$, $\Delta L$
  $\Delta \boldsymbol{w}^\ell_k = \boldsymbol{w}^\ell_k - \boldsymbol{w}^\ell_{k-1}$                    ▷ Compute previous step
  $\boldsymbol{w}^\ell_{k+1}$ ←step($\boldsymbol{w}^\ell_{k-1}$,$\Delta \boldsymbol{w}^\ell_k$,$\Delta L$)       ▷ Compute new solution
  $\Delta s^\ell_{k+1}$ ←distance($\boldsymbol{w}^\ell_{k+1}$, $\boldsymbol{w}^\ell_k$)         ▷ Compute curve coordinate
Output: $\boldsymbol{w}^\ell_{k+1}$, $\Delta s^\ell_{k+1}$
```

Algorithm 5 Parallel initialisation (`initializeMap`). Within this algorithm, the maps $\mathscr{U}$, $\mathscr{U}'$, $\mathscr{S}$, and Ξ are constructed from a series of solutions and corresponding curve length coordinates, $\{\boldsymbol{w}^\ell_i\}^I_{i=0}$ and $\{s^\ell_i\}^I_{i=0}$, respectively, both on level $\ell = 0$. Furthermore, the queue Q is initialised by adding all subintervals to the queue.

```
Input: $\{\boldsymbol{w}^\ell_i\}^I_{i=0}$, $\{s^\ell_i\}^I_{i=0}$
  Add $\boldsymbol{w}^\ell_0$ to $\mathscr{U}$, $s^\ell_0$ to $\mathscr{S}$, and $\xi^\ell_0$ to $\Xi$     ▷ Add the start of the level 0 solutions to the map $\mathscr{U}$
  for $k = 1, \dots, I$ do
      Add $\boldsymbol{w}^\ell_k$ to $\mathscr{U}$ and $\boldsymbol{w}^\ell_{k-1}$ to $\mathscr{U}'$.
      Add $s^\ell_k$ to $\mathscr{S}$ and $\xi^0_k$ to $\Xi$.
      Add $Q_{k-1} = [\xi^\ell_{k-1}, \xi^\ell_k)$ to Q.                  ▷ Construct elements of the queue $Q$
  end for
Output: $Q = \{Q_k = [\xi^\ell_k, \xi^\ell_{k+1}]\}_{k=0,\dots,I}$
```

After initialisation, the computation of the sub-intervals can take place. This requires the routine `correctQueueSerial` as defined in algorithm 6 for fully serial computations. That is, no communication between manager and worker takes place since everything will be done on the same node.

Algorithm 6 The routine that solves the queue (`correctQueueSerial`). Given a queue Q, this algorithm takes an entry from the queue using pop and solves the defined interval using `correct`. The new solution is added to the solution maps, and if required, new jobs are added to the queue Q using `submit`. The final solutions are collected from the maps using the `collectSolutions` routine.

```
Input: $Q$
  while $Q \neq \emptyset$ do
      $Q$, ID, $\Delta L$, $\boldsymbol{w}^\ell_i$, $\boldsymbol{w}^\ell_{i-1}$, $\boldsymbol{w}^\ell_{i+1}$← pop($Q$)
      $\{d^{\ell+1}_k\}_{k=0,\dots,N-1}$, $\{\boldsymbol{w}^{\ell+1}_k\}_{k=0,\dots,N}$, $\overline{\Delta L}$, $\delta L$← correct($N$,$\Delta L_0$, $\boldsymbol{w}^\ell_i$, $\boldsymbol{w}^\ell_{i-1}$, $\boldsymbol{w}^\ell_{i+1}$)
      $Q$← submit(ID,$\{d^{\ell+1}_k\}_{k=0,\dots,N-1}$,$\{\boldsymbol{w}^{\ell+1}_k\}_{k=0,\dots,N}$,$\overline{\Delta L}$,$\delta L$,$Q$)   ▷ Submit the job; adds new jobs to $Q$ if needed
  end while
  $\{\boldsymbol{w}_i\}$, $\{s_i\}$← collectSolutions
Output: $\{\boldsymbol{w}_i\}$, $\{s_i\}$
```

The computation of the sub-interval takes place in the `correct` routine of algorithm 7 and can be used both in a serial and a parallel implementation. Given a number of sub-intervals N, the original distance between the end-points of the interval ΔL_0 and the start point, previous solutions, and reference solutions $\boldsymbol{w}^\ell_i$, $\boldsymbol{w}^\ell_{i-1}$ and $\boldsymbol{w}^\ell_{i+1}$, respectively, this routine computes a series of solutions of the sub-interval $\{\boldsymbol{w}^{\ell+1}_k\}_{k=0,\dots,N}$, their distances $\{d_k\}^{\ell+1}_{k=0,\dots,N-1}$ and the distances $\overline{\Delta L}$ and δL for error computation. Note that the distance $\Delta L'$ can be computed by taking the sum of the distances.

The `correctQueueSerial` routine includes the pop, `submit`, and `collectSolutions` rou-

Algorithm 7 The routine that solves an interval (`correct`). This routine takes a number of subintervals N, the desired step length for the total interval, the start point $\boldsymbol{w}_i^\ell$, the previous point $\boldsymbol{w}_{i-1}^\ell$, and the next point $\boldsymbol{w}_{i+1}^\ell$. It returns the solutions on the subinterval and the distances between them, respectively $\{d_j^{\ell+1}\}_{j=0,\dots,N-1}$ and $\{\boldsymbol{w}_j^{\ell+1}\}_{j=0,\dots,N}$, as well as the distances $\overline{\Delta L}$ and δL. When the `step` does not converge, it is assumed that step size modification takes place and that the data points are adjusted accordingly.

```
Input: $N$, $\Delta L_0$, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, $\boldsymbol{w}_{i+1}^\ell$
  Initialise output vectors $\{d_k^{\ell+1}\}_{k=0,\dots,N-1}$, $\{\boldsymbol{w}_k^{\ell+1}\}_{k=0,\dots,N}$
  $\Delta L = \Delta L_0 / N$                                  ▷ Defines the size of the sub-intervals
  $\boldsymbol{w}_0^{\ell+1} = \boldsymbol{w}_i^\ell$
  $\Delta\boldsymbol{w}^{\ell+1} = \boldsymbol{w}_i^\ell - \boldsymbol{w}_{i-1}^\ell$        ▷ Determine previous arc-length step, to be used to predict the step on $\ell+1$
  for $k = 0,\dots,N-1$ do
      $\boldsymbol{w}_{k+1}^{\ell+1} \leftarrow$ step($\boldsymbol{w}_k^{\ell+1}$,$\Delta\boldsymbol{w}$,$\Delta L$)          ▷ Perform the ALM iteration
      $\Delta\boldsymbol{w} = \boldsymbol{w}_{k+1}^{\ell+1} - \boldsymbol{w}_k^{\ell+1}$                     ▷ Update the solution step
      $d_k \leftarrow$ distance($\boldsymbol{w}_{k+1}^{\ell+1}$, $\boldsymbol{w}_k^{\ell+1}$)                ▷ Gets the distance
  end for
  $\delta L \leftarrow$ distance($\boldsymbol{w}_{i+1}^\ell$, $\boldsymbol{w}_N^{\ell+1}$)
  $\overline{\Delta L} \leftarrow$ distance($\boldsymbol{w}_N$, $\boldsymbol{w}_0$)
Output: $\{d_j^{\ell+1}\}_{j=0,\dots,N-1}$, $\{\boldsymbol{w}_j^{\ell+1}\}_{j=0,\dots,N}$, $\overline{\Delta L}$, $\delta L$
```

tines. These routines mainly involve read and write operations for the mappers defined in figure 6.5; hence, only a brief description is provided:

- Q, `ID`, ΔL, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, $\boldsymbol{w}_{i+1}^\ell \leftarrow$ `pop`(Q): Takes the first available interval from the queue Q and returns a job `ID`, an interval length ΔL, the start solution $\boldsymbol{w}_i^\ell$, the previous solution $\boldsymbol{w}_{i-1}^\ell$ and the next available solution $\boldsymbol{w}_{i+1}^\ell$. It also updates the queue Q internally by removing the current entry.

- $Q \leftarrow$ `submit`(`ID`,$\{d_k^{\ell+1}\}_{k=0,\dots,N-1}$,$\{\boldsymbol{w}_k^{\ell+1}\}_{k=0,\dots,N}$,$\overline{\Delta L}$,$\delta L$,$Q$): Takes a job `ID`, the series of solutions $\{\boldsymbol{w}_k^{\ell+1}\}_{k=0,\dots,N}$ and their distances $\{d_k^{\ell+1}\}_{k=0,\dots,N-1}$ and the distances $\overline{\Delta L}$ and δL. Using equations (6.6) to (6.8), the errors are computed and solution intervals are added to the queue Q if needed.

- $\{\boldsymbol{w}_i\}$, $\{s_i\} \leftarrow$ `collectSolutions`: Based on the underlying mappers, the solutions of all levels are collected into $\{\boldsymbol{w}_i\}$ and $\{s_i\}$.

Hybrid implementation

The hybrid implementation of the APALM is referred to as the Adaptive Serial-Parallel Arc-Length Method (ASPALM), since it is a two-stage method with a serial initalisation and a parallel correction. This concept is similar to the concept presented for the ASALM, but communication between the manager and worker processes is added so that the correction stage can be performed in parallel. To this end, the `correctQueueSerial` routine is re-defined into `correctQueueParallel` and communications between workers and the manager are defined. In the following, the global solution algorithm for the ASPALM is defined in algorithm 8. As for the ASALM (see algorithm 2), the initialisation is performed in serial by the `serialSolve` routine, and the maps are initialised, both by the manager process. As soon as queue Q is established, the queue can be processed in parallel. Indeed, this implies that the worker processes are idle until the queue Q is fully available.

Algorithm 8 Global ASPALM routine (ASPALM). The ASPALM first consists of a serial solve of the whole curve length domain, followed by a parallel evaluation of subintervals. This algorithm is specified for simple manager-worker parallelisation; more advanced parallelisation schemes, e.g., with multiple managers, are easily achieved.

```
Input: $\Delta L, I$
if manager then
    $\{\boldsymbol{w}_i^\ell\}_{i=0}^I, \{s_i^\ell\}_{i=0}^I \leftarrow$ serialSolve($\Delta L, I$)        ▷ See algorithm 3
    $Q \leftarrow$ initializeMap($\{\boldsymbol{w}_i^\ell\}_{i=0}^I, \{s_i^\ell\}_{i=0}^I$)
    $\{\boldsymbol{w}_i^\ell\}, \{s_i^\ell\} \leftarrow$ correctQueueParallel($Q$)
else
    Initialise stop=false
    while stop = false do
        stop $\leftarrow$ receiveStop
        workerCorrect()
    end while
end if
Output: $\{\boldsymbol{w}_i\}, \{s_i\}$
```

As seen in algorithm 8, the manager process uses the `correctQueueParallel` routine; see algorithm 10. This routine is called by the manager process and sends and receives data to and from the workers, updates the queue, and heavily relies on communication functions as defined in table 6.1. The communication function `sendMetaData` can be omitted for the ASPALM since it is primarily used in the APALM to distinguish between initiation and correction jobs. Furthermore, as described in table 6.1, the routine `sendJob` can be called with and without the reference solution $\boldsymbol{w}_{i+1}^\ell$, depending whether it is available (correction) or not (initiation). In the case of the ASPALM, all jobs that are popped from the queue Q in the `correctQueueParallel` routine are by definition correction jobs.

As shown in algorithm 8, the worker processes will perform the `workerCorrect` from algorithm 12. This algorithm contains the correction step for any interval that is received from the communications coming from the manager process. The `workerCorrect` is executed until a stop signal is received from the manager process. The latter is broadcast to all workers as soon as queue Q is empty. Table 6.1 gives an overview of the communication functions that are used for communications between the manager and worker processes in the hybrid and parallel implementations.

Parallel implementation

Contrary to the serial and hybrid implementations, the fully parallel implementation does not work with a two-staged procedure of serial initialisation and parallel correction. Instead, the fully parallel solve consists of one stage with a single queue consisting of initialisation and correction jobs. For the Adaptive Parallel Arc-Length Method (APALM), the global routine is provided in algorithm 9.

As can be seen in algorithm 9, the manager process in the APALM only initialises the queue using the `initializeQueue` routine, and it contains the `correctQueueParallel` routine. The former is not specified explicitly since it only initialises a map with zero points and allocates the maximum number of intervals I as well as the interval length ΔL. The `correctQueueParallel` routine is given in algorithm 10. It applies meta-data communica-

Table 6.1: Required communications between manager and worker processes for the ASPALM and APALM.

Send/Receive	From	To	Data	Description
`sendMetaData` `receiveMetaData`	Manager	Worker	`ID`, ℓ	Communicates meta-data between the manager and the worker processes. In this case, only the `ID` and the level ℓ are needed.
`sendJob` `receiveJob`	Manager	Worker	`ID`, ΔL_0, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, ,$(\boldsymbol{w}_{i+1}^\ell)$	Communicates information to perform the computation of an interval between the manager and the worker processes. The reference solution $\boldsymbol{w}_{i+1}^\ell$ is optional since it is not available for initiation jobs.
`sendData` `receiveData`	Worker	Manager	W_j, `ID`, $\{d_j^{\ell+1}\}_{j=0,\dots,N-1}$, $\{\boldsymbol{w}_j^{\ell+1}\}_{j=0,\dots,N}$, $\overline{\Delta L}$, δL	Communicates the data resulting from a sub-interval computation between the manager and the worker processes. The receive communication also provides the worker from whom the data is received.
`sendStop` `receiveStop`	Manager	Worker (all)	Boolean	Communicates a stop signal between the manager and the worker processes.

tion, see table 6.1, to the worker processes so that the distinction based on the data level ℓ in algorithm 9 can be made by the workers. On the side of the worker processes, the meta-data is received, and if the level is equal to 0, an interval is initiated using `workerInitiate` (see algorithm 11), and if the level is larger than 0, an interval is corrected using `workerCorrect` (see algorithm 12); see algorithm 9. Inside the `workerInitiate` and `workerCorrect` routines, communications from the worker to the manager process (see table 6.1) are included.

6.4.3 Arc-length Exploration

To enable multi-branch parallelisation of APALM, small modifications are required to the data structure and the algorithms presented in sections 6.4.1 and 6.4.2. The easiest multi-branch parallelisation is achieved by identifying branch switches only in the serial solve, such that the initialisation of the APALM can be done across different branches. In this case, the serial solve is performed on the main branch, and any bifurcation point is stored such that a restart can be performed from this point; see [630] for details. As soon as such a bifurcation point is identified, a branch switch can be performed, and a new serial solve can be started from that point. As a result, a series of solutions $\{\boldsymbol{w}_i\}_{i=0}^I$ is computed for each branch. Similar to the single-branch case, a data structure and a queue can be initialised using algorithm 5 per branch. Depending on the parallel configuration, the queues Q^b, $b = 0, \dots, n_{\text{branches}}$ of all branches can be treated separately by multiple manager processes, or they can be combined into one large queue Q and handled by one single manager process. In the latter case, each job will also contain a branch identifier to refer to the corresponding data structure.

The advantages of the above approach combining multi-branch and within-branch par-

Algorithm 9 Global APALM routine (`APALM`). The APALM first consists of a serial solve of the whole curve length domain, followed by a parallel evaluation of subintervals. This algorithm is specified for simple `manager-worker` parallelisation; more advanced parallelisation schemes, e.g., with multiple managers, are easily achieved.

```
Input: $\Delta L, I$
if manager then
    $Q \longleftarrow$ initializeQueue($\Delta L, I$)
    $\{w_i^\ell\},\{s_i^\ell\} \longleftarrow$ correctQueueParallel($Q$)
    $\{w_i\}, \{s_i\} \longleftarrow$ collectSolutions
else
    while true do
        stop $\longleftarrow$ receiveStop
        if stop then
            Break loop
        end if
        ID, $\ell \longleftarrow$ receiveMetaData
        if $\ell = 0$ then
            workerInitiate()
        else
            workerCorrect()
        end if
    end while
end if
Output: $\{w_i\}, \{s_i\}, \{\xi_i\}$
```

allelisation using the APALM are that the extension from a single-branch APALM to a multi-branch APALM is straightforward. The disadvantage, however, is that the identification of bifurcations is only taken into account in the serial solve step; hence, any bifurcations that are identified in the parallel solve will not be taken into account. A remedy would be to rebuild the map and the data structure on the manager process as soon as a worker process finds a bifurcation point; this requires all active workers to terminate.

Algorithm 10 The `correctQueueParallel` routine, accompanied by the `workerCorrect` routine from algorithm 12 and communication functions defined in table 6.1. This routine takes the queue Q and assigns jobs from the queue to the available workers. Then, while the queue Q is non-empty, data is communicated to and from the workers, and solutions are submitted. Note that the `pop` and `submit` routines are equivalent to the ones in algorithm 6.

```
Input: $Q$
Initialise a pool of worker processes $W = \{W_j,\ j = 1, \dots, N_{\text{workers}}\}$
while $Q \neq \emptyset$ and $W \neq \emptyset$ do
    $Q$, ID, $\Delta L$, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, $\boldsymbol{w}_{i+1}^\ell \leftarrow$ pop($Q$)        ▷ See line 2 of algorithm 6
    sendStop(false)
    sentMetaData(ID,$\ell$)
    sendJob(ID,$\Delta L_0$, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, $\boldsymbol{w}_{i+1}^\ell$, $W_j$)
    Remove $Q_i$ from $Q$ and $W_j$ from $W$
end while
                                                  ▷ Send jobs to workers when they are available
while $|W| \neq N_{\text{workers}}$ do
    $W_j$, ID, $\{d_j^{\ell+1}\}_{j=0,\dots,N-1}$, $\{\boldsymbol{w}_j^{\ell+1}\}_{j=0,\dots,N}$, $\overline{\Delta L}$, $\delta L \leftarrow$ receiveDataWorker2Manager
    $Q \leftarrow$ submit(ID,$\{d_j^{\ell+1}\}_{j=0,\dots,N-1}$,$\{\boldsymbol{w}_j^{\ell+1}\}_{j=0,\dots,N}$,$\overline{\Delta L}$, $\delta L$, $Q$)        ▷ See line 4 of algorithm 6
    Add $W_j$ to $W$
    while $Q \neq \emptyset$ and $W \neq \emptyset$ do
        $Q$, ID, $\Delta L$, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, $\boldsymbol{w}_{i+1}^\ell \leftarrow$ pop($Q$)
        sendStop(false)        ▷ The worker always expects a stop signal, now it is false.
        sentMetaData(ID,$\ell$)
        sendJob(ID,$\Delta L_0$, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, $\boldsymbol{w}_{i+1}^\ell$, $W_j$)
        Remove $Q_i$ from $Q$ and $W_j$ from $W$
    end while
end while
sendStopManager2All(true)
Output: $\{\boldsymbol{w}_i\}$, $\{s_i\}$
```

Algorithm 11 Solve routine for a worker (`workerInitiate`). This routine performs the initiation steps (see algorithm 4) on jobs received from the manager, until a stop signal is received. More information on the communication functions can be found in table 6.1.

```
Input:
(ID,$\Delta L_0$, $\boldsymbol{w}_i$, $\boldsymbol{w}_{i-1}$, $\boldsymbol{w}_{\text{ref}}$) $\leftarrow$ receiveJob
$\boldsymbol{w}_{k+1}^\ell$,$\Delta s_{k+1}^\ell \leftarrow$initiate($\boldsymbol{w}_k^\ell$,$\Delta\boldsymbol{w}_{k-1}^\ell$,$\Delta L$)
sendDataWorker2Manager(ID,$\{d_j\}_{j=0,\dots,N-1}^{\ell+1}$,$\{\boldsymbol{w}_j\}_{j=0,\dots,N}^{\ell+1}$,$\overline{\Delta L}$)
Output:
```

Algorithm 12 Solve routine for a worker (`workerCorrect`). This routine performs correction steps (see algorithm 7) on jobs received from the manager, until a stop signal is received. More information on the communication functions can be found in table 6.1.

```
Input:
(ID,$\Delta L_0$, $\boldsymbol{w}_i$, $\boldsymbol{w}_{i-1}$, $\boldsymbol{w}_{\text{ref}}$) $\leftarrow$ receiveJob
$\{d_k^{\ell+1}\}_{k=0,\dots,N-1}$, $\{\boldsymbol{w}_k^{\ell+1}\}_{k=0,\dots,N}$, $\overline{\Delta L}$, $\delta L \leftarrow$ correct($N$,$\Delta L_0$, $\boldsymbol{w}_i^\ell$, $\boldsymbol{w}_{i-1}^\ell$, $\boldsymbol{w}_{i+1}^\ell$)
sendDataWorker2Manager(ID,$\{d_j\}_{j=0,\dots,N-1}^{\ell+1}$,$\{\boldsymbol{w}_j\}_{j=0,\dots,N}^{\ell+1}$,$\overline{\Delta L}$)
Output:
```

6.5 Numerical Experiments

In this section, the APALM scheme is demonstrated on a series of benchmark problems. The first two benchmark problems are structural analysis problems with limit-point instabilities and complex collapsing mechanisms involving strongly curved solution paths. The third benchmark is a buckling problem containing a bifurcation with multiple branches to illustrate the concept of the APALM in a multi-branch setting. All benchmark problems are computed using isogeometric Kirchhoff–Love shell elements based on the works [319, 320, 587]. Furthermore, a scaling test with respect to the number of worker processes is performed for all benchmark problems. Here, a scaling analysis of the ASPALM is performed to demonstrate the relative computational costs of the serial initialisation phase compared to the parallel correction phase. Furthermore, a scaling analysis of the APALM is performed to show the advantage of the fully parallel APALM scheme over the two-stage ASPALM scheme. For the scaling tests, the communications from table 6.1 are performed using the Message Passing Interface (MPI), and they are performed on the Delft High Performance Computing Centre (DHPC) [151] with Intel XEON E5-6248R 24C 3.0GHz nodes with 96GB of memory per CPU. The code is made available within the Geometry + Simulation modules [294] library. More information can be found in chapter 8.

6.5.1 Collapse of a Shallow Roof

The first benchmark problem involves a shallow roof subject to a point load at the midpoint. The roof is discretized with 4 × 4 NURBS elements of degree 3. The roof is composed of a lay-up of composites with material properties as presented in figure 6.6, inspired by [344]. It is modelled using isogeometric Kirchhoff-Love shell elements [319] supporting composite laminates [238]. A Crisfield ALM is used with an initial arc length of 30 and a scaling parameter of $\Psi = 1$. The tolerance of the APALM is set to $\varepsilon_l = \varepsilon_u = 10^{-2}$. This tolerance implies that intervals are marked for refinement when the traversed length is deviates than 1% of the original interval length. A smaller tolerance would imply that more elements are marked for refinement and that the corrections are performed up to lower levels. The material, and load parameters for this benchmark can be found in figure 6.6. Reference solutions are obtained using a serial arc-length method with a sufficiently fine increment size.

Figure 6.7 provides the results of the APALM applied to the collapse of the shallow roof. As can be seen from this figure, the serial computation provides a coarse estimate of the reference curve. Especially on the first limit point (between $\lambda \times w_A \in [-15, -8] \times [20, 25]$), the data is sparse, similar to the collapse itself (see inset in figure 6.7). The results of the APALM show that a lot of refinements are needed to represent the collapsing behaviour correctly in the region of the inset in figure 6.7. These regions do not necessarily involve extremely curved paths in the axes of figure 6.7, but the solutions $\boldsymbol{w}$ are most likely curved in the higher-dimensional solution space.

In order to assess the parallelisation of the APALM in this example, the test from figure 6.7 is performed with an increasing number of workers using the ASPALM and the APALM schemes. The results in table 6.2a show that for the computation with arc-length parameter $\Delta L = 30$, the parallel correction step of the ASPALM scales optimally (i.e., with a

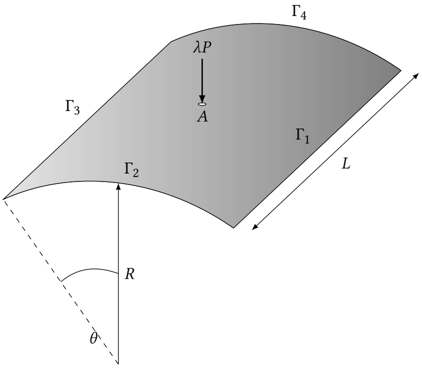


Figure 6.6: The problem definition for the benchmark of the collapsing roof with length $L = 508$ [mm], with radius $R = 2540$ [mm], an angle $\theta = 10$ [rad] with a thickness of $t = 6.35$ [mm]. The boundaries Γ_1 and Γ_3 have fixed displacements, and the other sides are free. The material is modelled using a Saint-Venant Kirchhoff laminate with $E_{11} = 3300$ [N/mm^2], $E_{22} = E_{33} = 1100$ [N/mm^2], $G_{12} = G_{13} = 660$ [N/mm^2], $E_{23} = 440$ [N/mm^2] and $\nu_{12} = \nu_{13} = \nu_{23} = 0.25$ [−] and with lay-up angles $[0/90/0]^\circ$. The load is variable with a magnitude $P = 10$ [N] and magnification factor λ.

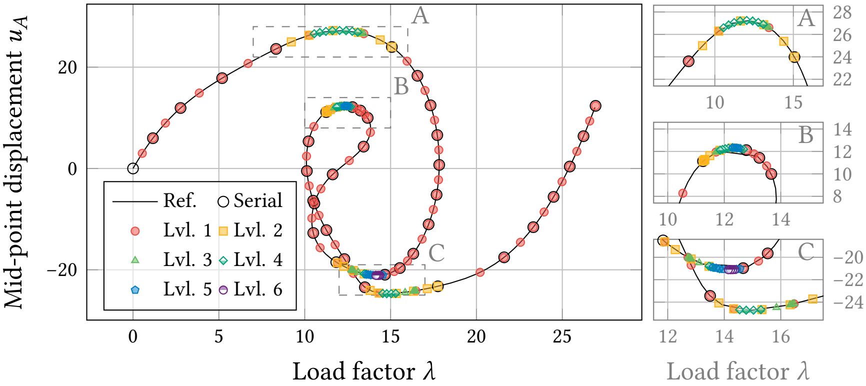


Figure 6.7: Results of the collapsing roof. The figure on the left indicates the full solution path, and the figures on the right depict the insets indicated in the left figure. The reference and serial solutions are represented by the solid line and the black markers, respectively. The solutions computed by the APALM are indicated per level. The simulation is performed with a tolerance of $\varepsilon_l = \varepsilon_u = 10^{-2}$ and an increment length of $\Delta L = 30$.

Table 6.2: Computational time in [s] for the benchmark of the collapsing roof for the ASPALM and APALM for different numbers of worker processes. The times for the ASPALM are presented for the serial initialisation and the parallel correction phases, and the sum of the two is given as the total computational time. The numbers in the *Serial* column should theoretically be the same, but they provide a representation of the variation in the time measurements. The results are presented for simulations with increment lengths $\Delta L = 30$ (a) and $\Delta L = 2.5$ (b), and the italic row with 0 workers denotes the ASALM method.

(a) $\Delta L = 30$

	ASPALM			APALM
#	Serial +	Parallel =	Total	Parallel
0	*115.7*	*195.3*	*311.1*	*287.1*
1	119.2	209.0	328.2	318.8
2	114.0	100.8	214.8	162.4
4	109.5	46.1	155.6	115.8
8	115.0	27.0	142.1	115.9
16	115.1	17.8	132.9	116.3
32	114.9	15.9	130.8	113.0
64	114.5	13.3	127.8	116.0

(b) $\Delta L = 2.5$

	ASPALM			APALM
#	Serial +	Parallel =	Total	Parallel
0	*507.2*	*1,778.1*	*2,285.3*	*2,187.1*
1	500.5	1,757.7	2,258.2	2,310.2
2	447.5	835.3	1,282.9	1,114.0
4	493.4	449.4	942.8	558.1
8	496.8	223.2	720.0	453.9
16	503.3	113.0	616.2	483.6
32	493.2	58.1	551.3	510.9
64	504.2	29.2	533.4	498.3
128	501.0	20.2	521.3	494.7
256	505.5	18.8	524.3	509.6

factor 2) up to around 8 workers, after which the scalability decreases and the parallel correction phase takes around 15% of the total computational time. Using the APALM scheme, the total computational time is decreased compared to the ASPALM scheme, and parallel corrections can be started as soon as the first interval has been initialized. The computational time of the APALM stagnates around 4 workers, at a computational time similar to the serial initialisation time for the ASPALM method, showing that adaptive parallel corrections can be performed without significantly more computational costs compared to a serial arc-length method. When the number of intervals is increased, e.g., by decreasing the arc-length parameter to $\Delta L = 2.5$ (table 6.2b), it can be seen that the parallel stage of the ASPALM scales up to a higher number of workers, in this case 64, up to the point that it takes around 5% of the total computational time for 256 workers. The APALM again provides computational times similar to a serial ALM without corrections. The improved scalability is explained by the fact that the queue is in general longer; therefore, the time that workers are idle waiting for the last job to be finished is smaller relative to the total computational time.

6.5.2 Collapse of a Truncated Cone

The second benchmark example is based on [587] and involves the collapsing behaviour of a truncated cone with a hyperelastic Mooney-Rivlin material model. This benchmark is based on [23], but in the work of [587], the full collapsing behaviour was revealed using arc-length methods. The geometry, material, and load specifications can be found in figure 6.8.

The truncated cone is modelled using a quarter geometry using symmetry conditions to represent the axisymmetry as used in the original case of [23]. The geometry is modelled with 32 NURBS elements of degree 2 over the height. Further, an initial arc length of 0.5

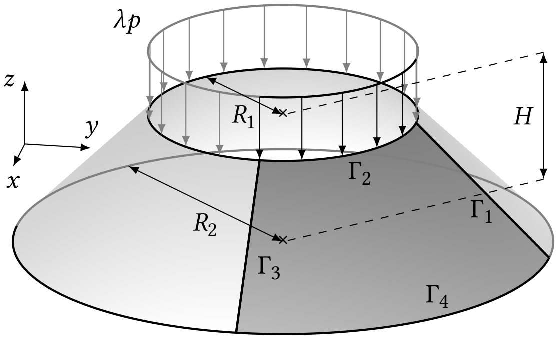


Figure 6.8: The problem definition for the benchmark of the collapsing truncated cone with inner radii $R_1 = 1\,[\mathrm{m}]$ and $R_2 = 2\,[\mathrm{m}]$ and height $H = 1\,[\mathrm{m}]$. The thickness of the cone is $t = 0.1\,[\mathrm{m}]$. The cone is modelled by using a quarter of the geometry, using symmetry conditions on Γ_1 and Γ_3. The displacements at the bottom boundary (Γ_4) are fixed, and on the top boundary, a variable line load is applied and is variable with magnitude $p = 1\,[\mathrm{N/mm}]$ and magnification factor λ. The material of the cone is modelled using an incompressible Mooney-Rivlin model with parameters $\mu = c_1 + c_2 = 4.225\,[\mathrm{N/mm^2}]$, $c_1/c_2 = 7$.

is used, and the scaling factor is $\Psi = 0$. The top boundary Γ_2 is free, and on the bottom boundary Γ_4, all displacements are fixed. The other boundaries have symmetric boundary conditions. The governing material model is an incompressible Mooney-Rivlin material model with a strain energy density function (with a slight abuse of notation)

$$\Psi(\mathbf{C}) = \frac{c_1}{2}(I_1 - 3) + \frac{c_2}{2}(I_2 - 3), \tag{6.18}$$

with I_1 and I_2 the first and second invariants of the deformation tensor $\mathbf{C} = \mathbf{F}^\top\mathbf{F}$. More information on the problem set-up and the material models can be found in [587]. The reference results are obtained from a serial ALM computation with a sufficiently small increment size.

The results of the collapsing truncated cone problem are presented in figure 6.9. As seen in this picture, the serial initialisation provides a coarse approximation of the path but leaves out details, e.g., the rotated “S”-shaped curve in the inset in figure 6.9. From the results, it is clear that the APALM focuses its refinement on the curved parts of the path and reveals the “S”-shaped curve among other features of the path.

Similar to the collapse of the roof, a scaling analysis of the parallel evaluations is performed. The results in table 6.3a verify that, as for the benchmark example with the collapsing roof, the scalability of the parallel correction phase scales optimally up to 8 workers, where the parallel correction phase takes around 15 % of the total computational time when using 64 workers. When using the APALM scheme, the collapsing cone also shows that the computation times of the APALM are similar to the times needed for serial initialisation, in other words, a classical ALM without adaptive corrections. When the number of intervals is increased, i.e., when the arc-length parameter is decreased to $\Delta L = 0.0625$ (table 6.3b), the scalability of the parallel phase of the ASPALM and of the full APALM reaches further, up to 64 workers.

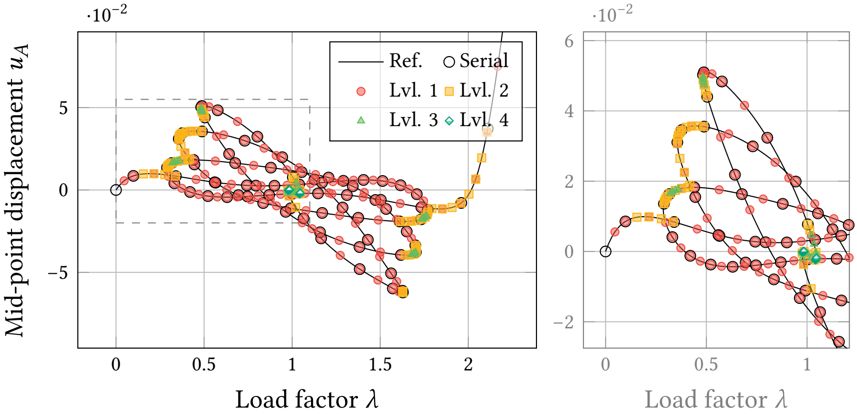


Figure 6.9: Results of the collapsing truncated cone. The figure on the left depicts the full solution path, and the figure on the right depicts the inset indicated in the left figure. The reference and serial solutions are represented by the solid line and the black markers, respectively. The solutions computed by the APALM are indicated per level. The simulation is performed with a tolerance of $\varepsilon_l = \varepsilon_u = 10^{-2}$ and an increment length of $\Delta L = 0.5$.

Table 6.3: Computational time in [s] for the benchmark of the collapsing truncated cone for the ASPALM and APALM for different numbers of worker processes. The times for the ASPALM are presented for the serial initialisation and the parallel correction phases, and the sum of the two is given as the total computational time. The numbers in the *Serial* column should theoretically be the same, but they provide a representation of the variation in the time measurements. The results are presented for simulations with increment lengths $\Delta L = 0.5$ (a) and $\Delta L = 0.0625$ (b), and the italic row with 0 workers denotes the ASALM method.

(a) $\Delta L = 0.5$

	ASPALM			**APALM**
#	Serial +	Parallel =	Total	Parallel
0	*160.2*	*244.0*	*404.2*	*436.6*
1	162.5	247.2	409.7	424.8
2	169.5	130.1	299.6	207.1
4	170.6	68.1	238.7	172.9
8	162.6	43.0	205.6	160.5
16	175.3	32.0	207.3	173.3
32	175.5	27.3	202.8	170.8
64	170.1	23.3	193.4	169.5

(b) $\Delta L = 0.0625$

	ASPALM			**APALM**
#	Serial +	Parallel =	Total	Parallel
0	*499.7*	*2,575.9*	*3,075.6*	*3,055.3*
1	467.5	2,232.5	2,700.0	2,783.8
2	496.3	1,337.0	1,833.2	1,573.4
4	467.8	654.4	1,122.2	789.5
8	490.1	322.6	812.7	489.4
16	467.6	167.6	635.1	496.0
32	494.1	97.1	591.1	483.9
64	491.4	55.7	547.1	493.6
128	485.0	41.5	526.5	494.5
256	493.8	32.9	526.7	491.4
512	491.8	25.8	517.6	488.5

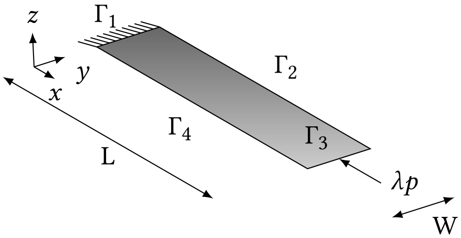


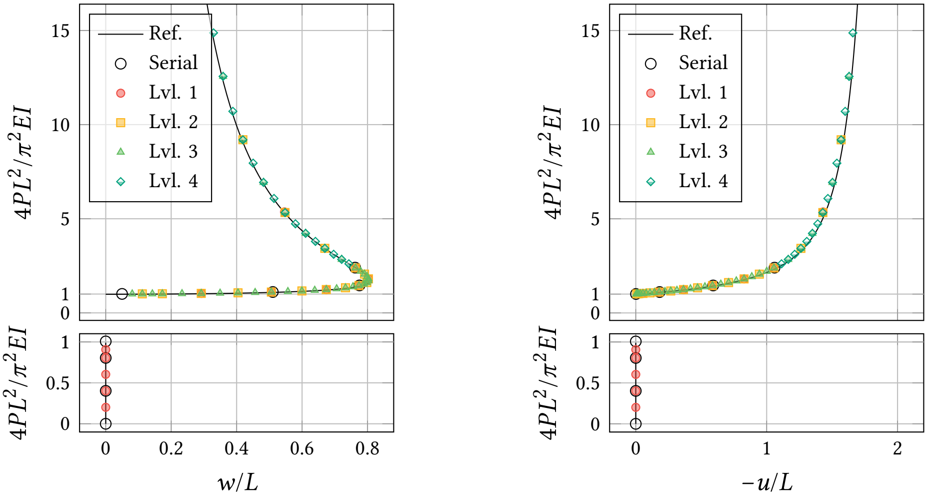
Figure 6.10: The problem definition for the benchmark of the buckling of a strip with length $L = 1$ [m], width $W = 0.01$ [m] and thickness $t = 0.01$ [m] subject to a horizontal load with magnitude $p = 0.1$ [N] with magnification factor λ. The strip has fixed displacements and rotations at Γ_1 and fixed displacements in y-direction on Γ_2 and Γ_4. The material is modelled using a Saint-Venant Kirchhoff material model with Young's modulus $E = 75 \cdot 10^6$ [N/mm^2] and Poisson ratio $\nu = 0$ [−].

6.5.3 Strip Buckling

The third example involves a benchmark problem consisting of a bifurcation instability. The problem consists of a flat strip that is clamped on one edge and free on all the others, with an in-plane compressive load applied on the free end opposite to the clamped edge; see figure 6.10 for the problem set-up and [423] for the reference results. The ALM that is used is a Crisfield method with $\Psi = 0$, with a pre-buckling arc-length of $5 \cdot 10^{-5}$, a post-buckling arc-length of 5, and a tolerance of the APALM of $\varepsilon_l = \varepsilon_u = 10^{-3}$. The serial ALM is equipped with an extension for the computation of singular points (Wriggers 1988); see [583] for more details on this implementation. Using these methods, an initially flat strip is compressed until the bifurcation point has been computed. As soon as the strip becomes unstable, the bifurcation point is computed, and a branch switch is performed, marking the transition between the pre-buckling and post-buckling branches.

The results for the buckled strip are presented in figure 6.11. In this figure, the non-dimensional horizontal and vertical displacements of the end point are plotted with respect to the non-dimensional applied load. In the plots, the pre- and post-buckling branches are plotted separately for clarity, but the branches should obviously be connected at the bifurcation point. As can be seen from the results, a rather coarse serial approximation of the post-buckling branch gives a good starting point for a multi-level approximation of the curve, providing additional detail in the sharp corner in $W/L \in [0.7, 0.8]$. In addition, it can be seen that the pre-buckling branch requires no more levels than the first, as the behaviour there is just a linear axial compression, hence the solution path is straight.

As for the previous two benchmark examples, a scaling analysis of the parallel evaluations is performed. The main difference between the previous two examples is that the present example involves a bifurcation point. However, since the job queue includes the jobs from all branches together, there is no idle time to wait for a branch to finish before starting a new branch; hence, it is expected that the parallel scaling for a bifurcation problem should have the same scaling properties. Indeed, table 6.4a shows that optimal scaling is achieved in the parallel correction phase of the ASPALM up to 8 nodes, after which the idle time to wait for the last job to finish significantly impacts the scaling, as



(a) Non-dimensional out-of-plane displacement of the beam with respect to the non-dimesional buckling load.

(b) Non-dimensional in-of-plane displacement (length direction) of the beam with respect to the non-dimensional buckling load.

Figure 6.11: Results of the buckling of a clamped strip. The left figure provides the out-of-plane displacement of the free end with respect to the non-dimensional load $4PL^2/\pi^2EI$, and the right figure represents the horizontal displacement of the free end with respect to the same non-dimensional load. In both figures, buckling occurs when $4PL^2/\pi^2EI = 1$ and the axes are split on this point to make the pre- and post-buckling branches both visible. The simulation is performed with a tolerance of $\varepsilon_l = \varepsilon_u = 10^{-3}$ and a step length of $\Delta L = 5 \cdot 10^{-5}$ (pre-buckling) and $\Delta L = 5$ (post-buckling).

Table 6.4: Computational time in [s] for benchmark of the buckled strip using the ASPALM and APALM with different numbers of worker processes. The times for the ASPALM are presented for the serial initialisation and the parallel correction phases, and the sum of the two is given as the total computational time. The numbers in the *Serial* column should theoretically be the same, but they provide a representation of the variation in the time measurements. The results are presented for simulations with increment lengths $\Delta L = 2.5$ (a) and $\Delta L = 0.025$ (b), and the italic row with 0 workers denotes the ASALM method.

(a) $\Delta L = 2.5$

	ASPALM			**APALM**
#	Serial +	Parallel =	Total	Parallel
0	*50.7*	*171.8*	*222.0*	*244.0*
1	58.7	198.1	257.0	244.0
2	58.5	102.6	161.0	112.0
4	59.0	51.1	110.0	77.0
8	59.0	27.9	87.0	67.0
16	59.3	24.9	84.0	68.0
32	60.3	23.0	83.0	68.0
64	58.6	24.4	83.0	66.0

(b) $\Delta L = 0.025$

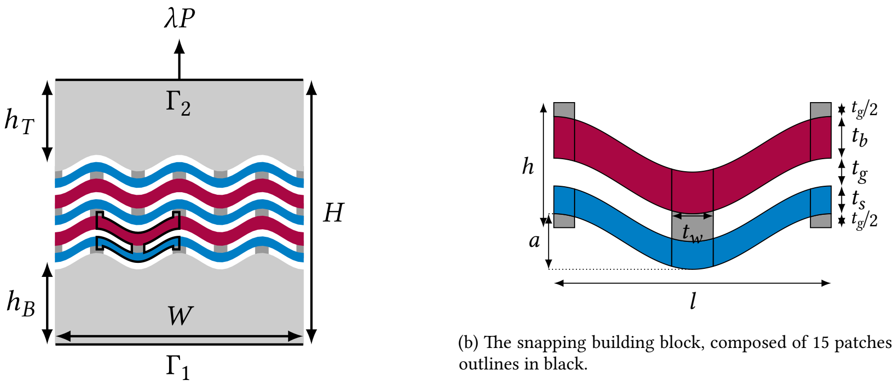
	ASPALM			**APALM**
#	Serial +	Parallel =	Total	Parallel
0	*963.3*	*1,963.3*	*2,926.6*	*3,017.6*
1	1,022.3	2,065.7	3,088.0	3,020.6
2	1,025.0	1,053.7	2,078.7	1,509.7
4	943.9	464.0	1,407.9	1,034.1
8	1,019.5	256.0	1,275.5	1,028.2
16	1,006.2	129.3	1,135.5	1,027.7
32	1,026.6	68.8	1,095.4	1,028.3
64	1,032.7	33.6	1,066.4	1,028.0
128	935.1	18.9	954.0	1,023.5
256	1,012.6	12.0	1,024.6	1,026.8

observed in the other benchmarks. In addition, it is found that the APALM reaches efficient computation of the full adaptive load-displacement curve within the time of a serial ALM computation, using 8 workers. When increasing the number of curve segments by decreasing the arc-length parameter to $\Delta L = 0.025$, it can again be seen that the parallel scalability increases. Optimal scaling of the parallel correction of the ASPALM is achieved up to 256 workers, with the correction phase taking only 1% of the total computational time. The APALM provides an adaptively refined solution curve in the time of a serial ALM computation, using only 4 to 8 workers.

6.5.4 Snapping Meta-Material

As a final example, the APALM is applied to a problem of larger scale. In particular, the snapping behaviour of a snapping meta-material is modelled, inspired by [455]. The meta-material consists of $N_x \times N_Y = 3 \times 2.5$ building blocks (see figure 6.12a) with a snapping and a bearing segment (see figure 6.12b), and the material is modelled with a compressible Neo-Hookean material model. The full problem details are provided in figure 6.12. The snapping behaviour of the meta-material is investigated by using arc-length methods on the varying load λP, with a step size of $\Delta L = 5 \cdot 10^{-2}$, until 1.5% strain. The simulation is modelled using 2D elasticity equations using the plane-stress assumption, which are discretised using B-splines with mixed degrees 2 and 3 and maximal regularity. The mesh is uniformly distributed, and the full system of equations has 16563 degrees of freedom. The simulations are performed using shared-memory parallelisation of the assembly routines and distributed memory parallelisation of the ASPALM and APALM. For reference, a displacement-controlled (DC) simulation is performed.

Figure 6.13 depicts the stress-strain curve for the snapping meta-material depicted



(b) The snapping building block, composed of 15 patches outlines in black.

(a) A snapping meta-material with 3 × 2.5 building blocks, of which one is outlined. The total multi-patch consists of 132 patches.

Figure 6.12: The problem definition for the snapping meta material using a grid of 3 × 2.5 elements (a) with the element geometry as defined in (b). The element dimensions are defined using the thickness of the load-bearing part $t_b = 1.5$ [mm] and the thickness of the snapping part $t_s = 1.0$ [mm], the thickness of the gap $t_g = 1.0$ [mm] and the thickness of the connectors $t_w = 1.5$ [mm], such that the height $h = t_b + t_s + 2t_g$. The length of the element is $\ell = 10$ [mm], and the amplitude of the cosine wave defining the element shape is given by $a = 0.3l$. Since the meta-material has 3 × 2.5 elements, the total width is $W = 3\ell$. The height of the total metamaterial is given by $H = 3h + 2t_g + t_s + h_B + h_T$, where $h_B = h_T = 5t_g$ are the buffer zones on the top and the bottom. The thickness of the specimen (in out-of-plane direction) is $b = 3$ [mm]. The material is defined using a compressible Neo-Hookean material model with Young's modulus $E = 78$ [N/mm^2] and Poisson ratio $\nu = 0.4$ [−]. The bottom boundary Γ_1 is fixed using $u_x = u_y = 0$, and the top boundary Γ_2 is fixed in the horizontal direction ($u_x = 0$) and coupled in the vertical direction u_y. The load applied on the top boundary is a variable defined by λP.

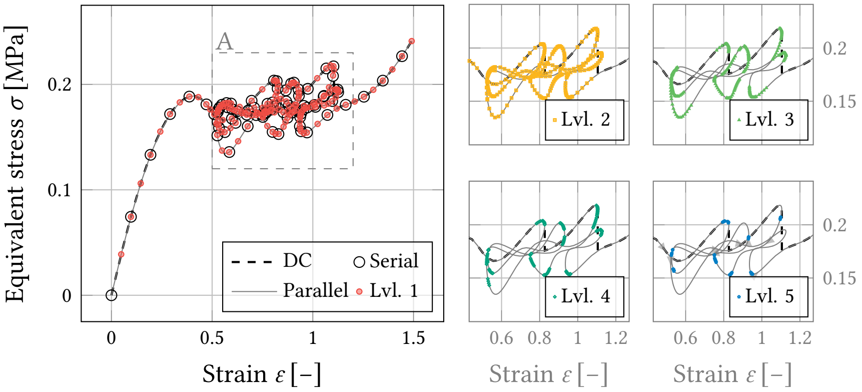


Figure 6.13: Stress-strain diagram for the snapping meta-material from figure 6.12. The vertical axis depicts the equivalent stress $\sigma = \lambda P/(bW)$, and the horizontal axis represents the strain $\varepsilon = u_y/H$, where u_y is the displacement of the top boundary Γ_2. The complete curve with the displacement-controlled (DC) results, the points obtained in serial initialisation, and the line obtained by parallel corrections are presented on the left. The figures on the right present the points from different hierarchical levels at the inset depicted in the left diagram. The simulation is performed with a tolerance of $\varepsilon_l = \varepsilon_u = 10^{-3}$ and an increment length of $\Delta L = 0.05$.

in figure 6.12. Firstly, it can be seen that the curve computed using a serial ALM coincides with the curves obtained from DC simulations, but the ALM shows additional snap-through behaviour on the points where the DC curve has kinks. These kinks coincide with the instability in the metamaterial. Furthermore, the figure shows that the initial course approximation at level 0 is refined up to level 4 in the adaptive arc-length method scheme proposed in this chapter, provided the tolerance of $\varepsilon_l = \varepsilon_u = 10^{-3}$. The refinements of the adaptive scheme are mainly present in the highly curved segments of the load displacement curve. The reader is referred to Video 1 from the supplementary material of [584] for the deformations corresponding to the stress-strain curve in figure 6.13.

As for the other numerical experiments, the parallelisation properties of the ASPALM and APALM schemes are investigated. For the snapping meta-material simulation, the computational times are presented in table 6.5. The results in the table show high computational times for the ASALM (i.e., the ASPALM and APALM with 0 workers) compared to the DC simulation. However, the scalability observed in the previous benchmark problems can also be observed in the simulation of the snapping metamaterial. In fact, the APALM with 8 workers requires a factor of 4 less computational time, again equivalent to the computational time required for only the serial initialisation phase of the ASPALM. Lastly, the case of the snapping meta-material shows that, compared to a naturally serial displacement-controlled method, the APALM achieves a speed-up of a factor of 2.5 while providing snapping behaviour with greater accuracy.

Table 6.5: Computational time in [s] for the example of the snapping meta-material for the ASPALM and APALM for different numbers of worker processes. The computational time for a displacement-controlled (DC) simulation with step $\Delta u_y = 0.0005\,[mm]$ is provided as a reference. The times for the ASPALM are presented for the serial initialisation and the parallel correction phases, and the sum of the two is given as the total computational time. The numbers in the *Serial* column should theoretically be the same, but they provide a representation of the variation in the time measurements. The italic row with 0 workers denotes the ASALM method.

	ASPALM			APALM	DC
#	Serial +	Parallel =	Total	Parallel	Serial
0	*1,571.6*	*5,204.8*	*6,776.4*	*7,022.9*	*4,400.8*
1	1,686.9	4,593.2	6,280.1	5,319.1	
2	1,237.5	3,005.9	4,243.4	3,827.9	
4	1,742.7	1,548.2	3,290.9	2,137.3	
8	1,445.4	717.4	2,162.8	1,711.8	
16	1,931.1	352.2	2,283.3	1,632.9	
32	1,746.9	219.7	1,966.6	1,755.6	

6.6 Conclusions and Outlook

In this chapter, an Adaptive Parallel Arc Length Method (APALM) is presented. Contrary to existing parallel implementations of the Arc-Length Method (ALM), the present method provides within-branch parallelisation, hence providing scalable parallelisation independent of the physics of the problem, i.e., the number of branches. The method employs a multi-level approach, where parallel corrections are performed on solution intervals that have been initialised before. Given the sub-intervals provided by the serial computation, computations with finer arc lengths can be performed and evaluated using suitable error measures, marking intervals for further refinement when needed. Employing the multi-level approach, their implementations are discussed: the Adaptive Serial ALM (ASALM), the Adaptive Serial-Parallel ALM (ASPALM), and the APALM. The ASALM is a serial implementation, employing only the inherent adaptivity of the concept provided in this chapter. The ASPALM is a two-stage approach, separating a serial initalisation of the full equilibrium path from the parallel corrections. The APALM is a fully parallel implementation, where parallel corrections are performed as soon as the first path segments have been initialized. Conceptually, the APALM has a higher degree of parallelisation since the workers are not idle until the full solution curve is obtained. Given a basic step function and distance computation, the present chapter provides all algorithms necessary for implementing the APALM with manager-worker parallelisation.

The implementation of the APALM is evaluated using three benchmark problems and an application example. The first problem involves the collapse of a composite shallow cylindrical shell. The second problem involves the collapse of a truncated conical rubber shell, and the third example involves the bifurcation problem of a strip subject to an in-plane load. Moreover, the method is applied to the modelling of a snapping metamaterial to investigate its performance on a larger-scale problem. In all examples, it can be observed that the APALM provides an accurate description of the reference solution, given a (sufficiently) coarse serial initialisation of the curve. Through refinement, the APALM provides refinements (hence details in the solution), typically on sharp corners in the load-displacement diagrams. In addition, the bifurcation example also shows that the APALM is able to work within an exploration scheme for bifurcations. In all benchmark problems, the ASPALM and APALM have been used to evaluate the parallelisation of the schemes. The natural separation of the serial and parallel stages of the ASPALM reveals the scalability of the parallel correction with respect to the number of workers, showing that the parallel correction can take only a fraction of the total computational time for a larger number of workers. Furthermore, the comparison between the ASPALM and the APALM shows that the full parallelisation of the APALM provides a more efficient scheme than the two-stage approach of the ASPALM, as expected. The benchmarks and example also show that the APALM provides a full solution curve – including adaptive refinements – in the same computational time needed to compute only the initialisation of the ASPALM. This reveals the potential of the APALM: it can provide detailed solution paths without significantly increasing the computational time. The coarser the initial step size, the more arc-length intervals are computed during the parallel corrections of the method until a sufficient; hence, the higher the computational merit of the method to reach a desired level of detail. Moreover, the scaling analyses also show that the benefits of the APALM are already achieved with a small number of workers, e.g., 8 workers, making the APALM interesting on a desktop scale. For larger clusters, the APALM can be employed using dynamic load balancing within OpenMP.

As the APALM enables parallelisation in the arc-length domain, future applications of this method include quasi-static computations for solid and fluid dynamics, among other problems, especially those with a large number of load steps. Therefore, future works with this method include automatic exploration of solution spaces, e.g., following the work of [561, 630], or applications with large numbers of degrees of freedom, for instance with phase-field models for fracture mechanics [58]. Other future work includes combining the APALM with a spatial refinement scheme to enable space-quasi-time refinements. MPI scalability and distribution of cores per worker are topics to investigate for different applications. Another topic for further investigation is the convergence of the underlying arc-length method for large steps. Since a fewer number of initial intervals reduces the serial initialisation time of the APALM, the parallel performance can be increased significantly when the initial step size is maximized. For example, the Mixed Integration Point (MIP) method increases the convergence of the ALM, allowing for larger step sizes. The performance of the MIP is demonstrated for isogeometric Kirchhoff–Love shells in [340, 343, 371, 372]. Lastly, since the presented APALM is developed for path-independent problems, an extension to path-dependent problems is a natural direction for future research.

6.A Result Reproduction

For the sake of reproducibility of the results in this chapter, this appendix provides brief instructions on the use of the software developed along with this thesis. The full software is available as part of the Geometry + Simulation Modules. For more detail on the contributions to this software library, and its installation, the reader is referred to chapter 8.

Table 6.6 provides per figure in this chapter the name of the file to run along with the arguments to be passed to obtain these figures. problem. For the APALM to work in parallel, G+Smo needs to be compiled with MPI.

Table 6.6: File name and run arguments required for the reproducibility of the figures in this chapter. Arguments with a single dash (-) require an argument. See chapter 8 for more detail about the software and installation instructions.

Figure	**Run File**		
	Arg.	**Description**	**Values**
Figure 6.7	benchmark_Roof_APALM		
Table 6.2	-N	Number of arc-length steps	50: Table 6.2a 600: Table 6.2b
	-L	Arc-length	30: Table 6.2a, 2.5: Table 6.2b
	-l	Maximum number of APALM levels	6: Table 6.2a, 10: Table 6.2b
	-T	APALM tolerance	1e-2: Table 6.2a, 1e-4: Table 6.2b
Figure 6.9	benchmark_Frustrum_APALM		
Table 6.3	-N	Number of arc-length steps	80: Table 6.3a 640: Table 6.3b
	-L	Arc-length	0.5: Table 6.3a, 0.0625: Table 6.3b
	-l	Maximum number of APALM levels	6: Table 6.3a, 10: Table 6.3b
	-T	APALM tolerance	1e-2: Table 6.3a, 1e-4: Table 6.3b
Figure 6.11	benchmark_Beam_APALM		
Table 6.4	-N	Number of arc-length steps	7: Table 6.4a 550: Table 6.4b
	-L	Pre-buckling arc-length	5e-5: Table 6.4a, 5e-7: Table 6.4b
	-l	Post-buckling arc-length	2.5: Table 6.4a, 2.5e-2: Table 6.4b
	-l	Maximum number of APALM levels	6: Table 6.4a, 10: Table 6.4b
	-T	APALM tolerance	1e-3: Table 6.4a, 1e-6: Table 6.4b
Figure 6.13	snapping_example_shell_APALM		
Table 6.5	snapping_example_shell_DC		
	-L	Load-step size	5e-4: DC, 5e-2: ALM

7

A Comparison of Smooth Multi-Patch Basis Constructions for Isogeometric Analysis

Using spline as a basis for analysis, Isogeometric Analysis (IGA) provides high efficiency in terms of degrees of freedom compared to Finite Element Methods (FEMs). Furthermore, the high continuity of spline bases allows for variational formulations with higher-order derivatives, such as the isogeometric Kirchhoff–Love shell, which requires a C^1-continuous basis. The construction of a C^1-smooth spline basis on a simply connected domain is straightforward by using the tensor-product basis construction, as highlighted also in section 2.2.1. However, for complex domains, e.g., domains with holes, C^1-smooth isogeometric analysis is more complicated and can be done using several approaches. In this chapter, a review of methods to perform IGA on complex domains is presented, and a set of four unstructured spline constructions for IGA – in particular, the D-Patch, Almost-C^1, Analysis-Suitable G^1, and the Approximate C^1 constructions – are compared qualitatively and quantitatively. The goal of this comparison is to provide insights into the selection of methods for practical problems as well as directions for future research. In the qualitative comparison, the properties of each method are evaluated and compared. In the quantitative comparison, a selection of numerical examples is used to highlight the different advantages and disadvantages of each method. In brief, this chapter concludes that among the considered methods, there is not one best method and some recommendations for future developments in unstructured spline methods for isogeometric analysis are given. If necessary, the reader is referred to section 2.2.1 for preliminary information on splines.

This chapter is published as:
[589] **H. M. Verhelst**, P. Weinmüller, A. Mantzaflaris, T. Takacs & D. Toshniwal, “A Comparison of Smooth Basis Constructions for Isogeometric Analysis”, *Computer Methods in Applied Mechanics and Engineering* **419**, 116659 (2024)

7.1 Introduction

Present-day engineering disciplines depend on Computer-aided design (CAD) and numerical simulation models for physics for design and analysis. Typically, geometries designed in CAD are converted to meshes for analysis with numerical techniques like Finite Element Methods (FEMs). Since the geometry description in CAD is based on splines, whereas meshes for simulation are based on linear geometry approximations, geometric data is lost during this conversion. Isogeometric analysis [268] is the bridge between CAD and Computer-Aided Engineering (CAE), since it employs splines as a basis for geometric design and numerical analysis. In practice, an isogeometric analysis and optimisation workflow can be seen as depicted in figure 7.1. Starting with a geometry from CAD, as well as material parameters, boundary conditions, et cetera from CAE, isogeometric simulations, and eventually geometry or topology optimisation can be performed. The step connecting the inputs from CAD and CAE is referred to as *IGA Setup* in figure 7.1. This step takes care of the preparation for the simulation step, including the pre-processing of the geometry, if needed, and the construction of the isogeometric discretisation space.

Due to the arbitrary smoothness of spline basis functions, isogeometric analysis has several advantages over conventional finite element methods. For example: (i) the introduction of k-refinements, which are proven to provide high accuracy per degree of freedom [72, 488]; (ii) high accuracy in eigenvalue problems, e.g., for structural vibrations [122, 269, 270]; or (iii) geometric exactness in parametric design and interface problems, e.g., applied to the parametric design of prosthetic heart valves [646]. Furthermore, the C^1-smooth discretisation spaces allow to solve equations such as the biharmonic equation, the Cahn–Hilliard equations, or the Kirchhoff–Love shell equations without introducing auxiliary variables. However, due to the tensor-product structure of the spline basis, higher-order smoothness can be enforced easily only on domains that allow simple patch partitions (e.g., an L-shape or an annulus), whereas on geometrically and topologically more complicated domains, alternative approaches are required to solve equations that require basis functions of higher-order continuity.

For more complicated domains, the *IGA setup* block in figure 7.1 involves a pre-processing step of either the geometry, the system of equations, or the solution space to solve the original system of equations. In figure 7.2, this pre-processing step is subdivided into three options: trimmed domain approaches, unstructured splines, and variational coupling methods. Given an initial geometry (cf. figure 7.3a), the trimmed domain approaches alter the tensor-product domain by defining parts of the domain that are physical or non-physical (cf. figure 7.3b). In the case of unstructured splines or variational coupling methods, the geometry is decomposed into multiple different patches (cf. figure 7.3c), on which continuity conditions are enforced by constructing a smooth basis (unstructured splines) or by adding extra terms to the system of equations (variational coupling approaches). In section 7.2 of this chapter, a review of trimmed domain approaches, unstructured splines, and variational coupling methods is provided. Examples include immersed methods, degenerate patches, and Nitsche's method, respectively. In the case of simple geometries (and given the right inputs), the methods are identical.

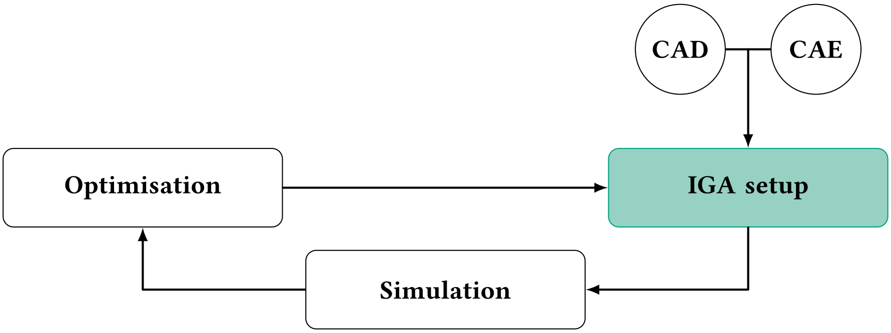


Figure 7.1: General workflow for solving a physics problem and optimising a geometry or topology coming from CAD and CAE processes. Starting from *CAD* and *CAE*, the *IGA Setup* is performed. In this block, a computational basis is extracted from the geometry to be used for simulation. Then, the *Simulation* block involves the assembly of the operators of the physics problem on the computational basis coming from the *IGA Setup*. In cases of shape or topology optimisation problems, the simulation results are evaluated, and the shape or topology is modified. From this changed shape and topology, a new computational basis can be obtained, and the process can be repeated. The *IGA Setup* block is marked to be elaborated further on in figure 7.2.

As shown in figure 7.2, each class of methods has its own characteristics, and previous work has provided several comparisons of methods among each other, which are elaborated more in section 7.2. In the context of the workflow sketched in figure 7.1, unstructured splines provide a valuable alternative to the other methods since they are constructed for a fixed topology, and hence the computational costs of their construction are not related to changing shapes or moving domains. However, recent developments have mainly focused on different unstructured spline methods separately, rather than providing a valuable comparison. Therefore, qualitative and quantitative comparisons of a selection of unstructured spline constructions are provided. Finite, piece-wise polynomial spline constructions are considered, hence not including rational constructions or infinite representations, such as subdivision surfaces. More precisely, examples of (globally) G^1-smooth multi-patch constructions (the Analysis-Suitable G^1 construction of [181] and the Approximate C^1 construction of [618]), the D-Patch method of [569] and the Almost-C^1 construction of [534] are compared, motivated in section 7.2.3. The selected methods are qualitatively compared based on their properties and quantitatively based on several different examples with biharmonic and Kirchhoff–Love shell equations. The aim of this chapter is to provide a fair comparison[1] of these methods, providing a good overview of the strengths and weaknesses of each method in different cases.

The chapter is outlined as follows: In section 7.2, a detailed overview of the methods appearing in figure 7.2 is provided. In section 7.3, a qualitative analysis of the four constructions that are discussed in this chapter is provided, while section 7.4 provides a quantitative analysis of all methods. There, five benchmark problems solving either a biharmonic or a Kirchhoff–Love equation are provided. These benchmark problems serve

[1] The authors of the original chapter believe that a comparison like the one presented in this chapter is never fully unbiased, since the authors have contributed to different methods in previous publications and do not represent the entire research community.

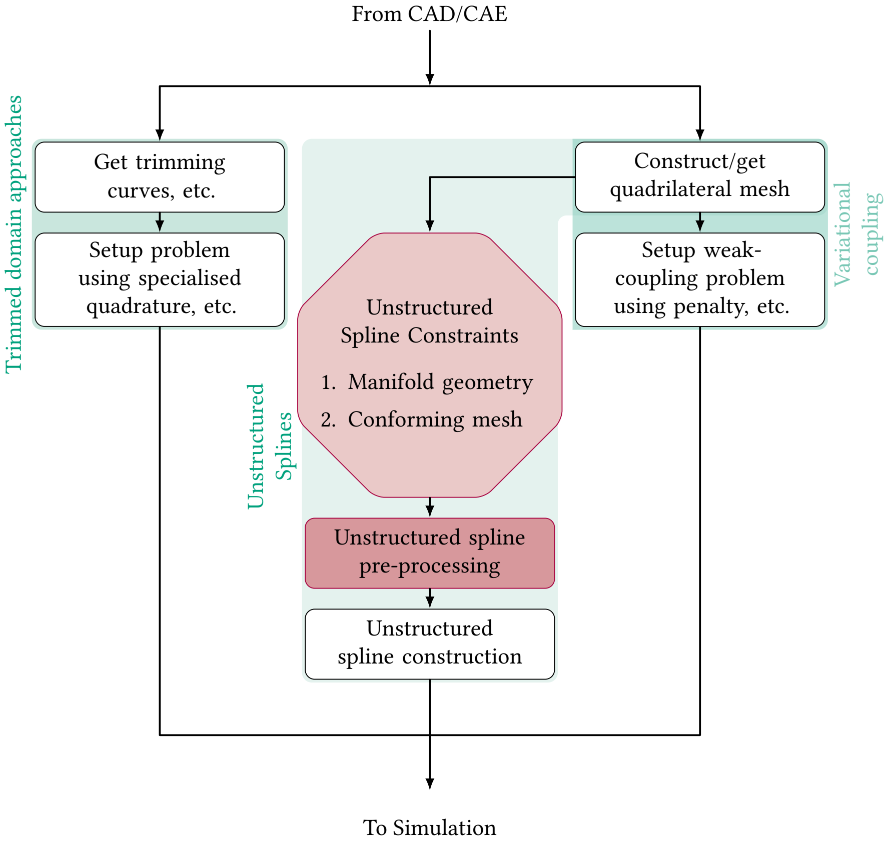

From CAD/CAE
Trimmed domain approaches
Get trimming curves, etc.
Setup problem using specialised quadrature, etc.
Construct/get quadrilateral mesh
Setup weak-coupling problem using penalty, etc.
Variational coupling
Unstructured Spline Constraints
1. Manifold geometry
2. Conforming mesh
Unstructured Splines
Unstructured spline pre-processing
Unstructured spline construction
To Simulation

Figure 7.2: Inside the *IGA Setup* block from figure 7.1, three methods are distinguished. Firstly, *trimmed domain approaches* use trimming curves or surfaces to identify parts of a tensor-product domain as the actual domain. However, since elements can be trimmed poorly, *specialised quadrature* rules and solver preconditioners are typically needed. Alternatives to trimming are *weak coupling* or *unstructured spline* methods. For both classes of methods, a geometry with a given topology needs to be decomposed into multiple sub-domains (i.e., patches) via *quadrilateral meshing*. Given a quadrilateral mesh, *weak methods* assemble extra penalty terms into the equation to be solved or add extra equations to be solved to satisfy continuity constraints. Lastly, *unstructured spline* constructions can be used to couple multiple domains by constructing a continuous basis. These methods, however, can only be used on *manifold geometries* and *conforming meshes*. When these requirements are satisfied, *unstructured spline pre-processing* is required before the *unstructured spline construction* can take place. The *pre-processing* is highlighted and will be elaborated on more in figure 7.7 in section 7.3.

different purposes, and they are compared to determine which method, in which setting, performs best. In section 7.5, this chapter is concluded based on the findings from the previous sections, and directions for future research are provided.

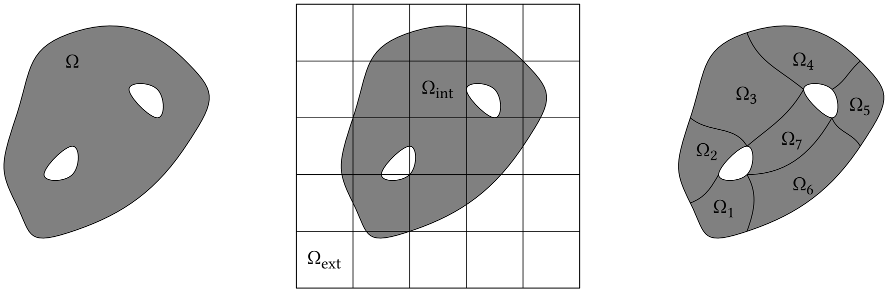
7.2 Multi-Patch Isogeometric Analysis: Literature Review

As discussed in the introduction of this chapter, in particular in figure 7.2, three classes of methods for the modelling of complicated domains can be characterised: trimmed domain approaches, variational coupling methods, and unstructured splines. The goal of all methods is to achieve a certain level of continuity across the whole analysis domain such that multi-patch isogeometric analysis can be performed, for example, for the Kirchhoff–Love shell model [319], the biharmonic equation, or the Cahn–Hilliard equation [207]. As shown in figure 7.3, trimmed domain approaches use the fact that parts of tensor-product geometries are trimmed away, using trimming curves to separate regions of interest and regions that should be omitted (see figure 7.3b). Variational coupling approaches and unstructured splines are defined on multi-patch domains, typically following from a segmentation of the original domain; see figure 7.3c. In the case of variational coupling methods, the system of equations is enriched with terms that will enforce continuity (typically in a weak sense) between the patches. In the case of unstructured spline constructions, a basis is constructed on the multi-patch object, where certain smoothness is enforced strongly. When starting from a trimmed geometry, the step of creating a multi-patch domain decomposition (i.e., *untrimming*) from an arbitrary geometry with an arbitrary topology is a very important step in the application of weak coupling methods and unstructured spline constructions, as can be seen in the flowchart in figure 7.2. In this chapter, however, the topic of untrimming will not be discussed as it is outside the scope of this study. Hence, the reader is referred to [243, 381] for an overview of these methods.

In this section, an overview of the trimmed domain approaches (section 7.2.1), variational coupling methods (section 7.2.2), and unstructured splines (section 7.2.3) is provided. A fourth method, which will not be discussed in this section, is to introduce auxiliary variables for derivatives of the solution so that C^1 continuity requirements are reduced to C^0 and standard interface coupling can be used. These so-called mixed formulations are common in conventional FEM, although recent advances have also been made for Kirchhoff–Love plates and shells and the biharmonic eigenvalue problem [326, 453, 454].

7.2.1 Trimming Approaches

Trimming is a technique where so-called trimming curves or surfaces separate parts of tensor-product spline domains to define a geometry. Trimming is a common technique to represent complex geometries in CAD, and typically, geometries consist of multiple trimmed patches with boundary and interface curves trimming the actual patches. The reader is referred to the work [379] for an overview of trimming methods in isogeometric analysis. Generalising the idea of trimming to techniques where curves or surfaces are used to define the domain of interest as *trimmed domain approaches*, several approaches have been proposed to perform simulations on complex geometries, including the finite



(a) Initial geometry (b) Trimming (c) Domain segmentation

Figure 7.3: Given an initial geometry Ω (a), trimming (b) uses the curves of the boundary of the original geometry to define the interior domain Ω_{int} and the exterior domain Ω_{ext}. An alternative approach for modelling the domain is to use domain segmentation (c). Here, the domain is decomposed into several patches Ω_i which together define the full domain Ω.

cell method [173, 431, 492], Cut-FEM [81] or immersed methods [261, 297]. The advantage of these methods is that the trimmed CAD geometries could be directly used for analysis. However, when only small parts of the physical domain are cut, leading to small cut elements, numerical difficulties can occur in the conditioning of the system, leading to solver instabilities or accuracy problems [145]. Therefore, the analysis of complex trimmed geometries via methods like the FCM typically requires special quadrature schemes to take into account small cut cells [156] or preconditioners to stabilise the numerical analysis [144]. In the context of Kirchhoff–Love shell modelling, isogeometric analysis on trimmed geometries has been performed in several studies [116, 219, 223], including some with a focus on multi-patch coupling [115, 116, 118, 224, 335].

7.2.2 Variational Coupling Methods

In this chapter, variational coupling methods are defined as methods that modify the system of equations to enforce certain continuity across patch interfaces. Examples of these methods are penalty methods, Nitsche's methods, mortar methods, or Lagrangian penalised methods. In the context of Kirchhoff–Love shell analysis, these weak coupling methods have received a lot of attention in previous studies, and an overview is provided by [433]. Firstly, an in-plane coupling was proposed in [316] together with a method for coupling non-manifold patches using the so-called bending strip method [318]. Later, weak coupling approaches were developed for multi-patch domains. Here, coupling terms can be added inside the existing variational formulation (referred to as Nitsche's or penalty methods) or imposed by Lagrange multipliers (referred to as mortar methods).

Several works on Nitsche techniques (cf. [412]) for isogeometric analysis have been published, starting from the imposition of boundary conditions [481], towards multi-patch coupling and the coupling of patches [482], and later using a non-symmetric parameter-free Nitsche's method [491]. Nitsche's methods have been applied to Kirchhoff plates [262], Kirchhoff–Love shells [47, 222, 223], hyperelastic 2D elasticity [170] and the biharmonic equation [618, 619] and for modelling local subdomains [61] for elasticity simulations. The advantages of Nitsche's methods are that the formulation is variationally consistent and requires only mild stabilisation, which can be performed automatically by estimating the stability parameter. However, the involved integral terms are complicated expressions that impose high implementation and assembly efforts. Therefore, coupling approaches using only penalisation have been developed [69, 172, 238, 342, 432]. Although several improvements have been made in these works, the main disadvantage of penalty methods is that a suitable penalty parameter has to be chosen. Using the super penalty approach [118, 119], the computation of the penalty parameter can be automated. However, this method has not yet been tested for non-linear shell problems or on 'dirty' geometries. Both Nitsche's and penalty methods can be used to couple geometries that are non-manifold, i.e., geometries that have out-of-plane connections like stiffened structures, by penalising changes in the angle of patches on an interface. Furthermore, the methods can handle interfaces with non-matching parameterisations.

Instead of adding coupling terms in the variational form, as is done in Nitsche's and penalty methods, mortar methods [48] add extra degrees of freedom by introducing Lagrange multipliers, which are required to resolve additional coupling conditions. The use of mortar methods to couple non-conforming isogeometric sub-domains was first done by [240]. In [165], the FEA-based approach of [48] was extended for NURBS-based IGA, but the aim was to develop a method for C^0-coupling for Reissner-Mindlin shells, hence insufficient for isogeometric Kirchhoff–Love shells. A mortar method aiming to establish C^1 coupling is given in [62], and a method that provides C^n continuity was given by [154, 155]. Furthermore, G^1 mortar coupling, referred to as extended mortar coupling, was presented in [496] for Kirchhoff–Love shells, based on a coupling in the least squares sense. On the other hand, in [46] a mortar method to enforce C^1 coupling for the biharmonic equation was developed, where the Lagrange multiplier spaces are constructed similarly to [73] for C^0-coupling. An approach to reduce computational costs involved in finding Lagrange multipliers is called dual mortaring [623], where Lagrange multipliers are eliminated using a compact dual basis. This approach has been developed for Bezier elements [664] and it has been applied for Kirchhoff–Love shells [386], and a bi-orthogonal spline space has been presented for weak dual mortaring for patch coupling [636]. In [255], a hybrid method was provided and applied to Kirchhoff plates, which combines mortar methods and penalty methods. Lastly, a comparison of Nitsche, penalty, and mortar methods is given by [14]. For a more complete overview of mortar methods for isogeometric analysis, the reader is referred to [241]. In general, mortar methods have the advantage over Nitsche's methods that there are no parameters involved and that the implementation efforts are lower. However, the disadvantage is that a suitable spline space needs to be found for the Lagrange multipliers [46, 73, 164]. Like Nitsche's and penalty methods, mortar methods can handle non-matching parameterisations and non-manifold interfaces, the latter by similar penalisation of interfacing patches.

7.2.3 Unstructured Splines

Compared to weak coupling methods, unstructured spline constructions do not alter the system of equations to be solved. Instead, the computational basis is modified so that it satisfies continuity conditions across patch interfaces. Unstructured splines are typically constructed for in-plane (i.e., manifold) interfaces and not for out-of-plane (i.e., non-manifold) interfaces, since the notion of smoothness is uniquely defined only in the former setting. However, unstructured spline constructions for non-manifold interfaces are possible, e.g., as in [101, 398, 653] in the context of subdivision. Furthermore, unstructured spline constructions are typically constructed on conforming interfaces, i.e., interfaces with matching meshes, but, as long as the patch parameterisations are matching, this can be overcome by taking the knot vector union of the interface patches. However, the advantage of unstructured spline constructions is that as soon as the basis is constructed for a certain untrimmed geometry, there are no additional costs involved other than evaluation costs for changing shapes, which makes unstructured spline bases suitable for shape optimisation problems. In the case of topology changes or large changes in shape, however, the mesh topology of the unstructured spline space has to be changed as well. Unlike weak methods, which are typically based on the introduction of penalties (e.g., in terms of energy), unstructured spline constructions are typically provided as generic geometric methods that are applicable to any equation that requires C^1 coupling across multi-patch interfaces. With the advance of isogeometric analysis, interest in parametrically C^1 and geometrically G^1 splines has grown. An overview of smooth multi-patch discretisations for isogeometric analysis can be found in [271], and a small overview is provided below. Methods enforcing parametric continuity, i.e., the type of continuity between mesh elements within a regular tensor-product spline patch, and methods providing general geometric continuity, cf. [213], are distinguished. In the following, three types of constructions are classified, depending on their continuity on patch interfaces, around vertices, and in the patch interior:

- Patch coupling with geometric continuity on patch interfaces and parametric continuity inside patches.
- Patch coupling with parametric continuity everywhere.
- Patch coupling with parametric continuity almost everywhere.

Although other constructions outside of these categories exist, e.g., [373, 654], our review is restricted to the aforementioned categories since the methods considered in sections 7.3 and 7.4 fall into these categories.

Geometric continuity on patch interfaces and parametric continuity inside patches

This first category of unstructured spline constructions assumes that a fixed C^0-matching multi-patch parametrisation is given. On this multi-patch domain, a C^1-smooth isogeometric space is constructed. As shown in [213], for any isogeometric function, the C^1 condition over each interface is equivalent to the G^1 geometric continuity condition of the graph surface corresponding to the function. If the domain is planar and the patches are bilinear, then the C^1 constraints can be resolved, and a C^1 spline space was constructed by [308] and applied to the isogeometric analysis of the biharmonic equation in [302]. It could be shown in [306] and [214] that C^1 splines over bilinear quadrilaterals and mixed (bi)linear quadrilateral/triangle meshes possess optimal approximation properties. Furthermore, the work [307] studied the arbitrary C^n-smooth spline space for bi-linear multi-patch parameterisations, based on their previously published findings.

Considering general C^0-matching multi-patch domains, the work of [109] introduces the class of analysis-suitable G^1 (AS-G^1) multi-patch parameterisations which includes bi-linear patches. This AS-G^1 condition is in general required to obtain optimal approximation properties. The condition implies that the glueing data for G^1 continuity is linear, which is explained in more detail in section 7.3.1. While it could be shown in [304] that all planar multi-patch domains possess AS-G^1 reparametrisations, creating AS-G^1 surface domains is more difficult. Several strategies to achieve this were introduced in [181], thus making C^1-smooth multi-patch parameterisations applicable to biharmonic equations and isogeometric Kirchhoff–Love shell models [182]. In the work of [463], the construction of [109] is used to develop a scaled-boundary model for smooth Kirchhoff–Love shells, similar to the approach of [15] for Kirchhoff plates.

Alternatively, by constructing an AS-G^1 parametrisation, one can relax the smoothness conditions. This was done in [618], where the construction of an approximate C^1 (Approx. C^1) space is presented. The basis construction is explicit and possesses the same degree-of-freedom structure as an AS-G^1 space, but the C^1 condition is not satisfied exactly but only approximately. It defaults to the AS-G^1 construction when the AS-G^1 requirements are met. In [619], a comparison of the presented space with Nitsche's method was performed, yielding optimal convergence results without the need for coupling terms. More details on the Approx. C^1 method are provided in section 7.3.2.

Parametric continuity everywhere

The starting point for this class of constructions is different from the previous. Here, smooth splines are created in a parametric sense between neighbouring mesh elements. Such parametric C^1 conditions are easy to resolve, but they lead to singularities at vertices of valencies other than four, so-called extraordinary vertices. This is due to the conflicting coupling conditions on partial derivatives around the EVs, which lead to all partial derivatives vanishing there. Inspired by the Degenerate Patch (D-Patch) approach from [464], the works of [408, 569] provide C^1 smooth spline spaces for multi-patch geometries with parametric smoothness everywhere. On extraordinary vertices (EVs), which is a junction between 3 or 5 or more patches (i.e., valence $\nu > 2, \nu \neq 4$), the original D-Patch method shows a singularity of the basis in EVs combined with a reduction of degrees of freedom at this point. An improvement of the D-Patch method was presented in [408] by splitting elements around the EVs such that every element is associated with four degrees of freedom. However, this construction does not have non-negativity and is based on PHT splines, which have limited smoothness. A new design and analysis framework for multi-patch geometries was presented in [569], based on D-Patches with T-splines for refinement and non-negative splines yielding optimal convergence properties. This was also demonstrated in [89] for isogeometric Kirchhoff–Love shells. In the work [615], it is motivated that this construction can also be used if only one element around the EV is isolated. More details on the D-Patch method are provided in section 7.3.3.

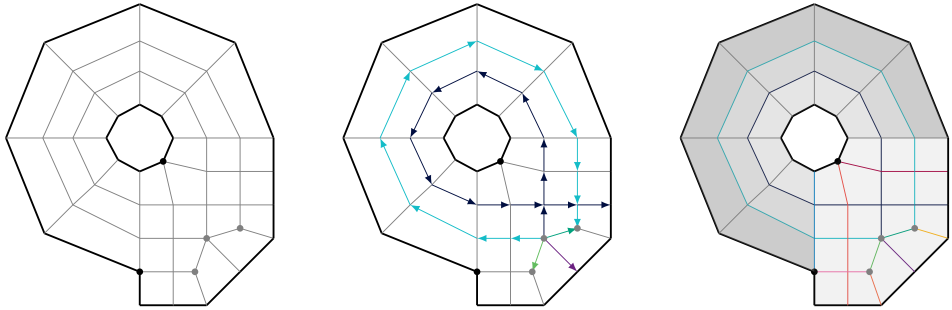
Alternatively, subdivision surface-based constructions lead to unstructured splines that are parametrically continuous everywhere, cf. [30, 31, 426, 468, 656]. However, such approaches require an infinite number of polynomial pieces around each EV. Thus, they are discarded from the comparison. Moreover, in general, their approximation properties are severely reduced near EVs [533].

Parametric continuity almost everywhere
As mentioned previously, imposing parametric continuity everywhere leads to singularities at all EVs. Thus, instead of constructing a space with full parametric continuity, spaces with parametric continuity almost everywhere except around the EVs can also be considered. This way, one ends up with regular, smooth rings around EVs, which then need to be filled in some way. Such so-called *hole-filling* techniques are commonplace in geometric modelling and can also be used to construct smooth spaces for isogeometric analysis, cf. [309–313, 407]. This chapter focuses here on the simplest possible way of resolving this issue, which is to enforce only C^0-smoothness near the EVs and G^1 at the EV, namely the Almost-C^1 construction proposed in [534]. Similar constructions, which enforce no additional smoothness near EVs, were proposed for mixed quadrilateral and triangle meshes in [567] and for arbitrary degree multi-patch B-splines with enhanced smoothness (MPBES) in [74].

The Almost-C^1 construction considered here yields piece-wise biquadratic splines that are C^1 in regular regions and which have reduced smoothness around extraordinary vertices, independent of the valence or the location (i.e., interior or boundary EVs). In contrast to that, most commonly used hole-filling approaches yield exactly C^1-smooth spaces but introduce locally polynomials of higher degree or require a higher degree to start with, such as the construction presented in [378], which converts Catmull–Clark subdivision surfaces to G^1-smooth piece-wise biquintic elements. While exact smoothness is relevant for geometric modelling, it is not necessary from an analysis point of view.

7.3 Qualitative Comparison

In the qualitative comparison of this chapter, the focus is on the properties of different unstructured spline constructions and their implication on the application of these constructions in a workflow, as in figure 7.1. More precisely, the continuity of each construction and their nestedness properties are discussed, and the aim is to provide a set of requirements for the *unstructured spline pre-processing* block in figure 7.2. Since the qualitative comparison of the considered methods in this chapter mostly covers the properties of the methods and their implications, mathematical details about the construction or convergence properties are not provided. For more details, the reader is referred to [181] for the Analysis-Suitable G^1 (AS-G^1) method, which extends the 2D construction from [305], to [618] for the Approximate C^1 (Approx. C^1) method, to [569] for the Degenerate Patches (D-Patch), and to [534] for the Almost-C^1 method. However, for the qualitative comparison, some key terms are introduced as preliminaries.



(a) A simple mesh with boundary edges in black and interior edges in gray. The boundary extraordinary vertices (bEVs), i.e., the vertices on a boundary with valence $v \geq 3$, are denoted by a black circle, and the interior extraordinary vertices (iEVs), i.e., interior vertices with valence $v \geq 4, v \neq 4$, are denoted by grey circles.

(b) Illustration of the interface tracing procedure. From each EV, all outgoing edges are traced as illustrated until another EV or a boundary is hit.

(c) Result of interface tracing from all the EVs. Every patch is now bounded by a set of boundary and traced interface curves. All patch corners are corners where a traced interface and/or a boundary edge form a corner. Along the hole, different patches are indicated with different shades of gray. In the part bottom-right of the hole, every face forms a patch since all traced curves denoted by colours intersect with other traced curves.

Figure 7.4: Procedure to find a multi-patch segmentation from a given mesh. The original mesh in (a) has 46 vertices, 81 edges, and 45 faces, and the final multi-patch (c) has 20 patches.

Firstly, a *quadrilateral mesh* (*quad mesh*) is a mesh of quadrilateral elements, representing a (planar) surface geometry. The quadrilaterals can be represented by tensor B-splines of any degree, which can be mapped onto a parametric unit square. Typically, when the tensor B-spline quadrilaterals have different sizes in different directions or even different refinement levels, assemblies of these *patches* are typically referred to as *multi-patches*. An example of a multi-patch is given in figure 7.3c. The conversion of a quad-mesh with many elements to a multi-patch with a smaller number of patches derived from groups of elements can be done using the procedure described in figure 7.4. Here, a half-edge mesh is traversed, and elements are collected into groups corresponding to final patches. The vertices of the elements in one group (i.e., patch) form the control net of the bi-linear patch.

Secondly, for parametrically smooth constructions, different classes of vertices are considered. For so-called *extraordinary vertices* (*EVs*) , these constructions are typically different. An *interior extraordinary vertex* (*interior EV*) is a vertex on a quad mesh on which three or more than four patches meet. The number of patches coming together at a vertex is referred to as the *valence*, denoted by v. Furthermore, a *boundary extraordinary vertex* (*boundary EV*) is a vertex on the boundary of the quad mesh with valence $v \geq 3$. For geometrically smooth constructions, the construction depends on the geometry around the vertex rather than the valence of the vertex. Hence, for these constructions, the notion of EVs is irrelevant.

Lastly, a refinement of a spline space is called *nested* if the refined spline space is fully contained in the unrefined space. As a consequence, the geometry is exact under element refinement, which is beneficial from an analysis point of view.

7.3.1 Analysis-suitable G^1

The analysis-suitable G^1 (AS-G^1) construction is a novel approach in isogeometric analysis that was introduced for planar geometries and surfaces in [109], but a construction that extends [305] for planar domains to surfaces is detailed in [181]. This construction ensures that basis functions at interfaces have C^1 continuity, while basis functions at vertices have C^2 continuity. The approach is based on the concept that G^k-smooth surfaces can produce C^k-smooth isogeometric functions [213]. When dealing with general C^0-matching multi-patch domains, the so-called AS-G^1 conditions must be satisfied to ensure optimal approximation. If these conditions are met, a C^1-smooth subspace of the isogeometric space can be constructed that is sufficiently large. Such geometries are referred to as analysis-suitable geometries. However, it should be noted that the C^1-smooth multi-patch isogeometric space generally depends on the geometry, as discussed in [303]. To overcome this issue, an Argyris-like space was proposed in [305], which has a dimension that is independent of the geometry.

Given an interface between two patches, the C^1 continuity condition at the interface is defined by a linear combination of tangent vectors and transversal derivatives, which is referred to as gluing data [109]. The C^1 smooth basis functions at the interface, or more generally at the edge, can be described by the first-order Taylor expansion of the trace and the transversal derivative. It is shown in [109] that the ideal choice for the space-representation of the trace and transversal derivative is $\mathbb{S}(\boldsymbol{p}, \boldsymbol{r}-1, \boldsymbol{h})$[2] and $\mathbb{S}(\boldsymbol{p}-1, \boldsymbol{r}-2, \boldsymbol{h})$, respectively. These basis functions have local support and are linearly independent, but they depend on the glueing data and, therefore, on the geometry reparameterisation itself. To ensure that the basis functions form a C^1-smooth subspace of the isogeometric space and to maintain the nestedness of the spline spaces, it is necessary to have glueing data as a linear function that fulfils all analysis-suitable geometries. For instance, all bi-linear patches meet this requirement. However, if a geometry is not analysis-suitable, it can be reparameterized using the technique presented in [304].

For any vertices in the quad mesh, describing the C^1 condition is not that straightforward. In order to keep it general, the vertex basis functions are constructed by the C^2 interpolation using the C^1 basis functions from the corresponding edges. As a consequence, the vertex basis functions also have local support and are linearly independent.

Summarising, the AS-G^1 construction can be constructed by three different, linearly independent sub-spaces: interior, edge, and vertex space. They can be described as follows:

- Interior space: basis functions that have zero values and derivatives on the patch edges and vertices.
- Interface space: basis functions that have vanishing function values up to the second derivatives at the vertices.

[2] The notation $\mathbb{S}(\boldsymbol{p} = (p,p), \boldsymbol{r} = (r,r), \boldsymbol{h} = (h,h))$ indicates a two-dimensional spline space with p as the polynomial degree, r as the regularity, and h as the mesh size in both directions.

- Vertex space: C^2 interpolating functions at the vertex, i.e., basis functions that have non-vanishing C^2 data at the vertex.

The AS-G^1 construction with the interface and vertex constructions as described above is fully C^1 over the whole domain. In addition, the AS-G^1 construction can only be constructed when the degree of the basis is $p \geq 3$ and the regularity is reduced to $r \leq p-2$.

Figure 7.5a presents a local region around and EV with valence five with line styles indicating different continuity levels on patch or element boundaries (see the caption of figure 7.5). For the AS-G^1 construction, the continuity at the vertex is C^2 by construction. Furthermore, the continuity at the interior element interfaces is C^{p-2} due to the restriction on keeping the isoparametric concept. Lastly, since the AS-G^1 construction provides a G^1 surface, the patch interfaces are C^1 by construction [213].

In sum, the core ideas behind the AS-G^1 construction are as follows:

- **Degree, regularity, continuity**
  The spline space is fully C^1, hence suitable to solve fourth-order problems. However, the computation of the space requires analysis-suitability of the parameterisation as well as degree $p \geq 3$ and regularity $r \leq p-2$ for the basis functions.

- **Limitations on construction**
  The space can be constructed on fully unstructured quadrilateral meshes with both interior and boundary extraordinary vertices. The construction of the basis functions is independent of the location or valence of the EVs. However, the analysis-suitability condition imposes a requirement on the geometries on which the construction can be constructed. Furthermore, the geometry parameterisation is not changed.

- **Nestedness**
  The spline spaces are nested.

- **Refinement procedure**
  The refinement procedure is standard (by knot insertion) since the parameterisation does not change.

7.3.2 Approximate C^1

The Approximate C^1 construction [618] provides, as the name suggests, approximately C^1 continuity on interfaces and vertices; more precisely, the construction provides C^1 continuity in the refinement limit. The Approx. C^1 construction shares similarities with the AS-G^1 construction, but the main difference between the construction of the Approx. C^1 and the AS-G^1 spaces is that it relaxes the AS-G^1 condition on the geometry, i.e., it allows geometries with non-linear glueing data. In fact, the exact glueing data are splines with a higher polynomial degree and lower regularity, or even piecewise rational. As a consequence, trying to extend the construction for AS-G^1 parameterisations directly to non-AS-G^1 geometries yields complicated basis functions that are challenging to evaluate and integrate accurately. To overcome this issue and obtain a construction with more easily definable basis functions, the glueing data are approximated. However, this approximation means that the C^1 condition is no longer satisfied exactly but only approximately.

By utilising the approximation of the glueing data, the Approximate C^1 construction incorporates the concept of different spline spaces found in the AS-G^1 construction. In this case, the interior, vertex, and interface basis functions fulfil the same conditions as in the AS-G^1 construction, but the degree and regularity differ between these spaces. Specifically, the sub-spaces for the AS-G^1 construction have $p \geq 3$ and $r \leq p-2$, while the Approximate C^1 construction employs an interior space with $p \geq 3$ and $r \leq p-1$, along with vertex and interface spaces that have locally reduced smoothness based on the approximation of the glueing data. Consequently, on the one hand, the Approximate C^1 construction restores the potential for maximal smoothness of isogeometric functions in the refinement limit, but the nestedness of the basis is lost. On the other hand, the approximation of the glueing data in the Approximate C^1 construction does not require analysis-suitability for the optimal convergence rate, unlike the AS-G^1 construction. This feature makes the method applicable to more complex geometries. When the Approximate C^1 construction is applied to an analysis-suitable geometry with $p \geq 3$ and $r \leq p-2$, and the glueing data approximation is exact, the construction becomes equivalent to the AS-G^1 construction.

Figure 7.5b presents a local region around and EV with valence five with line styles indicating different continuity levels on patch or element boundaries (see the caption of figure 7.5). For the Approx. C^1 construction on a fully smooth basis ($p \geq 3$ and $r = p-1$), the interior basis recovers full smoothness on element boundaries, hence C^{p-1} continuity. In the shaded region around the interfaces and the EV, the continuity is locally reduced by the construction of the locally reduced continuous subspace and the approximation of the glueing data. Similar to the AS-G^1 construction, the continuity on the EV is C^2 by construction, and the element boundaries are C^1 approximately.

In sum, the core ideas behind the Approx. C^1 construction are as follows:

- **Degree, regularity, continuity**
  The spline space is approximately C^1 and fully C^1 in the limit of refinement. This makes the spline space suitable to solve fourth-order problems. Contrary to the AS-G1 construction, the spline space approximates the glueing data, allowing maximal smoothness in the interior space ($r = p-1$) for degrees $p \geq 3$.

- **Limitations on construction**
  The space can be constructed on fully unstructured quadrilateral meshes with both interior and boundary extraordinary vertices. The construction of the basis functions is independent of the location or valence of the vertices. Contrary to AS-G^1, the analysis-suitability condition is not needed. However, the construction requires a G^1 condition at the interfaces of surfaces.

- **Nestedness**
  The spline spaces are not nested.

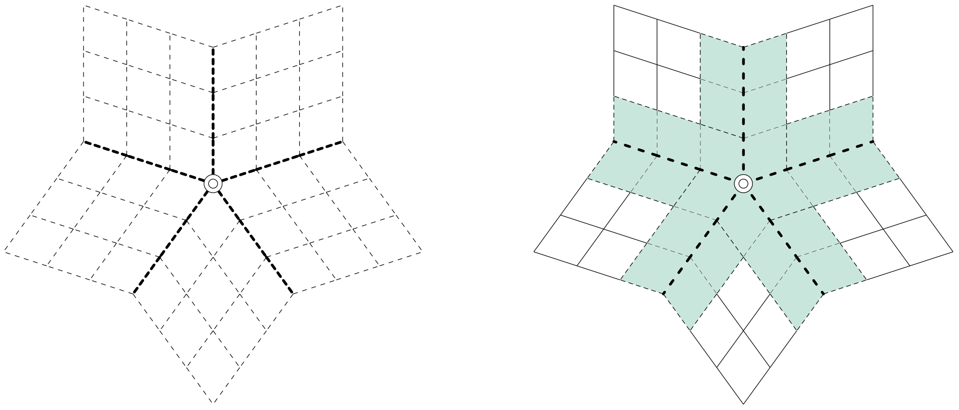


Figure 7.5: Schematic representation of the continuity across element boundaries and patch interfaces for the (a) AS-G^1 construction and (b) Approx. C^1 constructions. Thin lines indicate element boundaries, and thick lines indicate patch interfaces. Solid lines represent C^{p-1} continuity, dashed lines represent C^{p-2} continuity, thick dashed lines represent C^1 interfaces, and loosely dashed lines represent approximate C^1 interfaces. A double-lined circle represents a C^2 continuous vertex; a filled circle represents a singular vertex; and a white filled circle with a single line represents a C^1 continuous vertex. The shaded area for the Approx. C^1 represents local reduced continuity.

- **Refinement procedure**
  The refinement procedure is standard since the parameterisation does not change.

7.3.3 D-patch

The relative ease of imposing parametric smoothness for splines has led to the development of degenerate Bezier patches, or D-patches [464], which can be used to build C^1 smooth splines on unstructured quadrilateral meshes with no boundary extraordinary vertices. The constructions can be formulated for splines of any bi-degree [271], and there are no restrictions on their smoothness in the locally-structured regions of the mesh. In the locally unstructured regions of the mesh (i.e., in a neighbourhood of an extraordinary vertex), the splines are C^1 smooth and first-order degenerate. Note that this degeneracy means that the spline spaces are not necessarily H^2-conforming, but numerical evidence shows that they can still be used to solve fourth-order problems.

Specifically, the imposition of strong C^1 smoothness around an extraordinary vertex requires that the splines vanish up to first order at the extraordinary vertex. This degeneracy trivially implies matching first derivatives at the extraordinary vertex (since all of them vanish) but does not imply C^1 smoothness of the resulting spline functions and the geometries built using them. As shown in [464], additional conditions can be imposed upon certain higher-order mixed derivatives to ensure this desired C^1 smoothness. Furthermore, the effect of these additional constraints can be localised to a neighbourhood of the extraordinary vertex by imposing them on a subdivided representation of the splines [408]. This means that a patch-based representation of C^1 D-patch splines takes functions that are in $\mathbb{S}(\boldsymbol{p}, \boldsymbol{r}, \boldsymbol{h}/2)$ on each patch, where almost all basis functions are in $\mathbb{S}(\boldsymbol{p}, \boldsymbol{r}, \boldsymbol{h})$, except a few basis functions supported in a neighbourhood of extraordinary points (the number of basis functions depends on the valence).

The D-patch construction allows for nested refinements of the spline spaces [464]. If different orders of smoothness are being imposed in locally structured and locally unstructured regions of the mesh, then nested refinements produce spline spaces with a higher number of basis functions supported in the vicinity of extraordinary points (the number depends on the refinement level); see [569] for instance. On the other hand, a patch-based approach allows for a simpler implementation by limiting the smoothness across patch interfaces to C^1; the smoothness in patch interiors can still be arbitrarily chosen. However, special care should be taken when using D-patches with nested refinements – the degeneracy of the splines near extraordinary vertices means that, with mesh refinements, the shape regularity of the mesh starts to worsen with refinements, and the finite element matrices become very ill-conditioned.

In sum, the core ideas behind the D-patch spline construction are the following:

- **Degree, regularity, continuity**
  The spline space is fully C^1. In general, the degeneracy of derivatives means that the spaces are H^2-nonconforming, however, numerical evidence supports their use in solving fourth-order problems. The construction can be formulated for splines of any degree, and the smoothness away from extraordinary vertices can be chosen arbitrarily.
- **Limitations on construction**
  The space can be constructed on unstructured quadrilateral meshes with no extraordinary vertices on the boundary.
- **Nestedness**
  The spline spaces can be refined in a nested manner; however, the resulting mesh has poor shape regularity, and the corresponding finite element matrices may be very ill-conditioned.
- **Refinement procedure**
  Refinement procedures can be derived from standard B-spline knot insertion.

7.3.4 Almost C^1

Almost-C^1 splines are defined on a general, conforming quadrilateral mesh. They are piece-wise biquadratic and possess mixed smoothness, i.e., they are C^1 in regular regions, while the smoothness near extraordinary vertices, i.e., vertices with valence different from four, is reduced. To be precise, they are C^1 smooth at all vertices (including extraordinary vertices) and across all edges except for the ones emanating from an extraordinary vertex. Moreover, while they are defined to be biquadratic on all regular elements, they are piecewise biquadratic splines (with one inner knot in each direction) on all elements that are neighbouring an extraordinary vertex. Details can be found in [534]. As a consequence, a patch-based representation of Almost-C^1 splines takes functions that are in $\mathbb{S}(2, 1, \boldsymbol{h}/2)$ on each patch, where almost all basis functions are in $\mathbb{S}(2,1,\boldsymbol{h})$, except a few basis functions supported in a 1-ring neighbourhood of extraordinary points (the number depends on the valence).

A central feature of Almost-C^1 splines is the mixed smoothness imposition described above. In particular, this choice of mixed smoothness only depends on the current refinement level of the mesh. That is, standard C^1-smoothness is enforced across all edges at the current refinement level except the ones that are incident upon extraordinary vertices, where only C^0 smoothness is enforced. Additionally, these smoothness conditions are combined with G^1 smoothness imposition at each extraordinary vertex. This means that almost-C^1 splines do not yield nested spaces when refining. As a result, the refinement process essentially amounts to a projection of coarse Almost-C^1 splines onto the refined Almost-C^1 spline space. This projection can be chosen in many different ways and can have a significant impact on the limit surface description as well as isogeometric simulations using these spaces. In [534], a smoothing and refinement procedure is proposed that results in a C^1-smooth limit surface for sufficiently regular input data.

Let us briefly summarise the refinement procedure here. It is assumed that a quad mesh and a control point associated with each face of the mesh are given. The initial smoothing step guarantees that all control points associated to the one ring around an extraordinary vertex are coplanar. Having given such an initial control point grid, the geometry is refined using explicit subdivision rules as specified in [534, 567]. The rules are the same as for quadratic tensor-product B-splines in regular regions and maintain the coplanarity near extraordinary vertices.

In sum, the core ideas behind the Almost-C^1 spline construction are the following:

- **Degree, regularity, continuity**
  The spline space locally reproduces biquadratic polynomials, and it is sufficiently smooth to be able to solve fourth-order problems.
- **Limitations on construction**
  The splines can be constructed on fully unstructured quadrilateral meshes, in particular those that contain both interior and boundary extraordinary vertices.
- **Nestedness**
  Since the spaces are not nested, the convergence behaviour of Almost-C^1 splines depends on how the geometry parameterisation is refined.
- **Refinement procedure**
  An initial geometry and a refinement procedure can be constructed in such a way that the limit geometry parameterisation is normal and continuous everywhere.

Thus, the concept introduced in [534] is quite flexible, since the initial smoothing procedure and the refinement procedure are not unique and can be tailored to the needs coming from geometric modeling, e.g., one may want to reproduce Doo-Sabin subdivision surfaces, thus having to modify the subdivision rule for refinement accordingly. The

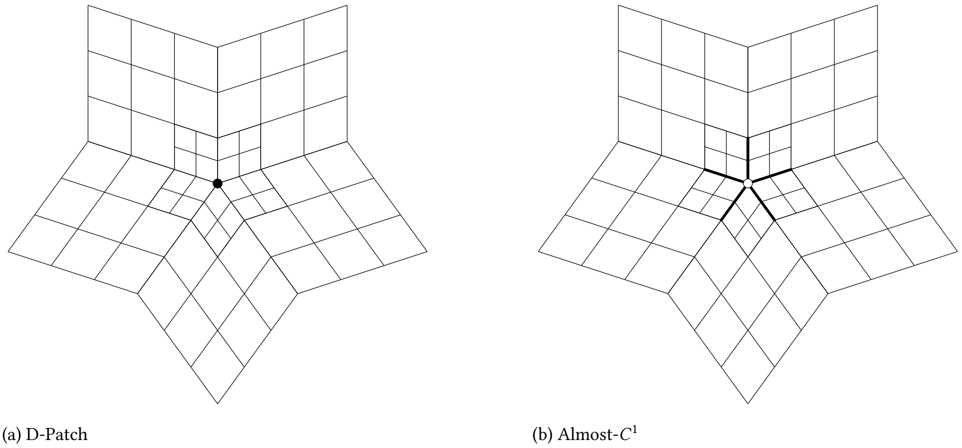

(a) D-Patch (b) Almost-C^1

Figure 7.6: Schematic representation of the continuity across element boundaries and patch interfaces for the (a) D-Patch and (b) Almost-C^1. Line styles are as in figure 7.5.

spline space that is introduced on each refinement level can be seen as a simple hole-filling construction, which is sufficient for numerical analysis.

7.3.5 Conclusions

In this section, a summary of the construction and the properties of the analysis-suitable G^1 (AS-G^1), the approximate C^1 (Approx. C^1), the degenerate patches (D-patch), and the Almost-C^1 methods have been provided, referring to the relevant publications for the mathematical details. For each method, comments have been provided on the degree, regularity, and continuity of the space, on the limitations of the construction in terms of the quadrilateral mesh, on the nestedness for refinement, and on the refinement procedure itself. In addition, figures 7.5 and 7.6 provides detailed information on the local continuity of the constructions around an extraordinary vertex.

The aim of the qualitative analysis of the methods in this chapter is to provide a comparison of a set of properties and requirements of each method and their implications for their applicability. While the subsections presented before provide a brief description of the properties of the methods and the reasons behind these properties and requirements, table 7.1 provides a side-by-side comparison of each method based on the subsections before. In particular, the table lists the *(i,ii)* requirements on degree and regularity for the constructions, *iii* geometrical or topological limitations if applicable, *(iv,v)* the continuity of the constructed bases in the interior and on the interfaces and element boundaries, and *vi* nestedness of the constructed basis.

Following from table 7.1, the requirements for construction of the unstructured spline bases are summarized in figure 7.7 as pre-processing conditions that have to be satisfied for each unstructured spline construction in the process depicted in figure 7.2. The degree and regularity conditions (cf. *i,ii* in table 7.1) must be satisfied for each construction, e.g.,

Table 7.1: Summary of the requirements for the construction and the properties for each of the considered bases. The construction requirements include the degree and regularity of the basis used for construction as well as the geometrical or topological properties of the input geometry. The properties include continuity on interfaces, vertices, and in the interior of the unstructured spline construction, as well as the nestedness property.

Requirements	**AS-**G^1	**Approx.** C^1	**D-Patch**	**Almost-**C^1
(*i*) Degree	$p \geq 3$	$p \geq 3$	$p \geq 3$	$p = 2$
(*ii*) Regularity	$r \leq p-2$	$r \leq p-1$	$r \leq p-1$	$r = 1$
(*v*) Geometrical / topological limitations	Analysis-suitability	G^2 continuity	BEVs: $v \leq 3$, C^1 continuity	C^1 continuity
Properties	**AS-**G^1	**Approx.** C^1	**D-Patch**	**Almost-**C^1
(*iii*) Interface & Vertex Continuity	C^1	C^1 in the limit	C^1	C^1 in the limit
(*iv*) Interior continuity	C^{p-2}	C^{p-1}	C^{p-1}	C^1
(*vi*) Nestedness	Yes	No	Yes	No

by performing projections on suitable spline spaces or by knot insertion routines. Furthermore, the geometric or topological limitations (cf. *iii* in table 7.1) impose additional constraints that the geometry must satisfy.

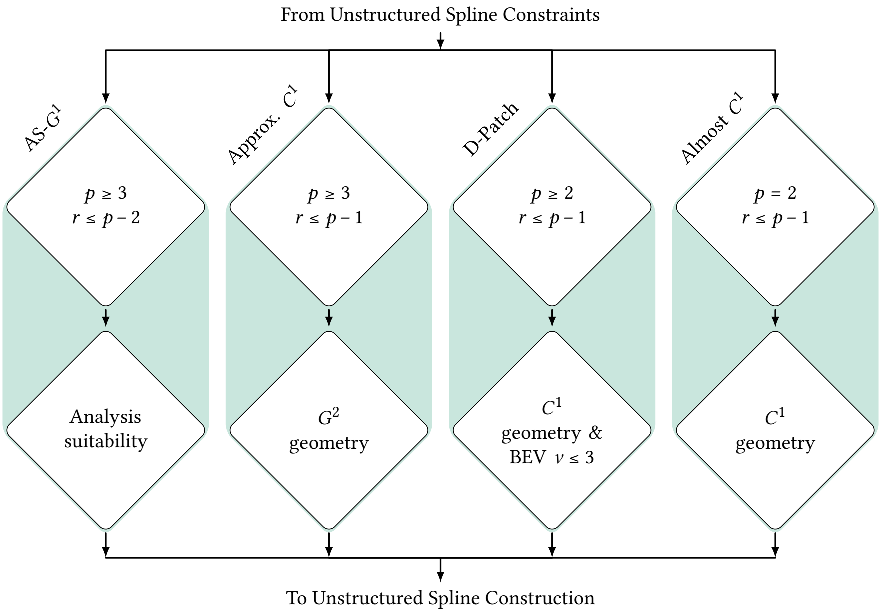


Figure 7.7: Inside the *unstructured spline pre-processing* block from figure 7.2. The unstructured spline requirements are depicted in diamond-shaped blocks for methods AS-G^1, Approx. C^1, D-Patch and Almost C^1. The first row represents requirements on degree p and regularity r. If not satisfied, the geometry can be projected onto a space that satisfies the requirement, or degree elevation or reduction steps can be performed together with refinement operations. The second row depicts the requirements on the geometry parameterization; these blocks can be satisfied by changing the geometry.

7.4 Quantitative Comparison

In this section, a quantitative comparison between the methods provided in section 7.3 is provided. In addition, variational coupling methods are compared, if applicable. The quantitative comparison is composed of various benchmark problems, each providing a different conclusion with respect to the methods considered:

Biharmonic problem on a planar domain (section 7.4.1) The first example entails solving the biharmonic problem on a planar domain. The goal of this example is to assess the convergence properties of all considered unstructured spline constructions; hence, the problem will be solved on a simple analysis-suitable geometry without EVs on the boundary, such that every method from section 7.3 can be applied and compared to the manufactured solution.

Linear Kirchhoff–Love shell analysis on a surface (section 7.4.2) The second example entails solving the Kirchhoff–Love shell equation on curved domains. The goal of this example is to demonstrate the performance of the unstructured spline construction for simple shell problems. Therefore, comparisons will be made to single-patch results and penalty coupling from [238].

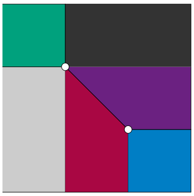


Figure 7.8: Multi-patch decomposition of a simple domain into six patches. The domain has two EVs in the interior (valences 3 and 5) and no boundary EVs.

Table 7.2: Degree p and regularity r constraints for each considered method from section 7.3, see table 7.1.

	$p = 2$	$p = 3$	$p = 3$
	$p = 2$	$r = 1$	$r = 2$
D-patch	★	★	★
Almost-C^1	★		
Approx. C^1		★	★
AS-G^1		★	
Nitsche/Penalty	★	★	★

Spectral analysis on a planar domain (section 7.4.3) In the third example, spectral analysis of a plate equation is performed. The goal of this example is to assess the spectral properties of the unstructured spline methods compared to a variational approach and a single patch, since the spectral properties of highly continuous bases have been demonstrated to be superior over non-smooth bases [121].

Modal analysis of a complex geometry (section 7.4.4) In the fourth example, a modal analysis is performed on a complex geometry extracted from a quad-mesh. The goal of this example is to demonstrate the applicability and performance of the unstructured spline methods on a large-scale, complicated geometry.

Stress analysis in a curved shell (section 7.4.5) Lastly, the fifth example involves the analysis of stress fields in shells. The goal of this example is to assess the performance of unstructured spline constructions and a penalty method when it comes to stress reconstruction in shells. For the Kirchhoff–Love shell, the stresses are obtained by taking gradients of the deformed geometry, hence of the solution. This means that for C^1 bases, stresses are C^0. This might be unfavourable in engineering applications where local stress fields are of importance, e.g., fatigue analysis.

In all examples except the complex geometry in section 7.4.4, the domain decomposition from figure 7.8 is used to decompose a simple domain into a domain with extraordinary vertices in the interior. Domains with EVs on the boundary are left out of scope since the D-patch construction would change the outer boundaries of the domain; hence, the comparison would involve a significantly different geometry. Since different methods have different constraints on the degree and regularity of the basis, different combinations of the degree p and regularity r are tested throughout the benchmark problems. In table 7.2, the combinations of p and r and the methods that are compared for these bases are provided. For the biharmonic problem and the spectral analysis (sections 7.4.1 and 7.4.3), Nitsche's method is used for comparison; see [619] for more details. When solving the Kirchhoff–Love shell equations, the penalty method is used for comparison; see [238] for more details. In all examples, Dirichlet boundary conditions are applied at the control points and clamped boundary conditions are applied weakly as in [238]. All results are obtained using the Geometry + Simulation modules [294, 376] and will be published in a separate publication.

As discussed in section 7.3, the D-patch and Almost-C^1 constructions involve a pre-smoothing of the geometry. In the case of mesh convergence results, refinements can be performed in different ways. On the one hand, the original geometry can be refined, and a new construction with a new geometry approximation can be performed. On the other hand, the geometry resulting from the construction at the first refinement level can be refined in a nested way, such that the geometry does not change after the first mesh. In the quantitative comparison, all refinements are performed in a nested way, unless specified otherwise.

7.4.1 Biharmonic Equation on a Planar Domain

The first benchmark entails the biharmonic equation on a planar domain. The purpose of this example is to assess the convergence properties of the unstructured spline methods described in section 7.3, following the structure of [619]. The biharmonic equation is solved on a unit square $\Omega = [0,1]^2$ with the patch segmentation from figure 7.8. The biharmonic equation is defined by

$$\Delta^2 \varphi = f. \tag{7.1}$$

In the present example, convergence is analysed with respect to a manufactured solution

$$\tilde{\varphi}(x_1,x_2) = (\cos(4\pi x_1)-1)(\cos(4\pi x_2)-1), \tag{7.2}$$

such that the right-hand-side function becomes:

$$f(x_1,x_2) = 256\pi^4(4\cos(4\pi x)\cos\big(4\pi y\big) - \cos(4\pi x) - \cos\big(4\pi y\big)) \tag{7.3}$$

Furthermore, on all boundaries of the domain, the manufactured solution and its derivatives are imposed as Dirichlet and Neumann boundary conditions, respectively:

$$\left.\begin{aligned} \varphi &= \tilde{\varphi}(x_1,x_2) \\ \partial_{\boldsymbol{n}}\varphi &= \partial_{\boldsymbol{n}}\tilde{\varphi} \end{aligned}\right\}\text{on } \Gamma, \tag{7.4}$$

where $\Gamma = \partial\Omega$, $\boldsymbol{n}$ is the unit outward normal vector on Γ. The biharmonic equation from equation (7.1) with boundary conditions equation (7.4) can be discretised by obtaining the weak formulation (see [619]), inserting equation (7.4), and defining an approximation of the solution φ as φ_h. Furthermore, a weak coupling can be established through Nitsche's method. For the mathematical details behind the discretisation of the biharmonic equation and optionally adding Nitsche interface coupling terms, the reader is referred to [618, 619]. For the D-Patch and Almost C^1 constructions, the geometry is smoothed upon construction. The geometry used for evaluation of the weak formulation is constructed by using an L_2-projection of the geometry from the coarsest space, which is projected onto the smooth basis of each refinement level. For the D-Patch, the non-negative smoothness matrix for vertex smoothing is used. Although this matrix produces non-nested meshes, it provides the highest rates of convergence. Furthermore, the factor β (cf. [569, sec. 5.1]) is chosen as $\beta = 0.4$ as used by [569], or $\beta = 1.2$, and halved in each refinement level.

To evaluate the unstructured spline constructions from section 7.3, the numerical approximation φ_h is compared to the manufactured solution $\tilde{\varphi}$ in the L^2-, H^1-, and H^2-norms on the multi-patch segmentation from figure 7.8. The bi-linear segmentation is refined and degree elevated until the desired degree p and regularity r from table 7.2 are obtained. In addition, a Nitsche coupling of the patches is employed for comparison.

The results for the comparison are presented in figure 7.9. For degree $p = 2$ and regularity $r = 1$, the Almost-C^1, D-patch, and Nitsche coupling methods are compared. As expected, the results show consistency between the Almost-C^1, D-patch, and Nitsche's methods with expected convergence. The results also show a slight dependency on the factor β for the D-Patch. For degree $p = 3$ and regularity $r = 1$, the Approx. C^1, AS-G^1, D-patch, and Nitsche's method can be compared. The results of the Approx. C^1 and AS-G^1 are exactly the same, since the original geometry is analysis-suitable and contains only bi-linear patches. Then, applying the Approx. C^1 to an analysis-suitable geometry with regularity $p - 2$, the approximate glueing data becomes exact, hence the same as in the AS-G^1 construction. The D-patch in this case shows better convergence of the L_2, H_1, and H_2 errors for $\beta = 1.2$ than for $\beta = 0.4$. Though, for both choices of β, the convergence is suboptimal, as was observed in the work by [89]. Furthermore, the L_2-norm increases at the last point of the D-Patch results due to the ill-conditioning of the system of equations. Lastly, for degree $p = 3$ and regularity $r = 2$, the Approx. C^1, D-patch, and Nitsche's methods are compared. The observations are as for the $p = 3$ and $r = 1$ cases.

Overall, the results show expected convergence behaviour for all considered spline constructions compared to theoretical results and compared to a Nitsche coupling method. However, the D-Patch method does not converge for very fine meshes, due to ill-conditioning of the system matrix.

7.4.2 Linear Kirchhoff–Love Shell Analysis on a Surface

The linear Kirchhoff–Love shell equations are solved on two geometries to demonstrate the convergence behaviour of the methods on curved surfaces. To this end, two benchmark examples are considered. Firstly, a hyperbolic paraboloid surface is constructed with shape, inspired by [182]:

$$\boldsymbol{r}(\xi_1, \xi_2) = \begin{bmatrix} \xi_1 & \xi_2 & \xi_1^2 - \xi_2^2 \end{bmatrix} \tag{7.5}$$

The left side of the hyperbolic paraboloid is clamped ($\boldsymbol{u} = 0$), and the other sides are free. Furthermore, a distributed load with a magnitude of $8000t$ is applied with t the thickness; see figure 7.10. Secondly, an elliptic paraboloid-shaped domain is modelled, with equation

$$\boldsymbol{r}(\xi_1, \xi_2) = \begin{bmatrix} \xi_1 & \xi_2 & 1 - 2\left(\xi_1^2 + \xi_2^2\right) \end{bmatrix} \tag{7.6}$$

For this shape, a point load with a magnitude of $10^8 t$ is applied in the middle of the domain. The corners of the domain are only fixed in the vertical z direction to allow sliding in the xy-plane. One corner is fixed in all directions to create a well-posed problem. For both hyperbolic paraboloid (figure 7.10) and elliptic paraboloid (figure 7.11), the multi-patch segmentation from figure 7.8 is used. In both cases, the shells are modelled with a thickness of $t = 0.01$ [mm] and with a Saint-Venant Kirchhoff material with Young's modulus $E = 200$ [GPa] and Poisson's ratio $\nu = 0.3$ [-]. The refinement procedure as described in section 7.4.1 is used for the D-Patch and Almost-C^1 constructions.

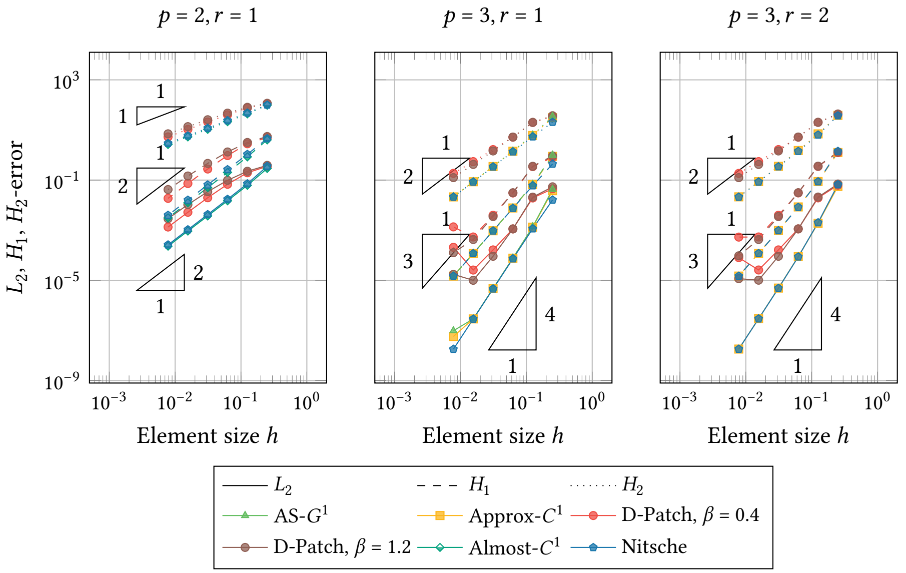


Figure 7.9: Errors for the AS-G^1, Approx. C^1, D-Patch and Almost-C^1 construction for the biharmonic problem on the domain in figure 7.8. The L_2, H_1 and H_2 errors with respect to the analytical solution are plotted with different line styles in the top row. Furthermore, all results are plotted against the element size h and the expected convergence rates are given by the triangles.

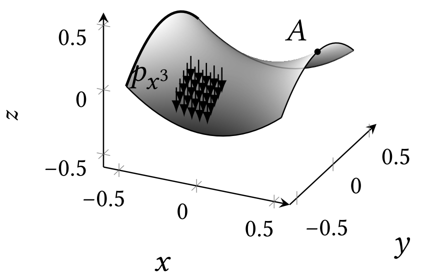


Figure 7.10: Hyperbolic paraboloid shell geometry with coordinates $\boldsymbol{r}(\xi_1,\xi_2) = \left[\xi_1 \quad \xi_2 \quad \xi_1^2 - \xi_2^2\right]$, $\xi_1, \xi_2 \in [-1/2.1/2]$. The left-edge of the hyperbolic paraboloid is clamped, i.e., the displacements and rotations are zero ($\boldsymbol{u} = 0$ and $\frac{\partial u_z}{\partial x} = 0$).

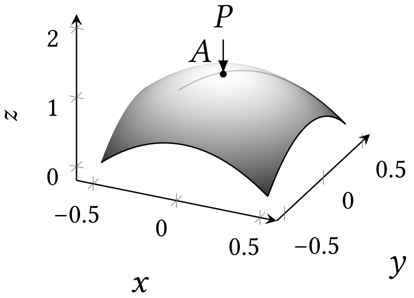


Figure 7.11: Elliptic paraboloid shell geometry with coordinates $\boldsymbol{r}(\xi_1,\xi_2) = \left[\xi_1 \quad \xi_2 \quad 1 - 2\left(\xi_1^2 + \xi_2^2\right)\right]$, $\xi_1, \xi_2 \in [-1/2.1/2]$. On the corners of the domain, the vertical displacements are set to zero $u_z = 0$ and one corner is fixed in-plane as well. Furthermore, a point load with a magnitude $P = 10^8 t$ is applied in the middle of the geometry.

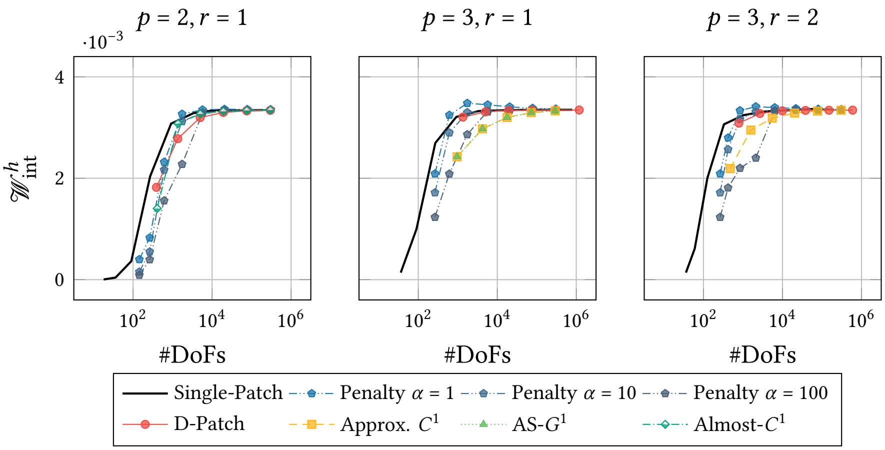


Figure 7.12: Bending energy norm $\mathscr{W}_{\text{int}}^h = \frac{1}{2}\boldsymbol{u}_h^\top K_L \boldsymbol{u}_h$ for the hyperbolic paraboloid geometry from figure 7.10 with a patch segmentation as in figure 7.8. The results are presented for different combinations of the degree p and regularity r for all unstructured spline constructions. In addition, the results for a penalty method with parameter $\alpha \in \{1, 10, 100\}$ are provided for comparison.

The results of both analyses are given in figures 7.12 and 7.13. Here, different unstructured spline constructions are tested on patch bases with different degrees and regularities, as reported in table 7.2. For each combination of degree p and regularity r, the energy norm $\mathscr{W}_{\text{int}}^h = \frac{1}{2}\boldsymbol{u}_h^\top K_L \boldsymbol{u}_h$ is plotted against the number of degrees of freedom, with $\boldsymbol{u}_h$ the discrete displacement vector and K_h the discrete linear stiffness matrix. From the results in figures 7.12 and 7.13, a few observations can be made. Firstly, the Approx. C^1 and AS-G^1 methods show slow convergence on the hyperbolic paraboloid geometry, while the convergence on the elliptic paraboloid geometry is similar to the single-patch convergence. The slow convergence for the hyperbolic paraboloid shell is also observed in [181]. Since the results of the same constructions on the elliptic paraboloid geometries do not show slower convergence, the slow convergence is hypothetically a result of the double curvature with different signs of the shell. Secondly, the D-Patch and Approx. C^1 show comparable convergence to the penalty method on both geometries, which is slightly slower than the convergence of the single-patch results. This is explained by the fact that the degrees of freedom are more optimally allocated for the single-patch parameterisation. Lastly, the results obtained by the penalty method for different penalty parameters α show convergence with a rate similar to the D-Patch and Almost-C^1 methods for penalty parameters $\alpha \in \{1, 10\}$. For $\alpha = 100$, the penalty method is still converging to the same solution, but convergence starts after a few refinement steps

7.4.3 Spectral Analysis on a Planar Domain

In this example, the spectral properties of the unstructured spline constructions on a multi-patch domain are considered. From [122], it is known that isogeometric analysis has the advantage over C^0 Finite Element Analysis with respect to spectra for eigenvalue problems. Smooth isogeometric discretisation provides converging spectra with spline degree

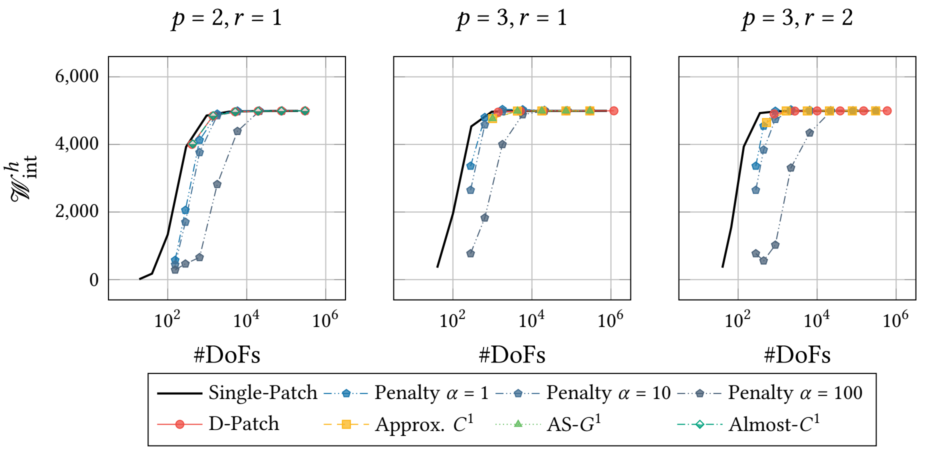


Figure 7.13: Bending energy norm $\mathcal{W}^h_{\text{int}} = \frac{1}{2}\boldsymbol{u}_h^\top K_L \boldsymbol{u}_h$ for the elliptic paraboloid geometry from figure 7.11 with a patch segmentation as in figure 7.8. The results are presented for different combinations of the degree p and regularity r for all unstructured spline constructions. In addition, the results for a penalty method with parameter $\alpha \in \{1, 10, 100\}$ are provided for comparison.

p, whereas the spectra obtained by C^0 FEA diverge with p and typically have optical branches. Similarly, when patches with C^0 continuity are considered, optical branches are introduced, and the accuracy of the spectral approximation decreases [406]. In this benchmark problem, the basis constructions from table 7.2 on figure 7.8 are compared on their spectral properties. For Nitsche's method, different values for the coupling parameter are used to assess its influence on the spectrum.

For the problem at hand, a unit-square domain with parametric layout from figure 7.8 is considered for simplicity. Modal analysis of the plate equation is considered. The stiffness operator of the free vibration plate equation is similar to the biharmonic equation from equation (7.1), and the inertia is included on the right-hand side:

$$D\Delta^2 w = -\rho t \frac{\partial^2 w}{\partial \tau^2} \tag{7.7}$$

Assuming that $w(x, y, \tau)$ is harmonic, i.e. $w(x, y, \tau) = \hat{w}(x, y)\exp\{i\omega\tau\}$ with ω a frequency, the equation simplifies to

$$D\Delta^2 \hat{w} = \omega^2 \frac{\partial^2 \hat{w}}{\partial \tau^2}. \tag{7.8}$$

Here, $D = Et^3/(12(1-\nu^2))$ is the flexural rigidity of the plate with $E = 10^5$ [Pa] the Young's modulus of the plate, $t = 10^{-2}$ [m] the thickness and $\nu = 0.2$ [-] the Poisson's ratio. Furthermore, $\rho = 10^5$ [kg] is the material density. Equation (7.8) is a generalised eigenvalue problem with eigenpairs (ω_i, ν_i) where ω_i is the i$^{\text{th}}$ eigenfrequency and ν_i the i$^{\text{th}}$ mode shape. The mode shape for a simply supported unit plate with $n \times m$ half-waves is given by

$$\nu_{nm}(x, y) = \sin(n\pi x)\sin\left(m\pi y\right) \tag{7.9}$$

with the corresponding eigenfrequency

$$\omega_{nm} = (n^2 + m^2)\pi^2 \sqrt{\frac{D}{\rho t}}. \tag{7.10}$$

In addition, the numerical solution to equation (7.8) is obtained by solving the following generalised eigenvalue problem:

$$D \int_{\Omega} \Delta w \Delta \varphi \, \mathrm{d}\Omega = \omega^2 \rho t \int_{\Omega} u \varphi \, \mathrm{d}\Omega \tag{7.11}$$

With φ a test function, see section 7.4.1. In further representation of the solutions, the index i is employed such that $\omega_i < \omega_{i+1}$ and the subscript h is used for numerically obtained solutions.

Figure 7.14 presents the spectra for different degrees, regularities, and methods. Here, the vertical axis represents the ratio of the numerically obtained eigenfrequency over the analytical eigenfrequency with index i, thus $\omega_{h,i}/\omega_i$. The horizontal axis represents the fraction of the eigenfrequency index i over the total number of eigenmodes. The total number of eigenmodes is equal to the number of degrees of freedom in the system. The results are presented for the degrees and regularities as in table 7.2.

Firstly, the $p = 2$, $r = 1$ plot shows that Nitsche's method oscillates for all considered values of the penalty parameter. Furthermore, in the part where it is not oscillating, the ratio $\omega_{i.h}/\omega_i$ is higher than for the D-patch and Almost-C^1 method. Additionally, the D-patch and Almost-C^1 methods show a significant difference with respect to the single patch result, which is due to the non-Cartesian multi-patch segmentation of figure 7.8 and the fact that the analytical mode shapes are Cartesian. For the $p = 3$, $r = 1$ and $p = 3$, $r = 2$ bases, similar conclusions can be drawn. Although for the $p = 3$, $r = 1$ case the Approx. C^1 method seems worse than the D-patch method, the opposite is true for $p = 3$, $r = 2$. Hence, it can be concluded that no method outperforms another, but that all unstructured spline constructions perform better than Nitsche's method.

7.4.4 Modal Analysis of a Complex Geometry

The next example of the quantitative analysis in this chapter involves modal analysis on a larger-scale complex geometry, depicted in figure 7.15a. The goal of this example is to show the usability of the considered constructions on an off-the-shelf industrial geometry. The geometry is represented as a mesh consisting of 15895 vertices, 31086 edges, and 62172 faces. This geometry is converted to bi-linear patches using the procedure discussed in figure 7.4 in section 7.3. The interface and boundary curves of the patches are given in figure 7.15b, and the final multi-patch object is given in figure 7.15c. The latter has 3 EVs of valence 3, 10 EVs of valence 5 and 16 bEVs. Moreover, the material parameters specified for a steel material. That is, the density of the material is $\rho = 7850 \cdot 10^{-6}$ [tonnes/mm^3], and the shell thickness is $t = 10$ [mm], the Young's modulus is $E = 210 \cdot 10^3$ [MPa] and the Poisson's ratio is $\nu = 0.3$ [−]. All the sides of the geometry are kept free, meaning that the

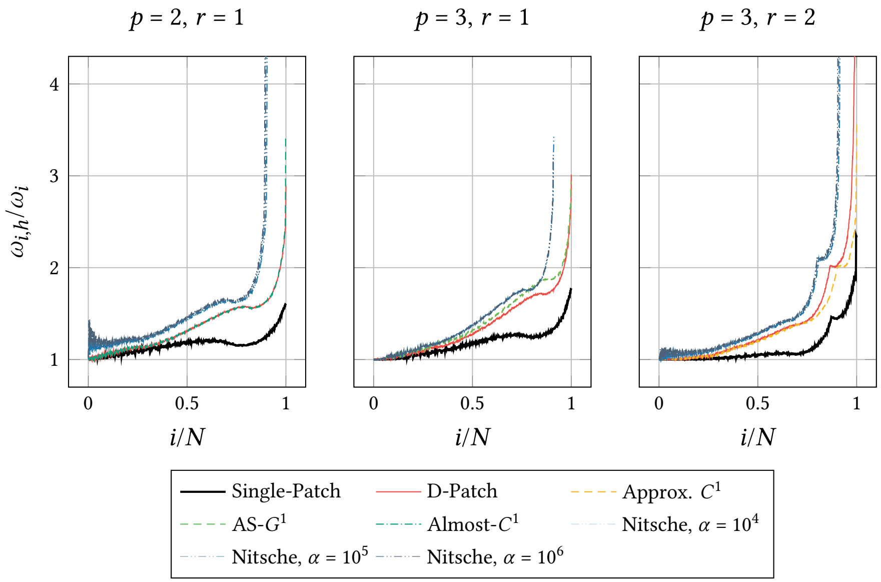


Figure 7.14: Eigenvalue spectra for the biharmonic eigenvalue problem on the domain from figure 7.8. The horizontal axes depict the eigenvalue index i over the total number of eigenvalues N. The vertical axes represent the numerical eigenvalue $\omega_{i,h}$ over the analytical eigenvalue ω_i, both with index i. The results are plotted for different combinations of the degree p and regularity r of the basis. The results for a Nitsche method are given for different penalty parameters α.

modal analysis results will consist of six modes with zero eigenfrequencies: the rigid body modes. In the sequel, only the results for deformation modes are listed.

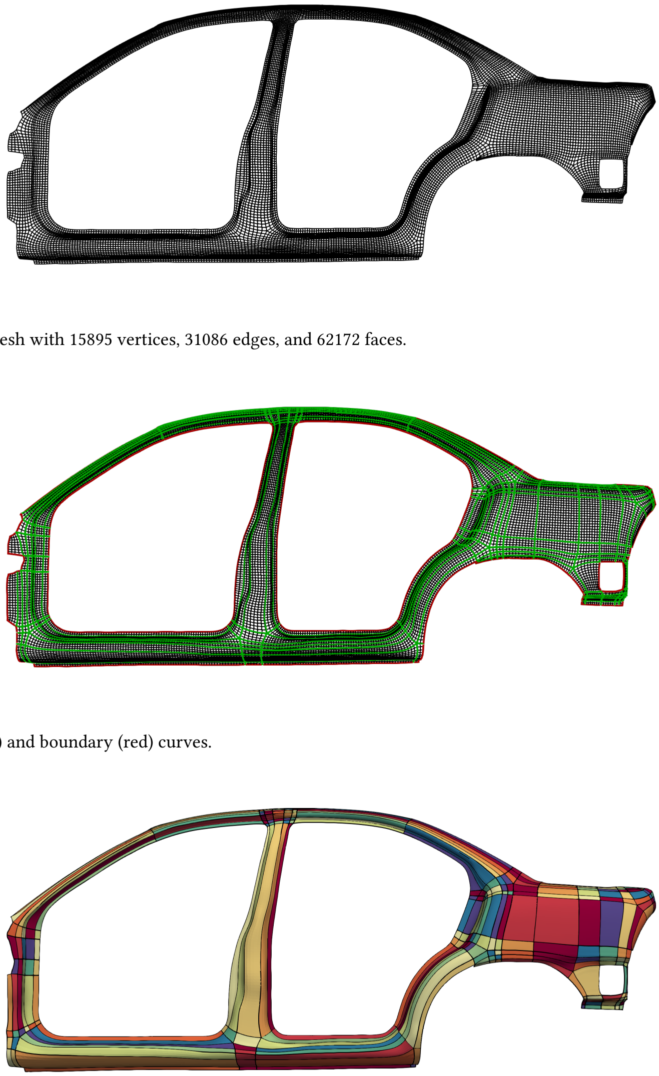
After the creation of the linear multi-patch object, h-, p-, and k-refinement steps can be performed to construct a multi-basis corresponding to the patch layout on which unstructured splines can be constructed. For the Almost-C^1 and D-Patch constructions, the bases are constructed by refining and elevating the initial linear basis up to the desired degree and regularity, after which the the Almost-C^1 and D-Patch basis and geometry are computed. An Almost-C^1 geometry is provided in figure 7.15c.

The AS-G^1 construction requires an analysis-suitable geometry, which can be constructed following [181], is based on the planar construction developed in [304]. However, the geometry from figure 7.15b is only C^0-smooth due to the original linear mesh it is constructed from. An algorithm to automatically pre-process the geometry to obtain an analysis-suitable G^1 surface is not yet developed. The algorithm from [181] requires AS-G^1 gluing data, which cannot be prescribed directly on a C^0 surface. If the surface is not pre-processed to be AS-G^1, no suitable gluing data can be found and the basis construction is not applicable. Although the Approx. C^1 construction does not require an analysis-suitable re-parameterization, it does require G^1 smoothness at the interfaces. If this condition is not satisfied there exists no C^1 construction that can be approximated by this method. For both methods, the required pre-processing efforts are non-trivial or not demonstrated on industrial geometries, and therefore left out of the scope of this chapter[3].

Furthermore, penalty methods have been used in the context of modal analysis on a 27 patch composite wind-turbine blade in [238], where the variation of the element size of interface elements seems rather small. In the present chapter, an attempt was made to apply the penalty method on the geometry in figure 7.15c, but unidentifiable vibration modes were obtained, possibly because of the large variation of element lengths across the interfaces of the domain, challenging the determination of a suitable penalty parameter α.

Table 7.3 presents the eigenfrequencies for the first four deformation modes of the car side panel for the D-Patch and the Almost-C^1 constructions with degree $p = 2$ and regularity $r = 1$ for the Almost-C^1 construction and with $(p, r) = (2, 1)$, $(p, r) = (3, 1)$ and $(p, r) = (3, 2)$ for the D-Patch. Figure 7.16 provides the corresponding mode shapes on the D-Patch geometry with $p = 3$, $r = 2$ and the mode shapes have been qualitatively matched to construct table 7.3. From these results, it can be observed that the Almost-C^1 and D-Patch methods provide eigenfrequencies in the same range and that the eigenfrequencies are mostly converging in the second digit. Moreover, the eigenfrequencies of the D-Patch and Almost-C^1 methods for coarse meshes and $p = 2$, $r = 1$ already provide reasonable estimates compared to higher degrees and refinements. On the other hand, the results obtained using an ABAQUS S4R element show convergence in the second digit and slightly lower frequencies than the IGA results, possibly because the FEM uses a different geom-

[3] In the case of a different starting point for this benchmark, such as a smooth mesh composed of higher-order quadrilateral elements, e.g. derived from a subdivision surface, instead of a bi-linear mesh, the pre-processing efforts required for the AS-G^1 and Approx. C^1 will be different.



(a) Original quad mesh with 15895 vertices, 31086 edges, and 62172 faces.

(b) Interface (green) and boundary (red) curves.

(c) Final multi-patch segmentation with 307 patches.

Figure 7.15: Geometry of the side panel of a car. The original mesh (a) is traced with the procedure from figure 7.4, yielding a set of boundary and interface curves (b). From these curves, the multi-patch segmentation (c) for isogeometric analysis is constructed following figure 7.4c.

Table 7.3: Eigenfrequencies of the Almost-C^1 and D-Patch constructions for the car geometry in figure 7.15. The results of an ABAQUS FEA simulation using the S4R element are provided as a reference. The mode shapes are plotted in figure 7.16.

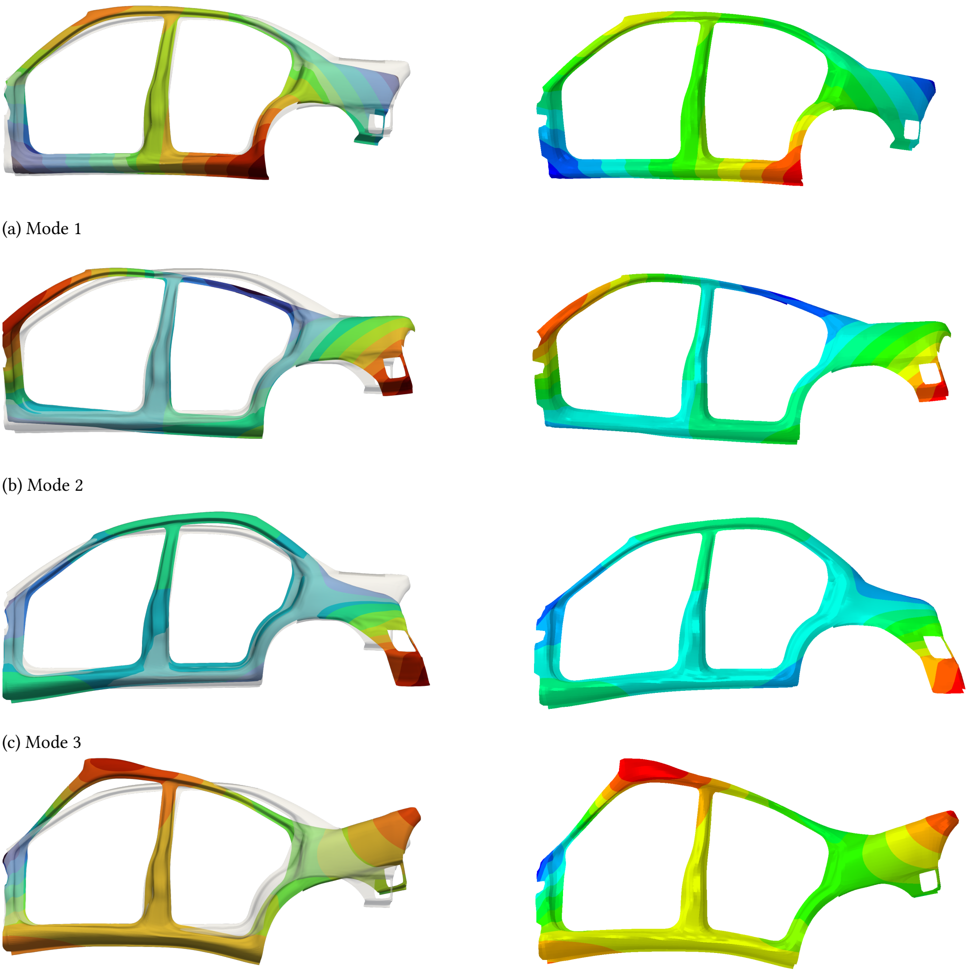
Method		# DoFs	Mode 1	Mode 2	Mode 3	Mode 4
Almost-C^1, $p = 2$, $r = 1$		13,731	15.740	25.567	43.829	56.654
		49,758	15.762	25.564	43.429	56.778
		189,654	15.776	25.552	43.269	56.785
		740,814	15.774	25.531	43.177	56.746
D-Patch, $p = 2$, $r = 1$		49,437	15.785	25.607	43.641	56.902
		189,333	15.780	25.561	43.323	56.807
		740,493	15.775	25.533	43.191	56.748
D-Patch, $p = 3$, $r = 1$		136,839	15.749	25.593	43.348	56.786
		630,459	15.760	25.581	43.231	56.801
D-Patch, $p = 3$, $r = 2$		71,760	15.771	25.539	43.224	56.744
		226,524	15.755	25.582	43.235	56.807
ABAQUS S4R	10mm	126,966	15.303	24.881	42.629	54.887
	5mm	440,076	15.224	24.780	42.516	54.627
	2.5mm	1,653,030	15.119	24.640	42.338	54.277

etry approximation. Overall, it can be concluded from this benchmark problem that the Almost-C^1 and D-Patch are more robust for industrial and large scale geometries, that are represented by at least C^0-conforming quadrilateral meshes, compared to the Approx. C^1 and AS-G^1 methods due to the pre-processing efforts required by the latter. Furthermore, these methods are parameter-free, making them robust also with respect to penalty methods.

7.4.5 Stress Analysis in a Curved Shell

An interesting application for smooth, unstructured spline construction is the use of thin shell analysis for engineering applications. Not only displacements (section 7.4.2) or vibrations (section 7.4.4) are of interest, but also stress evaluations, for example, fatigue analysis. In the last example, the performance of all methods in table 7.2 is demonstrated on the evaluation of stresses in a curved Kirchhoff–Love shell. Since the Kirchhoff–Love shell formulation is displacement-based, the displacements are C^1 continuous across patch interfaces for C^1 constructions. The stresses, however, are based on the gradients of the displacements; hence, their continuity theoretically is C^0 for a perfect C^1 coupling. In this example, the Von Mises membrane stress field resulting from the 6-patch elliptic paraboloid from figure 7.13 is considered. The stress fields are plotted for bases with degree and regularity from table 7.2 and additionally for a basis with $p = 4$, $r = 2$. Note that the regularity r of these bases is the regularity in the patch interior.

In figure 7.17, the stress fields for the elliptic paraboloid example from figure 7.13 are provided. From these results, it can immediately be seen that the stress field for a single



(a) Mode 1

(b) Mode 2

(c) Mode 3

(d) Mode 4

Figure 7.16: Out-of-plane deformations of the first four vibration modes of the side of the car from figure 7.15. The results on the left represent the results obtained by the D-Patch construction and the results on the right represent results obtained using ABAQUS (10mm). The mode shapes are all deformation modes warped by the deformation vector and plotted over the undeformed (transparent) geometry.

patch parameterisation with basis $p = 2$, $r = 1$ exposes the elements of the basis because of the C^0 continuity across elements. Similar effects are seen for the D-patch, Almost-C^1 and the penalty method. Increasing the degree of the basis while keeping the regularity the same results in a $p = 3$, $r = 1$ basis. The element continuity is still C^0 for the stresses, but the higher continuity of the basis within the element results in a slightly improved stress field, as can be seen from the single patch, the D-patch, and penalty methods. The Approx. C^1 and AS-G^1 methods in addition show a better stress field around the EVs compared to the D-patch with only small wiggles in the inner contour. Increasing the smoothness by going to $p = 3$, $r = 2$ shows that the Approx. C^1 method predicts the stress field very well over the whole domain, but with the wiggles in the inner contour, and that the D-patch suffers from the singularity at the EVs. Lastly, the $p = 4$, $r = 2$ plots show that the wiggles in the inner contour are eliminated for the Approx. C^1 and the AS-G^1 methods and that the artefacts of the D-patch around the EV are still there, but to a lesser extent. Finally, the results of the penalty method in figure 7.17 show it is able to provide an accurate representation of the stress fields. As seen from figure 7.13, penalty factors $\alpha = 1$ and $\alpha = 10$ provide good convergence in the bending energy norm. Indeed, the stress fields for the fixed 64 × 64 element meshes in figure 7.17 confirm that for these penalty factors, the stress fields accurately represent the single patch stress fields, despite small artefacts around the EVs for $\alpha = 1$. For a higher penalty factor of $\alpha = 100$, the stress fields following from the penalty method are not guaranteed to be accurate, showing the downside of this method.

Overall, the stress analysis for multiple combinations and regularities shows that the Almost-C^1 method is generally unfavourable since it is only applicable for $p = 2$, $r = 1$ hence C^0 stress fields, suffering from a lack of continuity over the whole domain. This also makes the D-patch as applicable as the Approx. C^1 method in terms of degree and regularity combinations. Comparing the D-patch with the Approx. C^1 and the AS-G^1 methods, it is shown that the D-patch suffers from the singularity in the EVs when reconstructing stresses, whereas the other two methods are able to recover the stress fields without problems. Moreover, this example has also shown the advantage of smooth, unstructured spline constructions for stress analyses since their continuity across (almost) all of the domain is ensured, contrary to the penalty method. Lastly, this example shows the advantage of IGA in general over lower-order methods like FEA, since the higher-degree bases (e.g., $p = 4$, $r = 2$) provide smooth stress fields compared to lower-degree bases ($p = 2$, $r = 1$).

7.4.6 Conclusions

In this section, a quantitative comparison of the AS-G^1, the Approx. C^1, the D-Patch, and the Almost-C^1 constructions is provided. The methods have been assessed in different aspects: i) convergence of the biharmonic equation (section 7.4.1); ii) convergence of the linear Kirchhoff–Love shell (section 7.4.2); iii) eigenvalue spectrum approximation (section 7.4.3); iv) application to a large-scale complex geometry (section 7.4.4); and v) the reconstruction of stress fields (section 7.4.5). From these analyses, the following conclusions can be drawn:

- All methods converge in a theoretical setting to the same solution for the biharmonic

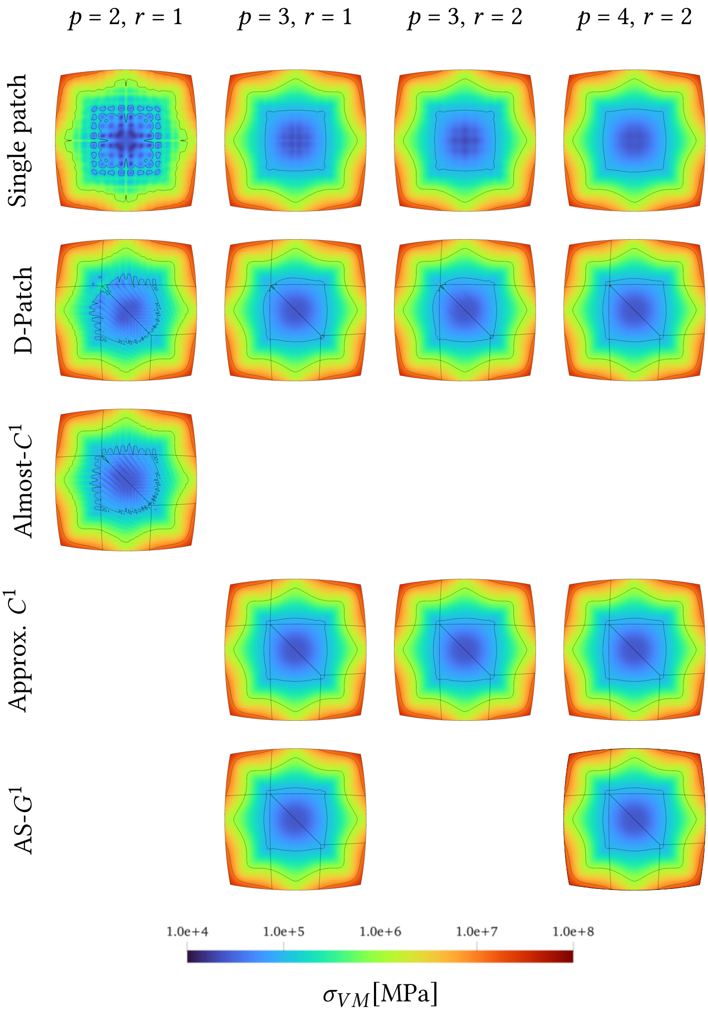


Figure 7.17: (Caption on next page).

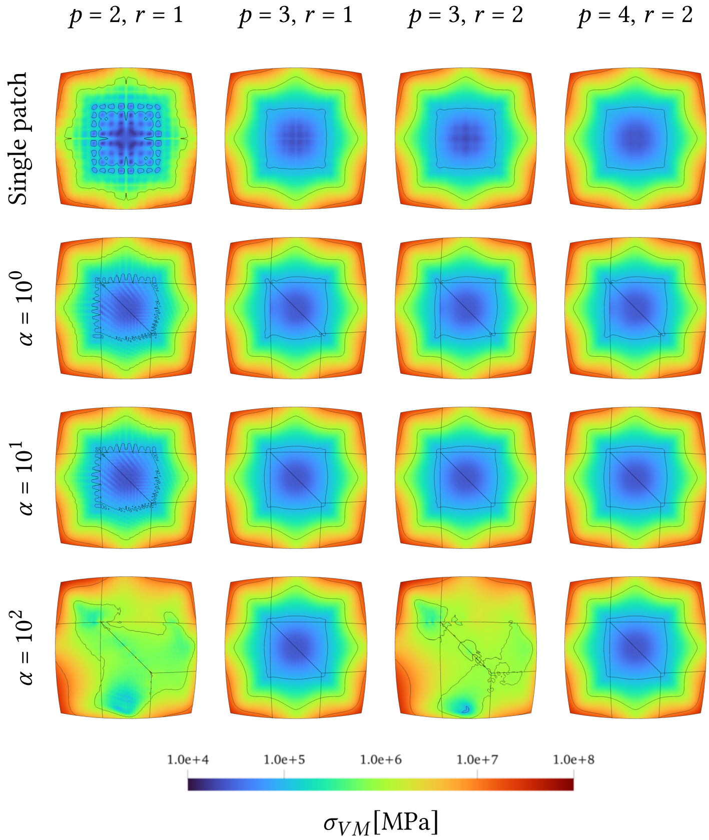


Figure 7.17: Von Mises membrane stress fields for the single patch, unstructured splines and penalty-coupled multi-patch paraboloid from section 7.4.2 and figure 7.13 with 64 × 64 elements per patch. The results are provided for different combinations of degree p and regularity r. The colour bar represents the stress and the contours are plotted for stress levels $\sigma_{VM} \in \{10^5, 10^6, 10^7\}$ [MPa].

equation (sections 7.4.1 and 7.4.2). However, the convergence behaviour of the D-Patch method is suboptimal and affected by conditioning issues for large meshes. Furthermore, the Approx. C^1 and AS-G^1 methods give worse convergence compared to other methods for the hyperbolic paraboloid shell but good convergence rates for the elliptic paraboloid shell example.

- From a spectral analysis of the biharmonic equation section 7.4.3, it can be concluded that there is no best unstructured spline construction. Depending on the degree and regularity, small differences in the eigenvalue spectra are observed between the methods. Comparing with Nitsche's method, however, it is concluded that the unstructured spline constructions considered in this chapter perform consistently better. This is also confirmed by the applied modal analysis on the car geometry section 7.4.4, where the penalty method fails to find accurate eigenfrequencies, possibly because of an unsuitable penalty parameter.

- From the applied modal analysis on a complex geometry, it can also be concluded that the Almost-C^1 and D-Patch constructions are more straight-forward to apply to a complex geometry extracted from a mesh. This is due to the fact that the Approx. C^1 and AS-G^1 constructions require, respectively, a G^2 geometry and an analysis-suitable geometry, which are both not trivial to construct from an originally C^0-continuous mesh. Instead, the D-Patch and Almost-C^1 constructions require a C^1 geometry, which is easier to construct in general.

- From the stress fields presented in section 7.4.5 and the analysis in section 7.4.2, it can be concluded that the AS-G^1 and Approx. C^1 methods provide excellent stress fields. The D-Patch also provides good stress fields, but inaccuracies are found around the EVs, possibly because of the singularity close to the EV. The Almost-C^1 method is considered inaccurate for stress analysis because of a lack of higher-degree generalisations. Lastly, comparison with penalty methods shows that the unstructured spline constructions generally provide a robust parameter-free approach for coupling, whereas the penalty method requires careful selection of the penalty parameter.

Overall, our findings suggest that the Almost-C^1 and D-Patch are generally easier to construct, but for certain problems, they have limited accuracy. On the other hand, the AS-G^1 or Approx. C^1 discretisations require more pre-processing efforts but provide optimal convergence, hence accuracy. This, however, depends on the input geometry: generic quad-meshes might require more pre-processing efforts than C^1-matching parameterisations. Lastly, the results provided in this section have shown that strong coupling methods have certain advantages over weak methods and therefore provide an interesting alternative.

7.5 Conclusions and Future Work

This chapter provides a qualitative and quantitative comparison of unstructured spline constructions for smooth multi-patches in isogeometric analysis. The general advantage of unstructured spline constructions over trimming or variational coupling methods is that they are parameter-free, do not require specialised solvers, and are typically constructed once in a shape optimisation workflow. The goal of this chapter is to compare the analysis-suitable G^1 (AS-G^1), the approximate C^1 (Approx. C^1), the degenerate patches (D-Patch), and the Almost-C^1 constructions with respect to qualitative aspects (i.e., constraints for application) and quantitative aspects (i.e., numerical performance).

From the qualitative analysis, it followed that each method required a different set of constraints to be satisfied before the constructions could be applied, see figure 7.7 and table 7.1. Degree and regularity constraints can be satisfied by knot insertion routines or refitting, which are relatively straight-forward. The constraint on analysis-suitability for the AS-G^1 and the constraint on G^2 continuity for the Approx. C^1 method require dedicated reparametrisation routines, such as the one presented by [304]. The fact that D-Patches are restricted to geometries without boundary extraordinary vertices requires redefinition of the quadrilateral mesh. Lastly, the fact that the Almost-C^1 method is only defined for bi-quadratic bases ($p = 2$) restricts the inter-element continuity to C^1 through the whole domain. Depending on the application and the availability of existing routines in software, different unstructured spline constructions are favourable, depending on the geometric flexibility or desired degree and regularity.

From the quantitative analysis, some conclusions can be drawn on the considered unstructured spline constructions and between unstructured spline constructions compared to variational methods such as Nitsche's method or a penalty method. From the analysis, it was in general observed that depending on the problem type, the different methods have their advantages and disadvantages. Firstly, simple biharmonic equations (see section 7.4.1) and linear shells (see section 7.4.2) provided good results for all methods. However, the AS-G^1 and Approx. C^1 methods showed slow convergence for the double-curved shell and the D-patch suffered from ill-conditioning for fine meshes. The Almost-C^1 provided good results in general, however it is only applicable on bi-quadratic splines. Secondly, all methods showed superiority over Nitsche's method for the computation of an eigenvalue spectrum for plate vibrations (see section 7.4.3) and no significant differences between the unstructured spline constructions have been observed. Thirdly, the D-Patch and Almost-C^1 showed straight-forward applicability on the problem of a complex geometry (see section 7.4.4), whereas the analysis-suitability requirement of the AS-G^1 method and the smoothness requirement of the Approx. C^1 method are non-trivial to satisfy on off-the-shelf industrial geometries.

For the penalty method, no suitable penalty parameter was found, and probably optimal penalty parameters should be chosen per interface rather than globally. Lastly, the AS-G^1 and Approx. C^1 methods provided superior results for stress reconstruction, where the D-Patch suffered around the EVs due to its singular parameterization and the Almost-C^1 method provided bad results due to a lack of higher degrees.

In conclusion, both comparisons give an overview of the applicability of the methods with respect to the requirements needed to construct them, on the notions of nestedness and in general on the performance of the methods. Overall, it can be concluded from both analyses that among the compared methods, there is no general best construction. More precisely, the quantitative analysis shows that different methods perform differently in different applications, given that they can be constructed. Furthermore, with the backgrounds and properties provided in the qualitative analysis section, we hope that the present chapter provides valuable insights for application of the considered methods to multi-patch problems.

In addition, the comparisons in the present chapter give directions for the improvements of the considered methods. For the AS-G^1 and Approx. C^1 methods, restrictions on geometry and parameterization are a bottleneck in the industrial applications. Therefore, it is recommended to expand the applicability of these methods by developing dedicated geometric pre-processing routines. For the D-Patch construction, the limitation of the construction of the basis near $v > 3$ boundary EVs calls for the development of routines to eliminate these EVs in quadrilateral multi-patches, as discussed in the qualitative comparison. Furthermore, the example of the bi-harmonic equation has shown that the D-Patch can suffer from ill-conditioned system, hence development of pre-conditioners for D-Patch constructions is advised. Lastly, although the Almost-C^1 resolves the downsides of the D-Patch construction, its restriction on the degree of the spline-space is a major disadvantage when plotting stress fields in shell analysis. Therefore, for the Almost-C^1 construction it is recommended to explore expansion to higher degrees.

7.A Result Reproduction

For the sake of reproducibility of the results in this chapter, this appendix provides brief instructions on the use of the software developed along with this thesis. The full software is available as part of the Geometry + Simulation Modules. For more detail on the contributions to this software library, and its installation, the reader is referred to chapter 8.

Table 7.4 provides per figure in this chapter the name of the file to run along with the arguments to be passed to obtain these figures.

Table 7.4: File name and run arguments required for the reproducibility of the figures in this chapter. Arguments with a single dash (-) require an argument while double-dashed arguments (--) are switches. See chapter 8 for more detail about the software and installation instructions. All executables in this table are from the gsUnstructuredSplines module, and the path to the XML files mentioned is gsUnstructuredSplines/filedata/.

Figure	Run File		
	Arg.	Description	Values
Figure 7.9	biharmonic_planar_example		
	-m	Method to use	1: D-Patch, 2: Approx. C^1, 3: AS-G^1, 4: Almost-C^1, 5: Nitsche
	-p	Degree	2 or 3
	-s	Regularity	1 or 2
	-r	Number of uniform refinement loops	5
	-B	β value for D-Patch	0.4 or 1.2
	-y	Nitsche parameter	1e2
	-f	Geometry file	planar/6p_square_linear.xml
Figure 7.12 Figure 7.13 Figure 7.17	kirchhoff-Love_multipatch_example		
	-m	Method to use	1: D-Patch, 2: Approx. C^1, 3: AS-G^1, 4: Almost-C^1
	-p	Degree	2 or 3
	-s	Regularity	1 or 2
	-r	Number of uniform refinement loops	6
	-G	Geometry file	surfaces/shell/... 6p_hyperboloid.xml 6p_paraboloid2.xml
	-B	Problem definition file	pde/shell/... 6p_hyperboloid_bvp.xml 6p_paraboloid2_bvp.xml
	--stress	Plot stresses	
	kirchhoff-Love_weak_multipatch_example		
	-s,-r, -G, -B	Same as above	
	-d	Penalty parameter	1e0, 1e1, 1e2
Figure 7.14	biharmonic_planar_eigenvalue_example		
	-m	Method to use	1: D-Patch, 2: Approx. C^1, 3: AS-G^1, 4: Almost-C^1, 5: Nitsche
	-p	Degree	2 or 3
	-s	Regularity	1 or 2
	-r	Number of uniform refinements	3: $p = 3, s = 1$ 4: $p = 2, s = 1$ and $p = 3, s = 2$
	-y	Nitsche parameter	1e4, 1e5, 1e6
	-f	Geometry file	planar/6p_square_linear.xml
Figure 7.16 Table 7.3	kirchhoff-Love_multipatch_vibration_XML_example		
	-G	Geometry file surfaces/neon/dpatch_p2_s1_r1_geom.xml surfaces/neon/dpatch_p2_s1_r2_geom.xml ... surfaces/neon/almostC1_p2_s1_r1_geom.xml surfaces/neon/almostC1_p2_s1_r2_geom.xml ...	
	-b	Basis file surfaces/neon/dpatch_p2_s1_r1_basis.xml surfaces/neon/dpatch_p2_s1_r2_basis.xml ... surfaces/neon/almostC1_p2_s1_r1_basis.xml surfaces/neon/almostC1_p2_s1_r2_basis.xml ...	
	-B	Problem definition file	pde/shell/car_bvp.xml

8

Isogeometric Structural Analysis in G+Smo

The last chapter of the core of this dissertation provides background on the implementation of the methods developed in chapters 3 to 7, aiming for reproducibility of the results presented in this dissertation as well as facilitating future developments on the topics presented herein. The methods presented in this dissertation are all implemented in the open-source Geometry + Simulation modules (`G+Smo`*), written in C++ with Python bindings. For detailed information on classes and functions implemented in* `G+Smo`*, the reader is referred to the documentation. Instead, this chapter provides a high-level overview of three new modules in* `G+Smo`*: a module for Kirchhoff–Love shells, a module for structural analysis, and a module for unstructured spline constructions. As motivated in this chapter, the modules are implemented with the aim of being compatible with future developments. For example, by providing base implementations of material laws, by using black-box functions for the structural analysis module, or by providing a standardised approach for the implementation of unstructured spline constructions, The high-level overview that is given for each module relates scientific literature and mathematical formulations to the classes implemented in the novel modules, and it elaborates on the reasoning behind the design of the modules. Through several examples with code snippets, simple routines are highlighted to illustrate how one can interact with the off-the-shelf routines provided within the modules. Overall, this chapter demonstrates that the new modules contribute to a versatile ecosystem for the modelling of multi-patch shell problems through fast off-the-shelf solvers with a simple interface, designed to be extended in future research. If necessary, the reader is referred to sections 2.2 and 2.4 for background information related to isogeometric analysis and numerical structural analysis.*

This chapter will be extended with more content out of the scope of this dissertation before submission.

8.1 Introduction

With the advent of Isogeometric Analysis [268], the fields of computer-aided design (CAD) and computer-aided engineering (CAE) slowly unify. By sharing the same mathematical foundation based on splines, highly smooth bases are introduced in classic Galerkin frameworks in CAE, and concepts like analysis suitability, water tightness, and de-featuring become important topics in the CAD community, supporting better analyses [36, 76, 77, 106]. Since its introduction, many developments have occurred within the isogeometric analysis paradigm, providing a viable alternative to Finite Element Analysis (FEA) in the engineering discipline. With the aim of unifying the CAD and CAE pipelines, IGA substantially influences both communities. For example, new spline constructions with refinement properties suitable for CAE have been developed [149, 158, 202, 203]. In addition, mathematical analysis of the properties of various existing spline constructions and their influence on CAE have been assessed [78]. On the other hand, the application of IGA to computational mechanics has increased the interest in problems with high continuity requirements, such as phase-field modelling [446, 447], as well as shape-optimisation problems benefiting from the smooth and local geometric parametrisation of spline geometries, e.g., for patient-specific heart valve design [646].

With the increasing complexity of problems solved with isogeometric analysis, the demand for versatile software libraries increases. Since these software libraries operate in the CAD and CAE domains, advanced geometric and physical modelling capabilities are essential in performing state-of-the-art simulations. Since the advent of IGA in 2005, several software libraries have been developed, ranging from commercial closed-source libraries and free open-source libraries to in-house codes. In the following, an exhaustive yet incomplete overview of publicly available (closed-source and open-source) software libraries with IGA features is presented. On the one hand, IGA software libraries are based on existing FEA libraries that have been extended with IGA capabilities, such as the commercial software LS-DYNA [227] with the ANSA pre-processor [336] or the open-source libraries Kratos multi-physics [130, 187] (C++), PetIGA (built on Petc in C++) [131], MFEM (C++)[9] and tIGAr (built on FEniCS) (Python) [296]. On the other hand, general-purpose IGA libraries have been presented, such as the open-source Nutils (Python) [667], GeoPDEs (MATLAB/Octave) [576], the Geometry + Simulation Modules (`G+Smo` , C++ with Python bindings) [294, 376] and PSYDAC (Python) [216] for general problems, and Bembel (C++) [159] for BEM-IGA, YETI (Python) [174] for structural optimisation and the closed-source software developed by Coreform LLC. In general, the FEA-based libraries are extensive libraries with large communities and heavily optimised routines (for FEA), whereas the IGA libraries have a much lower technology readiness level and emerge mainly from academic research projects.

The aim of this chapter is to provide an overview of implementation aspects and reproducibility of previously published results [182, 584–589, 618, 619] through an open-source code for multi-patch shell structural analysis using IGA. The software is provided as modules within the library of the Geometry + Simulation Modules (`G+Smo`), the latter providing basic routines for geometric modelling and isogeometric system assembly. In particular, this chapter describes a module for Kirchhoff–Love shell analysis including hyperelasticity

[587], a hyperelastic tension-field-based membrane model [585], as well as error estimation and adaptivity [586]. This gsKLShell module can be used as an off-the-shelf solver that does not require the user to implement PDEs manually. For unstructured spline-based multi-patch modelling, this chapter presents the gsUnstructuredSpline module, which has recently been used to provide a thorough comparison of unstructured spline constructions for isogeometric analysis [181, 182, 589, 618, 619]. Lastly, this chapter elaborates on a module for structural analysis in G+Smo named gsStructuralAnalysis, which includes the novel Adaptive Parallel Arc-Length Method, presented in [584]. Most of the features presented in the present chapter are off-the-shelf high-fidelity routines, aiming to be used in engineering applications. The goal of this chapter goal is to provide a versatile software architecture for the IGA paradigm and demonstrate that demanding problems such as shell analysis can be solved with higher-order IGA methods, both efficiently and with superior quality in the numerical results.

In the remainder of this chapter, the design of the modules for shell and structural analysis in G+Smo are discussed. In section 8.2, an overview of G+Smo is given to support the novel modules presented in this chapter. Section 8.3 elaborates on the shell assembler in particular, describing how the isogeometric Kirchhoff–Love shell formulation is implemented into the module. Section 8.4 describes the mathematical details behind different structural analysis routines and highlights the features of the implemented module. Section 8.5 elaborates on different unstructured spline constructions for multi-patch analysis in G+Smo . In section 8.6, results for different benchmark problems are provided. Some of these results are adopted from previous publications, but in this chapter, the aim is to elaborate on the model features behind them. Lastly, section 8.7 provides a concluding summary.

8.2 The Geometry + Simulation Modules

Since this chapter presents three novel modules for the Geometry + Simulation Modules library (G+Smo), a brief overview of G+Smo is provided here. The reader is referred to section 8.A for download and installation instructions. G+Smo is a header-only C++ library dedicated to isogeometric analysis. Being a collection of different modules, G+Smo is a versatile library for both geometric modelling and isogeometric analysis, based purely on the mathematical foundations of spline modelling. Leaving an extensive overview of G+Smo for another publication, this chapter focusses on the relevant features of G+Smo for the multi-patch shell analysis modules presented later (see sections 8.3 to 8.5).

8.2.1 Geometric Modelling

Geometric modelling in G+Smo can be performed on a wide range of spline bases and corresponding geometries. This flexibility in spline modelling is due to the fact that any basis in G+Smo inherits from gsBasis, for example tensor B-spline basis (gsTensorBSplineBasis), their NURBS counterpart (gsTensorNurbsBasis), or (Truncated) Hierarchical B-splines, or (T)HB splines [202, 203], (gsTHBSplineBasis and gsHBSplineBasis). These abstract basis objects can be collected in a gsMultiBasis to perform analysis on complex topologies. Similarly, the gsGeometry is an abstract definition of the geometric counterparts of the implemented bases in G+Smo , and the gsMultiPatch provides a container for them. The gsBasis and gsGeometry, gsMultiBasis and gsMultiPatch are widely accepted classes throughout G+Smo , meaning that multiple basis types can be used in all implemented solvers, e.g., for solving PDEs.

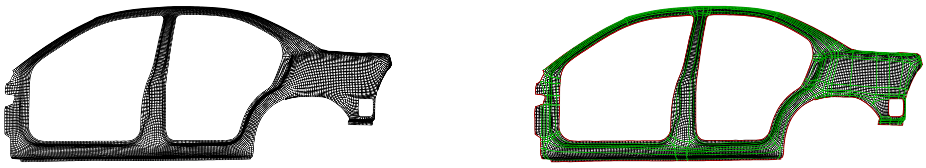
Geometric modelling in G+Smo can be used for the pre-processing of geometries and bases for structural analysis with shells. Within G+Smo , different spline geometry can be constructed and pre-processed in several ways. Firstly, CAD geometries can be imported through files, e.g., through the 3dm file format used in Rhinoceros, using the openNURBS plugin, through the IGES format, or via the Parasolid x_t. Secondly, geometries can be imported as a mesh, e.g., using the off format, which can be transformed to a bi-linear gsMultiPatch; see example 8.2.1. Thirdly, geometries can be imported as point clouds and converted to splines using (adaptive) fitting routines [66]. Lastly, boundary-represented geometries can be meshed using Coon's patches or using optimisation or PDE-based parameterisation techniques [245, 246, 284, 287].

Example 8.2.1 (Geometry import using a mesh)**.** *This example demonstrates the import of a mesh into* G+Smo *, after which it is converted to bi-linear patches. Firstly, the* gsFileData *class is used to read a* gsSurfMesh *from an* off *file; see figure 8.1a for the mesh:*

```
gsFileData<> fd("car.off");
gsSurfMesh HEmesh;
fd.getFirst<gsSurfMesh>(HEmesh);
```

The gsSurfMesh *uses the half-edge mesh structure. Using the algorithm presented in figure 7.4 (see section 7.3 of this dissertation) [589], the half-edge mesh is converted into a smaller number of patches; see figure 8.1b.*

```
gsMultiPatch<> mp = HEmesh.linear_patches();
```



(a) Mesh of the geometry stored in car.off. The mesh consists of 31086 edges, 15895 vertices, and 62172 faces.

(b) Interfaces (green) and boundaries (red) of the bi-linear patches obtained from the mesh in (a)

Figure 8.1: Mesh (a) and patch boundaries and interfaces (b) of the side of a car, read from the file car.off. The plots are obtained via the Paraview [177] plotting functions in G+Smo .

8.2.2 Assembly and Linear Algebra

Besides the geometric modelling capabilities of G+Smo , it features routines for the assembly and solving of linear systems for isogeometric analysis. Among the assembly routines available in G+Smo is the novel gsExprAssembler, used for assembly in the gsKLShell module from section 8.3. This assembler allows one to assemble linear systems based on the expressions of a weak formulation. It is shared-memory parallelised using OpenMP and supports exception handling. Furthermore, it uses abstract classes for mathematical functions, geometries (i.e., gsGeometry), and bases (i.e., gsBasis), such that it can assemble equations on different geometry definitions (e.g., a mathematical representation of a sphere or a THB-spline) using any available basis representation (e.g., gsBSplineBasis, gsTHBSplineBasis) and any implemented quadrature rule (e.g., the Gauß-Lobatto or Gauß-Legendre rules, or patch-wise quadrature [289]).

For a given linear system, G+Smo primarily uses Eigen [217] for linear algebra. For example, the gsVector, gsMatrix, and gsSparseMatrix provide interfaces to vectors and dense and sparse matrices. Furthermore, gsSparseSolver provides an interface to linear solvers in Eigen and third-party solvers such as Pardiso [452]. Eigenvalue problems can be solved using the eigenvalue solvers available in Eigen, but for sparse systems, G+Smo provides an interface to Spectra [452].

8.3 Kirchhoff–Love Shell Module

One of the novel modules presented in this chapter deals with isogeometric Kirchhoff-Love shells, as originally presented in [319]. The module is referred to as the gsKLShell module and is featured in recent works of the authors for (i) modelling stretch-based hyperelastic materials in the isogeometric Kirchhoff-Love shell framework [587], (ii) modelling wrinkling using isogeometric tension-field theory [585], (iii) the modelling of multi-patch shell problems using several unstructured spline constructions [182, 589], and (iv) adaptive isogeometric Kirchhoff-Love shell analysis with THB splines [586]. As stated in the introduction of this chapter (section 8.1), the gsKLShell module is developed within G+Smo , hence using the functionalities described in section 8.2.

In this section, a brief mathematical background of the isogeometric Kirchhoff-Love shell model is provided; see section 8.3.1. This background is provided as a reference to show which equations are assembled into systems of equations. Thereafter, section 8.3.2 elaborates on the design of the module. Table 8.4 in section 8.C provides a list of the main classes in the gsKLShell module.

8.3.1 Mathematical Background

The isogeometric Kirchhoff-Love shell was first presented in [319]. More details on this shell model were later provided in [316], and various extensions of the model have been published by other authors, e.g. stretch-based hyperelasticity for Kirchhoff–Love shells [587] (chapter 3), implicit wrinkling modelling of hyperelastic membranes using tension field theory [585] (chapter 4) or goal-oriented error estimators used for mesh adaptivity [586] (chapter 5). In this section, a brief derivation of the isogeometric Kirchoff-Love shell formulation is provided for the sake of reference, and the reader is referred to section 2.3.2 in this dissertation for a complete overview.

The variational formulation of the isogeometric Kirchhoff-Love shell model is given by:

$$\delta_{\boldsymbol{u}}\mathscr{W}^{\text{ext}}(\boldsymbol{u},\boldsymbol{v}) - \delta_{\boldsymbol{u}}\mathscr{W}^{\text{int}}(\boldsymbol{u},\boldsymbol{v}) = 0, \tag{8.1}$$

where $\delta_{\boldsymbol{u}}\mathscr{W}^{\text{ext}}(\boldsymbol{u},\boldsymbol{v})$ and $\delta_{\boldsymbol{u}}\mathscr{W}^{\text{int}}(\boldsymbol{u},\boldsymbol{v})$ are the variations of the external and internal virtual work, respectively, with respect to the displacement field $\boldsymbol{u}$ and a variation $\boldsymbol{v}$, and are given in [316, 319], among others. Using a discretization of $\boldsymbol{u}$ using B-splines, the r^{th} component of the residual vector $\boldsymbol{R}(\boldsymbol{u})$ is given by:

$$R_r(\boldsymbol{u}) = \int_\Omega \mathrm{N}(\boldsymbol{u}) : \boldsymbol{\varepsilon}_{,r}(\boldsymbol{u}) + \mathrm{M}(\boldsymbol{u}) : \boldsymbol{\kappa}_{,r}(\boldsymbol{u})\,\mathrm{d}\Omega - \int_\Omega \boldsymbol{f}\cdot\boldsymbol{u}_{,r}\,\mathrm{d}\Omega - \int_{\partial\Omega} \boldsymbol{g}\cdot\boldsymbol{u}_{,r}\,\mathrm{d}\Gamma, \tag{8.2}$$

where the product $:$ denotes the inner-product for second-order tensors, $\boldsymbol{\varepsilon}$ and $\boldsymbol{\kappa}$, $\mathrm{N}(\boldsymbol{u})$ and $\mathrm{M}(\boldsymbol{u})$ are the membrane strain, bending strain, force and bending moment tensors, respectively; see equations (2.22) and (2.35). Taking the second variation of equation (8.1) with respect to the displacement, the Jacobian matrix $K(\boldsymbol{u})$ is found, of which the entry with indices r, s is given by:

$$K_{rs} = \int_\Omega \mathrm{N}_{,s}(\boldsymbol{u}) : \boldsymbol{\varepsilon}_{,r}(\boldsymbol{u}) + \mathrm{N}(\boldsymbol{u}) : \boldsymbol{\varepsilon}_{,rs}(\boldsymbol{u}) + \mathrm{M}_{,s}(\boldsymbol{u}) : \boldsymbol{\kappa}_{,r}(\boldsymbol{u}) + \mathrm{M}(\boldsymbol{u}) : \boldsymbol{\kappa}_{,rs}(\boldsymbol{u})\,\mathrm{d}\Omega. \tag{8.3}$$

Here, subscripts $_r$ and $_s$ denote first variations, and $_{rs}$ denotes second variations of the aforementioned tensors; see equation (2.38). For linear structural analysis, the external force vector P is obtained by

$$P_r = -R_r(0) = \int_\Omega \boldsymbol{f}\cdot\boldsymbol{u}_{,r}\,\mathrm{d}\Omega + \int_{\partial\Omega} \boldsymbol{g}\cdot\boldsymbol{u}_{,r}\,\mathrm{d}\Gamma, \tag{8.4}$$

and the linear stiffness matrix K^L is obtained by

$$K^L_{rs} = K_{rs}(0) = \int_\Omega \mathrm{N}_{,s}(0) : \boldsymbol{\varepsilon}_{,r}(0) + \mathrm{M}_{,s}(0) : \boldsymbol{\kappa}_{,r}(0)\,\mathrm{d}\Omega. \tag{8.5}$$

For quasi-static and dynamic analyses, the external force vector and residual contain an extra parameter. In particular, quasi-static analysis involves the analysis of the problem with respect to a load scaled with load magnification factor λ, i.e., $\boldsymbol{f} \equiv \lambda\boldsymbol{f}$ or $\boldsymbol{g} \equiv \lambda\boldsymbol{g}$, yielding $\boldsymbol{R}(\boldsymbol{u},\lambda)$. For dynamic loading, the loads can be time-dependent, i.e., $\boldsymbol{f} \equiv \boldsymbol{f}(t)$ or $\boldsymbol{g} \equiv \boldsymbol{g}(t)$, yielding $\boldsymbol{R}(\boldsymbol{u},t)$. Furthermore, neglecting rotational inertia, the mass matrix M for a Kirchhoff-Love shell with constant density ρ and thickness t is:

$$M_{rs} = \rho t \int_\Omega u_{,r} u_{,s}\,\mathrm{d}\Omega. \tag{8.6}$$

8.3.2 Implementation

The implementation of the isogeometric Kirchhoff-Love shell model is done in the `gsKLShell` module in `G+Smo`. The module consists of several classes, of which the most important are highlighted in table 8.4; for a complete overview, the reader is referred to the documentation of `G+Smo`. In general, the module is designed so that it is fast, versatile, and forward-compatible.

Firstly, the gsThinShellAssembler is designed in such a way that the constitutive law is decoupled from the gsThinShellAssembler. The constitutive law, provided by the gsMaterialMatrixBase and its derived classes, operates as a black box within the gsThinShellAssembler. As a consequence, constitutive models can be developed as a family of models (i.e., like the gsBasis-family), providing forward compatibility with user-defined or new developments in constitutive models.

Secondly, the gsThinShellAssembler uses the gsExprAssembler for matrix assembly, providing flexibility with respect to the basis and geometry types, as well as the quadrature rules. This provides versatility in the basis and geometry types that can be used to assemble the KL shell (e.g., THB splines); it uses optimised parallel routines and patch rules from the gsExprAssembler; and it provides forward compatibility with respect to any future development that will be made to improve the gsExprAssembler. In addition, since all operations in the gsThinShellAssembler rely on generic gsBasis and gsGeometry classes, it is straightforward to apply the gsThinShellAssembler on geometries from other file types such as IGES or 3dm.

Lastly, all functionality of the gsKLShell module is binded to Python, meaning that the functionalities are not only available inside C++ but can also be called in Python. As it concerns bindings, the performance of the routines called in Python is similar to the C++ performance.

Example 8.3.1 (Non-linear analysis using Kirchhoff-Love shells)**.** *To show the use of the* gsThinShellAssembler *and the* gsMaterialMatrix*, an example code is provided. It is assumed that the geometry* (mp)*, the basis* (basis)*, the boundary conditions* (bcs)*, the body force* (force)*, the optional point loads* (pointLoads)*, and the material parameters* (E*,* nu*, and* rho) *are provided. Then, the* gsThinShellAssembler *and the* gsMaterialMatrix *are simply constructed as:*

```
// Provided mp, basis, bcs, force, pointLoads, E, nu, rho
gsMaterialMatrixLinear<3,real_t> materialMatrix(mp,thickness,E,nu,rho)
gsThinShellAssembler<3,real_t,true> assembler(mp,basis,bcs,force,materialMatrix);
assembler.addPointLoads(pointLoads);
```

Using the assembler, a linear system can be assembled using assemble()*, and it can be solved using sparse linear solvers to obtain the linear solution vector,* solution*. Then, the Jacobian matrix and the residual vector are obtained by using* assembleMatrix(solution) *and* assembleVector(solution)*, respectively, which can be used in Newton-Raphson iterations (see equation* (8.7)*) as shown below.*

```
assembler.assemble(); // Assembles equation (8.5) and equation (8.4)

gsSparseSolver<real_t>::LU solver(assembler.matrix());

// Solve the linear static problem (equation (8.7))
gsVector<real_t> solution = solver.solve(assembler.rhs());
```

```
// Solve the non-linear static problem (equation (8.7))
for (index_t it = 0; it!=10; it++)
{
  assembler.assembleMatrix(solution);        // Assembles equation (8.3)
  assembler.assembleVector(solution);        // Assembles equation (8.2)
  solver.compute(assembler.matrix());        // Factorises the matrix
  solution += solver.solve(assembler.rhs()); // Computes the update
}
```

Using solution, *a* gsMultiPatch *of the deformed shell can be constructed for post-processing:*

```
gsMultiPatch<> mp_def = assembler.constructSolution(solution);
```

8.4 Structural Analysis Module

The gsThinShellAssembler from the gsKLShell module, see section 8.3, can provide matrices and vectors resolving Kirchhoff-Love shell mechanics in isogeometric analysis. For different structural analysis applications, these matrices and vectors are used in a different way, depending on the type of analysis that is solved. For example, the analysis of eigenfrequencies of a structure requires modal analysis, and the analysis of structural stability requires linear buckling or quasi-static analysis. The gsStructuralAnalysis module implements routines for structural analysis in G+Smo . The main purpose of the module is to provide a broad range of structural analysis routines that can be reused in several applications. Moreover, the module is also home to the novel Adaptive Paralllel Arc-Length Method (APALM), as published in [584].

In this section, a brief mathematical background of different structural analysis routines is provided in section 8.4.1, for reference in later presented functions. Thereafter, section 8.4.2 elaborates on the design of the gsStructuralAnalysis module in G+Smo . In general, different structural analyses require different types of operators, for example, the linear stiffness matrix K or the force residual vector $\boldsymbol{R}(\boldsymbol{u})$, depending on the discrete displacements $\boldsymbol{u}$. Table 8.1 lists the discrete operators relevant in the structural analysis module, together with their type and a description. As will be explained in section 8.4.2, the code is designed in such a way that the operators are independent of the assemblers or discretisation method, such that the gsStructuralAnalysis module can be used with the gsKLShell module for Kirchhoff-Love shells, but also with the gsElasticity module for solids or with other modules providing discrete operators. Table 8.5 in section 8.C provides an overview of the most important classes in the gsStructuralAnalysis module.

8.4.1 Mathematical Background

A structural analysis problem typically involves a number of different operators providing matrices or vectors to construct a system to solve. As shown in section 8.3.1, the residual vector (equation (8.2)), the Jacobian (equation (8.3)), and the linearized Jacobian and force vector (equation (8.5)) can, for example, be derived. More precisely, the different linear operators that are required for performing the structural analyses described in the remainder of this section are given in table 8.1. This section provides a concise overview of the mathematics behind the structural analysis solvers implemented in the gsStructuralAnalysis module. For a complete overview, the reader is referred to section 2.4 of this dissertation,

Table 8.1: Operators for structural analysis, with their corresponding types in `G+Smo` and a description linking the operators to the mathematical theory for shells, see section 8.3.

Operator	Type	Description
$\boldsymbol{P}$	`Force_t`	The vector of external forces, see equation (8.4).
$\boldsymbol{R}(\boldsymbol{u})$ $\boldsymbol{R}(\boldsymbol{u},\lambda)$	`Residual_t` `ALResidual_t`	The vector of residual forces, depending on the discrete displacement vector $\boldsymbol{u}$ and optionally depending on the load factor λ, see equation (8.2).
K_L	`Matrix_t`	The linear stiffness matrix, see equation (8.5).
M	`Matrix_t`	The mass matrix, see equation (8.6).
$K(\boldsymbol{u})$ $K(\boldsymbol{u},\Delta\boldsymbol{u})$	`Jacobian_t` `dJacobian_t`	The Jacobian matrix, depending on the discrete displacement vector $\boldsymbol{u}$, see equation (8.3), and optionally on the displacement vector increment $\Delta\boldsymbol{u}$ for the Mixed Integration Point (MIP) method [341].

more precisely to section 2.4.1 for static analysis, to section 2.4.2 for modal analysis, to section 2.4.3 for buckling analysis and to to section 2.4.4 for post-buckling analysis.

Static Analysis

For (non-linear) static analysis, a time-independent boundary value problem with constant load magnitudes is solved. The most common approach is to solve the residual equation, i.e., the variational formulation from equation (8.1), using the Newton-Raphson method. This requires the discrete residual and the Jacobian of the system, such that the displacements $\boldsymbol{u}$ are found incrementally by:

$$K(\boldsymbol{u}^i)\Delta\boldsymbol{u} = -\boldsymbol{R}(\boldsymbol{u}^i), \quad \boldsymbol{u}^{i+1} = \boldsymbol{u}^i + \Delta\boldsymbol{u}, \quad i = 0, 1, \dots \tag{8.7}$$

Here, $\boldsymbol{u}^i$ is the solution in iteration i, and $\Delta\boldsymbol{u}$ is the solution increment. The initial solution $\boldsymbol{u}^0$ can be initialised by solving linear static analysis:

$$K_L\boldsymbol{u} = \boldsymbol{P}, \tag{8.8}$$

where $K_L = K(0)$ and $\boldsymbol{P}$ is the external load vector. The Newton-Raphson method is a well-known procedure to solve non-linear systems of equations. Although it requires the assembly of the Jacobian matrix and solving a linear system of equations in every iteration, it converges quadratically.

Alternative to the Newton-Raphson method, the Dynamic Relaxation method [419–421] provides an explicit method for the computation of static problems with slow convergence but low costs per iteration. It solves the structural dynamics equation by expanding the accelerations using finite differences and mass-proportional damping:

$$\dot{\boldsymbol{u}}_{t+\Delta t/2} = \frac{(2 - c\Delta t)}{(2 + c\Delta t)}\dot{\boldsymbol{u}}_{t-\Delta t/2} + \frac{2\Delta t}{(2 + c\Delta t)}M^{-1}\boldsymbol{R}(\boldsymbol{u}_t). \tag{8.9}$$

Where M is the mass matrix and c is the scaling coefficient of the proportional damping. Initialising the initial velocity with the zero-vector $\boldsymbol{u}_0 = 0$ and using central differences on the velocity gives the update of the displacement:

$$\boldsymbol{u}_{t+\Delta t} = \boldsymbol{u}_t + \Delta t\dot{\boldsymbol{u}}_{t+\Delta t/2}. \tag{8.10}$$

The mass and damping matrices are defined artificially, for example, using a scaled lumped mass matrix and a mass-proportional damping [32, 33, 566], introducing two parameters to the solver. Alternatively, the damping matrix can be omitted, and *kinetic damping* can be used [32, 33, 128, 508]. In this case, the kinetic energy is computed in each iteration, and when a peak occurs, the velocities are set to zero, and the iterations are restarted using the displacement vector $\boldsymbol{u}_{t+\Delta t}$ and the velocity vector $\dot{\boldsymbol{u}}_{t+\Delta t/2}$. After the peak, the displacements at the peak $\boldsymbol{u}_{t^\star}$ can be computed, along with the updated displacements and velocities after the peak [32]:

$$\boldsymbol{u}_{t^\star} = \boldsymbol{u}_{t+\Delta t} - \frac{3}{2}\dot{\boldsymbol{u}}_{t+\Delta t/2} + \frac{\Delta t}{2} M^{-1}\boldsymbol{R}(\boldsymbol{u}_t), \quad \dot{\boldsymbol{u}}_{\boldsymbol{u}_{t^\star}+\Delta t/2} = \frac{\Delta t}{2} M^{-1}\boldsymbol{R}(\boldsymbol{u}_{t^\star}.) \tag{8.11}$$

Modal Analysis

Modal analysis provides the eigenfrequencies and eigenmodes of a structure in free vibration. Starting from the structural dynamics equation, assuming no damping and a harmonic solution $\boldsymbol{u}(t) = \boldsymbol{u}_A e^{i\omega t}$, the structural dynamics equations simplify to a generalised eigenvalue problem:

$$\omega^2 M\boldsymbol{\phi} = K\boldsymbol{\phi} \tag{8.12}$$

With ω and $\boldsymbol{\phi}$ the eigenfrequency and modeshape, respectively. Given M and K_L as discrete linear operators with N degrees of freedom, the solution to equation (8.12) consists of N eigenpairs $(\omega_k, \boldsymbol{\phi}_k)$. For large N, K_L and M are sparse systems, and methods like the Shifted Block Lanczos Algorithm [212] can be used to find eigenpairs from a sparse general eigenvalue problem within a range.

Linear Buckling Analysis

Linear buckling analysis aims to find the critical load in a certain load configuration. Mathematically, this means that one needs to find the point where the stability of the structure changes, coinciding with the solution $\boldsymbol{u}$ for which $\det(K(\boldsymbol{u})) = 0$. In brief, linear buckling analysis requires a linear static solution $\boldsymbol{u}_L$ of the problem $K_L\boldsymbol{u}_L = \boldsymbol{P}$ in order to solve the following eigenvalue problem:

$$K_L\boldsymbol{\phi}_k = \lambda_k(K(\boldsymbol{u}_L) - K_L)\boldsymbol{\phi}_k. \tag{8.13}$$

Here, the sub-script $\cdot_L$ is used to emphasise the linear stiffness matrix K_L, and the eigenpair $(\lambda_k, \boldsymbol{\phi}_k)$ consists of the load magnification factor λ_k and the corresponding mode shape $\boldsymbol{\phi}_k$ for the k^{th} buckling mode, $k = 1, \dots, N$ for a system with N degrees of freedom. For more information regarding buckling analysis, the reader is referred to [70] among other references.

Post-Buckling Analysis

Post-buckling analysis involves the quasi-static analysis of a problem with time-independent but varying loads. The goal of quasi-static analysis is to solve $\boldsymbol{R}(\boldsymbol{u}, \lambda) = 0$, forming a so-called equilibrium path in the $\boldsymbol{w} = (\lambda, \boldsymbol{u})$-space. Instead of varying λ (load-control) or parts of $\boldsymbol{u}$ (displacement-control), arc-length methods find the solution to $\boldsymbol{R}(\boldsymbol{u}, \lambda) = 0$ by varying $\boldsymbol{u}$ and λ simultaneously, providing a method that allows to find paths with snap-through behaviour. Given N degrees of freedom, arc-length methods solve a system of $N+1$ equations iteratively by adding the constraint equation $f(\Delta\boldsymbol{u},\Delta\lambda)=0$ to the incremental solution $\Delta\boldsymbol{w}=(\Delta\boldsymbol{u},\Delta\lambda)$. For example, the constraint equation corresponding to Crisfield's [124] method is:

$$f(\Delta\boldsymbol{u},\Delta\lambda)=\Delta\boldsymbol{u}^\top\Delta\boldsymbol{u}+\Delta\lambda^2\boldsymbol{P}^\top\boldsymbol{P}-\Delta\ell^2=0 \tag{8.14}$$

Crisfield's method requires solving a quadratic equation, and therefore two solutions are typically found. In [588], a brief summary of techniques for robust arc-length analysis using Crisfield's method is provided, e.g., the handling of complex roots.
In case a singular point is detected, e.g., by monitoring the sign of the determinant of the Jacobian at every solution point, the singular point can be computed by solving the following system of equations [633]:

$$\begin{bmatrix}\boldsymbol{R}(\boldsymbol{u},\lambda)\\ K(\boldsymbol{u})\boldsymbol{\phi}\\ l(\boldsymbol{\phi})\end{bmatrix}=0. \tag{8.15}$$

Here, the first equation consists of the discrete residual of the system (see equation (8.2)), and the second equation implies finding the singular point by using the equivalence $\det K(\boldsymbol{u})=0 \iff K(\boldsymbol{u})\boldsymbol{\phi}=0$ for eigenvector $\boldsymbol{\phi}$, normalised using the condition $l(\boldsymbol{\phi})=\|\boldsymbol{\phi}\|-1=0$. Since the solution at the singular point is the buckling mode shape, this so-called *extended arc-length method* can be used to compute post-buckling simulations without *a priori* perturbation of the solution to switch branches described in [588].

8.4.2 Implementation

As mentioned in the beginning of this section 8.4, the `gsStructuralAnalysis` module is implemented in a way such that it is independent of the discretisation method. In other words, the `gsStructuralAnalysis` module requires the user to provide small interfaces to their discretisation methods, which will be used inside the module for the required computations. These discrete operators, summarised in table 8.1, can be used in the solvers provided in the `gsStructuralAnalysis` module, as listed in table 8.5. Furthermore, all routines in the module support exception handling, such that assembly errors or other errors can be passed robustly.

The operators listed in table 8.1, e.g., `ALResidual_t`, are used in different solvers throughout the `gsStructuralAnalysis` module and have different corresponding C++ types. The types `Force_t` and `Matrix_t` are straightforward, since they correspond to `gsVector<T>` and `gsMatrix<T>`. However, the other types require function arguments, e.g., the displacement vector $\boldsymbol{u}$ or the load magnification factor λ, and are therefore passed using functions. For example, the arc-length residual $\boldsymbol{R}(\boldsymbol{u},\lambda)$ or the Jacobian $K(\boldsymbol{u})$ have the following types (defined in `gsStructuralAnalysisOps.h`):

```
typedef std::function < bool ( gsVector<T> const &, T,     gsVector<T> & ) > ALResidual_t;
typedef std::function < bool ( gsVector<T> const &, gsSparseMatrix<T> & ) > Jacobian_t;
```

These definitions show that ALResidual_t is a function that returns a boolean (true if the assembly is successful, bool) and takes the vector of displacments ($\boldsymbol{u}$, gsVector<T> const &), a real number for the load magnification factor (λ, T), and in-place the residual vector ($\boldsymbol{R}(\boldsymbol{u},\lambda)$, gsVector<T> &) as input arguments. Similarly, the type Jacobian_t is a function returning a boolean for successful assembly that takes the vector of displacements ($\boldsymbol{u}$, gsVector<T> const &) and returns the Jacobian matrix in-place ($K(\boldsymbol{u})$, gsSparseMatrix<T> &). For the other operators with function arguments in table 8.1, the definitions work in a similar way. Eventually, the definition of an operator depends on the interface with the discretisation method. In example 8.4.1, an example is provided for the operator definitions using the gsThinShellAssembler to initialise the gsStaticNewton solver.

Example 8.4.1 (Linear operators for a non-linear static analysis using Kirchhoff-Love shells)**.** *When using the* gsThinShellAssembler *from the* gsKLShell *module, see section 8.3, the linear stiffness matrix K and the external force vector $\boldsymbol{P}$ are simply assembled by calling* assemble() *and obtained by calling* matrix() *and* rhs()*, respectively. As shown in example 8.3.1, the Jacobian and residual are assembled using* assembleMatrix(solution) *and* assembleVector(solution)*, respectively. Using this syntax, the function* ALResidual *of type* ALResidual_t *can be defined as follows:*

```
gsStructuralAnalysisOps<T>::ALResidual_t ALResidual;
ALResidual = [&assembler,&Force](gsVector<T> const &u, T lam, gsVector<T> & result)
{
  // Assemble eq. 8.2
  ThinShellAssemblerStatus status = assembler->assembleVector(u);
  result = Force - lam * Force - assembler->rhs();
  return status == ThinShellAssemblerStatus::Success;
};
```

Here, it can be seen that the ThinShellAssemblerStatus *returned by the assembly in the* gsThinShellAssembler *is used to verify whether the assembly is successful. Furthermore, an external force vector* Force *is used together with the residual $\boldsymbol{R}(\boldsymbol{u}) = \boldsymbol{F}_{int} - \boldsymbol{F}_{ext}$ to define $\boldsymbol{R}(\boldsymbol{u},\lambda) = \boldsymbol{R}(\boldsymbol{u}) + \boldsymbol{F}_{ext} - \lambda\boldsymbol{F}_{ext}$. Similarly, the Jacobian operator can be defined as:*

```
gsStructuralAnalysisOps<real_t>::Jacobian_t Jacobian;
Jacobian = [&assembler](gsVector<real_t> const &u, gsSparseMatrix<real_t> & m)
{
  // Assemble eq. 8.3
  ThinShellAssemblerStatus status = assembler->assembleMatrix(u);
  m = assembler->matrix();
  return status == ThinShellAssemblerStatus::Success;
};
```

The gsThinShellAssembler *also provides an implementation of the Mixed Integration Method (MIP) [341], which is a method to accelerate Newton iterations by decoupling the stress update from the displacements. In this case, the Jacobian matrix is given by $K(\boldsymbol{u},\Delta\boldsymbol{u})$ and defined as follows:*

```
gsStructuralAnalysisOps<real_t>::dJacobian_t dJacobian;
dJacobian = [&assembler](gsVector<real_t> const &u,
                         gsVector<real_t> const &Du,
                         gsSparseMatrix<real_t> & m)
{
  ThinShellAssemblerStatus status = assembler->assembleMatrix(u,Du);
  m = assembler->matrix();
  return status == ThinShellAssemblerStatus::Success;
};
```

The gsStaticNewton *and* gsALMCrisfield *solvers of the* gsStructuralAnalysis *module are compatible with the MIP method. Using the external load factor, the arc-length residual and Jacobian operators,* $\mathbf{P}$, $\mathbf{R}(\mathbf{u},\lambda)$, *and* $K(\mathbf{u})$, *respectively, an arc-length method can be setup. For example, a Crisfield arc-length solver is defined as follows:*

```
// Provided Force, ALResidual, Jacobian, options, dL
// Define a Crisfield method, with eq. 8.14
gsALMCrisfield<T> arcLength(Force,ALResidual,Jacobian);
// Set the length
arcLength.setReal("Length",dL);
```

8.5 Unstructured Splines Module

Although the Kirchhoff-Love shell implementation in the gsThinShellAssembler from gsKLShell, see section 8.3 contains an implementation of the penalty method [238], C^1 patch coupling using unstructured splines provides another possibility to model complex geometries. In this section, the gsUnstructuredSplines module is introduced. This module provides ready-to-use implementations to generate unstructured spline constructions on multi-patches. Section 8.5.1 provides a brief mathematical background to explain the idea behind the implementation presented in section 8.5.2. Table 8.6 in section 8.C gives an overview of the most important classes of the gsUnstructuredSplines module.

The unstructured splines module is intensively used in the previous work [589], where qualitative and quantitative comparisons are given of multiple unstructured spline routines. Furthermore, the work [378] develops a G^1 construction constructed in third-party software but imported into G+Smo as a combination of a sparse matrix and a local basis to perform different kinds of analyses. To the best of the author's knowledge, G+Smo has a unique position in unstructured splines modelling, being the only open-source code for isogeometric analysis and providing an interface for the development of unstructured splines together with advanced implementations of existing schemes.

8.5.1 Mathematical Background

The mathematical background behind the unstructured spline module is based on the concept of mapping splines defined on local patches into a global smooth space. In other words, a globally smooth basis function $\varphi_k \in \mathbb{S}^{\text{global}}$ can be represented by a weighted sum of locally defined basis functions $\psi_l \in \mathbb{S}^{\text{local}}$ through the coefficients A_{kl}. Therefore, provided a vector of evaluations ψ_l on a point ξ located in the parametric domain of one of the patches, $\boldsymbol{\psi}(\xi)$, the globally smooth basis at point ξ is:

$$\boldsymbol{\varphi}(\xi) = A\boldsymbol{\psi}(\xi). \tag{8.16}$$

For example, for a simple one-dimensional domain with two bases separated by a C^0 discontinuity, a C^1-smooth basis over the C^0 interface is constructed as shown in figure 2.8. In higher dimensions, interface smoothing, as illustrated in figure 2.8, can be performed to construct interface basis functions, but the construction of C^1 functions over vertices can be challenging. Examples of such constructions include the D-Patch [464, 569], the Almost-C^1 construction [534], the Approximate C^1 basis [618, 619], the Analysis-Suitable G^1 construction [109, 181, 182], Polar spline constructions [568] and constructions based on subdivision surfaces [30, 378].

8.5.2 Implementation

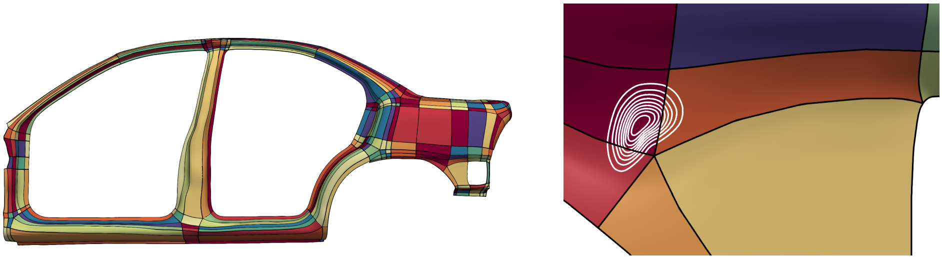
In `G+Smo` , unstructured splines can be modelled using the `gsMappedBasis` and `gsMappedSpline` classes, both providing functions to evaluate a basis or a geometry that is mapped as a linear combination of underlying bases; see equation (8.16). The `gsMappedBasis` and `gsMappedSpline` classes are part of `G+Smo` 's core and are respectively both derived from `gsFunctionSet`. This implies that both classes can be used as a basis or geometry, respectively, in assembly routines, for example.

The `gsUnstructuredSplines` provide the implementation of the D-Patch [464, 569], the Almost-C^1 construction [534], the Approximate C^1 basis [618, 619], and the Analysis-Suitable G^1 construction [109, 181, 182] constructions as ready-to-use classes; see table 8.6 for an overview. In other words, provided a multi-patch geometry, these classes provide a local basis as a `gsMultiBasis` and a mapping matrix `gsSparseMatrix`, or they directly construct a `gsMappedBasis` or `gsMappedSpline`. Example 8.5.1 provides an example of the construction of the Almost-C^1 basis and geometry.

Example 8.5.1 (Unstructured spline construction)**.** *Given the bi-linear multi-patch object* `mp` *from example 8.2.1, knot refinement and degree elevation steps can be performed to make the geometry suitable for the Almost-C^1 construction, after which this construction can be computed:*

```
mp.degreeIncrease(); // Increases the degree by 1 (default)
mp.uniformRefine();  // Refines the mesh by inserting 1 knot in each interval (default)

gsAlmostC1<2,real_t> almostC1(mp);
almostC1.options().setSwitch("SharpCorners",true); // Maintain C0 corners
almostC1.compute();
almostC1.matrix_into(global2local);

gsMappedBasis<2,real_t>  mbasis;
gsMappedSpline<2,real_t> mspline;
almostC1.globalBasis_into(mbasis);
almostC1.globalGeometry_into(mspline);
```

The resulting geometry, using the mesh from figure 8.1a in example 8.2.1, is given in figure 8.2a, with the contours of one Almost-C^1 basis around a valence $v = 5$ extraordinary vertex highlighted in figure 8.2b.



(a) Geometry of the side of the car.

(b) Contour lines of an Almost-C^1 vertex function around an extraordinary vertex (bottom-left of a).

Figure 8.2: The multi-patch geometry constructed using the Almost-C^1 method [534] (a) with a basis function highlighted using contour lines (a). The patches are coloured randomly for the sake of referencing.

8.6 Usage Examples

To demonstrate the capabilities of the software presented in this chapter, this section presents a number of usage examples. Most of the use cases are based on previous works, and for physical or mathematical interpretation, the reader is referred to these works. Instead, the usage examples are presented here with the aim of providing implementation background. Firstly, section 8.6.1 demonstrates different ways of defining materials in `G+Smo` , and is based on [587]. Secondly, section 8.6.2 provides examples of multi-patch shell analysis using the penalty method [238] or unstructured splines [589]. Thirdly, section 8.6.3 demonstrates the use of arc-length methods and the Adaptive Parallel Arc-Length Method (APALM) [584] for quasi-static analysis. Fourth, section 8.6.4 demonstrates adaptive meshing using goal-oriented error estimators. Lastly, section 8.6.5 provides an example of the Python bindings of the `gsKLShell` module, applied to a shape optimisation problem.

Throughout the usage examples, code snippets are provided to illustrate the use of the feature presented in the example. For the sake of brevity, the following variables are defined here and not repeated in the examples below:

- `mp`, `mp_def`: a (deformed) multi-patch geometry, using `gsMultiPatch`
- `basis`: a basis corresponding to a multi-patch, using `gsMultiBasis`
- `bcs`: a set of boundary conditions, using `gsBoundaryConditions`
- `rho`, `t`: the density ρ and thickness t of the shell, using `gsFunctionSet`
- `materialMatrix`: a constitutive model, using the classes in table 8.4

8.6.1 Material Modelling

One of the features of the `gsKLShell` module is its versatility with respect to constitutive modelling. By decoupling the constitutive models `gsMaterialMatrixBase` from the kinematics and assembly `gsThinShellAssembler`, see table 8.4, different material models can be used with the generic shell assembler with a minimal interface. In this section, an example of material selection is provided, after which the novel hyperelastic tension field theory-based material model is demonstrated.

Material Specification

Consider the uniaxial tension case depicted in figure 3.1 for a membrane with geometric dimensions $L \times W \times t = 1 \times 1 \times 0.001\,[\mathrm{m}^3]$ and material parameters defined via Lamé parameter $\mu = E/(2(1+\nu)) = 1.5 \cdot 10^6\,[\mathrm{N/m}^2]$ using a Poisson's ratio $\nu = 0.45\,[-]$ such that the Young's modulus E can be inferred. For a Mooney-Rivlin material, $\mu = \mu_1 + \mu_2$, with $\mu_1/\mu_2 = 7$. To investigate the uniaxial tension behaviour of a material using different material laws, material model specification of different models is necessary. In the following, a demonstration of defining different material models in different ways using the `gsKLShell` module is given. The uniaxial tension example is adopted from [587], and the model parameters are given in figure 3.1. Here, the example material definitions are given for compressible Neo-Hookean and Mooney-Rivlin material models.

To model the material laws for the uniaxial tension case, the material classes from table 8.4 need to be defined. As can be seen in example 8.3.1, a pointer to a material matrix, deriving from `gsMaterialMatrixBase`, needs to be passed to the `gsThinShellAssembler`. This can be done by explicitly constructing a material matrix and passing it into the assembler, as done in example 8.3.1. Alternatively, one can use the `getMaterialMatrix` helper function, for example:

```
// Provided mp, t, rho, ratio
// Define a MR material model with analytically implemented S and C tensors.
gsOptionList options;
options.addInt("Material","3 = Mooney-Rivlin",3);
options.addInt("Implementation","1 = Analytical",1);
gsConstantFunction<real_t> E(1.5e6*2*(1+0.45),3); // Young's modulus: E=mu*2*(1+nu)
gsConstantFunction<real_t> nu(0.45,3);            // Poisson's ratio
gsConstantFunction<real_t> Ratio(7.0,3);          // The material lives in the 3D domain
std::vector<gsFunctionSet<real_t> *> parameters{&E,&nu,&Ratio};
gsMaterialMatrixBase<real_t> * materialMatrix =
  getMaterialMatrix(mp,t,parameters,rho,options);
```

Here, the `parameters` object contains a list of functions defining the material properties. These functions, as well as the thickness `t` and the density `rho`, are passed as `gsFunctionSet` pointers, meaning that they can be defined using any type of function definition in `G+Smo` . This allows us to define material properties as splines, allowing optimisation of material properties in an isogeometric setting. In the example above, the material parameters are simply passed as `gsConstantFunction` objects, with the function value as the first argument and the geometric domain dimension as the second argument.

Alternative to the `getMaterialMatrix` routine, one can define a material matrix using `G+Smo` 's XML format, providing an XML item, for example, for a compressible Neo-Hookean material:

```
<MaterialMatrix type="IncompressibleNH3" id="0" TFT="false">
    <Thickness>
        <Function type="FunctionExpr" dim="3" index="0">0.0001</Function>
    </Thickness>
    <Density>
        <Function type=”FunctionExpr” dim=”3” index=”0”>0</Function>
    </Density>
    <Parameters>
        <!-- Youngs Modulus -->
        <Function type=”FunctionExpr” dim=”3” index=”0”>1.5e6*2*(1+0.45)</Function>
        <-- Poisson Ratio      -->
        <Function type=”FunctionExpr” dim=”3” index=”1”>0.45</Function>
    </Parameters>
</MaterialMatrix>
```

Using the material matrices defined above, either defined using getMaterialMatrix, constructed directly, or read via XML, the gsThinShellAssembler can be constructed as in example 8.3.1, and the Residual_t and Jacobian_t operators can be defined to perform non-linear arc-length analysis with an increasing load. The reader is referred to example 8.4.1 for the definition of the required solver. Finally, the results presented in figure 3.2 are obtained by using the material specifications from above.

Tension Field Theory

Consider the cylinder depicted in figure 4.7, with a radius of $R = 250\,[\text{mm}]$, length $H = 1.0\,[\text{mm}]$, thickness $t = 0.05\,[\text{mm}]$ and material parameters $E = 1.0\,[\text{GPa}]$ and $\nu = 0.5\,[-]$ subject to a boundary rotation $\theta = \pi/2\,[\text{rad}]$ and translation $u_x = H$. When modelling this example with a shell model, the deformation pattern is composed of diagonal wrinkles over the length of the cylinder; see figure 4.8. Since modelling the membrane wrinkles could require a large number of elements, an alternative way of modelling this benchmark is to model only the mid-plane of the wrinkled geometry using a modified membrane wrinkling model. Here, the use of the novel hyperelastic tension field theory-based model for general hyperelastic material models [585] is demonstrated, within the gsKLShell module.

Tension field theory-based membrane models are enabled by the gsMaterialMatrixTFT class. This class uses tension field – fields based on principal strains and stresses – to modify the stress and material tensors $\mathbf{S}$ and $\mathscr{C}$ according to the modification scheme from [401, 402] for linear elastic materials and from [585] for hyperelastic materials. For each quadrature point, the stress and material tensors are determined by the tension field: they are set to zero (or a small number) in the slack state, they are modified according to the schemes in [401, 402, 585] in the wrinkled state, and they are unmodified in the taut state.

Since the gsMaterialMatrixTFT class inherits from gsMaterialMatrixBase, it can be used in the gsThinShellAssembler like any other material model in gsKLShell. The class gsMaterialMatrixTFT can be initialised by setting the TFT flag in the XML specification to true, or by constructing the object as:

```
// Provided materialMatrix<d,real_t>
gsMaterialMatrixTFT<d,real_t,true> materialMatrixTFT(&materialMatrix);
materialMatrixTFT.options().setInt();
```

Here, the template parameter d specifies the geometric dimension, either 2 or 3, and the last template parameter specifies whether a linear modification (true) [401, 402] or a non-linear modification (false) [585] should be used. The results in figure 4.8 are obtained by a two-stage static solver composed of a dynamic relaxation method followed by a Newton-Raphson method, defined as follows:

```
// Provided F, M, K, Residual, Jacobian, alpha
gsStaticDR<real_t> DR(M,F,Residual);
DR.options().setReal(”alpha”,alpha);
DR.options().setReal(”tol”,1e-2);
DR.initialize();
gsStaticNewton<real_t> NR(K,F,Jacobian,Residual);
NR.options().setReal(”tol”,1e-6);
NR.initialize();
gsStaticComposite<real_t> solver({&DR,&NR});
solver.solve();
```

Here, the parameter `alpha` is the tuning parameter for the dynamic relaxation method. Furthermore, the tolerances specified are the relative residual tolerances.

8.6.2 Multi-Patch Analysis

In order to model Kirchhoff-Love shell problems on complex multi-patch geometries, C^1 continuity of the basis needs to be ensured on patch interfaces. To meet this continuity requirement, various techniques can be used, as reviewed in [589]. Among these techniques are the penalty coupling method for shells [238] and unstructured spline constructions. In this section, two examples for multi-patch shell analysis are given: using penalty coupling (section 8.6.2) and using unstructured spline constructions (section 8.6.2).

Weak Coupling

Consider the T-beam depicted in figure 8.3 fixed at its side, Γ. The beam has length $L = 10.0\,[\mathrm{m}]$, flange width $w = 2.0\,[\mathrm{m}]$, height $h = 2.0\,[\mathrm{m}]$ and thickness $t = 0.1\,[\mathrm{m}]$ and is exposed to a point load $F = 10.0\,[\mathrm{N}]$ and has material properties $E = 10.0\cdot 10^7\,[\mathrm{Pa}]$, $\nu = 0.0\,[-]$, modelled through a linear Saint-Venant Kirchhoff material model. The beam can be modelled using a multi-patch geometry with three patches: one for the web and two for the flange, as shown in the work by [238]. On the interface of these patches, penalty coupling for the isogeometric Kirchhoff-Love shell can be applied to find the response of the beam. Contrary to unstructured splines, the penalty method provides versatility in interface coupling, for instance,pling patches with non-matching interfaces or patches that are joined with C^0 continuity. However, the penalty method requires a penalty coefficient, which needs to be selected a priori.

Penalty coupling is established by adding extra energy contributions to the weak formulation of the isogeometric Kirchhoff-Love shell (see [238]). The penalty method is implemented using a single parameter for both rotations and displacements, according to the work by [238]. To activate penalty coupling on selected interfaces, stored in `interfaces`, of a multi-patch geometry, only the following options need to be set to the `gsThinShellAssembler`:

```
// Provided mp, assembler
std::vector<boundaryInterface> interfaces = mp.interfaces();
assembler.addWeakC0(interfaces);
assembler.addWeakC1(interfaces);
assembler.initInterfaces();
assembler.options().setReal(”PenaltyIfc”,1e3);
```

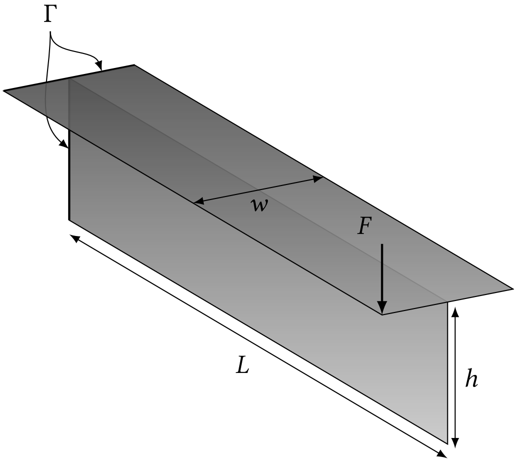


Figure 8.3: Problem setup for the T-beam example, inspired by [238].

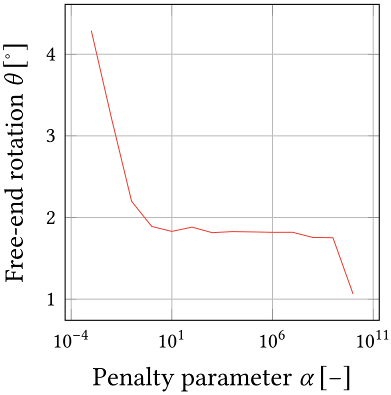


Figure 8.4: The rotation of the end-point of the beam with respect to the penalty parameter α.

Here, `interfaces` is a vector for storing all interfaces of the multi-patch geometry. This vector can be modified to apply penalty coupling to a selection of the patch interfaces. Furthermore, weak boundary conditions have also been implemented in the `gsThinShellAssembler`, as also proposed by [238]. Finally, using the lines provided, the response of the beam can be computed for different penalty parameters, as shown in figure 8.4.

Strong Coupling

Provided the geometry of the car from examples 8.2.1 and 8.5.1, with matching patches. On this geometry, an unstructured spline basis can be constructed such that C^1 isogeometric analysis can be performed. In particular, a modal analysis of this geometry is performed to infer the eigenfrequencies of the car part. The material used in this example is steel with material parameters $E = 210 \cdot 10^3$ [MPa], $\nu = 0.3$ [−], $\rho = 7850$ [tonnes/mm^3] and a thickness of $t = 10$ [mm]. This example is inspired by [589], to which the reader is referred for a comparison of multiple methods with respect to a commercial finite element code.

Strong coupling of multi-patches is enabled by using unstructured spline constructions, enabled through the `gsUnstructuredSplines` module (see section 8.5). To perform shell analysis on `gsMappedBasis` and `gsMappedSpline`, the `gsThinShellAssembler` is generalised to work on geometries defined by arbitrary functions (i.e., inherited from `gsFunctionSet`), which in turn support computation of first and second derivatives, normals, and other geometric data. Moreover, the `gsThinShellAssembler` allows to store a `gsMappedBasis` object with an underlying local basis provided as a `gsMultiBasis` used for quadrature definitions. Using the `mbasis` and `mspline` defined in example 8.5.1, the `gsThinShellAssembler` can be configured to use unstructured splines as follows:

```
// Provided bases, mbasis, mspline, and assembler
assembler.setSpaceBasis(bases);
assembler.setSmoothBasis(mbasis);
assembler.setGeometry(mspline);
```

Where the objects bases, mbasis and mspline follow, for example, from the Almost-C^1 method [534], as demonstrated in example 8.5.1. For the modal analysis, the gsModalSolver solver is used, compiled with Spectra. Using the shell assembler, this solver is simply set up as:

```
// Provided assembler
gsSparseMatrix<real_t> K, M;
assembler.assemble();         // Asseble eqs. 8.5 and 8.4
K = assembler.matrix();       // Collect eq. 8.5
assembler.assembleMass();     // Assemble eq. 8.6
M = assembler.matrix();       // Collect eq. 8.6
gsModalSolver<> solver(K,M); // Define the solver
solver.compute();             // Compute eq. 8.12
```

Using this solver with the almost-C^1 basis, the vibration modes of the car geometry are obtained, as presented in figure 7.16 and table 7.3. For more details, the reader is referred to section 7.4.4.

8.6.3 Structural Stability Analysis

In order to demonstrate the gsALMBase and gsAPALM solvers from the gsStructuralAnalysis class, two examples regarding the analysis of structural instabilities are presented. Firstly, the set-up of an arc-length simulation with singular point computation for solving a bifurcation problem without imposing initial perturbations is presented. This method relies on solving the system in equation (8.15). Thereafter, the use of the Adaptive Parallel Arc-Length Method from [584] is presented.

Bifurcation Instabilities

Provided a rectangular membrane with length $L = 280\,[\text{mm}]$, width $W = 140\,[\text{mm}]$ and thickness $t = 0.14\,[\text{mm}]$, with material parameters $\mu = \mu_1 + \mu_2 = 12.43 \cdot 10^4 + 31.62 \cdot 10^4\,[\text{Pa}]$ for an incompressible Mooney-Rivlin material model ($\nu = 0.5\,[-]$), which is fixed on the left end and clamped but pulled on the right end, as depicted in figure 3.11. When the load P is increased, tension is applied, and longitudinal wrinkles will appear and disappear as the strain ε increases. This benchmark example is based on the work of [427] and replicated from [587]. This benchmark example can be modelled using arc-length methods with the extended arc-length method to compute the bifurcation point; see section 8.4.1, which will be demonstrated in the following.

Provided a gsThinShellAssembler and a gsALMCrisfield arc-length method (see example 8.4.1), an arc-length simulation including bifurcation point detection, computation, and branch switching is performed by:

```
// Provided arcLength, Nsteps
index_t k=0;
while (k<Nsteps)
{
    gsStatus status = arcLength.step();
    if (status==gsStatus::NotConverged || status==gsStatus::AssemblyError)
    {
        arcLength.reduceLength(0.5);      // Multiplies the current length by 0.5
        continue;
    }
    arcLength.computeStability();
    if (arcLength.stabilityChange())      // if true, a limit point or bifurcation point
    {                                     // is found
        arcLength.computeSingularPoint(); // Compute eq. 8.15
        arcLength.switchBranch();
        }
    arcLength.resetLength();
    k++;
}
```

As can be seen in the code snippet above, the arc length is reduced when the step is unsuccessful. Furthermore, the singular point computation is activated if the stability of the structure changes, which is checked through internal stability computations based on what is described in section 8.4.1. In figure 3.12a, the resulting solution of the wrinkling benchmark is provided as the uniform meshing results. For more details, the reader is referred to section 3.5.4 or section 5.6.6.

Limit-Point Instabilities

Provided the geometry of a snapping meta-material inspired by [455], see figure 6.12. To model the response of this metamaterial, different strategies can be used. One can employ displacement-controlled simulation by incrementally fixing the displacement on the top boundary and computing the corresponding deformations in the domain. Alternatively, one can use arc-length methods to find the snapping behaviour of the metamaterial. In the latter case, the Adaptive Parallel Arc-Length Method can be used to speed up computations, as demonstrated in [584]. In the following, the setup of a displacement-controlled simulation is demonstrated, followed by the setup of an arc-length method, both using the `gsStructuralAnalysis` module.

Firstly, a displacement-controlled simulation can be done by incrementally increasing the Dirichlet boundary condition on Γ_2 and performing a static solve in each load step. This routine is simplified by the `gsControlDisplacement` class, taking a static solver as an input:

```
// Provided K, F, Jacobian, ALResidual, assembler, topBdr, b, W, H,
// probePatch, probePoint, Emax
gsStaticNewton<real_t> staticNR(K,F,Jacobian,ALResidual);
gsControlDisplacement<real_t> control(&staticNR);
real_t sigma, eps;
gsMultiPatch<real_t> deformed, displacement;
while (eps<=Emax)
{
    // Perform a step
    control.step(dy);
    // Construct the deformed geometry
    assembler.constructSolution(U,deformed);
    // Compute the equivalent stress on the top boundary
    sigma = assembler.boundaryForce(deformed,topBdr)[1] / (b*W);
    // Construct the displacement field
    assembler.constructDisplacement(solVector,displacement);
    // Compute epsilon based on the displacement at probePatch and probePoint;
    eps = displacement.patch(probePatch).eval(probePoint) / H;
}
```

Here, the probePoint defined on a probePatch is used to evaluate the displacement of the top boundary for the computation of the total strain ε of the metamaterial.

Secondly, the recently developed Adaptive Parallel Arc-Length Method (APALM) [584] allows to run quasi-static simulations in parallel and is implemented in gsAPALM. Since the APALM can be implemented in a non-intrusive way and therefore can be simply wrapped around a gsALMBase class. To keep track of the solutions, the gsAPALMData accompanies the gsAPALM class. In G+Smo , the APALM is used as follows:

```
// Provided arcLength
gsAPALMData<real_t,solution_t> apalmData;
const gsMpi & MPI               = gsMpi::init(argc, argv);
gsMpiComm             MPI_COMM = MPI.worldComm();
gsAPALM<real_t> apalm(arcLength,apalmData,MPI_COMM);
```

Here, the second template argument of the class gsAPALMData is the way solutions are provided. Here, solution_t is std::pair<gsVector<real_t>,real_t>, where the first vector is the discrete vector of solutions and the second is a scalar representing the load. Furthermore, it can be seen that the APALM algorithms are decoupled from the APALM data structure. The reason for this is that the data structure in gsAPALMData could be used for other applications beyond arc-length methods. The maximum refinement level of the APALM routine and the tolerance are specified in the options of the data structure gsAPALMData, since the data structure decides whether an interval should be refined or not. The number of sub-intervals per refinement is specified in the gsAPALM class, since this is the class submitting solutions into the data structure. When the APALM is defined, the function solve(N) can be called to perform fully parallel load-stepping using the APALM using N steps, or the functions serialSolve(N) and parallelSolve(N) can be called to perform serial initialisation and parallel correction, respectively, in a segregated way (see [584]). The gsAPALM class is compatible with singular point detection and approach, as described above.

The equilibrium paths following from the APALM and the displacement-controlled simulation are provided in figure 6.13. For a discussion of the results, the reader is referred to section 6.5.4.

8.6.4 Error Analysis and Adaptivity

Provided the wrinkling problem from sections 5.6.6 and 8.6.3, an adaptive meshing procedure can be set up. Here, the setup of the dual-weighted residual (DWR) error estimator for Kirchhoff-Love shells is presented, which allows for goal-oriented error estimation, hence goal-adaptive meshing. For more details, the reader is referred to the paper [586].

The DWR method provided in the gsKLShell module provides error estimates in terms of goal functionals such that adaptive meshing can be used. The use of the error estimation routines is implemented in a straight-forward way. The DWR method requires a primal and dual problem to be solved, defined on the basis used for analysis, $\mathbb{S}_h^p$, and an enriched basis, $\tilde{\mathbb{S}}_h^p$. Since the error estimator is defined on two spaces, the gsThinShellAssemblerDWR class is called the regular gsThinShellAssemblerDWR, but with two bases instead of one. The configuration of the gsThinShellAssemblerDWR works exactly the same as the configuration of a gsThinShellAssembler, but it adds the configuration of a goal functional via the setGoal:

```
// Provided mp, basisL, basisH, bcs, force, materialMatrix
gsThinShellAssemblerDWR<d,T,bool> DWRassembler(mp,basisL,basisH,bcs,force,materialMatrix);
DWRassembler.setGoal(GoalFunction::MembraneStress)
```

Here, the basisL and basisH are the bases $\mathbb{S}_h^p$ and $\tilde{\mathbb{S}}_h^p$, respectively, and the other variables are as in example 8.3.1. In order to estimate the error using the DWR, the (non-linear) primal problem has to be solved on $\mathbb{S}_h^p$, and two linear dual problems have to be solved on $\mathbb{S}_h^p$ and $\tilde{\mathbb{S}}_h^p$, respectively. The matrices and vectors for the dual problems are assembled for the provided goal functional using assembler.assembleDualL() and can be accessed by assembler.matrixL() and assembler.dualL() for the matrix and vector, respectively. Since this routine is the same for all problems involving DWR error estimation, the error estimation part has been integrated in the gsDWRHelper. Mesh adaptivity can be performed using the gsAdaptiveMeshing class, for which the reader is referred to the documentation. Given an arc-length method (see example 8.4.1), a gsDWRHelper, and an adaptive meshing class, the following loop can be used to adaptively refine a snapping example:

```
// Provided DWRassembler, arcLength, mp
gsAdaptiveMeshing<real_t>    mesher(mp);
gsThinShellDWRHelper<real_t> DWRhelper(DWRassembler);
std::vector<real_t>          elErrors;              // element-wise errors
gsVector<real_t>             solutionVector;
gsHBoxContainer<2,real_t>    markRef, markCrs;      // Containers for marked elements
for (index_t k=0; k!=Nsteps; k++)
{
    arcLength.step();                               // Perform an arc-length step
    solutionVector = arcLength.solutionU();         // Obtain the solution vector
    DWRhelper.computeError(solutionVector);         // Compute the errors (element-wise)
    elErrors = DWRhelper.sqErrors(true);            // Obtain squared errors, normalised
                                                    // w.r.t. the global squared error:
    mesher.markRef_into(elErrors,markRef);          // Mark elements for refinement
    mesher.markCrs_into(elErrors,markRef,markCrs);  // Mark elements for coarsening,
                                                    // provided markRef
    mesher.refine(markRef);                         // Refine the geometry
    mesher.unrefine(markCrs);                       // Unrefine the geometry
}
```

This code, which is a simplified version of the actual code used for the wrinkling example (see benchmark_Wrinkling_DWR.cpp), can be used for any goal functional chosen in gsThinShellAssemblerDWR and for refinement using hierarchical and truncated hierarchical B-splines, which are refined using the suitable grading algorithms from [67] by default within the gsAdaptiveMeshing and gsHBoxContainer classes. For more information, the reader is referred to the documentation of these classes and the corresponding examples. In figures 5.22 and 5.23, the results of the goal-adaptive meshing simulation of the tension wrinkling benchmark problem are provided. For more details, the reader is referred to section 5.6.6.

8.6.5 Python Interface

One feature of `G+Smo` is the fact that the classes and their functions are callable inside Python, powered by `Pybind11` [276]. This means that `G+Smo` routines can be used to conceptualise ideas inside Python, with the speed of C++. As the classes in the `gsKLShell` module provide off-the-shelf assembly of the isogeometric Kirchhoff-Love shell equations, the Python bindings of these classes allow users to assemble shell models inside Python. In this section, a brief example in the Python language is given to illustrate the use of `G+Smo` in Python through the `pygismo` module. In particular, this example demonstrates the use of the `gsKLShell` through `pygismo` to perform shape optimisation on the geometry defined in figure 8.5a through the following optimisation problem:

$$\begin{cases} \min_{B} \max_{\Omega} |u_z| \\ \text{s.t.} \int_{\Omega} |g_{\alpha\beta}| \, \mathrm{d}\Omega = \text{constant} \end{cases} \tag{8.17}$$

Here, B is the matrix of control points of the geometry and Ω is the domain. Hence, the objective function aims to find the control points B such that the maximum abosolute displacement in the domain is minimised. The nonlinear constraint keeps the surface area of Ω constant.

Firstly, a `gsThinShellAssembler` and a `gsMaterialMatrix` can be created inside Python as follows, similar to example 8.3.1:

```python
import pygismo as gs

# Define mp, basis, t, bcs, f, E, nu, pload
mm = gs.klshell.gsMaterialMatrixLinear3(mp,t)
mm.setYoungsModulus(E)
mm.setPoissonsRatio(nu)

assembler = gs.klshell.gsThinShellAssembler3(mp,basis,bcs,f,mm)
assembler.setPointLoads(pload)
```

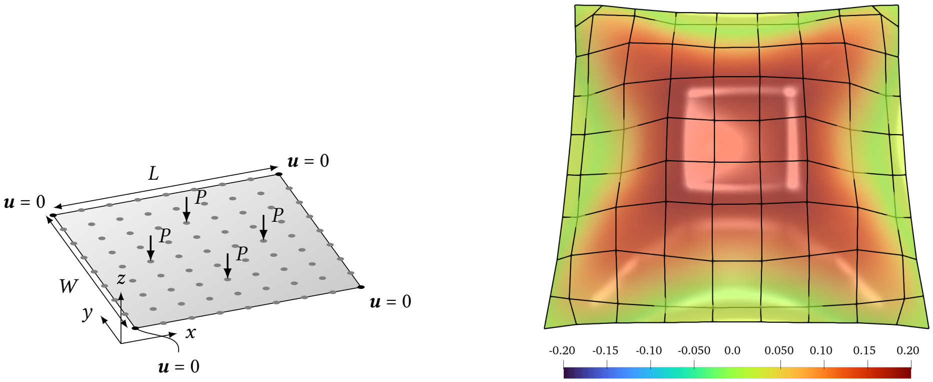
Given a `makeGeometry` function in Python, which creates a `gsMultiPatch` from the design variables, an objective for computing the minimum deformation by sampling 100 × 100 points in the domain is defined as follows:

```python
# Computes the optimisation objective
def computeObjective(design):
    mp_tmp = makeGeometry(design)

    assembler.setGeometry(mp_tmp)
    assembler.assemble()
    matrix = assembler.matrix()
    vector = assembler.rhs()
    solution = scipy.linalg.spsolve(matrix,vector[:,0])

    sol = assembler.constructDisplacement(solution)

    nx = ny = 100
    x = np.linspace(0, 1, nx)
    y = np.linspace(0, 1, ny)
    xv, yv = np.meshgrid(x,y,indexing='xy')
    pts = np.stack((xv.flatten(),yv.flatten()))

    deformation = sol.patch(0).eval(pts)
    return np.max(np.abs(deformation[2,:]))
```

Furthermore, the non-linear constraint equation to keep the area of the geometry constant is simply defined using the getArea() function of the gsThinShellAssembler:

```
# Computes the area constraint
def computeConstraint(design):
    mp_tmp = makeGeometry(design)
    return assembler.getArea(mp_tmp) - assembler.getArea(mp)
```

Finally, the computeObjective and computeConstraint can be passed to minimisation algorithms from SciPy to solve the optimisation problem from equation (8.17). The full file is given by example_shell3D_opt.py, and the result is provided in figure 8.5b.



(a) Problem setup for the optimised square plate.The vertical loads have magnitude P and the corners of the plate have restrained displacements $\boldsymbol{u} = 0$. The geometry has 9×9 control points, of which the grey ones are degrees of freedom for the shape optimisation and the black ones are fixed.

(b) Optimised shape of the square plate. The geometry is coloured by its z-coordinate and the control net is depicted by black lines.

Figure 8.5: Shape optimisation of a square plate with restrained corners and subject to out-of-plane point loads.

8.7 Summary

This chapter introduces open-source software for structural stability analysis using isogeometric membranes and Kirchhoff–Love shell elements. The software is developed as modules within the Geometry + Simulation modules library (G+Smo); this C++ library with a Python interface contains various routines for geometric processing and the analysis of systems of equations using isogeometric analysis. The novel modules in this chapter contain new methods from recent publications of the authors, including hyperelastic material models for shells and membranes [585, 587], goal-oriented error estimators for Kirchhoff–Love shells [586], an adaptive parallel arc-length method [584] and various unstructured spline constructions for isogeometric analysis [589]. The modules provide off-the-shelf solvers for isogeometric analysis, aimed at future extension with novel material models, fast solvers for structural analysis, and new unstructured spline constructions. Through brief code snippets, examples, and brief class details, this chapter aims to inspire the reader for the design of similar software packages and to stimulate the reader to use the presented modules for verification purposes or to use them in new developments. Overall, the experiments presented in this chapter demonstrate that isogeometic analysis is able to provide efficient modelling tools and robust solutions to shell problems and beyond.

8.A Installation Instructions

The software in this chapter is part of the Geometry + Simulation Modules (`G+Smo`). Therefore, the installation instructions for the software presented in this chapter are derived from `G+Smo` . The following instructions are for MacOS and Linux systems. For Windows, the reader is referred to the `G+Smo` documentation.

8.A.1 Downloading G+Smo

The examples provided in this chapter and in other chapters of the dissertation are reproduced using the versions as listed in table 8.2. To obtain the repository at this state, one can either clone the repository from GitHub, or download a permanently stored image from the Software Heritage.

Table 8.2: Versions for `G+Smo` and the submodules `gsKLShell`, `gsStructuralAnalysis`, `gsUnstructuredSplines` used in this dissertation.

Module	**Version**	**GitHub Repository** (preceded by `https://github.com/gismo/`)
`gismo`	`v23.12.0`	`gismo/releases/tag/v23.12.0`
`gsKLShell`	`v23.12`	`gsKLShell/releases/tag/v23.12`
`gsStructuralAnalysis`	`v23.12`	`gsStructuralAnalysis/releases/tag/v23.12`
`gsUnstructuredSplines`	`v23.12`	`gsUnstructuredSplines/releases/tag/v23.12`

Downloading G+Smo from GitHub

To obtain the `G+Smo` repository from GitHub, one can use the following command

```
git clone --branch v23.12.0 https://github.com/gismo/gismo
```

Within version `v23.12.0` of `G+Smo` , the submodules `gsKLShell`, `gsStructuralAnalysis` and `gsUnstructuredSplines` are automatically checkout in the versions listed in table 8.2. If this is not the case, they can be manually downloaded into the `optional/` directory of `G+Smo` :

```
# Navigate to the optional/ folder in gismo/
cd optional

# Clone the gsKLShell, gsStructuralAnalysis and gsUnstructuredSplines modules
git clone --branch v23.12 https://github.com/gismo/gsKLShell
git clone --branch v23.12 https://github.com/gismo/gsStructuralAnalysis
git clone --branch v23.12 https://github.com/gismo/gsUnstructuredSplines
```

Downloading G+Smo from Software Heritage

A static copy of the software from this thesis is stored in the Software Heritage[1] repository. This copy is not linked to the G+Smo GitHub repositories, hence one cannot access newer versions from this copy. To obtain G+Smo and the relevant submodules from the Software Heritage, one can use the following identifier:

```
swh:1:dir:dc870b8f2f07833a664c01cc87996e8cbbfb51a7;
origin=https://github.com/hverhelst/2023-12-gismo_PhD_thesis_HMVerhelst;
visit=swh:1:snp:7ca6040eb2bfa24f5bc9c240f0a419109fb6a301;
anchor=swh:1:rev:3ad3bdf9c15d791ed590bfc66cd16a6d3cf299e5
```

This identifier points to the GitHub repository 2023-12-gismo_PhD_thesis_HMVerhelst, storing the full code belonging to this dissertation.

8.A.2 Installing G+Smo

Installation of G+Smo is done through the following commands[2]:

```
# Navigate to the directory of gismo
cd <path/to/gismo>

# Make a build folder and go inside
mkdir build
cd build

# Initialize the build with the correct CMake flags
cmake . -DGISMO_WITH_OPENMP=ON
        -DGISMO_WITH_MPI=ON
        -DGISMO_SUBMODULES=”gsSpectra;gsKLShell;gsStructuralAnalysis;gsUnstructuredSplines”

# Build gismo
make gsKLShell-examples gsStructuralAnalysis-all gsUnstructuredSplines-examples
```

Optionally, the Python bindings can be built from source using Pybind11.

```
# Set the path to the Pybind11 directory
cmake .
-DGISMO_WITH_PYBIND11=ON # requires pybind11 installed.

# Build pygismo
make pygismo
```

The pygismo package installed from source are loaded into a Python script by using

[1] The Software Heritage is a project designed and built to preserve software. It can be accessed via www.softwareheritage.org

[2] The procedure is tested on a notebook with Ubuntu 22.04 LTS, an Intel i7 13-700H CPU and 32GB RAM. For other installation instructions, the reader is referred to the README.md file in the G+Smo repository.

```
# Add the path to gismo/build/lib
import sys
gismo_path="<path/to/gismo>/build/lib"
sys.path.append(gismo_path)

# Import pygismo
import pygismo
```

8.A.3 Verifying the Installation

The installation of G+Smo can be verified using the unit-tests provided within the library. The unit-tests in G+Smo can be compiled using the following commands:

```
  # Set the path to the Pybind11 directory
  cmake . -DGISMO_BUILD_UNITTESTS=ON

  # Build pygismo
  make unittests
```

Consequently, the unit-tests related to the modules presented in this chapter can be run as follows:

```
# From the build folder
./bin/unittests gsThinShellAssembler_test gsMaterialMatrix_test
```

8.B Result Reproduction

Most of the examples provided in this chapter are based on results from earlier chapters. For the reproducibility of the examples corresponding to earlier chapters the reader is therefore directed to the respective chapter for reproduction instructions. For the newly provided examples in this chapter, table 8.3 provides reproduction instructions.

Table 8.3: File name and run arguments required for the reproducibility of the figures in this chapter. Arguments with a single dash (-) require an argument.

Figure	**Run File**		
	Arg.	**Description**	**Values**
Figure 8.4	static_shell_XML		
	-i	Input file	gsStructuralAnalysis/filedata/pde/beam.xml
	-e	Number of degree elevation steps	1: $p = 2$
	-r	Number of uniform refinement steps	2: 4×4 elements per patch
Figure 8.5	gsKLShell/python_examples/example_shell3D_opt.py		
	Run this file with Python 3.X		

8.C Class overview per module

Table 8.4: Overview of the main classes in the gsKLShell module. The template argument d denotes the parametric dimension, and the argument T denotes the type for real numbers, e.g., double or long double. The symbol └ denotes inheritance.

Class	Description
gsMaterialMatrixBase	Base class providing the constitutive implementation for the gsThinShellAssembler.
└ gsMaterialMatrixBaseDim	Dimension-dependent base class inheriting from gsMaterialMatrixBase and providing all necessary basis computations for the constitutive relations.
└ gsMaterialMatrixLinear	Implementation of the Saint-Venant Kirchoff constitutive relation.
└ gsMaterialMatrixComposite	Uses the Saint-Venant Kirchoff constitutive relation for laminates with different orientations, described in [238]
└ gsMaterialMatrixNonlinear	Implementation of the Neo-Hookean, Mooney-Rivlin, and Ogden material models, following the works of [320] and [587]. The class is templated over the material model mat and over the implementation: (i) $\mathbf{S}$ and $\boldsymbol{\mathcal{C}}$ are given analytically; (ii) Ψ and its derivatives to $\mathbf{C}$ are given analytically; or (iii) Ψ and its derivatives to λ are given analytically.
└ gsMaterialMatrixTFT	A tension field theory-based material matrix for wrinkling modelling. This material matrix uses another material matrix as an input and applies the modification scheme from [401] for linear elastic materials or the scheme from [585] for hyperelastic materials to implicitly model wrinkling.
gsThinShellAssembler	The actual assembler for the Kirchhoff–Love shell model. The assembler takes a geometry (gsFunctionSet), a basis (gsMultiBasis or gsMappedBasis), a set of boundary conditions (gsBoundaryConditions), a body force (gsFunctionSet), and a material matrix (gsMaterialMatrixBase) as input. Optionally, point loads, follower pressures, or an elastic foundation can be provided. The class provides the linear stiffness matrix K, the external force vector $\boldsymbol{P}$, the residual vector $\boldsymbol{R}(\boldsymbol{u})$, the Jacobian $K(\boldsymbol{u})$, and the mass matrix M.
gsThinShellAssemblerDWR	The same as gsThinShellAssembler, but takes a primal and a dual space (both gsMultiBasis) and a goal functional. The class outputs operators for the Dual-Weighted Residual (DWR) method.

Table 8.5: Most important solvers in the `gsStructuralAnalysis` module. The symbol └ denotes inheritance.

Class	Description
`gsStaticBase`	Provides a base class for static analysis. The class provides common functions for the derived classes, such as functions accessing solutions or stability computations.
└ `gsStaticDR`	Class that implements the Dynamic Relaxation (DR) method. The class takes the residual vector $\boldsymbol{R}(\boldsymbol{u})$ and the (lumped) mass matrix M.
└ `gsStaticNewton`	Class that implements Newton-Raphson (NR) iterations for solving nonlinear static problems. The class takes the force vector $\boldsymbol{P}$, the residual vector $\boldsymbol{R}(\boldsymbol{u})$, the stiffness matrix K, and the Jacobian operator $K(\boldsymbol{u})$ or the MIP Jacobian $K(\boldsymbol{u},\Delta\boldsymbol{u})$.
└ `gsStaticComposite`	Class that implements a sequential static solver. Multiple static solvers with different tolerances are provided, and this class sequentially calls the static solvers and provides the updates.
`gsEigenProblemBase`	Base class for eigenvalue problems for structural analysis. The class provides the solution routines, whereas the derived classes provide the set-up of the eigenvalue problem. The eigenproblems are solved using the generalised eigensolver of `Eigen` [217] or using `Spectra` [452] for large problems, optionally with shifts.
└ `gsModalSolver`	A solver that performs modal analysis, i.e., that solves the eigenvalue problem equation (8.12), hence requires the mass and linear stiffness matrices M and K, respectively.
└ `gsBucklingSolver`	A solver that performs linear buckling analysis, i.e., that solves the eigenvalue problem equation (8.13). The solver can be enabled by providing the matrices K and $K(\boldsymbol{u}_0)$ or by providing the matrix K, the load vector $\boldsymbol{P}$, and the operator for the Jacobian $K(\boldsymbol{u})$.
`gsALMBase`	Base class for Arc-Length Methods (ALMs). This class also includes the extended ALM [633], used for computing singular points. All solvers require the external load vector $\boldsymbol{P}$, the arc-length residual $\boldsymbol{R}(\boldsymbol{u},\lambda)$, and the Jacobian matrix $K(\boldsymbol{u})$.
└ `gsALMLoadControl`	A class performing load-controlled simulations.
└ `gsALMRiks`	An implementation of Riks' arc-length method, see [470].
└ `gsALMCrisfield`	A class using Crisfield's method, see [124] and equation (8.14).

Table 8.6: Most important classes in the `gsUnstructuredSplines` module. The symbol └ denotes inheritance.

Class	Description
`gsDPatchBase`	The base class for smoothing methods that rely on local refinements. The class contains basic routines for interface and vertex smoothing and requires specifications to handle extraordinary vertices for derived classes.
└ `gsSmoothInterfaces`	The most basic class for smoothing interfaces. It provides C^1 smoothing on interfaces and ordinary vertices but does not enforce exact or almost C^1 smoothing at extraordinary vertices. The class can be initialised for tensor B-spline bases and geometries or THB-spline geometries with degree $p \geq 2$ and regularity $0 < r \leq p-1$
└ `gsDPatch`	Class implementing the degenerate patch (D-Patch) approach from [569]. It relies on local THB refinement around the extraordinary vertices, building a space with reduced local regularity and refinement. The class can be initialised for tensor B-spline bases and geometries with degree $p \geq 2$ and regularity $0 < r \leq p-1$
└ `gsAlmostC1`	Class implementing the almost-C^1 method from [534]. As the D-Patch, it relies on local THB refinement. The class can be initialised for tensor B-spline bases and geometries or THB-spline geometries with degree $p = 2$ and regularity $r = 1$
`gsC1SurfSpline`	Class implementing the analysis-suitable G^1 (AS-G^1) spline construction from [181, 182]. This class relies on different local bases for the vertex, interface, and interior spaces and requires a degree of $p \geq 3$ and a regularity of $r \leq p-2$
`gsApproxC1Spline`	Class implementing the approximate C^1 spline construction from [618, 619]. Its implementation is similar to the AS-G^1 construction, but it can be constructed for large regularity, namely, it requires a degree of $p \geq 3$ and a regularity of $r \leq p-1$
`gsMPBESSpline`	Class implementing the multi-patch B-splines with enhanced smoothness (MPBES) construction from [74]. The construction requires $p \geq 2$ and $r \leq p-1$. Note that this construction is C^{p-1} everywhere, but only C^0 in the vicinity of the EV.

9

Conclusions

This dissertation provides a broad range of mathematical techniques for the numerical modelling of structural instabilities in thin shells or membranes, with a special focus on membrane wrinkling. The primary goal of this dissertation, see section 1.1, is restated here for convenience:

> *Develop robust and efficient models for structural stability analysis using isogeometric analysis, with a focus on wrinkling.*

As stated in the introduction of this dissertation (see chapter 1), different aspects of numerical instability analysis are covered, related to the sub-goals mentioned in the introduction. Before concluding on the main goal, conclusions are drawn on the sub-goals:

(i) *Extend the isogeometric Kirchhoff-Love shell formulation with constitutive models required for implicit and explicit wrinkling analysis.*
Motivated by wrinkling of membranes under large strains, e.g., the tension wrinkling case investigated by Cerda *et al.* [92] (see figure 1.1a), chapters 3 and 4 present novel techniques for embedding hyperelastic material models into the isogeometric analysis framework. On the one hand, an extension of the hyperelastic material formulations for isogeometric Kirchhoff-Love shells from Kiendl *et al.* [320] to material formulations defined in terms of principal stretches instead of invariants is given in chapter 3. Among the results in this chapter, a new investigation of the snapping behaviour of a truncated cone inspired by Başar & Itskov [23] is given, and first results for isogeometric modelling of the tension wrinkling case inspired by Cerda *et al.* [92] are given.
Contrary to the modelling of membrane wrinkling with isogeometric shell elements, membrane elements cannot provide accurate wrinkling patterns due to the lack of bending stiffness (see section 2.5). However, using a modification scheme using tension field theory, chapter 4 provides an implicit wrinkling model for isogeometric membrane elements. That is, wrinkling patterns are modelled implicitly, such that the mid-surface of a wrinkled membrane can be modelled accurately without the need for fine mesh sizes to capture fine wrinkles. This model is an extension of the

models from Nakashino & Natori, Nakashino *et al.* [401, 402] and Liu *et al.* [356] to hyperelastic materials.

(ii) *Develop methods for adaptive meshing and adaptive load stepping tailored to stability analysis.*
Several techniques can be employed to improve the efficiency of isogeometric methods for solving PDEs. In chapters 5 and 6, two concepts of adaptivity are provided, respectively mesh adaptivity and adaptive load-stepping. Chapter 5 regards adaptive meshing, using a novel error estimator based on the Dual-Weighted Residual method to steer adaptive meshing with suitably graded THB splines. Since the DWR can be used with intuitive goal functions, mesh adaptivity based on engineering indicators (e.g., the critical buckling load or the principal stress at a point) can be realised. In this way, mesh degrees of freedom can be reduced based on a goal function of interest from an application perspective.
Besides spatial adaptivity, chapter 6 provides a novel idea for a parallel and adaptive arc-length method, the APALM. ALMs are commonly used methods for the computation of snapping or bifurcation instabilities in structural analysis. These methods are inherently serial, potentially creating a bottleneck for large computing systems. By starting with a coarse estimate of the equilibrium path, the APALM adaptively refines in parallel, improving speed and accuracy for structural stability problems, as demonstrated in chapter 6.

(iii) *Provide a qualitative and quantitative comparison of unstructured spline constructions for IGA.*
Besides the continuity requirements of the isogeometric Kirchhoff-Love shells, buckling problems benefit from high continuity in terms of accuracy [414]. However, to model complex geometries, either trimming approaches, variational coupling strategies, or unstructured spline constructions can be used to provide continuous spline constructions over multi-patches. Chapter 7 provides a qualitative and quantitative comparison of unstructured spline constructions for smooth multi-patch analysis. This comparison shows the benefits of using variational coupling methods, e.g., in terms of stress reconstruction or spectral problems, and it shows the benefits of different unstructured spline constructions under different conditions.

(iv) *Develop an efficient, robust and future proof open-source computational tool for isogeometric wrinkling analysis.*
Aiming for continuity and applicability of the methods developed in this dissertation, the last chapter of the core of this dissertation (chapter 8) provides an overview of the software developed along with this dissertation. The open-source software is part of the Geometry + Simulation modules [294], hence forward-compatible with new developments in assembly routines and linear algebra, geometric utilities, and interfaces to other software packages. The novel modules developed in this dissertation provide compatibility with future developments, as follows: First, the Kirchhoff-Love shell assembler is decoupled from the constitutive relations, enhancing the future development of new constitutive models. Secondly, the structural analysis module utilises black-box implementations of the Jacobian matrix and residual vector, such that they can be used with other elements and libraries. Thirdly, the un-

structured spline module builds on a generic framework for mapped bases and geometries, allowing user-defined unstructured spline constructions.

Based on the sub-goals mentioned above, the following conclusions about the main goal of this dissertation can be drawn: Based on the results from goal (i), isogeometric structural analysis is enabled explicitly and implicitly using shell and membrane elements, applicable to membrane wrinkling and other snapping problems. Efficient modelling is obtained by employing adaptive methods in space and load stepping (see goal (ii)), using the DWR method as an error estimator. An effort for robust methods is made through the use of unstructured splines for complex geometries (see goal (iii)), which do not require tuning parameters or specialised quadrature rules or solvers. Furthermore, by providing an open-source implementation based on an established software library, robustness is enhanced, and forward compatibility is provided by design.

10

Outlook

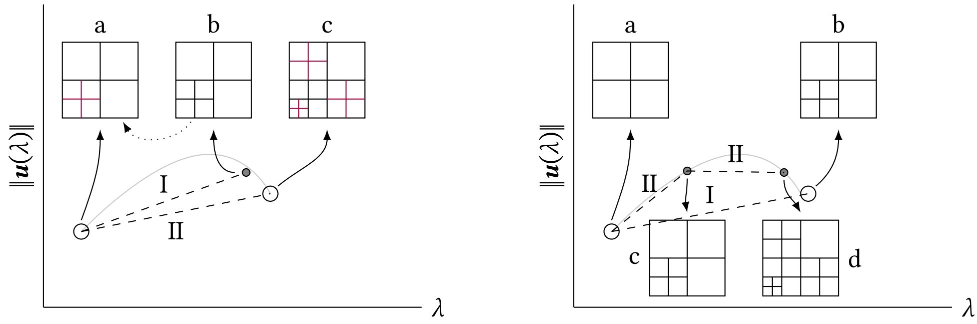
In the following, a number of concepts and ideas for future research directions are provided. The concepts are inspired by the methods developed in this dissertation, and using chapter 8 the software provided along with this dissertation can used as a starting point.

10.1 Goal-Oriented *r*-Adaptivity

Instead of adaptivity based on refinement of the underlying spaces, as is done in chapter 5, adaptive simulations can be performed using *r*-adaptivity on spline parametrisations [285, 286, 645] or the equivalent method of *moving meshes* within the FEA-context. Here, the mesh parametrisation is adjusted so that degrees of freedom are not added, but moved to regions of the domain where the error contributions are high. As a result, the underlying spline spaces are based on tensor product-definitions, hence benefiting from fast formation techniques, e.g., weighted quadrature [83, 489] or sum factorisation [12]. Since *r*-adaptivity does not increase the approximation space, the technique is limited in the reduction of the errors, due to the fact that no degrees of freedom are added and mesh bijectivity needs to be maintained. Therefore, goal-oriented *r*-adaptivity should be seen as a parametrisation-correction method for shell simulations, rather than a method to capture multi-scale effects. Depending on the underlying spaces, goal-oriented *r*-adaptivity can also be investigated for unstructured spline constructions, based on chapter 7 of this dissertation.

10.2 A Spatially Adaptive Parallel Arc-Length Method

The adaptive meshing framework presented in chapter 5 is applied on wrinkling and snapping problems. In these examples, however, serial arc-length methods have been used for continuation. Using the methods developed in chapters 5 and 6 fully adaptive methods for post-buckling analysis, hence including mesh adaptivity and adaptive arc-length method, can be developed. Figure 10.1 provides two ideas to do so: via an adaptive meshing scheme within the APALM where arc lengths are not refined but corrected with adaptive meshes; or, by adaptive meshing and arc length refinement simultaneously. In both cases, the arc



(a) An APALM segment subject to iterative mesh refinement. The segment is first started from mesh a, resulting in step I. Using an adaptive meshing algorithm, a new mesh can be constructed (b) and the computation can be restarted from the initial start point with a modified mesh a (red lines), resulting in interval II. On the end-point of interval II, the mesh can be modified again, resulting in mesh c, and the procedure can be restarted until a tolerance is reached.

(b) An APALM segment subject to arc-length and mesh adaptivity. Segment I is computed first, starting from mesh a and resulting in mesh b after refinement. In a next APALM iteration, the arc length is refined and two new intervals are created, with solutions on meshes c and d. The distance between the solution on mesh d and b can be computed using a continuous distance function.

Figure 10.1: Two proposals for an extension of the APALM with mesh adaptivity.

length distance needs to be computed as a continuous inner product, instead of a discrete one, such that the distance between solutions on different meshes can be computed.

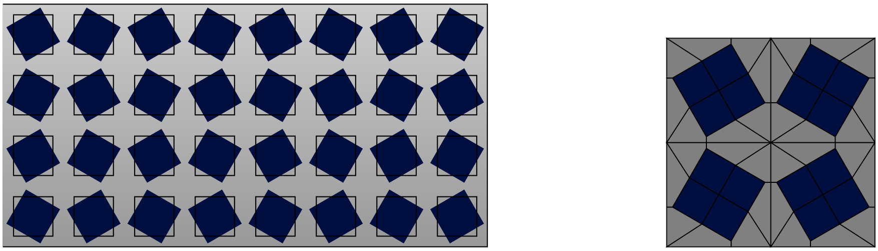
10.3 Variationally Consistent Tension Fields

As concluded in chapter 4, the modification method for hyperelastic membranes using the tension field theory provide good convergence behaviour within Newton-Raphson iterations for a fixed tension field within the iterations. However, when the tension field is subject to significant changes, the Newton-Raphson convergence of the method is negatively affected. To resolve this issue, it is proposes to consider the tension field as an unknown variable of the method, and to solve it using a level-set method or similar. It should be noted that the tension field consists of three categories ({Taut, Slack, Wrinkled}), which need to be captured by the level-set function or similar, for which inspiration can be found in multi-material topology optimisation, mixing problems or multi-phase problems.

10.4 Stiffened Membranes

Inspired by the work of de Bode [51] and Lavaerts [331] and the application of offshore solar panels, the methods provided in this dissertation can be employed to study the behaviour of membranes with embedded solar panels. As shown in the works of de Bode [51] and Lavaerts [331], the orientation of stiff zones and the application of holes in membranes can greatly influence the wrinkling behaviour of offshore solar platforms. By using unstructured spline constructions (chapter 7) in combination with tension field theory-based membrane models (chapter 4), a robust approach can be developed for design, analysis and optimisation of membranes with embedded solar panels. For example, the membrane in figure 10.2 represents an auxetic lay-up, inspired by the work of Bonfanti & Bhaskar [56].

For the development of membranes with embedded cables, for example used for offshore solar platforms, smart wearables or sails, the methods provided in this dissertation can be extended using embedding methods. By embedding cable or beam elements [35, 248, 602, 612], into the Kirchhoff-Love shell or the isogeometric membrane element provided in this dissertation, a novel isogeometric approach for reinforced membranes can be developed, for example benefiting from intuitive and efficient parametrisation of the embedded curves for shape optimisation.



(a) A rectangular sheet with an auxetic lay-up of solar panels (blue squares) compared to a Cartesian lay-up (outlined squares).

(b) A detail of a multi-patch segmentation of a 2 × 2 auxetic membrane element.

Figure 10.2: A sketch of an auxetic membrane structure (a) with a detailed element (b).

10

Bibliography

[1] C. Adam, S. Bouabdallah, M. Zarroug & H. Maitournam (2015). “Improved Numerical Integration for Locking Treatment in Isogeometric Structural Elements. Part II: Plates and Shells”. *Comput. Methods Appl. Mech. Eng.*, **284**:106–137.

[2] A. L. Adler, M. M. Mikulas & J. M. Hedgepeth (2000). “Static and dynamic analysis of partially wrinkled membrane structures”. *41st Struct. Struct. Dyn. Mater. Conf. Exhib.*, **4**:158–171.

[3] T. Akita, K. Nakashino, M. C. Natori & K. C. Park (2007). “A Simple Computer Implementation of Membrane Wrinkle Behaviour via a Projection Technique”. *Int. J. Numer. Meth. Eng.*, **71**(10):1231–1259.

[4] M. Alaydin, M. Behzadinasab & Y. Bazilevs (2022). “Isogeometric Analysis of Multilayer Composite Shell Structures: Plasticity, Damage, Delamination and Impact Modeling”. *Int. J. Solids Struct.*, **252**:111 782.

[5] M. Alaydin, D. Benson & Y. Bazilevs (2021). “An updated Lagrangian framework for Isogeometric Kirchhoff–Love thin-shell analysis”. *Comput. Methods Appl. Mech. Eng.*, **384**:113 977.

[6] R. Alberini, A. Spagnoli & M. Terzano (2021). “Numerical Modelling of Wrinkled Hyperelastic Membranes with Topologically Complex Internal Boundary Conditions”. *Int. J. Mech. Sci.*, **212**:106 816.

[7] V. Alic & K. Persson (2016). “Form Finding with Dynamic Relaxation and Isogeometric Membrane Elements”. *Comput. Methods Appl. Mech. Eng.*, **300**:734–747.

[8] M. Ambati, J. Kiendl & L. De Lorenzis (2018). “Isogeometric Kirchhoff–Love Shell Formulation for Elasto-Plasticity”. *Comput. Methods Appl. Mech. Eng.*, **340**:320–339.

[9] R. Anderson, J. Andrej, A. Barker, J. Bramwell, J.-S. Camier, J. Cerveny, V. Dobrev, Y. Dudouit, A. Fisher, T. Kolev, W. Pazner, M. Stowell, V. Tomov, I. Akkerman, J. Dahm, D. Medina & S. Zampini (2021). “MFEM: A Modular Finite Element Methods Library”. *Comput. Struct.*, **81**:42–74.

[10] C. Androulidakis, E. N. Koukaras, M. G. Pastore Carbone, M. Hadjinicolaou & C. Galiotis (2017). “Wrinkling Formation in Simply-Supported Graphenes under Tension and Compression Loadings”. *Nanoscale*, **9**(46):18 180–18 188.

[11] ANSYS Manual (2019). *ANSYS®Academic Research Mechanical, Release 19.1, Help System, SHELL181.* ANSYS, Inc., r19.1 ed.

[12] P. Antolin, A. Buffa, F. Calabrò, M. Martinelli & G. Sangalli (2015). “Efficient Matrix Computation for Tensor-Product Isogeometric Analysis: The Use of Sum Factorization”. *Comput. Methods Appl. Mech. Eng.*, **285**:817–828.

[13] P. Antolin, A. Buffa & L. Coradello (2020). “A hierarchical approach to the a posteriori error estimation of isogeometric Kirchhoff plates and Kirchhoff–Love shells”. *Comput. Methods Appl. Mech. Eng.*, **363**:112 919.

[14] A. Apostolatos, K.-U. Bletzinger & R. Wüchner (2019). “Weak Imposition of Constraints for Structural Membranes in Transient Geometrically Nonlinear Isogeometric Analysis on Multipatch Surfaces”. *Comput. Methods Appl. Mech. Eng.*, **350**:938–994.

[15] J. Arf, M. Reichle, S. Klinkel & B. Simeon (2023). “Scaled Boundary Isogeometric Analysis with C^1 Coupling for Kirchhoff Plate Theory”. *Comput. Methods Appl. Mech. Eng.*, **415**:116 198.

[16] J. H. Argyris & D. W. Scharpf (1968). “The SHEBA Family of Shell Elements for the Matrix Displacement Method”. *The Aeronautical Journal*, **72**(694):873–883.

[17] D. A. Aruliah, L. Van Veen & A. Dubitski (2016). "Algorithm 956: PAMPAC, a parallel adaptive method for pseudo-arclength continuation". *ACM Trans. Math. Softw.*, **42**(1).

[18] A. Atai & D. J. Steigmann (2012). "Modeling and Simulation of Sutured Biomembranes". *Mech. Res. Commun.*, **46**:34–40.

[19] A. Atai & D. J. Steigmann (2014). "Numerical Analysis of Wrinkled, Anisotropic, Nonlinearly Elastic Membranes". *Mech. Res. Commun.*, **57**:1–5.

[20] B. Audoly (2011). "Localized Buckling of a Floating Elastica". *Phys. Rev. E*, **84**(1):011 605.

[21] L. Azrar, B. Cochelin, N. Damil & M. Potier-Ferry (1993). "An Asymptotic-Numerical Method to Compute the Postbuckling Behaviour of Elastic Plates and Shells". *Int. J. Numer. Meth. Eng.*, **36**(8):1251–1277.

[22] Y. Başar & Y. Ding (1997). "Shear deformation models for large-strain shell analysis". *Int. J. Solids Struct.*, **34**(14):1687–1708.

[23] Y. Başar & M. Itskov (1998). "Finite element formulation of the Ogden material model with application to ruber-like shells". *Int. J. Numer. Meth. Eng.*

[24] Y. Başar & D. Weichert (2013). *Nonlinear Continuum Mechanics of Solids*. Springer-Verlag Berlin Heidelberg GmbH.

[25] F. Baginski (2005). "On the Design and Analysis of Inflated Membranes: Natural and Pumpkin Shaped Balloons". *SIAM J. Appl. Math.*, **65**(3):838–857.

[26] F. Baginski, K. A. Brakke & W. W. Schur (2006). "Stability of Cyclically Symmetric Strained Pumpkin Balloons and the Formation of Undesired Equilibria". *J. Aircraft*, **43**(5):1414–1423.

[27] A. Balu, S. Nallagonda, F. Xu, A. Krishnamurthy, M.-C. Hsu & S. Sarkar (2019). "A Deep Learning Framework for Design and Analysis of Surgical Bioprosthetic Heart Valves". *Sci. Rep.*, **9**(1):18 560.

[28] K. Bandara & F. Cirak (2018). "Isogeometric shape optimisation of shell structures using multiresolution subdivision surfaces". *Comput. Aided Des.*, **95**:62–71.

[29] W. Bangerth & R. Rannacher (2003). *Adaptive Finite Element Methods for Differential Equations*. Birkhäuser Basel, Basel, 1st ed. ISBN 978-3-7643-7009-1.

[30] P. J. Barendrecht (2013). "Isogeometric analysis for subdivision surfaces". Ph.D. thesis, Eindhoven University of Technology.

[31] P. J. Barendrecht, M. Bartoň & J. Kosinka (2018). "Efficient quadrature rules for subdivision surfaces in isogeometric analysis". *Comput. Methods Appl. Mech. Eng.*, **340**:1–23.

[32] M. R. Barnes (1988). "Form-Finding and Analysis of Prestressed Nets and Membranes". *Comput. Struct.*, **30**(3):685–695.

[33] M. R. Barnes (1999). "Form Finding and Analysis of Tension Structures by Dynamic Relaxation". *Int. J. Space Struct.*, **14**(2):89–104.

[34] R. Barsotti, S. S. Ligarò & G. F. Royer-Carfagni (2001). "The Web Bridge". *Int. J. Solids Struct.*, **38**(48):8831–8850.

[35] A. Bauer, M. Breitenberger, B. Philipp, R. Wüchner & K.-U. Bletzinger (2017). "Embedded Structural Entities in NURBS-based Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **325**:198–218.

[36] Y. Bazilevs, V. M. Calo, J. A. Cottrell, J. A. Evans, T. J. R. Hughes, S. Lipton, M. A. Scott & T. W. Sederberg (2010). "Isogeometric Analysis Using T-splines". *Comput. Methods Appl. Mech. Eng.*, **199**(5-8):229–263.

[37] Y. Bazilevs, M. C. Hsu, J. Kiendl, R. Wüchner & K. U. Bletzinger (2011). "3D simulation of wind turbine rotors at full scale. Part II: Fluid-structure interaction modeling with composite blades". *Int. J. Numer. Meth. Fluids*, **65**(1-3):236–253.

[38] Y. Bazilevs, K. Takizawa & T. E. Tezduyar (2013). *Computational Fluid-Structure Interaction.* John Wiley & Sons. ISBN 9781118483565.

[39] R. Becker & R. Rannacher (2001). "An optimal control approach to a posteriori error estimation in finite element methods". *Acta Numer.*, **10**:1–102.

[40] M. Behzadinasab, M. Alaydin, N. Trask & Y. Bazilevs (2022). "A General-Purpose, Inelastic, Rotation-Free Kirchhoff–Love Shell Formulation for Peridynamics". *Comput. Methods Appl. Mech. Eng.*, **389**:114 422.

[41] L. Beirão da Veiga, A. Buffa, C. Lovadina, M. Martinelli & G. Sangalli (2012). "An Isogeometric Method for the Reissner–Mindlin Plate Bending Problem". *Comput. Methods Appl. Mech. Eng.*, **209–212**:45–53.

[42] P. Bellini & A. Chulya (1987). "An improved automatic incremental algorithm for the efficient solution of nonlinear finite element equations". *Comput. Struct.*, **26**(1-2):99–110.

[43] D. Benson, Y. Bazilevs, M.-C. Hsu & T. Hughes (2011). "A large deformation, rotation-free, isogeometric shell". *Comput. Methods Appl. Mech. Eng.*, **200**(13-16):1367–1378.

[44] D. Benson, Y. Bazilevs, M. Hsu *et al.* (2010). "Isogeometric shell analysis: The Reissner–Mindlin shell". *Comput. Methods Appl. Mech. Eng.*, **199**(5-8):276–289.

[45] D. Benson, S. Hartmann, Y. Bazilevs, M.-C. Hsu & T. Hughes (2013). "Blended isogeometric shells". *Comput. Methods Appl. Mech. Eng.*, **255**:133–146.

[46] A. Benvenuti, G. Loli, G. Sangalli & T. Takacs (2023). "Isogeometric Multi-Patch C^1-Mortar Coupling for the Biharmonic Equation". *arxiv:2303.07255 [cs, math].*

[47] J. Benzaken, J. A. Evans, S. F. McCormick & R. Tamstorf (2021). "Nitsche's Method for Linear Kirchhoff–Love Shells: Formulation, Error Analysis, and Verification". *Comput. Methods Appl. Mech. Eng.*, **374**.

[48] C. Bernardi, Y. Maday & A. T. Patera (1993). "Domain Decomposition by the Mortar Element Method". In H. G. Kaper, M. Garbey & G. W. Pieper, eds., *Asymptotic and Numerical Methods for Partial Differential Equations with Critical Parameters*, NATO ASI Series, 269–286. Springer Netherlands, Dordrecht. ISBN 978-94-011-1810-1.

[49] M. Bischoff, K.-U. Bletzinger, W. A. Wall & E. Ramm (2004). "Models and Finite Elements for tws". In *Encyclopedia of cm*, chap. 3. John Wiley & Sons, Ltd. ISBN 978-0-470-09135-7.

[50] J. R. Blandino, J. D. Johnston & U. K. Dharamsi (2002). "Corner wrinkling of a square membrane due to symmetric mechanical loads". *J. Spacecr. Rockets*, **39**(5):717–724.

[51] S. de Bode (2021). "Wrinkling analysis and design of Offshore Flexible Floating Solar Structures". Master's thesis, Technische Universiteit Delft.

[52] W. Böhm (1980). "Inserting New Knots into B-spline Curves". *Comput. Aided Des.*, **12**(4):199–201.

[53] W. Böhm, G. Farin & J. Kahmann (1984). "A Survey of Curve and Surface Methods in CAGD". *Comput. Aided Geom. Des.*, **1**(1):1–60.

[54] W. Böhm & H. Prautzsch (1985). "The Insertion Algorithm". *Comput. Aided Des.*, **17**(2):58–59.

[55] W. Böhm & H. Prautzsch (1994). *Geometric Concepts for Geometric Design.* AK Peters, Ltd. ISBN 1-56881-004-0.

[56] A. Bonfanti & A. Bhaskar (2019). "Elastic Stabilization of Wrinkles in Thin Films by Auxetic Microstructure". *Extreme Mechanics Letters*, **33**:100 556.

[57] A. S. Bonin & K. A. Seffen (2014). "De-wrinkling of pre-tensioned membranes". *Int. J. Solids Struct.*, **51**(19-20):3303–3313.

[58] M. J. Borden, T. J. Hughes, C. M. Landis & C. V. Verhoosel (2014). "A higher-order phase-field model for brittle fracture: Formulation and analysis within the isogeometric analysis framework". *Comput. Methods Appl. Mech. Eng.*, **273**:100–118.

[59] R. Bouclier, T. Elguedj & A. Combescure (2013). "Efficient Isogeometric NURBS-based Solid-Shell Elements: Mixed Formulation and B¯-Method". *Comput. Methods Appl. Mech. Eng.*, **267**:86–110.

[60] R. Bouclier, T. Elguedj & A. Combescure (2015). "An Isogeometric Locking-Free NURBS-based Solid-Shell Element for Geometrically Nonlinear Analysis". *Int. J. Numer. Meth. Eng.*, **101**(10):774–808.

[61] R. Bouclier & J. C. Passieux (2018). "A Nitsche-based Non-Intrusive Coupling Strategy for Global/Local Isogeometric Structural Analysis". *Comput. Methods Appl. Mech. Eng.*, **340**:253–277.

[62] R. Bouclier, J. C. Passieux & M. Salaün (2017). "Development of a New, More Regular, Mortar Method for the Coupling of NURBS Subdomains within a NURBS Patch: Application to a Non-Intrusive Local Enrichment of NURBS Patches". *Comput. Methods Appl. Mech. Eng.*, **316**:123–150.

[63] E. H. Boutyour, H. Zahrouni, M. Potier-Ferry & M. Boudi (2004). "Bifurcation Points and Bifurcated Branches by an Asymptotic Numerical Method and Padé Approximants". *Int. J. Numer. Meth. Eng.*, **60**(12):1987–2012.

[64] R. Bouzidi & Y. Lecieux (2010). "Experimental Analysis on Membrane Wrinkling under Biaxial Load - Comparison with Bifurcation Analysis". *Int. J. Solids Struct.*, **47**(18-19):2459–2475.

[65] F. Box, D. Vella, R. W. Style & J. A. Neufeld (2017). "Indentation of a Floating Elastic Sheet: Geometry versus Applied Tension". *Proc. R. Soc. A. Math. Phys. Eng. Sci.*

[66] C. Bracco, C. Giannelli, D. Großmann, S. Imperatore, D. Mokriš & A. Sestini (2022). "THB-Spline Approximations for Turbine Blade Design with Local B-Spline Approximations". In *SEMA SIMAI Springer Series*, vol. 29, 63–82. Springer, Cham.

[67] C. Bracco, C. Giannelli & R. Vázquez (2018). "Refinement Algorithms for Adaptive Isogeometric Methods with Hierarchical Splines". *Axioms*, **7**(3):43.

[68] F. Brau, P. Damman, H. Diamant & T. A. Witten (2013). "Wrinkle to Fold Transition: Influence of the Substrate Response". *Soft Matter*, **9**(34):8177.

[69] M. Breitenberger, A. Apostolatos, B. Philipp, R. Wüchner & K. U. Bletzinger (2015). "Analysis in Computer Aided Design: Nonlinear Isogeometric B-Rep Analysis of Shell Structures". *Comput. Methods Appl. Mech. Eng.*, **284**:401–457.

[70] B. Brendel & E. Ramm (1982). "Nichtlineare Stabilitätsuntersuchungen Mit Der Methode Der Finiten Elemente". *Ingenieur-Archiv*, **51**(5):337–362.

[71] R. P. Brent (1971). "An Algorithm with Guaranteed Convergence for Finding a Zero of a Function". *The Computer Journal*, **14**(4):422–425.

[72] A. Bressan & E. Sande (2019). "Approximation in FEM, DG and IGA: a theoretical comparison". *Numer. Math.*, **143**(4):923–942.

[73] E. Brivadis, A. Buffa, B. Wohlmuth & L. Wunderlich (2015). "Isogeometric Mortar Methods". *Comput. Methods Appl. Mech. Eng.*, **284**:292–319.

[74] F. Buchegger, B. Jüttler & A. Mantzaflaris (2016). "Adaptively refined multi-patch B-splines with enhanced smoothness". *Appl. Math. Comput.*, **272**:159–172.

[75] N. Büchter, E. Ramm & D. Roehl (1994). "Three-dimensional extension of non-linear shell formulation based on the enhanced assumed strain concept". *Int. J. Numer. Meth. Eng.*, **37**(15):2551–2568.

[76] A. Buffa, O. Chanon & R. Vázquez (2022). “Adaptive Analysis-Aware Defeaturing”. *arxiv:2212.05183 [cs, math].*

[77] A. Buffa, O. Chanon & R. Vázquez (2022). “Analysis-Aware Defeaturing: Problem Setting and a Posteriori Estimation”. *Math. Models Methods Appl. Sci.*, **32**(02):359–402.

[78] A. Buffa, G. Gantner, C. Giannelli, D. Praetorius & R. Vázquez (2022). “Mathematical Foundations of Adaptive Isogeometric Analysis”. *Arch. Comput. Methods. Eng.*, **29**(7):4479–4555.

[79] A. Buffa & C. Giannelli (2016). “Adaptive isogeometric methods with hierarchical splines: Error estimator and convergence”. *Math. Models Methods Appl. Sci.*, **26**(01):1–25.

[80] R. L. Burden, J. D. Faires & A. M. Burden (2015). *Numerical Analysis.* Cengage Learning. ISBN 978-1-305-25366-7.

[81] E. Burman, S. Claus, P. Hansbo, M. G. Larson & A. Massing (2015). “CutFEM: Discretizing Geometry and Partial Differential Equations”. *Int. J. Numer. Meth. Eng.*, **104**(7):472–501.

[82] S. Cai, H. Zhang & W. Zhang (2023). “An Integrated Design Approach for Simultaneous Shape and Topology Optimization of Shell Structures”. *Comput. Methods Appl. Mech. Eng.*, **415**:116 218.

[83] F. Calabrò, G. Sangalli & M. Tani (2017). “Fast Formation of Isogeometric Galerkin Matrices by Weighted Quadrature”. *Comput. Methods Appl. Mech. Eng.*, **316**:606–622.

[84] M. Carraturo, C. Giannelli, A. Reali & R. Vázquez (2019). “Suitably graded THB-spline refinement and coarsening: Towards an adaptive isogeometric analysis of additive manufacturing processes”. *Comput. Methods Appl. Mech. Eng.*, **348**:660–679.

[85] E. Carrera (1994). “A study on arc-length-type methods and their operation failures illustrated by a simple model”. *Comput. Struct.*, **50**(2):217–229.

[86] J. F. Caseiro, R. A. F. Valente, A. Reali, J. Kiendl, F. Auricchio & R. J. Alves de Sousa (2014). “On the Assumed Natural Strain Method to Alleviate Locking in Solid-Shell NURBS-based Finite Elements”. *Comput. Mech.*, **53**(6):1341–1353.

[87] J. Caseiro, R. Valente, A. Reali, J. Kiendl, F. Auricchio & R. Alves de Sousa (2015). “Assumed Natural Strain NURBS-based Solid-Shell Element for the Analysis of Large Deformation Elasto-Plastic Thin-Shell Structures”. *Comput. Methods Appl. Mech. Eng.*, **284**:861–880.

[88] H. Casquero, L. Liu, Y. Zhang *et al.* (2017). “Arbitrary-degree T-splines for isogeometric analysis of fully nonlinear Kirchhoff–Love shells”. *Comput. Aided Des.*, **82**:140–153.

[89] H. Casquero, X. Wei, D. Toshniwal, A. Li, T. J. Hughes, J. Kiendl & Y. J. Zhang (2020). “Seamless Integration of Design and Kirchhoff–Love Shell Analysis Using Analysis-Suitable Unstructured T-splines”. *Comput. Methods Appl. Mech. Eng.*, **360**:112 765.

[90] P. V. Cavallaro, C. J. Hart & A. M. Sadegh (2014). “Mechanics of Air-Inflated Drop-Stitch Fabric Panels Subject to Bending Loads”. In *ASME 2013 International Mechanical Engineering Congress and Exposition.* American Society of Mechanical Engineers Digital Collection.

[91] E. Cerda & L. Mahadevan (2003). “Geometry and Physics of Wrinkling”. *Phys. Rev. Lett.*

[92] E. Cerda, K. Ravi-Chandar & L. Mahadevan (2002). “Wrinkling of an Elastic Sheet under Tension”. *Nature*, **419**(6907):579–580.

[93] G. Chagnon, J. Ohayon, J. L. Martiel & D. Favier (2017). “Hyperelasticity Modeling for Incompressible Passive Biological Tissues”. In *Biomechanics of Living Organs: Hyperelastic Constitutive Laws for Finite Element Modeling*, 3–30. Elsevier Inc. ISBN 9780128040607.

[94] M. Chasapi, L. Mester, B. Simeon & S. Klinkel (2022). "Isogeometric Analysis of 3D Solids in Boundary Representation for Problems in Nonlinear Solid Mechanics and Structural Dynamics". *Int. J. Numer. Meth. Eng.*, **123**(5):1228–1252.

[95] Z. Chen, H.-y. Chen, D. M. Kaufman, A. Research, H.-y. Chen, E. Vouga, D. M. Kaufman, H.-Y. Chen & M. Skouras (2021). "Fine Wrinkling on Coarsely Meshed Thin Shells". *ACM Trans. Graph.*, **40**(5).

[96] S. Chen, Q. Li, Q. Zhang, Y. Qu, H. Ji, R. S. Ruoff & W. Cai (2012). "Thermal Conductivity Measurements of Suspended Graphene with and without Wrinkles by Micro-Raman Mapping". *Nanomaterials*, **23**(36):365 701.

[97] L. Chen, N. Nguyen-Thanh, H. Nguyen-Xuan, T. Rabczuk, S. P. A. Bordas & G. Limbert (2014). "Explicit Finite Deformation Analysis of Isogeometric Membranes". *Comput. Methods Appl. Mech. Eng.*, **277**:104–130.

[98] A. B. Chmielewski (2001). *Overview Of Gossamer Structures.* American Institute of Aeronautics and Astronautics. ISBN 9781600866616.

[99] K.-J. Choi & H.-S. Ko (2002). "Stable but Responsive Cloth". In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, 604–611. Association for Computing Machinery, New York, NY, USA. ISBN 978-1-58113-521-3.

[100] J. Chopin, A. Panaitescu & A. Kudrolli (2018). "Corner singularities and shape of stretched elastic sheets". *Phys. Rev. E*, **98**(4):043 003.

[101] F. Cirak & Q. Long (2011). "Subdivision Shells with Exact Boundary Control and Non-Manifold Geometry". *Int. J. Numer. Meth. Eng.*, **88**(9):897–923.

[102] K. A. Cliffe, E. J. Hall & P. Houston (2010). "Adaptive discontinuous galerkin methods for eigenvalue problems arising in incompressible fluid flows". *SIAM J. Sci. Comput.*, **31**(6):4607–4632.

[103] B. Cochelin (1994). "A Path-Following Technique via an Asymptotic-Numerical Method". *Comput. Struct.*, **53**(5):1181–1192.

[104] B. Cochelin, N. Damil & M. Potier-Ferry (2007). *Méthode Asymptotique Numérique.* Hermes Lavoissier.

[105] B. Cochelin & M. Medale (2013). "Power Series Analysis as a Major Breakthrough to Improve the Efficiency of Asymptotic Numerical Method in the Vicinity of Bifurcations". *J. Comp. Phys.*, **236**:594–607.

[106] E. Cohen, T. Martin, R. M. Kirby, T. Lyche & R. F. Riesenfeld (2010). "Analysis-Aware Modeling: Understanding Quality Considerations in Modeling for Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **199**(5):334–356.

[107] N. Collier, L. Dalcin, D. Pardo & V. M. Calo (2013). "The Cost of Continuity: Performance of Iterative Solvers on Isogeometric Finite Elements". *SIAM J. Sci. Comput.*, **35**(2):A767–A784.

[108] N. Collier, D. Pardo, L. Dalcin, M. Paszynski & V. Calo (2012). "The Cost of Continuity: A Study of the Performance of Isogeometric Finite Elements Using Direct Solvers". *Comput. Methods Appl. Mech. Eng.*, **213–216**:353–361.

[109] A. Collin, G. Sangalli & T. Takacs (2016). "Analysis-Suitable G1 Multi-Patch Parametrizations for C1 Isogeometric Spaces". *Comput. Aided Geom. Des.*, **47**:93–113.

[110] C. D. Coman (2007). "On the applicability of tension field theory to a wrinkling instability problem". *Acta Mech.*, **190**(1-4):57–72.

[111] C. D. Coman & A. P. Bassom (2008). "Wrinkling of pre-stressed annular thin films under azimuthal shearing". *Math. Mech. Solids*, **13**(6):513–531.

[112] C. D. Coman & D. M. Haughton (2006). "Localized wrinkling instabilities in radially stretched annular thin films". *Acta Mech.*, **185**(3-4):179–200.

[113] A. Concha, J. W. McIver, P. Mellado, D. Clarke, O. Tchernyshyov & R. L. Leheny (2007). "Wrinkling of a Bilayer Membrane". *Phys. Rev. E*, **75**(1):016 609.

[114] P. Contri & B. A. Schrefler (1988). "A Geometrically Nonlinear Finite Element Analysis of Wrinkled Membrane Surfaces by a No-Compression Material Model". *Commun. Appl. Numer. Meth.*, **4**(1):5–15.

[115] L. Coradello, P. Antolin, R. Vázquez & A. Buffa (2020). "Adaptive Isogeometric Analysis on Two-Dimensional Trimmed Domains Based on a Hierarchical Approach". *Comput. Methods Appl. Mech. Eng.*, **364**:112 925.

[116] L. Coradello, D. D'Angella, M. Carraturo, J. Kiendl, S. Kollmannsberger, E. Rank & A. Reali (2020). "Hierarchically Refined Isogeometric Analysis of Trimmed Shells". *Comput. Mech.*, **66**(2):431–447.

[117] L. Coradello, ·. Davide D'angella, M. Carraturo, J. Kiendl, S. Kollmannsberger, ·. E. Rank & A. Reali (2020). "Hierarchically refined isogeometric analysis of trimmed shells". *Comput. Mech.*, **66**:431–447.

[118] L. Coradello, J. Kiendl & A. Buffa (2021). "Coupling of Non-Conforming Trimmed Isogeometric Kirchhoff–Love Shells via a Projected Super-Penalty Approach". *Comput. Methods Appl. Mech. Eng.*, **387**:114 187.

[119] L. Coradello, G. Loli & A. Buffa (2021). "A Projected Super-Penalty Method for the C^1-Coupling of Multi-Patch Isogeometric Kirchhoff Plates". *Comput. Mech.*, **67**(4):1133–1153.

[120] J. A. Cottrell, T. J. Hughes & Y. Bazilevs (2009). *Isogeometric Analysis: Toward Integration of CAD and FEA*. Wiley. ISBN 9780470748732.

[121] J. Cottrell, T. Hughes & A. Reali (2007). "Studies of Refinement and Continuity in Isogeometric Structural Analysis". *Comput. Methods Appl. Mech. Eng.*, **196**(41-44):4160–4183.

[122] J. Cottrell, A. Reali, Y. Bazilevs & T. Hughes (2006). "Isogeometric Analysis of Structural Vibrations". *Comput. Methods Appl. Mech. Eng.*, **195**(41-43):5257–5296.

[123] M. G. Cox (1972). "The Numerical Evaluation of B-splines". *IMA J. Appl. Math.*, **10**(2):134–149.

[124] M. M. Crisfield (1981). "A Fast Incremental/Iterative Solution Procedure That Handles "Snap-Through"". In *Computational Methods in Nonlinear Structural and Solid Mechanics*, 55–62. Pergamon. ISBN 978-0-08-027299-3.

[125] M. A. Crisfield (1983). "An arc-length method including line searches and accelerations". *Int. J. Numer. Meth. Eng.*, **19**(9):1269–1289.

[126] M. A. Crisfield (1991). *Non-linear finite element analysis of solids and structures - Volume 2: Advanced topics*. Wiley. ISBN 9780471929567.

[127] M. Crisfield, G. Jelenic, Y. Mi, H.-G. Zhong & Z. Fan (1997). "Some Aspects of the Non-Linear Finite Element Method". *Finite Elem. Anal. Des.*, **27**(1):19–40.

[128] P. Cundall (1976). "Explicit Finite-Difference Methods in Geomechanics". *Proc. 2nd Int. Cof. Num. Meth. Geomech., ASCE, New York*, 132–150.

[129] E. C. Cyr, S. Günther & J. B. Schroder (2019). "Multilevel initialization for layer-parallel deep neural network training". *arXiv preprint arXiv:1912.08974*.

[130] P. Dadvand, R. Rossi & E. Oñate (2010). "An Object-oriented Environment for Developing Finite Element Codes for Multi-disciplinary Applications". *Arch. Comput. Methods. Eng.*, **17**(3):253–297.

[131] L. Dalcin, N. Collier, P. Vignal, A. M. A. Côrtes & V. M. Calo (2016). "PetIGA: A Framework for High-Performance Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **308**:151–181.

[132] N. Damil & M. Potier-Ferry (1990). "A New Method to Compute Perturbed Bifurcations: Application to the Buckling of Imperfect Elastic Structures". *Int. J. Eng. Sci.*, **28**(9):943–957.

[133] N. Damil & M. Potier-Ferry (2006). “A Generalized Continuum Approach to Describe Instability Pattern Formation by a Multiple Scale Analysis”. *Comptes Rendus Mécanique*, **334**(11):674–678.

[134] N. Damil & M. Potier-Ferry (2008). “A Generalized Continuum Approach to Predict Local Buckling Patterns of Thin Structures”. *Eur. J. Comput. Mech.*, **17**(5-7):945–956.

[135] N. Damil & M. Potier-Ferry (2010). “Influence of Local Wrinkling on Membrane Behaviour: A New Approach by the Technique of Slowly Variable Fourier Coefficients”. *J. Mech. Phys. Solids*, **58**(8):1139–1153.

[136] N. Damil, M. Potier-Ferry & H. Hu (2014). “Membrane Wrinkling Revisited from a Multi-Scale Point of View”. *Adv. Model. Simul. Eng. Sci.*, **1**(1):6.

[137] D. D’Angella & A. Reali (2020). “Efficient extraction of hierarchical B-Splines for local refinement and coarsening of Isogeometric Analysis”. *Comput. Methods Appl. Mech. Eng.*, **367**:113 131.

[138] H. Darijani & R. Naghdabadi (2010). “Hyperelastic materials behavior modeling using consistent strain energy density functions”. *Acta Mech.*, **213**(3-4):235–254.

[139] B. Davidovitch, R. D. Schroll, D. Vella, M. Adda-Bedia & E. A. Cerda (2011). “Prototypical Model for Tensional Wrinkling in Thin Sheets”. *Proc. Natl. Acad. Sci.*, **108**(45):18 227–18 232.

[140] C. De Boor (1962). “Bicubic Spline Interpolation”. *J. Math. Phys.*, **41**(1-4):212–218.

[141] C. De Boor (1978). *A Practical Guide to Splines*, vol. 27. springer-verlag New York.

[142] R. de Borst, M. A. Crisfield, J. J. Remmers & C. V. Verhoosel (2012). *Non-Linear Finite Element Analysis of Solids and Structures: Second Edition*. John Wiley & Sons. ISBN 978-0-470-66644-9.

[143] L. De Lorenzis, P. Wriggers & T. J. Hughes (2014). “Isogeometric contact: a review”. *GAMM-Mitteilungen*, **37**(1):85–123.

[144] F. de Prenter, C. V. Verhoosel, E. H. van Brummelen, J. A. Evans, C. Messe, J. Benzaken & K. Maute (2020). “Multigrid Solvers for Immersed Finite Element Methods and Immersed Isogeometric Analysis”. *Comput. Mech.*, **65**(3):807–838.

[145] F. de Prenter, C. Verhoosel, H. van Brummelen, M. Larson & S. Badia (2022). “Stability and Conditioning of Immersed Finite Element Methods: Analysis and Remedies”. *arxiv:arXiv:2208.08538*.

[146] R. De Rooij & M. M. Abdalla (2015). “A Finite Element Interior-Point Implementation of Tension Field Theory”. *Comput. Struct.*, **151**:30–41.

[147] L. Dedè & H. A. Santos (2012). “B-spline goal-oriented error estimators for geometrically nonlinear rods”. *Comput. Mech.*, **49**(1):35–52.

[148] S. Deng & V. Berry (2016). “Wrinkled, Rippled and Crumpled Graphene: An Overview of Formation Mechanism, Electronic Properties, and Applications”. *Mater. Today*, **19**(4):197–212.

[149] J. Deng, F. Chen, X. Li, C. Hu, W. Tong, Z. Yang & Y. Feng (2008). “Polynomial Splines over Hierarchical T-meshes”. *Graph. Models*, **70**(4):76–86.

[150] U. K. Dharamsi, D. M. Evanchik & J. R. Blandino (2002). “Comparing photogrammetry with a conventional displacement measurement technique on a 0.5m square Kapton® membrane”. *Collect. Tech. Pap. - AIAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn. Mater. Conf.*, **1**(April):463–472.

[151] Delft High Performance Computing Centre (DHPC) (2022). “DelftBlue Supercomputer (Phase 1)”. `https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1`.

[152] A. Diaby, A. Le van & C. Wielgosz (2006). “Buckling and Wrinkling of Prestressed Membranes”. *Finite Elem. Anal. Des.*, **42**(11):992–1001.

[153] H. Diamant & T. A. Witten (2011). "Compression Induced Folding of a Sheet: An Integrable System". *Phys. Rev. Lett.*, **107**(16):164 302.

[154] M. Dittmann, S. Schuß, B. Wohlmuth & C. Hesch (2019). "Weak Cn Coupling for Multipatch Isogeometric Analysis in Solid Mechanics". *Int. J. Numer. Meth. Eng.*, **118**(11):678–699.

[155] M. Dittmann, S. Schuß, B. Wohlmuth & C. Hesch (2020). "Crosspoint Modification for Multi-Patch Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **360**:112 768.

[156] S. C. Divi, C. V. Verhoosel, F. Auricchio, A. Reali & E. H. van Brummelen (2020). "Error-Estimate-Based Adaptive Integration for Immersed Isogeometric Analysis". *Comput. Struct.*, **80**(11):2481–2516.

[157] H. Do, Y. Y. Tan, N. Ramos, J. Kiendl & O. Weeger (2020). "Nonlinear Isogeometric Multiscale Simulation for Design and Fabrication of Functionally Graded Knitted Textiles". *Compos. B Eng.*, **202**:108 416.

[158] T. Dokken, T. Lyche & K. F. Pettersen (2013). "Polynomial Splines over Locally Refined Box-Partitions". *Comput. Aided Geom. Des.*, **30**(3):331–356.

[159] J. Dölz, H. Harbrecht, S. Kurz, M. Multerer, S. Schöps & F. Wolf (2020). "Bembel: The Fast Isogeometric Boundary Element C++ Library for Laplace, Helmholtz, and Electric Wave Equation". *Software X*, **11**:100 476.

[160] W. Dörfler (1996). "A Convergent Adaptive Algorithm for Poisson's Equation". *SIAM J. Numer. Anal.*, **33**(3):1106–1124.

[161] W. Dornisch & S. Klinkel (2014). "Treatment of Reissner–Mindlin Shells with Kinks without the Need for Drilling Rotation Stabilization in an Isogeometric Framework". *Comput. Methods Appl. Mech. Eng.*, **276**:35–66.

[162] W. Dornisch, S. Klinkel & B. Simeon (2013). "Isogeometric Reissner–Mindlin Shell Analysis with Exactly Calculated Director Vectors". *Comput. Methods Appl. Mech. Eng.*, **253**:491–504.

[163] W. Dornisch, R. Müller & S. Klinkel (2016). "An Efficient and Robust Rotational Formulation for Isogeometric Reissner–Mindlin Shell Elements". *Comput. Methods Appl. Mech. Eng.*, **303**:1–34.

[164] W. Dornisch & J. Stöckler (2021). "An Isogeometric Mortar Method for the Coupling of Multiple NURBS Domains with Optimal Convergence Rates". *Numer. Math.*, **149**(4):871–931.

[165] W. Dornisch, G. Vitucci & S. Klinkel (2015). "The Weak Substitution Method - an Application of the Mortar Method for Patch Coupling in NURBS-based Isogeometric Analysis". *Int. J. Numer. Meth. Eng.*, **103**(3):205–234.

[166] B. Dortdivanlioglu, A. Javili & C. Linder (2017). "Computational Aspects of Morphological Instabilities Using Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **316**:261–279.

[167] B. Dortdivanlioglu & C. Linder (2019). "Diffusion-Driven Swelling-Induced Instabilities of Hydrogels". *J. Mech. Phys. Solids*, **125**:38–52.

[168] D. Draelants (2015). "PyNCT: Python Numerical Continuation Toolbox".

[169] D. Draelants, J. Broeckhove, G. T. S. Beemster & W. Vanroose (2013). "Numerical Bifurcation Analysis of the Pattern Formation in a Cell Based Auxin Transport Model". *J. Math. Biol.*, **67**(5):1279–1305.

[170] X. Du, G. Zhao, W. Wang & H. Fang (2020). "Nitsche's Method for Non-Conforming Multipatch Coupling in Hyperelastic Isogeometric Analysis". *Comput. Mech.*, **65**(3):687–710.

[171] G. Duffett & B. D. Reddy (1983). "The analysis of incompressible hyperelastic bodies by the finite element method". *Comput. Methods Appl. Mech. Eng.*, **41**(1):105–120.

[172] T. X. Duong, F. Roohbakhshan & R. A. Sauer (2017). "A New Rotation-Free Isogeometric Thin Shell Formulation and a Corresponding Continuity Constraint for Patch Boundaries". *Comput. Methods Appl. Mech. Eng.*, **316**:43–83.

[173] A. Düster, J. Parvizian, Z. Yang & E. Rank (2008). "The Finite Cell Method for Three-Dimensional Problems of Solid Mechanics". *Comput. Methods Appl. Mech. Eng.*, **197**(45):3768–3782.

[174] A. Duval, T. Hirschler, J. E. Cornejo Fuentes, M. Guerder & T. Elguedj (2023). "YETI: YET Another Iga Code". In *IGA 2023*. Lyon, France.

[175] R. Echter, B. Oesterle & M. Bischoff (2013). "A Hierarchic Family of Isogeometric Shell Finite Elements". *Comput. Methods Appl. Mech. Eng.*, **254**:170–180.

[176] M. Epstein & M. A. Forcinito (2001). "Anisotropic Membrane Wrinkling: Theory and Analysis". *Int. J. Solids Struct.*, **38**(30):5253–5272.

[177] N. Fabian, K. Moreland, D. Thompson, A. C. Bauer, P. Marion, B. Gevecik, M. Rasquin & K. E. Jansen (2011). "The Paraview Coprocessing Library: A Scalable, General Purpose in Situ Visualization Library". In *2011 IEEE Symposium on Large Data Analysis and Visualization*, 89–96. IEEE. ISBN 1-4673-0155-8.

[178] M. Fafard & B. Massicotte (1993). "Geometrical Interpretation of the Arc-Length Method". *Comput. Struct.*, **46**(4):603–615.

[179] S. Faghfouri & F. G. Rammerstorfer (2020). "Mode Transitions in Buckling and Post-Buckling of Stretched-Twisted Strips". *Int. J. Nonlin. Mech.*, **127**:103 609.

[180] R. D. Falgout, S. Friedhoff, T. V. Kolev, S. P. MacLachlan & J. B. Schroder (2014). "Parallel time integration with multigrid". *SIAM J. Sci. Comput.*, **36**(6):C635–C661.

[181] A. Farahat, B. Jüttler, M. Kapl & T. Takacs (2023). "Isogeometric Analysis with C1-smooth Functions over Multi-Patch Surfaces". *Comput. Methods Appl. Mech. Eng.*, **403**:115 706.

[182] A. Farahat, H. M. Verhelst, J. Kiendl & M. Kapl (2023). "Isogeometric Analysis for Multi-Patch Structured Kirchhoff–Love Shells". *Comput. Methods Appl. Mech. Eng.*, **411**:116 060.

[183] G. Farin, J. Hoschek & M.-S. Kim (2002). *Handbook of Computer Aided Geometric Design*. Elsevier. ISBN 0-444-51104-0.

[184] Y. Feng, D. Owen & D. Perić (1997). "On the Sign of the Determinant of the Structural Stiffness Matrix for Determination of Loading Increment in Arc-Length Algorithms". *Commun. Numer. Methods Eng.*, **13**(1):47–49.

[185] Y. Feng, D. Perić & D. Owen (1995). "Determination of Travel Directions in Path-Following Methods". *Math. Comput. Model.*, **21**(7):43–59.

[186] Y. Feng, D. Perić & D. Owen (1996). "A New Criterion for Determination of Initial Loading Parameter in Arc-Length Methods". *Comput. Struct.*, **58**(3):479–485.

[187] V. M. Ferrándiz, P. Bucher, R. Zorrilla, R. Rossi, S. Warnakulasuriya, A. Cornejo, jcotela, C. Roig, M. A. Celigueta, J. Maria, tteschemacher, M. Masó, G. Casas, M. Núñez, P. Dadvand, S. Latorre, I. de Pouplana, J. I. González, F. Arrufat, riccardotosi, AFranci, A. Ghantasala, K. B. Sautter, P. Wilson, dbaumgaertner, B. Chandra, A. Geiser, I. Lopez, lluís & J. Gárate (2023). "KratosMultiphysics/Kratos: Release 9.4". Zenodo.

[188] P. J. Flory (1961). "Thermodynamic relations for high elastic materials". *Transactions of the Faraday Society*.

[189] D. R. Forsey & R. H. Bartels (1988). "Hierarchical B-spline refinement". *Comput. Graph. (ACM)*, **22**(4):205–212.

[190] C. Fu, H. H. Dai & F. Xu (2021). "Computing Wrinkling and Restabilization of Stretched Sheets Based on a Consistent Finite-Strain Plate Theory". *Comput. Methods Appl. Mech. Eng.*, **384**:113 986.

[191] C. Fu, T. Wang, F. Xu, Y. Huo & M. Potier-Ferry (2019). "A Modeling and Resolution Framework for Wrinkling in Hyperelastic Sheets at Finite Membrane Strain". *J. Mech. Phys. Solids*, **124**:446–470.

[192] C. Fu, Y. Yang, T. Wang & F. Xu (2022). "A Consistent Finite-Strain Plate Model for Wrinkling of Stretched Anisotropic Hyperelastic Films". *Thin-Walled Struc.*, **179**:109 643.

[193] F. Fujii & E. Ramm (1997). "Computational bifurcation theory: path-tracing, pinpointing and path-switching". *Eng. Struct.*, **19**(5):385–392.

[194] D. Fußeder, B. Simeon & A. V. Vuong (2015). "Fundamental Aspects of Shape Optimization in the Context of Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **286**:313–331.

[195] D. Galhofo, N. Silvestre, A. M. de Deus, L. Reis, A. P. C. Duarte & R. Carvalho (2022). "Structural Behaviour of Pre-Tensioned Solar Sails". *Thin-Walled Struc.*, **181**:110 007.

[196] M. J. Gander (2015). "50 Years of Time Parallel Time Integration". `https://link.springer.com/chapter/10.1007/978-3-319-23321-5_3`.

[197] J. Gao, L. Gao, Z. Luo & P. Li (2019). "Isogeometric Topology Optimization for Continuum Structures Using Density Distribution Function". *Int. J. Numer. Meth. Eng.*, **119**(10):991–1017.

[198] J. Gao, M. Xiao, L. Gao, J. Yan & W. Yan (2020). "Isogeometric Topology Optimization for Computational Design of Re-Entrant and Chiral Auxetic Composites". *Comput. Methods Appl. Mech. Eng.*, **362**:112 876.

[199] E. M. Garau & R. Vázquez (2018). "Algorithms for the implementation of adaptive isogeometric methods using hierarchical B-splines". *Appl. Numer. Math.*, **123**:58–87.

[200] J. Gedicke & C. Carstensen (2013). "A posteriori error estimators for convection-diffusion eigenvalue problems q". *Comput. Methods Appl. Mech. Eng.*

[201] S. Giani, L. Grubišić & J. S. Ovall (2012). "Benchmark results for testing adaptive finite element eigenvalue procedures". *Appl. Numer. Math.*, **62**(2):121–140.

[202] C. Giannelli, B. Jüttler, S. K. Kleiss, A. Mantzaflaris, B. Simeon & J. Špeh (2016). "THB-splines: An Effective Mathematical Technology for Adaptive Refinement in Geometric Design and Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **299**:337–365.

[203] C. Giannelli, B. Jüttler & H. Speleers (2012). "THB-splines: The Truncated Basis for Hierarchical Splines". *Comput. Aided Geom. Des.*, **29**(7):485–498.

[204] C. Giannelli, T. Kanduč, M. Martinelli, G. Sangalli & M. Tani (2022). "Weighted quadrature for hierarchical B-splines". *Comput. Methods Appl. Mech. Eng.*, **400**:115 465.

[205] A. Giust, B. Jüttler & A. Mantzaflaris (2020). "Local (T)HB-spline projectors via restricted hierarchical spline fitting". *Comput. Aided Geom. Des.*, **80**:101 865.

[206] A.-K. Goldbach, A. M. Bauer, R. Wüchner & K.-U. Bletzinger (2020). "CAD-Integrated Parametric Lightweight Design With Isogeometric B-Rep Analysis". *Front. Built Environ.*, **6**:44.

[207] H. Gómez, V. M. Calo, Y. Bazilevs & T. J. R. Hughes (2008). "Isogeometric Analysis of the Cahn–Hilliard Phase-Field Model". *Comput. Methods Appl. Mech. Eng.*, **197**(49):4333–4352.

[208] A. Goyal (2015). "Isogeometric Shell Discretizations for Flexible Multibody Dynamics". doctoralthesis, Technische Universität Kaiserslautern.

[209] C. Graczykowski (2016). "Mathematical Models and Numerical Methods for the Simulation of Adaptive Inflatable Structures for Impact Absorption". *Comput. Struct.*, **174**:3–20.

[210] G. Greschik & M. M. Mikulas (2002). "Design study of a square solar sail architecture". *J. Spacecr. Rockets*, **39**(5):653–661.

[211] G. Greschik, C. V. White & M. A. Salama (2003). "On the precisely uniform and uniaxial tensioning of a film sheet via integrated catenary". *Collect. Tech. Pap. - AIAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn. Mater. Conf.*, **6**(April):4605–4615.

[212] R. G. Grimes, J. G. Lewis & H. D. Simon (1994). "A Shifted Block Lanczos Algorithm for Solving Sparse Symmetric Generalized Eigenproblems". *SIAM J. Matrix Anal. Appl.*, **15**(1):228–272.

[213] D. Groisser & J. Peters (2015). "Matched Gk-constructions Always Yield Ck-continuous Isogeometric Elements". *Comput. Aided Geom. Des.*, **34**:67–72.

[214] J. Grošelj, M. Kapl, M. Knez, T. Takacs & V. Vitrih (2020). "A Super-Smooth C1 Spline Space over Planar Mixed Triangle and Quadrilateral Meshes". *Comput. Struct.*, **80**(12):2623–2643.

[215] F. Gruttmann & R. L. Taylor (1992). "Theory and finite element formulation of rubberlike membrane shells using principal stretches". *Int. J. Numer. Meth. Eng.*, **35**(5):1111–1126.

[216] Y. Güçlü, S. Hadjout & A. Ratnani (2022). "PSYDAC: A High-Performance IGA Library in Python". In *8th European Congress on Computational Methods in Applied Sciences and Engineering*.

[217] G. Guennebaud, B. Jacob *et al.* (2010). "Eigen v3". http://eigen.tuxfamily.org.

[218] Y. Guo, H. Do & M. Ruess (2019). "Isogeometric Stability Analysis of Thin Shells: From Simple Geometries to Engineering Models". *Int. J. Numer. Meth. Eng.*

[219] Y. Guo, J. Heller, T. J. Hughes, M. Ruess & D. Schillinger (2018). "Variationally Consistent Isogeometric Analysis of Trimmed Thin Shells at Finite Deformations, Based on the STEP Exchange Format". *Comput. Methods Appl. Mech. Eng.*, **336**:39–79.

[220] Y. Guo, M. Pan, X. Wei, F. Luo, F. Sun & M. Ruess (2022). "Implicit Dynamic Buckling Analysis of Thin-Shell Isogeometric Structures Considering Geometric Imperfections". *Int. J. Numer. Meth. Eng.*, **n/a**(n/a).

[221] J. Guo, X. Qiu, Z. Zhang, L. Zhang & C. Wang (2022). "Inflated Pillow with Flexible Bistable Kink: Snap Design and Application". *AIAA J.*, **60**(11):6500–6505.

[222] Y. Guo & M. Ruess (2015). "Nitsche's Method for a Coupling of Isogeometric Thin Shells and Blended Shell Structures". *Comput. Methods Appl. Mech. Eng.*, **284**:881–905.

[223] Y. Guo & M. Ruess (2015). "Weak Dirichlet Boundary Conditions for Trimmed Thin Isogeometric Shells". *Comput. Struct.*, **70**(7):1425–1440.

[224] Y. Guo, M. Ruess & D. Schillinger (2017). "A Parameter-Free Variational Coupling Approach for Trimmed Isogeometric Thin Shells". *Comput. Mech.*, **59**(4):693–715.

[225] Y. Guo, Z. Zou & M. Ruess (2021). "Isogeometric Multi-Patch Analyses for Mixed Thin Shells in the Framework of Non-Linear Elasticity". *Comput. Methods Appl. Mech. Eng.*, **380**.

[226] V. Gupta, A. Jameel, S. K. Verma, S. Anand & Y. Anand (2023). "An Insight on NURBS Based Isogeometric Analysis, Its Current Status and Involvement in Mechanical Applications". *Arch. Comput. Methods. Eng.*, **30**(2):1187–1230.

[227] S. Hartmann, D. Benson & A. Nagy (2016). "Isogeometric Analysis with LS-DYNA". In *Journal of Physics: Conference Series*, vol. 734, 032125. IOP Publishing. ISBN 1742-6596.

[228] R. Hartmann, J. Held, T. Leicht & F. Prill (2010). "Error estimation and adaptive mesh refinement for aerodynamic flows". *Notes Numer. Fluid Mech. Multidiscip. Des.*

[229] R. Hartmann & P. Houston (2002). "Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations". *J. Comp. Phys.*, **183**(2):508–532.

[230] R. Hartmann & P. Houston (2003). "Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws". *SIAM J. Sci. Comput.*, **24**(3):979–1004.

[231] R. Hartmann & P. Houston (2006). “Symmetric Interior Penalty DG Methods for the Compressible Navier-Stokes Equations II: Goal-Oriented A Posteriori Error Estimation”. *Int. J. Numer. Anal. Model.*, **3**(1):141–162.

[232] E. M. Haseganu & D. J. Steigmann (1994). “Analysis of Partly Wrinkled Membranes by the Method of Dynamic Relaxation”. *Comput. Mech.*, **14**(6):596–614.

[233] T. J. Healey, Q. Li & R.-B. Cheng (2013). “Wrinkling Behavior of Highly Stretched Rectangular Elastic Films via Parametric Global Bifurcation”. *J. Nonlinear Sci.*, **23**(5):777–805.

[234] P. Hennig, M. Ambati, L. De Lorenzis & M. Kästner (2018). “Projection and transfer operators in adaptive isogeometric analysis with hierarchical B-splines”. *Comput. Methods Appl. Mech. Eng.*, **334**:313–336.

[235] P. Hennig, M. Kästner, P. Morgenstern & D. Peterseim (2017). “Adaptive mesh refinement strategies in isogeometric analysis— A computational comparison”. *Comput. Methods Appl. Mech. Eng.*, **316**:424–448.

[236] P. Hennig, S. Müller & M. Kästner (2016). “Bézier extraction and adaptive refinement of truncated hierarchical NURBS”. *Comput. Methods Appl. Mech. Eng.*, **305**:316–339.

[237] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski & E. T. Phipps (2005). “An Overview of the Trilinos Project”. *ACM Trans. Math. Softw.*, **31**(3):397–423.

[238] A. J. Herrema, E. L. Johnson, D. Proserpio, M. C. Wu, J. Kiendl & M.-C. Hsu (2019). “Penalty Coupling of Non-Matching Isogeometric Kirchhoff–Love Shell Patches with Application to Composite Wind Turbine Blades”. *Comput. Methods Appl. Mech. Eng.*, **346**:810–840.

[239] A. J. Herrema, N. M. Wiese, C. N. Darling, B. Ganapathysubramanian, A. Krishnamurthy & M.-C. Hsu (2017). “A Framework for Parametric Design Optimization Using Isogeometric Analysis”. *Comput. Methods Appl. Mech. Eng.*, **316**:944–965.

[240] C. Hesch & P. Betsch (2012). “Isogeometric Analysis and Domain Decomposition Methods”. *Comput. Methods Appl. Mech. Eng.*, **213–216**:104–112.

[241] C. Hesch, U. Khristenko, R. Krause, A. Popp, A. Seitz, W. Wall & B. Wohlmuth (2022). “Frontiers in Mortar Methods for Isogeometric Analysis”. In J. Schröder & P. Wriggers, eds., *Non-Standard Discretisation Methods in Solid Mechanics*, Lecture Notes in Applied and cm, 405–447. Springer International Publishing, Cham. ISBN 978-3-030-92672-4.

[242] A. Hessenthaler, R. D. Falgout, J. B. Schroder, A. de Vecchi, D. Nordsletten & O. Röhrle (2021). “Time-periodic steady-state solution of fluid-structure interaction and cardiac flow problems through multigrid-reduction-in-time”. *arXiv preprint arXiv:2105.00305.*

[243] R. R. Hiemstra, K. M. Shepherd, M. J. Johnson, L. Quan & T. J. R. Hughes (2020). “Towards Untrimmed NURBS: CAD Embedded Reparameterization of Trimmed B-rep Geometry Using Frame-Field Guided Global Parameterization”. *Comput. Methods Appl. Mech. Eng.*, **369**:113 227.

[244] J. Hinz, M. Abdelmalik & M. Möller (2020). “Goal-Oriented Adaptive THB-Spline Schemes for PDE-Based Planar Parameterization”. *2001.08874.*

[245] J. Hinz, A. Jaeschke, M. Möller & C. Vuik (2021). “The role of PDE-based parameterization techniques in gradient-based IGA shape optimization applications”. *Comput. Methods Appl. Mech. Eng.*, **378**:113 685.

[246] J. Hinz, M. Möller & C. Vuik (2018). “Elliptic grid generation techniques in the framework of isogeometric analysis applications”. *Comput. Aided Geom. Des.*, **65**:48–75.

[247] T. Hirschler, R. Bouclier, D. Dureisseix, A. Duval, T. Elguedj & J. Morlier (2019). “A Dual Domain Decomposition Algorithm for the Analysis of Non-Conforming Isogeometric Kirchhoff–Love Shells”. *Comput. Methods Appl. Mech. Eng.*, **357**:112 578.

[248] T. Hirschler, R. Bouclier, A. Duval, T. Elguedj & J. Morlier (2019). “The Embedded Isogeometric Kirchhoff–Love Shell: From Design to Shape Optimization of Non-Conforming Stiffened Multipatch Structures”. *Comput. Methods Appl. Mech. Eng.*

[249] C. Hofreither & S. Takacs (2017). “Robust Multigrid for Isogeometric Analysis Based on Stable Splittings of Spline Spaces”. *SIAM J. Numer. Anal.*, **55**(4):2004–2024.

[250] C. Hofreither, S. Takacs & W. Zulehner (2017). “A Robust Multigrid Method for Isogeometric Analysis in Two Dimensions Using Boundary Correction”. *Comput. Methods Appl. Mech. Eng.*, **316**:22–42.

[251] D. P. Holmes & A. J. Crosby (2010). “Draping Films: A Wrinkle to Fold Transition”. *Phys. Rev. Lett.*, **105**(3):038 303.

[252] G. Holzapfel (2000). *Nonlinear solid mechanics: A continuum approach for engineering*. John Wiley & Sons, ltd. ISBN 0471823198.

[253] G. A. Holzapfel, R. Eberlein, P. Wriggers & H. W. Weizsäcker (1996). “Large strain analysis of soft biological membranes: Formulation and finite element analysis”. *Comput. Methods Appl. Mech. Eng.*, **132**(1-2):45–61.

[254] C. O. Horgan & J. G. Murphy (2007). “Limiting chain extensibility constitutive models of Valanis-Landel type”. *J. Elast.*, **86**(2):101–111.

[255] T. Horger, A. Reali, B. Wohlmuth & L. Wunderlich (2019). “A Hybrid Isogeometric Approach on Multi-Patches with Applications to Kirchhoff Plates and Eigenvalue Problems”. *Comput. Methods Appl. Mech. Eng.*, **348**:396–408.

[256] S. Hosseini, J. J. C. Remmers, C. V. Verhoosel & R. de Borst (2013). “An isogeometric solid-like shell element for nonlinear analysis”. *Int. J. Numer. Meth. Eng.*, **95**(3):238–256.

[257] S. Hosseini, J. J. Remmers, C. V. Verhoosel & R. de Borst (2014). “An Isogeometric Continuum Shell Element for Non-Linear Analysis”. *Comput. Methods Appl. Mech. Eng.*, **271**:1–22.

[258] M. C. Hsu, D. Kamensky, Y. Bazilevs, M. S. Sacks & T. J. Hughes (2014). “Fluid–structure interaction analysis of bioprosthetic heart valves: significance of arterial wall deformation”. *Comput. Mech.*, **54**(4):1055–1071.

[259] M. C. Hsu, D. Kamensky, F. Xu, J. Kiendl, C. Wang, M. C. Wu, J. Mineroff, A. Reali, Y. Bazilevs & M. S. Sacks (2015). “Dynamic and fluid–structure interaction simulations of bioprosthetic heart valves using parametric design with T-splines and Fung-type material models”. *Comput. Mech.*, **55**(6):1211–1225.

[260] M. C. Hsu, C. Wang, A. J. Herrema, D. Schillinger, A. Ghoshal & Y. Bazilevs (2015). “An Interactive Geometry Modeling and Parametric Design Platform for Isogeometric Analysis”. In *Computers and Mathematics with Applications*.

[261] M.-C. Hsu, C. Wang, F. Xu, A. J. Herrema & A. Krishnamurthy (2016). “Direct Immersogeometric Fluid Flow Analysis Using B-rep CAD Models”. *Comput. Aided Geom. Des.*, **43**:143–158.

[262] Q. Hu, F. Chouly, P. Hu, G. Cheng & S. P. Bordas (2018). “Skew-Symmetric Nitsche’s Formulation in Isogeometric Analysis: Dirichlet and Symmetry Conditions, Patch Coupling and Frictionless Contact”. *Comput. Methods Appl. Mech. Eng.*, **341**:188–220.

[263] Q. Hu, Y. Xia, S. Natarajan, A. Zilian, P. Hu & S. P. Bordas (2020). “Isogeometric analysis of thin Reissner–Mindlin shells: locking phenomena and B-bar method”. *Comput. Mech.*, **65**(5):1323–1341.

[264] W. Huang, Q. Huang, Y. Liu, J. Yang, H. Hu, F. Trochu & P. Causse (2019). “A Fourier Based Reduced Model for Wrinkling Analysis of Circular Membranes”. *Comput. Methods Appl. Mech. Eng.*, **345**:1114–1137.

[265] J. Huang, M. Juszkiewicz, W. H. de Jeu, E. Cerda, T. Emrick, N. Menon & T. P. Russell (2007). “Capillary Wrinkling of Floating Thin Polymer Films”. *Science*, **317**(5838):650–653.

[266] Q. Huang, Z. Kuang, H. Hu & M. Potier-Ferry (2019). "Multiscale Analysis of Membrane Instability by Using the Arlequin Method". *Int. J. Solids Struct.*, **162**:60–75.

[267] T. J. R. Hughes (2000). *The Finite Element Method : Linear Static and Dynamic Finite Element Analysis*. Dover Publications. ISBN 0-486-41181-8.

[268] T. Hughes, J. A. Cottrell & Y. Bazilevs (2005). "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement". *Comput. Methods Appl. Mech. Eng.*, **194**(39-41):4135–4195.

[269] T. J. R. Hughes, J. A. Evans & A. Reali (2014). "Finite Element and NURBS Approximations of Eigenvalue, Boundary-Value, and Initial-Value Problems". *Comput. Methods Appl. Mech. Eng.*, **272**:290–320.

[270] T. J. R. Hughes, A. Reali & G. Sangalli (2008). "Duality and Unified Analysis of Discrete Approximations in Structural Dynamics and Wave Propagation: Comparison of p-Method Finite Elements with k-Method NURBS". *Comput. Methods Appl. Mech. Eng.*, **197**(49):4104–4124.

[271] T. J. Hughes, G. Sangalli, T. Takacs & D. Toshniwal (2021). "Smooth Multi-Patch Discretizations in Isogeometric Analysis". *Handb. Numer. Anal.*, **22**:467–543.

[272] T. Iwasa (2017). "Approximate Estimation of Wrinkle Wavelength and Maximum Amplitude Using a Tension-Field Solution". *Int. J. Solids Struct.*, **121**:201–211.

[273] T. Iwasa (2018). "Experimental Verification on Simplified Estimation Method for Envelope Curve of Wrinkled Membrane Surface Distortions". *Thin-Walled Struc.*, **122**:622–634.

[274] T. Iwasa (2018). "Wrinkle-Reduction Law for Rectangular Membranes under a Shear Load". *AIAA J.*, **56**(7):2870–2876.

[275] H. A. Jahangiry & A. Jahangiri (2019). "Combination of Isogeometric Analysis and Level-Set Method in Topology Optimization of Heat-Conduction Systems". *Appl. Therm. Eng.*, **161**:114 134.

[276] W. Jakob, J. Rhinelander & D. Moldovan (2017). "Pybind11–Seamless Operability between C++ 11 and Python". *URL: https://github. com/pybind/pybind11.*

[277] E. Jambon-Puillet, D. Vella & S. Protière (2016). "The Compression of a Heavy Floating Elastic Film". *Soft Matter*, **12**(46):9289–9296.

[278] A. Jarasjarungkiat, R. Wüchner & K. U. Bletzinger (2008). "A Wrinkling Model Based on Material Modification for Isotropic and Orthotropic Membranes". *Comput. Methods Appl. Mech. Eng.*, **197**(6-8):773–788.

[279] A. Jarasjarungkiat, R. Wüchner & K. U. Bletzinger (2009). "Efficient Sub-Grid Scale Modeling of Membrane Wrinkling by a Projection Method". *Comput. Methods Appl. Mech. Eng.*, **198**(9-12):1097–1116.

[280] A. Javili, B. Dortdivanlioglu, E. Kuhl & C. Linder (2015). "Computational Aspects of Growth-Induced Instabilities through Eigenvalue Analysis". *Comput. Mech.*

[281] C. H. Jenkins, F. Haugen & W. H. Spicher (1998). "Experimental measurement of wrinkling in membranes undergoing planar deformation". *Exp. Mech.*, **38**(2):147–152.

[282] Y. Ji, K. Chen, M. Möller & C. Vuik (2023). "On an Improved PDE-based Elliptic Parameterization Method for Isogeometric Analysis Using Preconditioned Anderson Acceleration". *Comput. Aided Geom. Des.*, **102**:102 191.

[283] Y. Ji, M. Möller & H. M. Verhelst (under review). "Design Through Analysis". In *Fluids Under Control.*

[284] Y. Ji, M. Y. Wang, M. D. Pan, Y. Zhang & C. G. Zhu (2022). "Penalty function-based volumetric parameterization method for isogeometric analysis". *Comput. Aided Geom. Des.*, **94**:1–21.

[285] Y. Ji, M.-Y. Wang, Y. Wang & C.-G. Zhu (2022). "Curvature-Based R-Adaptive Planar NURBS Parameterization Method for Isogeometric Analysis Using Bi-Level Approach". *Comput. Aided Des.*, **150**:103 305.

[286] Y. Ji, M. Wang, Y. Yu & C. Zhu (2023). "Curvature-Based r-Adaptive Isogeometric Analysis with Injectivity-Preserving Multi-Sided Domain Parameterization". *J. Syst. Sci. Complex.*, **36**(1):53–76.

[287] Y. Ji, Y. Y. Yu, M. Y. Wang & C. G. Zhu (2021). "Constructing high-quality planar NURBS parameterization for isogeometric analysis by adjustment control points and weights". *J. Comput. Appl. Math.*, **396**.

[288] N. Jin, W. Lu, Z. Geng & R. P. Fedkiw (2017). "Inequality Cloth". In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, 1–10. Association for Computing Machinery, New York, NY, USA. ISBN 978-1-4503-5091-4.

[289] K. A. Johannessen (2017). "Optimal Quadrature for Univariate and Tensor Product Splines". *Comput. Methods Appl. Mech. Eng.*, **316**:84–99.

[290] E. L. Johnson, M. C. Wu, F. Xu *et al.* (2020). "Thinner biological tissues induce leaflet flutter in aortic heart valve replacements". *Proc. Natl. Acad. Sci.*, **117**(32):19 007–19 016.

[291] G. R. Joldes, A. Wittek & K. Miller (2010). "Real-Time Nonlinear Finite Element Computations on GPU - Application to Neurosurgical Simulation". *Comput. Methods Appl. Mech. Eng.*, **199**(49-52):3305–3314.

[292] G. R. Joldes, A. Wittek & K. Miller (2011). "An Adaptive Dynamic Relaxation Method for Solving Nonlinear Finite Element Problems. Application to Brain Shift Estimation". *Int. J. Numer. Meth. Bio.*, **27**(2):173–185.

[293] R. M. Jones (2006). *Buckling of Bars, Plates, and Shells.* Bull Ridge Corporation, Blacksburg, Va. ISBN 978-0070085930.

[294] B. Jüttler, U. Langer, A. Mantzaflaris, S. E. Moore & W. Zulehner (2014). "Geometry + Simulation Modules: Implementing Isogeometric Analysis". *Proc. Appl. Math. Mech.*, **14**(1):961–962.

[295] C. Kadapa (2021). "A simple extrapolated predictor for overcoming the starting and tracking issues in the arc-length method for nonlinear structural mechanics". *Eng. Struct.*, **234**:111 755.

[296] D. Kamensky & Y. Bazilevs (2019). "tIGAr: Automating Isogeometric Analysis with FEniCS". *Comput. Methods Appl. Mech. Eng.*, **344**:477–498.

[297] D. Kamensky, M.-C. Hsu, D. Schillinger, J. A. Evans, A. Aggarwal, Y. Bazilevs & M. S. Sacks (2015). "An Immersogeometric Variational Framework for Fluid–Structure Interaction: Application to Bioprosthetic Heart Valves". *Comput. Methods Appl. Mech. Eng.*, **284**:1005–1053.

[298] T. Kanai, K. Takizawa, T. E. Tezduyar, T. Tanaka & A. Hartmann (2019). "Compressible-Flow Geometric-Porosity Modeling and Spacecraft Parachute Computation with Isogeometric Discretization". *Comput. Mech.*, **63**(2):301–321.

[299] H. Kang, F. Chen & J. Deng (2013). "Modified T-splines". *Comput. Aided Geom. Des.*, **30**(9):827–843.

[300] S. Kang & S. Im (1997). "Finite Element Analysis of Wrinkling Membranes". *J. Appl. Mech.*, **64**(2):263–269.

[301] S. Kang & S. Im (1999). "Finite Element Analysis of Dynamic Response of Wrinkling Membranes". *Comput. Methods Appl. Mech. Eng.*, **173**(1-2):227–240.

[302] M. Kapl, F. Buchegger, M. Bercovier & B. Jüttler (2017). "Isogeometric Analysis with Geometrically Continuous Functions on Planar Multi-Patch Geometries". *Comput. Methods Appl. Mech. Eng.*, **316**:209–234.

[303] M. Kapl, G. Sangalli & T. Takacs (2017). "Dimension and Basis Construction for Analysis-Suitable G1 Two-Patch Parameterizations". *Comput. Aided Geom. Des.*, **52–53**:75–89.

[304] M. Kapl, G. Sangalli & T. Takacs (2018). "Construction of Analysis-Suitable G1 Planar Multi-Patch Parameterizations". *Comput. Aided Des.*, **97**:41–55.

[305] M. Kapl, G. Sangalli & T. Takacs (2019). "An Isogeometric C1 Subspace on Unstructured Multi-Patch Planar Domains". *Comput. Aided Geom. Des.*, **69**:55–75.

[306] M. Kapl, G. Sangalli & T. Takacs (2021). "A Family of C^1 Quadrilateral Finite Elements". *Adv. Comput. Math.*, **47**(6):82.

[307] M. Kapl & V. Vitrih (2021). "Cs-Smooth Isogeometric Spline Spaces over Planar Bilinear Multi-Patch Parameterizations". *Adv. Comput. Math.*, **47**(3):47.

[308] M. Kapl, V. Vitrih, B. Jüttler & K. Birner (2015). "Isogeometric Analysis with Geometrically Continuous Functions on Two-Patch Geometries". *Comput. Struct.*, **70**(7):1518–1538.

[309] K. Karčiauskas, T. Nguyen & J. Peters (2016). "Generalizing Bicubic Splines for Modeling and IGA with Irregular Layout". *Comput. Aided Des.*, **70**:23–35.

[310] K. Karčiauskas & J. Peters (2015). "Smooth Multi-Sided Blending of Biquadratic Splines". *Comput. Graph.*, **46**:172–185.

[311] K. Karčiauskas & J. Peters (2021). "Least Degree G1-Refinable Multi-Sided Surfaces Suitable For Inclusion Into C1 Bi-2 Splines". *Comput. Aided Des.*, **130**:102 927.

[312] K. Karčiauskas & J. Peters (2021). "Multi-Sided Completion of C2 Bi-3 and C1 Bi-2 Splines: A Unifying Approach". *Comput. Aided Geom. Des.*, **86**:101 978.

[313] K. Karčiauskas & J. Peters (2021). "Refinable Multi-Sided Caps for Bi-Quadratic Splines". *Vision, modeling, and visualization: 26th international symposium on vision, modeling, and visualization virtual-only event September 27-28, 2021 at the Faculty of Computer Science of the Technische Universität Dresden, German*, 1–8.

[314] S. Khalil, Y. Belaasilia, A. Hamdaoui, B. Braikat, N. Damil & M. Potier-Ferry (2020). "A Reduced-Order Modeling Based on Multi-Scale Method for Wrinkles with Variable Orientations". *Int. J. Solids Struct.*, **207**:89–103.

[315] S. Khalil, Y. Belaasilia, A. Hamdaoui, B. Braikat, M. Jamal, N. Damil & Z. Azari (2019). "ANM Analysis of a Wrinkled Elastic Thin Membrane". *Comptes Rendus Mécanique*, **347**(10):701–709.

[316] J. Kiendl (2011). "Isogeometric Analysis and Shape Optimal Design of Shell Structures". Ph.D. thesis, Technische Universität München.

[317] J. Kiendl, F. Auricchio, L. Beirão da Veiga, C. Lovadina & A. Reali (2015). "Isogeometric collocation methods for the Reissner–Mindlin plate problem". *Comput. Methods Appl. Mech. Eng.*, **284**:489–507.

[318] J. Kiendl, Y. Bazilevs, M.-C. Hsu, R. Wüchner & K.-U. Bletzinger (2010). "The Bending Strip Method for Isogeometric Analysis of Kirchhoff–Love Shell Structures Comprised of Multiple Patches". *Comput. Methods Appl. Mech. Eng.*, **199**(37-40):2403–2416.

[319] J. Kiendl, K.-U. Bletzinger, J. Linhard & R. Wüchner (2009). "Isogeometric Shell Analysis with Kirchhoff–Love Elements". *Comput. Methods Appl. Mech. Eng.*, **198**(49-52):3902–3914.

[320] J. Kiendl, M.-C. Hsu, M. C. Wu & A. Reali (2015). "Isogeometric Kirchhoff–Love shell formulations for general hyperelastic materials". *Comput. Methods Appl. Mech. Eng.*, **291**:280–303.

[321] G. Kikis, W. Dornisch & S. Klinkel (2019). "Adjusted Approximation Spaces for the Treatment of Transverse Shear Locking in Isogeometric Reissner–Mindlin Shell Analysis". *Comput. Methods Appl. Mech. Eng.*, **354**:850–870.

[322] T.-Y. Kim, E. Puntel & E. Fried (2012). "Numerical Study of the Wrinkling of a Stretched Thin Sheet". *Int. J. Solids Struct.*, **49**(5):771–782.

[323] G. Kiss, C. Giannelli, U. Zore, B. Jüttler, D. Großmann & J. Barner (2014). "Adaptive CAD model (re-)construction with THB-splines". *Graph. Models*, **76**(5):273–288.

[324] S. Klinkel & S. Govindjee (2002). "Using finite strain 3D-material models in beam and shell elements". *Engineering Computations*, **19**(3-4):254–271.

[325] T. Kondo, Kazou, Iai, Takeshi, Moriguti, Sigeiti, Marasaki (1954). “Tension field theory”. *Mem. unifying study basic Probl. Eng. Sci. by means Geom.*, **1**:61–85.

[326] V. Kosin, S. Beuchler & T. Wick (2023). “A new mixed method for the biharmonic eigenvalue problem”. *Comput. Struct.*, **136**:44–53.

[327] A. Kunoth, T. Lyche, G. Sangalli & S. Serra-Capizzano (2018). *Splines and PDEs: From Approximation Theory to Numerical Linear Algebra.* Springer. ISBN 3-319-94910-1.

[328] G. Kuru, C. V. Verhoosel, K. G. van der Zee & E. H. van Brummelen (2014). “Goal-adaptive Isogeometric Analysis with hierarchical splines”. *Comput. Methods Appl. Mech. Eng.*

[329] W. F. Lam & C. T. Morley (1992). “Arc-Length Method for Passing Limit Points in Structural Calculation”. *Journal of Structural Engineering*, **118**(1):169–185.

[330] D. Lathouwers (2011). “Spatially adaptive eigenvalue estimation for the SN equations on unstructured triangular meshes”. *Ann. Nucl. Energy*, **38**(9):1867–1876.

[331] E. Lavaerts (2020). “Framework to research and design wrinkle free very large flexible offshore solar platforms by adding permeability”. Master’s thesis, Technische Universiteit Delft.

[332] H. Le Meitour, G. Rio, H. Laurent, A. S. Lectez & P. Guigue (2021). “Analysis of Wrinkled Membrane Structures Using a Plane Stress Projection Procedure and the Dynamic Relaxation Method”. *Int. J. Solids Struct.*, **208–209**:194–213.

[333] K. S. Lee, S. E. Han & T. Park (2011). “A Simple Explicit Arc-Length Method Using the Dynamic Relaxation Method with Kinetic Damping”. *Comput. Struct.*, **89**(1-2):216–233.

[334] L. Leidinger (2020). “Explicit Isogeometric B-Rep Analysis for Nonlinear Dynamic Crash Simulations”. Ph.D. thesis, Technische Universität München.

[335] L. F. Leidinger, M. Breitenberger, A. M. Bauer, S. Hartmann, R. Wüchner, K. U. Bletzinger, F. Duddeck & L. Song (2019). “Explicit Dynamic Isogeometric B-Rep Analysis of Penalty-Coupled Trimmed NURBS Shells”. *Comput. Methods Appl. Mech. Eng.*, **351**:891–927.

[336] L. F. Leidinger, S. Hartmann, L. Rorris, M. Breitenberger, A. M. Bauer, R. Wüchner, K.-U. Bletzinger, F. Duddeck & L. Song (2019). “Connecting Design and Analysis: Explicit Isogeometric Analysis Using ANSA and LS-DYNA”. In *8th Before Reality Conference.*

[337] J. Leifer (2007). “Simplified computational models for shear-compliant borders in solar sails”. *J. Spacecr. Rockets*, **44**(3):571–581.

[338] J. Leifer & W. K. Belvin (2003). “Prediction of wrinkle amplitudes in thin film membranes using finite element modeling”. *Collect. Tech. Pap. - AIAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn. Mater. Conf.*, **7**(April):5319–5324.

[339] J. Leifer, J. Black, W. Belvin & V. Behun (2003). “Evaluation of shear compliant borders for wrinkle reduction in thin film membrane structures”. In *44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 1984.

[340] L. Leonetti & J. Kiendl (2023). “A Mixed Integration Point (MIP) Formulation for Hyperelastic Kirchhoff–Love Shells for Nonlinear Static and Dynamic Analysis”. *Comput. Methods Appl. Mech. Eng.*, **416**:116 325.

[341] L. Leonetti, F. Liguori, D. Magisano & G. Garcea (2018). “An Efficient Isogeometric Solid-Shell Formulation for Geometrically Nonlinear Analysis of Elastic Shells”. *Comput. Methods Appl. Mech. Eng.*, **331**:159–183.

[342] L. Leonetti, F. S. Liguori, D. Magisano, J. Kiendl, A. Reali & G. Garcea (2020). “A Robust Penalty Coupling of Non-Matching Isogeometric Kirchhoff–Love Shell Patches in Large Deformations”. *Comput. Methods Appl. Mech. Eng.*, **371**:113 289.

[343] L. Leonetti, D. Magisano, F. Liguori *et al.* (2018). "An isogeometric formulation of the Koiter's theory for buckling and initial post-buckling analysis of composite shells". *Comput. Methods Appl. Mech. Eng.*, **337**:387–410.

[344] L. Leonetti, D. Magisano, A. Madeo, G. Garcea, J. Kiendl & A. Reali (2019). "A simplified Kirchhoff–Love large deformation model for elastic shells and its effective isogeometric formulation". *Comput. Methods Appl. Mech. Eng.*, **354**:369–396.

[345] B. Li, Y. P. Cao, X. Q. Feng & H. Gao (2012). "Mechanics of Morphological Instabilities and Surface Wrinkling in Soft Materials: A Review". *Soft Matter*, **8**(21):5728–5745.

[346] Q. Li & T. J. Healey (2016). "Stability boundaries for wrinkling in highly stretched elastic sheets". *J. Mech. Phys. Solids*, **97**:260–274.

[347] B. Li, F. Jia, Y.-P. Cao, X.-Q. Feng & H. Gao (2011). "Surface Wrinkling Patterns on a Core-Shell Soft Sphere". *Phys. Rev. Lett.*, **106**(23):234 301.

[348] M. Li, Y. Li, C. Zhang, G. Qi, Y. Sui, Y. Luo & J. Liu (2022). "Stiffness Modulation-Driven Wrinkle-Free Membrane". *Appl. Eng. Sci.*, **9**:100 087.

[349] M. Li, Y. Niu, H. Wu, X. Zhang, Y. Luo & Z. Kang (2017). "Wrinkling and Wrinkling-Suppression in Graphene Membranes with Frozen Zone". *Thin Solid Films*, **638**:345–353.

[350] X. Li & M. A. Scott (2014). "Analysis-Suitable T-splines: Characterization, Refineability, and Approximation". *Math. Models Methods Appl. Sci.*, **24**(06):1141–1164.

[351] X. Li & T. W. Sederberg (2019). "S-Splines: A Simple Surface Solution for IGA and CAD". *Comput. Methods Appl. Mech. Eng.*, **350**:664–678.

[352] X. Li & J. Zhang (2018). "AS++ T-splines: Linear Independence and Approximation". *Comput. Methods Appl. Mech. Eng.*, **333**:462–474.

[353] F. S. Liguori, D. Magisano, L. Leonetti & G. Garcea (2021). "Nonlinear Thermoelastic Analysis of Shell Structures: Solid-Shell Modelling and High-Performing Continuation Method". *Compos. Struct.*, **266**:113 734.

[354] J. L. Lions, Y. Maday & G. Turinici (2001). "Résolution d'EDP par un schéma en temps « pararéel »". *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, **332**(7):661–668.

[355] Z. Liu, J. Cheng, M. Yang, P. Yuan, C. Qiu, W. Gao & J. Tan (2019). "Isogeometric Analysis of Large Thin Shell Structures Based on Weak Coupling of Substructures with Unstructured T-splines Patches". *Adv. Eng. Softw.*, **135**:102 692.

[356] X. Liu, C. H. Jenkins & W. W. Schur (2001). "Large Deflection Analysis of Pneumatic Envelopes Using a Penalty Parameter Modified Material Model". *Finite Elem. Anal. Des.*, **37**(3):233–251.

[357] N. Liu, E. L. Johnson, M. R. Rajanna, J. Lua, N. Phan & M. C. Hsu (2021). "Blended Isogeometric Kirchhoff–Love and Continuum Shells". *Comput. Methods Appl. Mech. Eng.*, **385**:114 005.

[358] J. Liu, M. Möller & H. M. Schuttelaars (2021). "Balancing Truncation and Round-off Errors in FEM: One-dimensional Analysis". *J. Comput. Appl. Math.*, **386**:113 219.

[359] F. Liu, F. Xu & C. Fu (2019). "Orientable Wrinkles in Stretched Orthotropic Films". *Extreme Mechanics Letters*, **33**:100 579.

[360] L. Liu, Y. J. Zhang & X. Wei (2015). "Weighted T-splines with Application in Reparameterizing Trimmed NURBS Surfaces". *Comput. Methods Appl. Mech. Eng.*, **295**:108–126.

[361] M. Loibl (2019). "Implementation and Validation of an Isogeometric Hierarchic Shell Formulation". Master's thesis, Technische Universität München.

[362] M. Loibl, L. Leonetti, A. Reali & J. Kiendl (2023). "Patch-Wise Quadrature of Trimmed Surfaces in Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **415**:116 279.

[363] G. Lorenzo, M. A. Scott, K. Tew, T. J. Hughes & H. Gomez (2017). "Hierarchically refined and coarsened splines for moving interface problems, with particular application to phase-field models of prostate tumor growth". *Comput. Methods Appl. Mech. Eng.*, **319**:515–548.

[364] LS-DYNA Manual (2018). *LS-DYNA®Keyword User's Manual Volume 1.* Livermore Software Technology Corporation, r11.0 ed.

[365] K. Lu, M. Accorsi & J. Leonard (2001). "Finite element analysis of membrane wrinkling". *Int. J. Numer. Meth. Eng.*, **50**:1017–1038.

[366] Y. Luo, J. Xing & Z. Kang (2020). "Topology Optimization Using Material-Field Series Expansion and Kriging-based Algorithm: An Effective Non-Gradient Method". *Comput. Methods Appl. Mech. Eng.*, **364**:112 966.

[367] Y. Luo, J. Xing, Z. Kang, J. Zhan & M. Li (2020). "Uncertainty of Membrane Wrinkling Behaviors Considering Initial Thickness Imperfections". *Int. J. Solids Struct.*, **191–192**:264–277.

[368] Y. Luo, J. Xing, Y. Niu, M. Li & Z. Kang (2017). "Wrinkle-Free Design of Thin Membrane Structures Using Stress-Based Topology Optimization". *J. Mech. Phys. Solids*, **102**:277–293.

[369] T. Lyche & K. Morken (2008). "Spline Methods Draft". *Department of Informatics, Center of Mathematics for Applications, University of Oslo, Oslo*, 3–8.

[370] L. Ma, L. He & Y. Ni (2020). "Tunable Hierarchical Wrinkling: From Models to Applications". *J. Appl. Phys.*, **127**(11):111 101.

[371] D. Magisano, L. Leonetti & G. Garcea (2017). "Advantages of the mixed format in geometrically nonlinear analysis of beams and shells using solid finite elements". *Int. J. Numer. Meth. Eng.*, **109**(9):1237–1262.

[372] D. Magisano, L. Leonetti & G. Garcea (2017). "How to Improve Efficiency and Robustness of the Newton Method in Geometrically Non-Linear Structural Problem Discretized via Displacement-Based Finite Elements". *Comput. Methods Appl. Mech. Eng.*, **313**:986–1005.

[373] M. Majeed & F. Cirak (2017). "Isogeometric Analysis Using Manifold-Based Smooth Basis Functions". *Comput. Methods Appl. Mech. Eng.*, **316**:547–567.

[374] E. H. Mansfield (1969). "Tension Field Theory, a New Approach Which Shows Its Duality with Inextensional Theory". In M. Hetényi & W. G. Vincenti, eds., *Applied Mechanics*, International Union of Theoretical and Applied Mechanics, 305–320. Springer, Berlin, Heidelberg. ISBN 978-3-642-85640-2.

[375] E. Mansfield (1970). "Load transfer via a wrinkled membrane". *Proc. R. Soc. London. A. Math. Phys. Sci.*, **316**(1525):269–289.

[376] A. Mantzaflaris (2020). "An Overview of Geometry plus Simulation Modules". In *Mathematical Aspects of Computer and Information Sciences: 8th International Conference, MACIS 2019, Gebze, Turkey, November 13–15, 2019, Revised Selected Papers 8*, 453–456. Springer. ISBN 3-030-43119-3.

[377] G. Marckmann & E. Verron (2006). "Comparison of hyperelastic models for rubber-like materials". *Rubber Chemistry and Technology*, **79**(5):835–858.

[378] M. Marsala, A. Mantzaflaris & B. Mourrain (2022). "G1 – Smooth Biquintic Approximation of Catmull-Clark Subdivision Surfaces". *Comput. Aided Geom. Des.*, **99**:102 158.

[379] B. Marussig & T. J. R. Hughes (2018). "A Review of Trimming in Isogeometric Analysis: Challenges, Data Exchange and Simulation Aspects". *Arch. Comput. Methods. Eng.*, **25**(4):1059–1127.

[380] R. Massabò & L. Gambarotta (2006). "Wrinkling of Plane Isotropic Biological Membranes". *J. Appl. Mech.*, **74**(3):550–559.

[381] F. Massarwi, B. van Sosin & G. Elber (2018). "Untrimming: Precise Conversion of Trimmed-Surfaces to Tensor-Product Surfaces". *Comput. Graph.*, **70**:80–91.

[382] D. A. Matoz-Fernandez, F. A. Davidson, N. R. Stanley-Wall & R. Sknepnek (2020). "Wrinkle Patterns in Active Viscoelastic Thin Sheets". *Phys. Rev. Res.*, **2**(1):013 165.

[383] S. Meduri, M. Cremonesi, A. Frangi & U. Perego (2022). "A Lagrangian Fluid–Structure Interaction Approach for the Simulation of Airbag Deployment". *Finite Elem. Anal. Des.*, **198**:103 659.

[384] K. Meerbergen, A. Spence & D. Roose (1994). "Shift-Invert and Cayley Transforms for Detection of Rightmost Eigenvalues of Nonsymmetric Matrices". *BIT Numer. Math.*, **34**(3):409–423.

[385] B.-A. Memon & X.-z. Su (2004). "Arc-Length Technique for Nonlinear Finite Element Analysis". *Journal of Zhejiang University-SCIENCE A*, **5**(5):618–628.

[386] D. Miao, Z. Zou, M. A. Scott, M. J. Borden & D. C. Thomas (2021). "Isogeometric Bézier Dual Mortaring: The Kirchhoff–Love Shell Problem". *Comput. Methods Appl. Mech. Eng.*, **382**:113 873.

[387] M. M. Mikulas (1964). *Behavior of a flat stretched membrane wrinkled by the rotation of an attached hub.* National Aeronautics and Space Administration.

[388] M. M. Mikulas & A. L. Adler (2003). "Rapid structural assessment approach for square solar sails including edge support cords". *Collect. Tech. Pap. - AIAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn. Mater. Conf.*, **1**(April):440–453.

[389] T. Miyamura (2000). "Wrinkling on stretched circular membrane under in-plane torsion:". *Eng. Struct.*, **22**(11):1407–1425.

[390] Y. Miyazaki (2006). "Wrinkle/Slack Model and Finite Element Dynamics of Membrane". *Int. J. Numer. Meth. Eng.*, **66**(7):1179–1209.

[391] M. Möller & D. Kuzmin (2006). "Adaptive mesh refinement for high-resolution finite element schemes". *Int. J. Numer. Meth. Fluids*, **52**(5):545–569.

[392] J. Montes, B. Thomaszewski, S. Mudur & T. Popa (2020). "Computational Design of Skintight Clothing". *ACM Trans. Graph.*, **39**(4):105:105:1–105:105:12.

[393] M. Mooney (1940). "A theory of large elastic deformation". *J. Appl. Phys.*

[394] S. Morganti, F. Auricchio, D. J. Benson *et al.* (2015). "Patient-specific isogeometric structural analysis of aortic valve closure". *Comput. Methods Appl. Mech. Eng.*, **284**:508–520.

[395] J. Mosler (2008). "A Fully Variational Algorithmic Formulation for Wrinkling at Finite Strains". *Proc. Appl. Math. Mech.*, **8**(1):10 447–10 448.

[396] J. Mosler (2008). "A Novel Variational Algorithmic Formulation for Wrinkling at Finite Strains Based on Energy Minimization: Application to Mesh Adaption". *Comput. Methods Appl. Mech. Eng.*, **197**(9-12):1131–1146.

[397] J. Mosler & F. Cirak (2009). "A Variational Formulation for Finite Deformation Wrinkling Analysis of Inelastic Membranes". *Comput. Methods Appl. Mech. Eng.*, **198**(27-29):2087–2098.

[398] M. Moulaeifard, F. Wellmann, S. Bernard, M. de la Varga & D. Bommes (2023). "Subdivide and Conquer: Adapting Non-Manifold Subdivision Surfaces to Surface-Based Representation and Reconstruction of Complex Geological Structures". *Math. Geosci.*, **55**(1):81–111.

[399] F. Muttin (1996). "A Finite Element for Wrinkled Curved Elastic Membranes, and Its Application to Sails". *Commun. Numer. Methods Eng.*, **12**(11):775–785.

[400] A. Najah, B. Cochelin, N. Damil & M. Potier-Ferry (1998). "A Critical Review of Asymptotic Numerical Methods". *Arch. Comput. Methods. Eng.*, **5**(1):31–50.

[401] K. Nakashino & M. C. Natori (2005). "Efficient Modification Scheme of Stress-Strain Tensor for Wrinkled Membranes". *AIAA J.*, **43**(1).

[402] K. Nakashino, A. Nordmark & A. Eriksson (2020). "Geometrically Nonlinear Isogeometric Analysis of a Partly Wrinkled Membrane Structure". *Comput. Struct.*, **239**:106 302.

[403] V. Nayyar, K. Ravi-Chandar & R. Huang (2011). "Stretch-Induced Stress Patterns and Wrinkles in Hyperelastic Thin Sheets". *Int. J. Solids Struct.*, **48**(25-26):3471–3483.

[404] M. Nazzal, M. Alkhader, A. Agha, F. Abu-Farha, Z. Ali & W. AlDamaty (2022). "Wrinkling Suppression in Thin Membranes Using Designed Geometrical Features". *Proc. Inst. Mech. Eng. C J. Mech. Eng. Sci.*, **236**(19):10 163–10 174.

[405] V. P. Nguyen, C. Anitescu, S. P. Bordas & T. Rabczuk (2015). "Isogeometric Analysis: An Overview and Computer Implementation Aspects". *Math. Comput. Simul.*, **117**:89–116.

[406] T.-H. Nguyen, R. R. Hiemstra, S. K. F. Stoter & D. Schillinger (2022). "A Variational Approach Based on Perturbed Eigenvalue Analysis for Improving Spectral Properties of Isogeometric Multipatch Discretizations". *Comput. Methods Appl. Mech. Eng.*, **392**:114 671.

[407] T. Nguyen, K. Karčiauskas & J. Peters (2016). "C1 Finite Elements on Non-Tensor-Product 2d and 3d Manifolds". *Appl. Math. Comput.*, **272**:148–158.

[408] T. Nguyen & J. Peters (2016). "Refinable C1 Spline Elements for Irregular Quad Layout". *Comput. Aided Geom. Des.*, **43**:123–130.

[409] C. Nguyen, X. Zhuang, L. Chamoin, X. Zhao, H. Nguyen-Xuan & T. Rabczuk (2020). "Three-Dimensional Topology Optimization of Auxetic Metamaterial Using Isogeometric Analysis and Model Order Reduction". *Comput. Methods Appl. Mech. Eng.*, **371**:113 306.

[410] B. H. Nguyen, X. Zhuang, P. Wriggers, T. Rabczuk, M. E. Mear & H. D. Tran (2017). "Isogeometric Symmetric Galerkin Boundary Element Method for Three-Dimensional Elasticity Problems". *Comput. Methods Appl. Mech. Eng.*, **323**:132–150.

[411] N. Nguyen-Thanh, N. Valizadeh, M. Nguyen, H. Nguyen-Xuan, X. Zhuang, P. Areias, G. Zi, Y. Bazilevs, L. De Lorenzis & T. Rabczuk (2015). "An Extended Isogeometric Thin Shell Analysis Based on Kirchhoff–Love Theory". *Comput. Methods Appl. Mech. Eng.*, **284**:265–291.

[412] J. Nitsche (1971). "Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind". *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, **36**(1):9–15.

[413] B. Oesterle, S. Bieber, R. Sachse, E. Ramm & M. Bischoff (2018). "Intrinsically Locking-Free Formulations for Isogeometric Beam, Plate and Shell Analysis". *Proc. Appl. Math. Mech.*, **18**(1):e201800 399.

[414] B. Oesterle, F. Geiger, D. Forster, M. Fröhlich & M. Bischoff (2022). "A Study on the Approximation Power of NURBS and the Significance of Exact Geometry in Isogeometric Pre-Buckling Analyses of Shells". *Comput. Methods Appl. Mech. Eng.*, **397**:115 144.

[415] B. Oesterle, E. Ramm & M. Bischoff (2016). "A Shear Deformable, Rotation-Free Isogeometric Shell Formulation". *Comput. Methods Appl. Mech. Eng.*, **307**:235–255.

[416] B. Oesterle, R. Sachse, E. Ramm & M. Bischoff (2017). "Hierarchic Isogeometric Large Rotation Shell Elements Including Linearized Transverse Shear Parametrization". *Comput. Methods Appl. Mech. Eng.*, **321**:383–405.

[417] Y. Ofir, D. Givoli & A. Libai (2011). "An Axisymmetric Parachute Model with Wrinkling". *J. Mech. Mat. Struct.*, **6**(1):417–442.

[418] R. W. Ogden (1997). *Non-linear elastic deformations.* Dover Publications. ISBN 9780486696485.

[419] J. R. H. Otter (1965). "Computations for Prestressed Concrete Reactor Pressure Vessels Using Dynamic Relaxation". *Nucl. Struct. Eng.*, **1**(1):61–75.

[420] J. R. H. Otter, A. C. Cassell, R. E. Hobbs & Poisson (1966). "Dynamic Relaxation". *Proceedings of the Institution of Civil Engineers*, **35**(4):633–656.

[421] J. Otter & A. Day (1960). "Tidal Computations". *The Engineer.*

[422] Oxford English Dictionary (2023). "Wrinkle, n.1".

[423] A. Pagani & E. Carrera (2018). "Unified formulation of geometrically nonlinear refined beam theories". *Mech. Adv. Mat. Struct.*, **25**(1):15–31.

[424] M. Pan, B. Jüttler & A. Mantzaflaris (2021). "Efficient matrix assembly in isogeometric analysis with hierarchical B-splines". *J. Comput. Appl. Math.*, **390**:113 278.

[425] M. Pan, B. Jüttler & F. Scholz (2022). "Efficient matrix computation for isogeometric discretizations with hierarchical B-splines in any dimension". *Comput. Methods Appl. Mech. Eng.*, **388**:114 210.

[426] Q. Pan, G. Xu, G. Xu & Y. Zhang (2016). "Isogeometric analysis based on extended Catmull–Clark subdivision". *Comput. Struct.*, **71**(1):105–119.

[427] A. Panaitescu, M. Xin, B. Davidovitch, J. Chopin & A. Kudrolli (2019). "Birth and Decay of Tensional Wrinkles in Hyperelastic Sheets". *Phys. Rev. E.*

[428] J. Panetta, F. Isvoranu, T. Chen, E. Siéfert, B. Roman & M. Pauly (2021). "Computational Inverse Design of Surface-Based Inflatables". *ACM Trans. Graph.*, **40**(4):40:1–40:14.

[429] M. Papadrakakis (1981). "A Method for the Automatic Evaluation of the Dynamic Relaxation Parameters". *Comput. Methods Appl. Mech. Eng.*, **25**(1):35–48.

[430] H. Parisch (1995). "A Continuum-based Shell Theory for Non-linear Applications". *Int. J. Numer. Meth. Eng.*, **38**(11):1855–1883.

[431] J. Parvizian, A. Düster & E. Rank (2007). "Finite Cell Method". *Comput. Mech.*, **41**(1):121–133.

[432] T. Pasch, L. F. Leidinger, A. Apostolatos, R. Wüchner, K. U. Bletzinger & F. Duddeck (2021). "A Priori Penalty Factor Determination for (Trimmed) NURBS-based Shells with Dirichlet and Coupling Constraints in Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **377**:113 688.

[433] K. Paul, C. Zimmermann, T. X. Duong & R. A. Sauer (2020). "Isogeometric Continuity Constraints for Multi-Patch Shells Governed by Fourth-Order Deformation and Phase Field Models". *Comput. Methods Appl. Mech. Eng.*, **370**.

[434] J. D. Paulsen (2019). "Wrapping Liquids, Solids, and Gases in Thin Sheets". *Ann. Rev. Condens. Matter. Phys.*, **10**(1):431–450.

[435] J. D. Paulsen, V. Démery, K. B. Toga, Z. Qiu, T. P. Russell, B. Davidovitch & N. Menon (2017). "Geometry-Driven Folding of a Floating Annular Sheet". *Phys. Rev. Lett.*, **118**(4):048 004.

[436] J. D. Paulsen, E. Hohlfeld, H. King, J. Huang, Z. Qiu, T. P. Russell, N. Menon, D. Vella & B. Davidovitch (2016). "Curvature-Induced Stiffness and the Spatial Variation of Wavelength in Wrinkled Sheets.". *Proc. Natl. Acad. Sci.*, **113**(5):1144–9.

[437] L. Piegl & W. Tiller (1995). *The NURBS Book.* Monographs in Visual Communications. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-97387-1.

[438] A. C. Pipkin (1986). "The Relaxed Energy Density for Isotropic Elastic Membranes". *IMA J. Appl. Math.*, **36**(1):85–99.

[439] A. C. Pipkin (1993). "Relaxed Energy Densities for Small Deformations of Membranes". *IMA J. Appl. Math.*, **50**(3):225–237.

[440] A. C. Pipkin (1994). "Relaxed Energy Densities for Large Deformations of Membranes". *IMA J. Appl. Math.*, **52**(3):297–308.

[441] P. Plucinsky & K. Bhattacharya (2017). "Microstructure-Enabled Control of Wrinkling in Nematic Elastomer Sheets". *J. Mech. Phys. Solids*, **102**:125–150.

[442] L. Pocivavsek, R. Dellsy, A. Kern, S. Johnson, B. Lin, K. Y. C. Lee & E. Cerda (2008). "Stress and fold localization in thin elastic membranes". *Science*.

[443] M. Potier-Ferry, F. Mohri, F. Xu, N. Damil, B. Braikat, K. Mhada, H. Hu, Q. Huang & S. Nezamabadi (Mon Feb 29 19:00:00 EST 2016). "Cellular Instabilities Analyzed by Multi-Scale Fourier Series: A Review". *Discrete Contin. Dyn. Syst. S*, **9**(2):585–597.

[444] H. Prautzsch, W. Böhm & M. Paluszny (2002). *Bézier and B-spline Techniques*, vol. 6. Springer.

[445] G. Pretti, W. M. Coombs & C. E. Augarde (2022). "A displacement-controlled arc-length solution scheme". *Comput. Struct.*, **258**:106 674.

[446] D. Proserpio, M. Ambati, L. De Lorenzis & J. Kiendl (2020). "A Framework for Efficient Isogeometric Computations of Phase-Field Brittle Fracture in Multipatch Shell Structures". *Comput. Methods Appl. Mech. Eng.*, **372**:113 363.

[447] D. Proserpio, M. Ambati, L. De Lorenzis & J. Kiendl (2021). "Phase-Field Simulation of Ductile Fracture in Shell Structures". *Comput. Methods Appl. Mech. Eng.*, **385**:114 019.

[448] D. Proserpio & J. Kiendl (2022). "Penalty Coupling of Trimmed Isogeometric Kirchhoff–Love Shell Patches". *J. Mech.*, **38**:156–165.

[449] E. Puntel, L. Deseri & E. Fried (2011). "Wrinkling of a Stretched Thin Sheet". *J. Elast.*, **105**(1-2):137–170.

[450] W. Punurai, W. Tongpool & J. Héctor Morales (2012). "Implementation of Genetic Algorithm for Optimum Cutting Pattern Generation of Wrinkle Free Finishing Membrane Structures". *Finite Elem. Anal. Des.*, **58**:84–90.

[451] Z. Qin, M. Taylor, M. Hwang, K. Bertoldi & M. J. Buehler (2014). "Effect of Wrinkles on the Surface Area of Graphene: Toward the Design of Nanoelectronics". *Nano Lett.*, **14**(11):6520–6525.

[452] Y. Qiu (2023). "Yixuan/Spectra".

[453] K. Rafetseder & W. Zulehner (2018). "A decomposition result for Kirchhoff plate bending problems and a new discretization approach". *SIAM J. Numer. Anal.*, **56**(3):1961–1986.

[454] K. Rafetseder & W. Zulehner (2019). "A New Mixed Approach to Kirchhoff–Love Shells". *Comput. Methods Appl. Mech. Eng.*, **346**:440–455.

[455] A. Rafsanjani, A. Akbarzadeh & D. Pasini (2015). "Snapping Mechanical Metamaterials under Tension". *Adv. Mater.*, **27**(39):5931–5935.

[456] S. A. Ragon, Z. Gürdal & L. T. Watson (2002). "A comparison of three algorithms for tracing nonlinear equilibrium paths of structural systems". *Int. J. Solids Struct.*, **39**(3):689–698.

[457] T. Raible, K. Tegeler, S. Löhnert & P. Wriggers (2005). "Development of a Wrinkling Algorithm for Orthotropic Membrane Materials". *Comput. Methods Appl. Mech. Eng.*, **194**(21-24):2550–2568.

[458] S. Raknes, X. Deng, Y. Bazilevs, D. Benson, K. Mathisen & T. Kvamsdal (2013). "Isogeometric Rotation-Free Bending-Stabilized Cables: Statics, Dynamics, Bending Strips and Coupling with Shells". *Comput. Methods Appl. Mech. Eng.*, **263**:127–143.

[459] E. Ramm (1981). “Strategies for Tracing the Nonlinear Response Near Limit Points”. In W. Wunderlich, E. Stein & K.-J. Bathe, eds., *Nonlinear Finite Element Analysis in Structural Mechanics*, 63–89. Springer, Berlin, Heidelberg. ISBN 978-3-642-81589-8.

[460] R. Rannacher (2004). “Adaptive finite element methods in flow computations”. *Recent Advances in Adaptive Computation. Contemporary Mathematics*, **383**:176–183.

[461] J. N. Reddy (2014). *An Introduction to Nonlinear Finite Element Analysis: With Applications to Heat Transfer, Fluid Mechanics, and Solid Mechanics.* Oxford University Press, Oxford. ISBN 0-19-964175-7.

[462] S. Reese & S. Govindjee (1998). “A theory of finite viscoelasticity and numerical aspects”. *Int. J. Solids Struct.*, **35**(26-27):3455–3482.

[463] M. Reichle, J. Arf, B. Simeon & S. Klinkel (2023). “Smooth Multi-Patch Scaled Boundary Isogeometric Analysis for Kirchhoff-Love Shells”. *arxiv:2304.05857 [cs, math].*

[464] U. Reif (1997). “A Refineable Space of Smooth Spline Surfaces of Arbitrary Topological Genus”. *J. Approx. Theory*, **90**(2):174–199.

[465] E. Reissner (1938). “On Tension Field Theory”. *Proc. of the 5th Int. Congr. for Applied Mechanics Harvard Univ. & MIT*, 88–92.

[466] H. F. Renzsch (2018). “Development of a System for the Investigation of Spinnakers Using Fluid Structure Interaction Methods”. Ph.D. thesis, Technische Universiteit Delft.

[467] M. Rezaiee-Pajand & H. Estiri (2017). “Geometrically Nonlinear Analysis of Shells by Various Dynamic Relaxation Methods”. *World J. Eng.*, **14**(5):381–405.

[468] A. Riffnaller-Schiefer, U. H. Augsdörfer & D. W. Fellner (2016). “Isogeometric shell analysis with NURBS compatible subdivision surfaces”. *Appl. Math. Comput.*, **272**:139–147.

[469] E. Riks (1972). “The Application of Newton’s Method to the Problem of Elastic Stability”. *J. Appl. Mech.*, **39**(4):1060.

[470] E. Riks (1979). “An Incremental Approach to the Solution of Snapping and Buckling Problems”. *Int. J. Solids Struct.*, **15**(7):529–551.

[471] M. M. Ripp, V. Démery, T. Zhang & J. D. Paulsen (2020). “Geometry Underlies the Mechanical Stiffening and Softening of an Indented Floating Film”. *Soft Matter*, **16**(17):4121–4130.

[472] M. Ritto-Corrêa & D. Camotim (2008). “On the arc-length and other quadratic control methods: Established, less known and new implementation procedures”. *Comput. Struct.*, **86**(11-12):1353–1368.

[473] M. Rivetti (2013). “Non-Symmetric Localized Fold of a Floating Sheet”. *Comptes Rendus Mécanique*, **341**(3):333–338.

[474] M. Rivetti & S. Neukirch (2014). “The Mode Branching Route to Localization of the Finite-Length Floating Elastica”. *J. Mech. Phys. Solids*, **69**:143–155.

[475] R. S. Rivlin (1947). “Torsion of a rubber cylinder”. *J. Appl. Phys.*, **18**(5):444–449.

[476] D. G. Roddeman (1991). “Finite-Element Analysis of Wrinkling Membranes”. *Commun. Appl. Numer. Meth.*, **7**(4):299–307.

[477] D. G. Roddeman, J. Drukker, C. W. Oomens & J. D. Janssen (1987). “The Wrinkling of Thin Membranes: Part II—Numerical Analysis”. *J. Appl. Mech.*, **54**(4):888–892.

[478] D. G. Roddeman, J. Drukker, C. W. J. Oomens & J. D. Janssen (1987). “The Wrinkling of Thin Membranes: Part I—Theory”. *J. Appl. Mech.*, **54**(4):884–887.

[479] J. Rodriguez, G. Rio, J. M. Cadou & J. Troufflard (2011). “Numerical Study of Dynamic Relaxation with Kinetic Damping Applied to Inflatable Fabric Structures with Extensions for 3D Solid Element and Non-Linear Behavior”. *Thin-Walled Struc.*, **49**(11):1468–1474.

[480] F. Roohbakhshan & R. A. Sauer (2017). “Efficient isogeometric thin shell formulations for soft biological materials”. *Biomech. Model. Mechanobiol.*, **16**(5):1569–1597.

[481] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn & E. Rank (2013). “Weakly Enforced Essential Boundary Conditions for NURBS-embedded and Trimmed NURBS Geometries on the Basis of the Finite Cell Method”. *Int. J. Numer. Meth. Eng.*, **95**(10):811–846.

[482] M. Ruess, D. Schillinger, A. I. Özcan & E. Rank (2014). “Weak Coupling for Isogeometric Analysis of Non-Matching and Trimmed Multi-Patch Geometries”. *Comput. Methods Appl. Mech. Eng.*, **269**:46–71.

[483] K. Rupp (2022). “Microprocessor Trend Data”. `https://github.com/karlrupp/microprocessor-trend-data`.

[484] H. Sakamoto, Y. Miyazaki & K. Park (2003). “Evaluation of Cable Suspended Membrane Structures for Wrinkle-Free Design”. In *44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. American Institute of Aeronautics and Astronautics, Norfolk, Virginia. ISBN 978-1-62410-100-7.

[485] H. Sakamoto, K. C. Park & Y. Miyazaki (2005). “Dynamic wrinkle reduction strategies for cable-suspended membrane structures”. *J. Spacecr. Rockets*, **42**(5):850–858.

[486] H. Sakamoto, K. C. Park & Y. Miyazaki (2007). “Evaluation of membrane structure designs using boundary web cables for uniform tensioning”. *Acta Astronaut.*, **60**(10-11):846–857.

[487] A. G. Salinger, N. M. Bou-Rabee, E. A. BURROUGHS, R. P. PAWLOWSKI, R. B. LEHOUCQ, LOUIS. ROMERO & E. D. WILKES (2002). “LOCA 1.0 Library of Continuation Algorithms: Theory and Implementation Manual”. *Tech. rep.*, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States); Sandia ....

[488] E. Sande, C. Manni & H. Speleers (2020). “Explicit error estimates for spline approximation of arbitrary smoothness in isogeometric analysis”. *Numer. Math.*, **144**(4):889–929.

[489] G. Sangalli & M. Tani (2018). “Matrix-Free Weighted Quadrature for a Computationally Efficient Isogeometric k-Method”. *Comput. Methods Appl. Mech. Eng.*, **338**:117–133.

[490] R. A. Sauer & T. X. Duong (2017). “On the theoretical foundations of thin solid and liquid shells”. *Math. Mech. Solids*, **22**(3):343–371.

[491] D. Schillinger, I. Harari, M.-C. Hsu, D. Kamensky, S. K. F. Stoter, Y. Yu & Y. Zhao (2016). “The Non-Symmetric Nitsche Method for the Parameter-Free Imposition of Weak Boundary and Coupling Conditions in Immersed Finite Elements”. *Comput. Methods Appl. Mech. Eng.*, **309**:625–652.

[492] D. Schillinger & M. Ruess (2015). “The Finite Cell Method: A Review in the Context of Higher-Order Structural Analysis of CAD and Image-Based Geometric Models”. *Arch. Comput. Methods. Eng.*, **22**(3):391–455.

[493] I. J. Schoenberg (1946). “Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions. Part B. On the Problem of Osculatory Interpolation. A Second Class of Analytic Approximation Formulae”. *Quart. Appl. Math.*, **4**(2):112–141.

[494] D. Schöllhammer & T. P. Fries (2019). “Reissner–Mindlin Shell Theory Based on Tangential Differential Calculus”. *Comput. Methods Appl. Mech. Eng.*, **352**:172–188.

[495] R. D. Schroll, M. Adda-Bedia, E. Cerda, J. Huang, N. Menon, T. P. Russell, K. B. Toga, D. Vella & B. Davidovitch (2013). “Capillary Deformations of Bendable Films”. *Phys. Rev. Lett.*, **111**(1):014 301.

[496] S. Schuß, M. Dittmann, B. Wohlmuth, S. Klinkel & C. Hesch (2019). “Multi-Patch Isogeometric Analysis for Kirchhoff–Love Shell Elements”. *Comput. Methods Appl. Mech. Eng.*, **349**:91–116.

[497] K. Schweizerhof & P. Wriggers (1986). “Consistent linearization for path following methods in nonlinear fe analysis”. *Comput. Methods Appl. Mech. Eng.*, **59**(3):261–279.

[498] T. W. Sederberg, J. Zheng, A. Bakenov & A. Nasri (2003). “T-splines and T-NURCCs”. *ACM Trans. Graph.*, **22**(3):477–484.

[499] A. Shamanskiy, M. H. Gfrerer, J. Hinz & B. Simeon (2020). “Isogeometric Parametrization Inspired by Large Elastic Deformation”. *Comput. Methods Appl. Mech. Eng.*, **363**:112 920.

[500] M. H. Shariff (2000). “Strain energy function for filled and unfilled rubberlike material”. *Rubber Chemistry and Technology*, **73**(1):1–18.

[501] B. Shawcroft & K. M. Shepherd (2023). “Establishing Connectivity for Isogeometric and Hybrid FEA/Isogeometric Analyses with Multiple Parts through Beam Element Projection”. In *2023 Intermountain Engineering, Technology and Computing (IETC)*, 272–277.

[502] G. Shen, Z. Bi, Y. Xia & G. Zheng (2023). “An Orthotropic Peridynamic Shell Model for Linear Elastic Deformation and Crack Propagation”. *Eng. Fract. Mech.*, **282**:109 174.

[503] K. M. Shepherd (2023). “Isogeometric Boundary-Conforming Body-In-White Crash Model Construction, Analysis, and Comparison With Fem Model”. *Faculty Publications*.

[504] K. M. Shepherd, X. D. Gu & T. J. R. Hughes (2022). “Isogeometric Model Reconstruction of Open Shells via Ricci Flow and Quadrilateral Layout-Inducing Energies”. *Eng. Struct.*, **252**:113 602.

[505] J. Shi (1996). “Computing Critical Points and Secondary Paths in Nonlinear Structural Stability Analysis by the Finite Element Method”. *Comput. Struct.*, **58**(1):203–220.

[506] J. Shi & M. Crisfield (1994). “A Semi-Direct Approach for the Computation of Singular Points”. *Comput. Struct.*, **51**(1):107–115.

[507] S. G. Shin & C. O. Lee (2020). “Splitting Basis Techniques in Cloth Simulation by Isogeometric Analysis”. *Comput. Methods Appl. Mech. Eng.*, **362**:112 871.

[508] T. A. Shugar (1990). *Automated Dynamic Relaxation Solution Algorithms for Compliant Systems*. Naval Civil Engineering Laboratory.

[509] E. Siéfert, J. Bico, E. Reyssat & B. Roman (2020). “Geometry and Mechanics of Inextensible Curvilinear Balloons”. *J. Mech. Phys. Solids*, **143**:104 068.

[510] E. Siéfert, E. Reyssat, J. Bico & B. Roman (2019). “Bio-Inspired Pneumatic Shape-Morphing Elastomers”. *Nat. Mater.*, **18**(1):24–28.

[511] E. Siéfert, E. Reyssat, J. Bico & B. Roman (2019). “Programming Curvilinear Paths of Flat Inflatables”. *Proc. Natl. Acad. Sci.*, **116**(34):16 692–16 696.

[512] N. Silvestre (2016). “Wrinkling of stretched thin sheets: Is restrained Poisson’s effect the sole cause?”. *Eng. Struct.*, **106**:195–208.

[513] J. C. Simo & R. L. Taylor (1991). “Quasi-incompressible finite elasticity in principal stretches. continuum basis and numerical algorithms”. *Comput. Methods Appl. Mech. Eng.*

[514] A. A. Sipos & E. Fehér (2016). “Disappearance of Stretch-Induced Wrinkles of Thin Sheets: A Study of Orthotropic Films”. *Int. J. Solids Struct.*, **97–98**:275–283.

[515] M. Skouras, B. Thomaszewski, P. Kaufmann, A. Garg, B. Bickel, E. Grinspun & M. Gross (2014). “Designing Inflatable Structures”. *ACM Trans. Graph.*, **33**(4):63:1–63:10.

[516] P. M. Sobota, W. Dornisch, R. Müller & S. Klinkel (2017). "Implicit dynamic analysis using an isogeometric Reissner-Mindlin shell formulation". *Int. J. Numer. Meth. Eng.*, **110**(9):803–825.

[517] J. Sogn & S. Takacs (2019). "Robust Multigrid Solvers for the Biharmonic Problem in Isogeometric Analysis". *Comput. Struct.*, **77**(1):105–124.

[518] J. Sogn & S. Takacs (2023). "Multigrid Solvers for Isogeometric Discretizations of the Second Biharmonic Problem". *Math. Models Methods Appl. Sci.*, **33**(09):1803–1828.

[519] H. Speleers & C. Manni (2016). "Effortless quasi-interpolation in hierarchical spaces". *Numer. Math.*, **132**(1):155–184.

[520] D. J. Steigmann (2013). "A Well-Posed Finite-Strain Model for Thin Elastic Sheets with Bending Stiffness". *Math. Mech. Solids*, **18**(1):103–112.

[521] D. J. Steigmann & A. E. Green (1990). "Tension-Field Theory". *Proc. R. Soc. London. A. Math. Phys. Sci.*, **429**(1876):141–173.

[522] P. Steinmann, M. Hossain & G. Possart (2012). "Hyperelastic models for rubber-like materials: Consistent tangent operators and suitability for Treloar's data". *Arch. Appl. Mech.*, **82**(9):1183–1217.

[523] N. Stoop, R. Lagrange, D. Terwagne, P. M. Reis & J. Dunkel (2015). "Curvature-Induced Symmetry Breaking Determines Elastic Surface Patterns". *Nat. Mater.*, **14**(3):337–342.

[524] N. Stoop & M. M. Müller (2015). "Non-Linear Buckling and Symmetry Breaking of a Soft Elastic Sheet Sliding on a Cylindrical Substrate". *Int. J. Nonlin. Mech.*, **75**:115–122.

[525] P. Sun, J. Huang, J. Zhang & F. Meng (2022). "Wrinkling Patterns and Stress Analysis of Tensile Membrane with Rigid Elements". *Appl. Sci.*, **12**(13).

[526] T. Sussman & K. J. Bathe (1987). "A finite element formulation for nonlinear incompressible elastic and inelastic analysis". *Comput. Struct.*, **26**(1-2):357–409.

[527] D. Swain & A. Gupta (2015). "Interfacial Growth during Closure of a Cutaneous Wound: Stress Generation and Wrinkle Formation". *Soft Matter*, **11**(32):6499–6508.

[528] D. Swain & A. Gupta (2016). "Mechanics of Cutaneous Wound Rupture". *J. Biomech.*, **49**(15):3722–3730.

[529] D. Swain & A. Gupta (2019). "Mechanochemical Aspects of Skin Wound Healing in Microgravity". *Mech. Res. Commun.*, **96**:87–93.

[530] K. Sze, X. Liu & S. Lo (2004). "Popular Benchmark Problems for Geometric Nonlinear Analysis of Shells". *Finite Elem. Anal. Des.*, **40**(11):1551–1569.

[531] M. Taffetani & D. Vella (2017). "Regimes of Wrinkling in Pressurized Elastic Shells.". *Philos. Trans. Royal Soc. A*, **375**(2093):20160 330.

[532] S. Takacs (2018). "Robust Approximation Error Estimates and Multigrid Solvers for Isogeometric Multi-Patch Discretizations". *Math. Models Methods Appl. Sci.*, **28**(10):1899–1928.

[533] T. Takacs (2023). "Approximation Properties over Self-Similar Meshes of Curved Finite Elements and Applications to Subdivision Based Isogeometric Analysis". *arxiv:2307.10403 [cs, math].*

[534] T. Takacs & D. Toshniwal (2023). "Almost-C1 Splines: Biquadratic Splines on Unstructured Quadrilateral Meshes and Their Application to Fourth Order Problems". *Comput. Methods Appl. Mech. Eng.*, **403**:115 640.

[535] A. Takei, F. Brau, B. Roman & J. Bico (2011). "Stretch-Induced Wrinkles in Reinforced Membranes: From out-of-Plane to in-Plane Structures". *Europhys. Lett.*, **96**(6):64 001.

[536] K. Takizawa, M. Fritze, D. Montes, T. Spielman & T. E. Tezduyar (2012). "Fluid–Structure Interaction Modeling of Ringsail Parachutes with Disreefing and Modified Geometric Porosity". *Comput. Mech.*, **50**(6):835–854.

[537] K. Takizawa, D. Montes, M. Fritze, S. Mcintyre, J. Boben & T. E. Tezduyar (2013). "Methods for Fsi Modeling of Spacecraft Parachute Dynamics and Cover Separation". *Math. Models Methods Appl. Sci.*, **23**(02):307–338.

[538] K. Takizawa, C. Moorman, S. Wright, T. Spielman & T. E. Tezduyar (2011). "Fluid–Structure Interaction Modeling and Performance Analysis of the Orion Spacecraft Parachutes". *Int. J. Numer. Meth. Fluids*, **65**(1-3):271–285.

[539] K. Takizawa, T. Spielman, C. Moorman & T. E. Tezduyar (2012). "Fluid-Structure Interaction Modeling of Spacecraft Parachutes for Simulation-Based Design". *J. Appl. Mech.*, **79**(1):010 907.

[540] K. Takizawa, T. Spielman & T. E. Tezduyar (2011). "Space–Time FSI Modeling and Dynamical Analysis of Spacecraft Parachutes and Parachute Clusters". *Comput. Mech.*, **48**(3):345–364.

[541] K. Takizawa & T. E. Tezduyar (2012). "Computational Methods for Parachute Fluid–Structure Interactions". *Arch. Comput. Methods. Eng.*, **19**(1):125–169.

[542] K. Takizawa, T. E. Tezduyar, J. Boben, N. Kostov, C. Boswell & A. Buscher (2013). "Fluid–Structure Interaction Modeling of Clusters of Spacecraft Parachutes with Modified Geometric Porosity". *Comput. Mech.*, **52**(6):1351–1364.

[543] K. Takizawa, T. E. Tezduyar, C. Boswell, R. Kolesar & K. Montel (2014). "FSI Modeling of the Reefed Stages and Disreefing of the Orion Spacecraft Parachutes". *Comput. Mech.*, **54**(5):1203–1220.

[544] K. Takizawa, T. E. Tezduyar, C. Boswell, Y. Tsutsui & K. Montel (2015). "Special Methods for Aerodynamic-Moment Calculations from Parachute FSI Modeling". *Comput. Mech.*, **55**(6):1059–1069.

[545] K. Takizawa, T. E. Tezduyar & T. Kanai (2017). "Porosity Models and Computational Methods for Compressible-Flow Aerodynamics of Parachutes with Geometric Porosity". *Math. Models Methods Appl. Sci.*, **27**(04):771–806.

[546] K. Takizawa, T. E. Tezduyar & R. Kolesar (2015). "FSI Modeling of the Orion Spacecraft Drogue Parachutes". *Comput. Mech.*, **55**(6):1167–1179.

[547] K. Takizawa, T. E. Tezduyar, R. Kolesar, C. Boswell, T. Kanai & K. Montel (2014). "Multiscale Methods for Gore Curvature Calculations from FSI Modeling of Spacecraft Parachutes". *Comput. Mech.*, **54**(6):1461–1476.

[548] K. Takizawa, T. E. Tezduyar & T. Terahara (2016). "Ram-air parachute structural and fluid mechanics computations with the Space–Time Isogeometric Analysis (ST-IGA)". *Comput. Fluids*, **141**:191–200.

[549] K. Takizawa, S. Wright, C. Moorman & T. E. Tezduyar (2011). "Fluid–Structure Interaction Modeling of Parachute Clusters". *Int. J. Numer. Meth. Fluids*, **65**(1-3):286–307.

[550] Y. Tan, B. Hu, J. Song, Z. Chu & W. Wu (2020). "Bioinspired Multiscale Wrinkling Patterns on Curved Substrates: An Overview". *Nanomicro Lett.*, **12**(1):1–42.

[551] M. Taylor, K. Bertoldi & D. J. Steigmann (2014). "Spatial Resolution of Wrinkle Patterns in Thin Elastic Sheets at Finite Strain". *J. Mech. Phys. Solids*, **62**:163–180.

[552] M. Taylor, B. Davidovitch, Z. Qiu & K. Bertoldi (2015). "A Comparative Analysis of Numerical Approaches to the Mechanics of Elastic Sheets". *J. Mech. Phys. Solids*, **79**:92–107.

[553] M. Taylor & M. Shirani (2020). "Simulation of Wrinkling in Incompressible Anisotropic Thin Sheets with Wavy Fibers". *Int. J. Nonlin. Mech.*, **127**:103 610.

[554] M. Taylor, M. Shirani, Y. Dabiri, J. M. Guccione & D. J. Steigmann (2019). "Finite Elastic Wrinkling Deformations of Incompressible Fiber-Reinforced Plates". *Int. J. Eng. Sci.*, **144**:103 138.

[555] A. B. Tepole, H. Kabaria, K.-U. Bletzinger *et al.* (2015). "Isogeometric Kirchhoff–Love shell formulations for biological membranes". *Comput. Methods Appl. Mech. Eng.*, **293**:328–347.

[556] T. Terahara, K. Takizawa, R. Avsar & T. E. Tezduyar (2023). "T-Splines Computational Membrane–Cable Structural Mechanics with Continuity and Smoothness: II. Spacecraft Parachutes". *Comput. Mech.*, **71**(4):677–686.

[557] T. Terahara, K. Takizawa & T. E. Tezduyar (2023). "T-Splines Computational Membrane–Cable Structural Mechanics with Continuity and Smoothness: I. Method and Implementation". *Comput. Mech.*, **71**(4):657–675.

[558] T. Teschemacher, A. M. Bauer, R. Aristio, M. Meßmer, R. Wüchner & K.-U. Bletzinger (2022). "Concepts of Data Collection for the CAD-integrated Isogeometric Analysis". *Eng. Comput.*, **38**(6):5675–5693.

[559] T. Teschemacher & K.-U. Bletzinger (2023). "CAD-integrated Parametric Modular Construction Design". *Eng. Rep.*, **5**(8):e12 632.

[560] A. Tessler, D. W. Sleight & J. T. Wang (2005). "Effective modeling and nonlinear shell analysis of thin membranes exhibiting structural wrinkling". *J. Spacecr. Rockets*, **42**(2):287–298.

[561] J. Thies, M. Wouters, R. S. Hennig & W. Vanroose (2021). "Towards Scalable Automatic Exploration of Bifurcation Diagrams for Large-Scale Applications". In *Lecture Notes in Computational Science and Engineering*, vol. 139, 981–989. Springer, Cham. ISBN 978-3-030-55873-4.

[562] D. C. Thomas, L. Engvall, S. K. Schmidt, K. Tew & M. A. Scott (2022). "U-splines: Splines over unstructured meshes". *Comput. Methods Appl. Mech. Eng.*, **401**:115 515.

[563] J. C. Thomas & C. Wielgosz (2004). "Deflections of Highly Inflated Fabric Tubes". *Thin-Walled Struc.*, **42**(7):1049–1066.

[564] H. Tian, M. Potier-Ferry & F. Abed-Meraim (2021). "Buckling and Wrinkling of Thin Membranes by Using a Numerical Solver Based on Multivariate Taylor Series". *Int. J. Solids Struct.*, **230–231**:111 165.

[565] R. Tielen, M. Möller, D. Göddeke & C. Vuik (2020). "P-Multigrid Methods and Their Comparison to h-Multigrid Methods within Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **372**:113 347.

[566] B. H. Topping & A. I. Khan (1994). "Parallel Computation Schemes for Dynamic Relaxation". *Engineering Computations*, **11**(6):513–548.

[567] D. Toshniwal (2022). "Quadratic Splines on Quad-Tri Meshes: Construction and an Application to Simulations on Watertight Reconstructions of Trimmed Surfaces". *Comput. Methods Appl. Mech. Eng.*, **388**:114 174.

[568] D. Toshniwal, H. Speleers, R. R. Hiemstra & T. J. R. Hughes (2017). "Multi-Degree Smooth Polar Splines: A Framework for Geometric Modeling and Isogeometric Analysis". *Comput. Methods Appl. Mech. Eng.*, **316**:1005–1061.

[569] D. Toshniwal, H. Speleers & T. J. Hughes (2017). "Smooth Cubic Spline Spaces on Unstructured Quadrilateral Meshes with Particular Emphasis on Extraordinary Points: Geometric Design and Isogeometric Analysis Considerations". *Comput. Methods Appl. Mech. Eng.*, **327**:411–458.

[570] L. R. G. Treloar (1975). *The Physics of Rubber Elasticity.* Oxford University Press, Oxford, UK, 1st ed.

[571] P. Underwood (1983). "Dynamic Relaxation". *Computational method for transient analysis*, 265.

[572] K. G. Van Der Zee & C. V. Verhoosel (2011). "Isogeometric Analysis-Based Goal-Oriented Error Estimation for Free-Boundary Problems". *Finite Elem. Anal. Des.*, **47**:600–609.

[573] C. van Engelen (2022). “Modelling of a Flexible Inflatable Floater: Analysis of the Stiffness Behaviour of a Drop-Stitch Panel for Offshore Floating Photovoltaics”. Master’s thesis, Technische Universiteit Delft.

[574] H. Vandeparre, M. Piñeirua, F. Brau, B. Roman, J. Bico, C. Gay, W. Bao, C. N. Lau, P. M. Reis & P. Damman (2011). “Wrinkling Hierarchy in Constrained Thin Sheets from Suspended Graphene to Curtains”. *Phys. Rev. Lett.*, **106**(22):224 301.

[575] P. Vannucci, B. Cochelin, N. Damil & M. Potier-Ferry (1998). “An Asymptotic-Numerical Method to Compute Bifurcating Branches”. *Int. J. Numer. Meth. Eng.*, **41**(8):1365–1389.

[576] R. Vázquez (2016). “A New Design for the Implementation of Isogeometric Analysis in Octave and Matlab: GeoPDEs 3.0”. *Comput. Struct.*, **72**(3):523–554.

[577] D. Vella, A. Ajdari, A. Vaziri & A. Boudaoud (2011). “The Indentation of Pressurized Elastic Shells: From Polymeric Capsules to Yeast Cells”. *J. R. Soc. Interface*, **9**(68):448–455.

[578] D. Vella, A. Ajdari, A. Vaziri & A. Boudaoud (2011). “Wrinkling of Pressurized Elastic Shells”. *Phys. Rev. Lett.*, **107**(17):174 301.

[579] D. Vella & B. Davidovitch (2018). “Regimes of Wrinkling in an Indented Floating Elastic Sheet”. *Phys. Rev. E*, **98**(1):013 003.

[580] D. Vella, H. Ebrahimi, A. Vaziri & B. Davidovitch (2015). “Wrinkling Reveals a New Isometry of Pressurized Elastic Shells”. *Europhys. Lett.*, **112**(2):24 007.

[581] D. Vella, J. Huang, N. Menon, T. P. Russell & B. Davidovitch (2015). “Indentation of Ultrathin Elastic Films and the Emergence of Asymptotic Isometry”. *Phys. Rev. Lett.*, **114**(1):014 301.

[582] R. Veltz (2020). “BifurcationKit.Jl”. GitHub.

[583] H. M. Verhelst (2019). “Modelling Wrinkling Behaviour of Large Floating Thin Offshore Structures: An application of Isogeometric Structural Analysis for Post-Buckling Analyses”. Master’s thesis, Delft University of Technology.

[584] H. M. Verhelst, J. H. D. Besten & M. Möller (2023). “An Adaptive Parallel Arc-Length Method”. *arxiv:2303.01075 [cs, math].*

[585] H. M. Verhelst, J. H. D. Besten & M. Möller (in preparation). “Tension-Field Theory for General Hyperelastic Materials in Isogeometric Analysis”.

[586] H. M. Verhelst, A. Mantzaflaris, M. Möller & J. H. D. Besten (2023). “Goal-Adaptive Meshing of Isogeometric Kirchhoff-Love Shells”. *arxiv:2307.08356 [cs, math].*

[587] H. M. Verhelst, M. Möller, J. Den Besten, A. Mantzaflaris & M. Kaminski (2021). “Stretch-Based Hyperelastic Material Formulations for Isogeometric Kirchhoff–Love Shells with Application to Wrinkling”. *Comput. Aided Des.*, **139**:103 075.

[588] H. M. Verhelst, M. Möller, J. Den Besten, F. J. Vermolen & M. L. Kaminski (2021). “Equilibrium Path Analysis Including Bifurcations with an Arc-Length Method Avoiding A Priori Perturbations”. In *Lecture Notes in Computational Science and Engineering*, vol. 139, 1109–1117. Springer, Cham. ISBN 978-3-030-55873-4.

[589] H. M. Verhelst, P. Weinmüller, A. Mantzaflaris, T. Takacs & D. Toshniwal (2023). “A Comparison of Smooth Basis Constructions for Isogeometric Analysis”. *arxiv:2309.04405 [cs, math].*

[590] C. V. Verhoosel, G. J. Van Zwieten, B. Van Rietbergen & R. De Borst (2015). “Image-based goal-oriented adaptive isogeometric analysis with application to the micro-mechanical modeling of trabecular bone”. *Comput. Methods Appl. Mech. Eng.*, **284**:138–164.

[591] A.-V. Vuong, C. Giannelli, B. Jüttler & B. Simeon (2011). “A hierarchical approach to adaptive local refinement in isogeometric analysis”. *Comput. Methods Appl. Mech. Eng.*, **200**(49-52):3554–3567.

[592] H. Wagner (1929). "Flat sheet metal girders with very thin metal web Part I: General theories and assumptions". *Tech. Rep. 604*, National Advisory Committee for Aeronautics.

[593] H. Wagner (1931). "Flat sheet metal girders with very thin metal web: Part II: Sheet metlab girders with sparce resistance to bending - oblique uprights - stiffness". *Natl. Advis. Comm. Aeronaut.*

[594] T. J. W. Wagner & D. Vella (2011). "Floating carpets and the delamination of elastic sheets". *Phys. Rev. Lett.*, **107**(4):044 301.

[595] W. Wagner & P. Wriggers (1988). "A Simple Method for the Calculation of Postcritical Branches". *Engineering computations*, **5**(2):103–109.

[596] W. A. Wall, M. A. Frenzel & C. Cyron (2008). "Isogeometric Structural Shape Optimization". *Comput. Methods Appl. Mech. Eng.*, **197**(33):2976–2988.

[597] Y. Wang & D. J. Benson (2016). "Geometrically Constrained Isogeometric Parameterized Level-Set Based Topology Optimization via Trimmed Elements". *Front. Mech. Eng.*, **11**(4):328–343.

[598] Y. Wang & D. J. Benson (2016). "Isogeometric Analysis for Parameterized LSM-based Structural Topology Optimization". *Comput. Mech.*, **57**(1):19–35.

[599] T. Wang, Z. Dai, M. Potier-Ferry & F. Xu (2022). "Curvature-Regulated Multiphase Patterns in Tori". *Phys. Rev. Lett.*, **130**:048 201.

[600] Z. Wang & M. Devel (2011). "Periodic Ripples in Suspended Graphene". *Phys. Rev. B*, **83**(12):125 422.

[601] T. Wang, C. Fu, F. Xu *et al.* (2019). "On the wrinkling and restabilization of highly stretched sheets". *Int. J. Eng. Sci.*, **136**:1–16.

[602] Y. Wang, L. Jin, H. Yang, P. Hao, Y. Ji & B. Wang (2023). "Isogeometric-Based Mapping Modeling and Buckling Analysis of Stiffened Panels". *Thin-Walled Struc.*, **186**:110 676.

[603] T. Wang, F. Liu, C. Fu, X. Zhang, K. Wang & F. Xu (2021). "Curvature Tunes Wrinkling in Shells". *Int. J. Eng. Sci.*, **164**:103 490.

[604] C. Wang, Y. Liu, L. Lan & H. Tan (2013). "Graphene Wrinkling: Formation, Evolution and Collapse". *Nanoscale*, **5**(10):4454–4461.

[605] Y. Wang, Q. Liu, J. Zhang, T. Hong, W. Sun, L. Tang, E. Arnold, Z. Suo, W. Hong, Z. Ren & C. F. Guo (2019). "Giant Poisson's Effect for Wrinkle-Free Stretchable Transparent Electrodes". *Adv. Mater.*, **31**(35):1902 955.

[606] C. Y. Wang, K. Mylvaganam & L. C. Zhang (2009). "Wrinkling of Monolayer Graphene: A Study by Molecular Dynamics and Continuum Plate Theory". *Phys. Rev. B*, **80**(15):155 445.

[607] Z. P. Wang, L. H. Poh, J. Dirrenberger, Y. Zhu & S. Forest (2017). "Isogeometric Shape Optimization of Smoothed Petal Auxetic Structures via Computational Periodic Homogenization". *Comput. Methods Appl. Mech. Eng.*, **323**:250–271.

[608] F.-F. Wang, T. Wang, X. Zhang, Y. Huang, I. Giorgio & F. Xu (2022). "Wrinkling of Twisted Thin Films". *Int. J. Solids Struct.*, 112075.

[609] T. Wang, Y. Yang, C. Fu, F. Liu, K. Wang & F. Xu (2020). "Wrinkling and Smoothing of a Soft Shell". *J. Mech. Phys. Solids*, **134**:103 738.

[610] T. Wang, Y. Yang, C. Fu & F. Xu (2022). "Competition between Mullins and Curvature Effects in the Wrinkling of Stretched Soft Shells". *Int. J. Solids Struct.*, **241**:111 473.

[611] T. Wang, Y. Yang & F. Xu (2022). "Mechanics of Tension-Induced Film Wrinkling and Restabilization: A Review". *Proc. R. Soc. A. Math. Phys. Eng. Sci.*, **478**(2263).

[612] Y. Wang, Y. Yu & Y. Lin (2022). "Isogeometric Analysis with Embedded Stiffened Shells for the Hull Structural Mechanical Analysis". *J. Mar. Sci. Technol.*, **27**(1):786–805.

[613] Q. Wang & X. Zhao (2015). "A Three-Dimensional Phase Diagram of Growth-Induced Surface Instabilities". *Sci. Rep.*, **5**(1):8887.

[614] O. Weeger, B. Narayanan & M. L. Dunn (2019). "Isogeometric Shape Optimization of Nonlinear, Curved 3D Beams and Beam Structures". *Comput. Methods Appl. Mech. Eng.*, **345**:26–51.

[615] X. Wei, X. Li, K. Qian, T. J. R. Hughes, Y. J. Zhang & H. Casquero (2022). "Analysis-Suitable Unstructured T-splines: Multiple Extraordinary Points per Face". *Comput. Methods Appl. Mech. Eng.*, **391**:114 494.

[616] X. Wei, Y. Zhang, L. Liu & T. J. R. Hughes (2017). "Truncated T-splines: Fundamentals and Methods". *Comput. Methods Appl. Mech. Eng.*, **316**:349–372.

[617] J. Weingarten (1861). "Ueber Eine Klasse Auf Einander Abwickelbarer Flächen.". *Journal Für Mathematik*, **1861**(59):382–393.

[618] P. Weinmüller & T. Takacs (2021). "Construction of Approximate C1 Bases for Isogeometric Analysis on Two-Patch Domains". *Comput. Methods Appl. Mech. Eng.*, **385**:114 017.

[619] P. Weinmüller & T. Takacs (2022). "An Approximate C1 Multi-Patch Space for Isogeometric Analysis with a Comparison to Nitsche's Method". *Comput. Methods Appl. Mech. Eng.*, **401**:115 592.

[620] G. A. Wempner (1971). "Discrete Approximations Related to Nonlinear Theories of Solids". *Int. J. Solids Struct.*, **7**(11):1581–1599.

[621] C. Wielgosz & J.-C. Thomas (2002). "Deflections of Inflatable Fabric Panels at High Pressure". *Thin-Walled Struc.*, **40**(6):523–536.

[622] C. Wielgosz & J. C. Thomas (2003). "An Inflatable Fabric Beam Finite Element". *Commun. Numer. Methods Eng.*, **19**(4):307–312.

[623] B. I. Wohlmuth (2000). "A Mortar Finite Element Method Using Dual Spaces for the Lagrange Multiplier". *SIAM J. Numer. Anal.*, **38**(3):989–1012.

[624] Y. W. Wong & S. Pellegrino (2002). "Amplitude of Wrinkles in Thin Membranes". In *Solid Mechanics and Its Applications*, vol. 104, 257–270. Springer, Dordrecht.

[625] Y. W. Wong & S. Pellegrino (2002). "Computation of Wrinkle Amplitudes in Thin Membrane". In *43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 1369.

[626] W. Wong & S. Pellegrino (2006). "Wrinkled Membranes I: Experiments". *J. Mech. Mat. Struct.*, **1**(1):3–25.

[627] W. Wong & S. Pellegrino (2006). "Wrinkled Membranes II: Analytical Models". *J. Mech. Mat. Struct.*, **1**(1):27–61.

[628] W. Wong & S. Pellegrino (2006). "Wrinkled Membranes III: Numerical Simulations". *J. Mech. Mat. Struct.*, **1**(1):63–95.

[629] Y. Wong, S. Pellegrino & K. Park (2003). "Prediction of Winkle Amplitudes in Square Solar Sails". In *44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 1982.

[630] M. Wouters & W. Vanroose (2019). "Automatic Exploration Techniques of Numerical Bifurcation Diagrams Illustrated by the Ginzburg–Landau Equation". *SIAM J. Appl. Dyn. Syst.*, **18**(4):2047–2098.

[631] P. Wriggers (2008). *Nonlinear Finite Element Methods.* Springer Science and Business Media. ISBN 978-3-540-71000-4.

[632] P. Wriggers & J. C. Simo (1990). "A general procedure for the direct computation of turning and bifurcation points". *Int. J. Numer. Meth. Eng.*, **30**(1):155–176.

[633] P. Wriggers, W. Wagner & C. Miehe (1988). "A Quadratically Convergent Procedure for the Calculation of Stability Points in Finite Element Analysis". *Comput. Methods Appl. Mech. Eng.*, **70**(3):329–347.

[634] M. C. Wu, H. M. Muchowski, E. L. Johnson, M. R. Rajanna & M.-C. Hsu (2019). "Immersogeometric Fluid–Structure Interaction Modeling and Simulation of Transcatheter Aortic Valve Replacement". *Comput. Methods Appl. Mech. Eng.*, **357**:112 556.

[635] M. C. Wu, R. Zakerzadeh, D. Kamensky *et al.* (2018). "An anisotropic constitutive model for immersogeometric fluid–structure interaction analysis of bioprosthetic heart valves". *J. Biomech.*, **74**:23–31.

[636] L. Wunderlich, A. Seitz, M. D. Alaydın, B. Wohlmuth & A. Popp (2019). "Biorthogonal Splines for Optimal Weak Patch-Coupling in Isogeometric Analysis with Applications to Finite Deformation Elasticity". *Comput. Methods Appl. Mech. Eng.*, **346**:197–215.

[637] Y. Xia, X. Meng, G. Shen, G. Zheng & P. Hu (2021). "Isogeometric Analysis of Cracks with Peridynamics". *Comput. Methods Appl. Mech. Eng.*, **377**:113 700.

[638] Y. Xia, H. Wang, G. Zheng, W. Li & G. Shen (2023). "Mesh-Free Discretization of Peridynamic Shell Structures and Coupling Model with Isogeometric Analysis". *Eng. Fract. Mech.*, **277**:108 997.

[639] Y. Xia, H. Wang, G. Zheng, G. Shen & P. Hu (2022). "Discontinuous Galerkin Isogeometric Analysis with Peridynamic Model for Crack Simulation of Shell Structure". *Comput. Methods Appl. Mech. Eng.*, **398**:115 193.

[640] X. Xiao, M. Sabin & F. Cirak (2019). "Interrogation of Spline Surfaces with Application to Isogeometric Design and Analysis of Lattice-Skin Structures". *Comput. Methods Appl. Mech. Eng.*, **351**:928–950.

[641] M. Xin & B. Davidovitch (2021). "Stretching Hookean Ribbons Part I: Relative Edge Extension Underlies Transverse Compression and Buckling Instability". *Eur. Phys. J. E*, **44**(7):1–14.

[642] J. Xing, Y. Luo, J. Zhan & Z. Kang (2018). "Global shape optimization of fixtures to suppress wrinkles in large-displacement membrane structures". *Int. J. Solids Struct.*, **144-145**:301–312.

[643] F. Xu, C. Fu & Y. Yang (2020). "Water Affects Morphogenesis of Growing Aquatic Plant Leaves". *Phys. Rev. Lett.*, **124**(3):038 003.

[644] F. Xu, E. L. Johnson, C. Wang *et al.* (2020). "Computational investigation of left ventricular hemodynamics following bioprosthetic aortic and mitral valve replacement". *Mech. Res. Commun.*, 103604.

[645] G. Xu, B. Li, L. Shu, L. Chen, J. Xu & T. Khajah (2019). "Efficient R-Adaptive Isogeometric Analysis with Winslow's Mapping and Monitor Function Approach". *J. Comput. Appl. Math.*, **351**:186–197.

[646] F. Xu, S. Morganti, R. Zakerzadeh, D. Kamensky, F. Auricchio, A. Reali, T. J. R. Hughes, M. S. Sacks & M.-C. Hsu (2018). "A Framework for Designing Patient-Specific Bioprosthetic Heart Valves Using Immersogeometric Fluid-Structure Interaction Analysis". *Int. J. Numer. Meth. Bio.*, **34**(4):e2938.

[647] F. Xu, M. Potier-Ferry, S. Belouettar & H. Hu (2015). "Multiple Bifurcations in Wrinkling Analysis of Thin Films on Compliant Substrates". *Int. J. Nonlin. Mech.*, **76**:203–222.

[648] F. Xu, S. Zhao, C. Lu & M. Potier-Ferry (2020). "Pattern Selection in Core-Shell Spheres". *J. Mech. Phys. Solids*, **137**:103 892.

[649] D. Yan, D. Huangfu, K. Zhang & G. Hu (2016). "Wrinkling of the membrane with square rigid elements". *Europhys. Lett.*, **116**(2):1–7.

[650] D. Yan, K. Zhang, F. Peng & G. Hu (2014). "Tailoring the Wrinkle Pattern of a Microstructured Membrane". *Appl. Phys. Lett.*, **105**(7):071 905.

[651] Y. Yang, H. H. Dai, F. Xu & M. Potier-Ferry (2018). "Pattern Transitions in a Soft Cylindrical Shell". *Phys. Rev. Lett.*, **120**(21):215 503.

[652] Y. Yang, C. Fu & F. Xu (2020). “A Finite Strain Model Predicts Oblique Wrinkles in Stretched Anisotropic Films”. *Int. J. Eng. Sci.*, **155**:103 354.

[653] L. Ying & D. Zorin (2001). “Nonmanifold Subdivision”. In *Proceedings Visualization, 2001. VIS ’01.*, 325–569.

[654] X. Yuan & W. Ma (2014). “Mapped B-spline Basis Functions for Shape Design and Isogeometric Analysis over an Arbitrary Parameterization”. *Comput. Methods Appl. Mech. Eng.*, **269**:87–107.

[655] Y. Yuan & Y. Xing (2022). “A Separation-of-Variable Method for the Wrinkling Problems of Orthotropic Rectangular Stretched Sheets”. *Compos. Struct.*, **300**:116 104.

[656] Q. Zhang, M. Sabin & F. Cirak (2018). “Subdivision surfaces with isogeometric analysis adapted refinement weights”. *Comput. Aided Des.*, **102**:104–114.

[657] H. Zhao, X. Liu, A. H. Fletcher, R. Xiang, J. T. Hwang & D. Kamensky (2022). “An Open-Source Framework for Coupling Non-Matching Isogeometric Shells with Application to Aerospace Structures”. *Comput. Struct.*, **111**:109–123.

[658] L. Zheng (2008). “Wrinkling of Dielectric Elastomer Membranes”. Ph.D. thesis, California Institute of Technology.

[659] F. Zhi-Fang & H. Jimin (2001). *Modal Analysis.* Butterworth-Heinemann. ISBN 0-7506-5079-6.

[660] Z. Zhou & D. Murray (1995). “An incremental solution technique for unstable equilibrium paths of shell structures”. *Comput. Struct.*, **55**(5):749–759.

[661] J. Zhu, X. Zhang & T. Wierzbicki (2018). “Stretch-Induced Wrinkling of Highly Orthotropic Thin Films”. *Int. J. Solids Struct.*, **139–140**:238–249.

[662] R. Ziegler, W. Wagner & K.-U. Bletzinger (2003). “A Finite Element Model for the Analysis of Wrinkled Membrane Structures”. *Int. J. Space Struct.*, **18**(1):1–14.

[663] Z. Zou, T. J. Hughes, M. A. Scott, R. A. Sauer & E. J. Savitha (2021). “Galerkin Formulations of Isogeometric Shell Analysis: Alleviating Locking with Greville Quadratures and Higher-Order Elements”. *Comput. Methods Appl. Mech. Eng.*, **380**:113 757.

[664] Z. Zou, M. A. Scott, M. J. Borden, D. C. Thomas, W. Dornisch & E. Brivadis (2018). “Isogeometric Bézier Dual Mortaring: Refineable Higher-Order Spline Dual Bases and Weakly Continuous Geometry”. *Comput. Methods Appl. Mech. Eng.*, **333**:497–534.

[665] Z. Zou, M. A. Scott, D. Miao, M. Bischoff, B. Oesterle & W. Dornisch (2020). “An Isogeometric Reissner–Mindlin Shell Element Based on Bézier Dual Basis Functions: Overcoming Locking and Improved Coarse Mesh Accuracy”. *Comput. Methods Appl. Mech. Eng.*, **370**:113 283.

[666] J. Zwar, G. Elber & S. Elgeti (2022). “Shape Optimization for Temperature Regulation in Extrusion Dies Using Microstructures”. *J. Mech. Des.*, **145(**012004).

[667] G. van Zwieten, J. van Zwieten & W. Hoitinga (2022). “Nutils”. Zenodo.

Curriculum Vitæ

Hugo Maarten Verhelst

07-02-1995 Born in Wijchen, The Netherlands.

Education

2007–2013 Voorbereidend Wetenschappelijk Onderwijs (VWO)
Maaswaal College Wijchen

2013–2016 Bachelor of Science in Maritime Engineering (*cum laude*, *honours*)
Technische Universiteit Delft

2016–2019 Master of Science in Maritime Engineering (*cum laude*, *honours*)
Master of Science in Applied Mathematics (*cum laude*)
Technische Universiteit Delft
Thesis: Modelling Wrinkling Behaviour of Large Floating Thin Offshore Structures
Supervisors: Dr. M. Möller
Dr. ir. J.H. Den Besten
Dr. ir. F.J. Vermolen
Prof. Dr. ir. M.L. Kaminski

2019–2024 PhD. Computational Mechanics
Technische Universiteit Delft
Thesis: Isogeometric Analysis of Wrinkling
Promotor: Dr. M. Möller
Co-promotor: Dr. ir. J.H. Den Besten

Academic Experience

11/2023–present Post-doctoral researcher
Dipartimento di Matematica e Informatica "U. Dini"
Università degli Studi di Firenze

11/2023–present Post-doctoral researcher (part-time)
Department of Maritime & Transport Technology
Technische Universiteit Delft

Research Visits

05/2023 – 06/2023	Inria Sophia-Antipolis (4 weeks) Visiting Dr. A. Mantzaflaris
03/2023 – 04/2023	Università degli Studi di Firenze, Italy (5 weeks) Visiting Dr. C. Giannelli
11/2022	Universität der Bundeswehr Munich, Germany (1 week) Visiting Prof. Dr. J. Kiendl
07/2022	Inria Sophia-Antipolis, France (3 weeks) Visiting Dr. A. Mantzaflaris
11/2021 – 12/2019	Inria Sophia-Antipolis, France (5 weeks) Visiting Dr. A. Mantzaflaris
10/2021	Johannes Keppler Universität Linz, Austria (1 week) Visiting P. Weinmüller, A. Farahat and Dr. T. Takacs
10/2019 – 11/2019	Inria Sophia-Antipolis, France (9 weeks) Visiting Dr. A. Mantzaflaris

Conference Contributions

8	Invited talks at international workshops and conferences
15	Contributed talks at international workshops and conferences
4	Poster presentations at international workshops and conferences

Awards

11/2023	Best presentation award in the session "Advanced Discretization Methods" *26th Engineering Mechanics Symposium*
11/2022	Best presentation award in the session "High Performance Computing" *24th Engineering Mechanics Symposium*
09/2021	SIAM Travel Award for the conference "Solid and Physical Modelling 2021" *Society of Industrial and Applied Mathematics (SIAM)*
12/2019	Best MSc. thesis in Maritime Technology *Koninklijke Nederlandse Vereniging voor Technici op Scheepvaartgebied (KNVTS)*

List of Publications

Journal Articles (part of this dissertation)

11. **H. M. Verhelst**, A. Mantzaflaris, M. Möller & J. H. Den Besten, "Isogeometric multi-patch shell analysis using Geometry + Simulation Modules", in preparation.

10. **H. M. Verhelst**, J. H. Den Besten & M. Möller, "Tension-Field Theory for General Hyperelastic Materials in Isogeometric Analysis", in preparation.

9. **H. M. Verhelst**, M. Möller, A. Mantzaflaris & J. H. Den Besten, "Goal-Adaptive Meshing of Isogeometric Kirchhoff–Love Shells", arXiv:2307.08356, under revision for *Engineering with Computers*

8. **H. M. Verhelst**, J. H. Den Besten & M. Möller, "An Adaptive Parallel Arc-Length Method", arXiv:2303.01075, under revision for *Computers & Structures*

7. **H. M. Verhelst**, P. Weinmüller, A. Mantzaflaris, T. Takacs & D. Toshniwal, "A Comparison of Smooth Basis Constructions for Isogeometric Analysis", *Computer Methods in Applied Mechanics and Engineering* **419**, 116659(2024)

6. **H. M. Verhelst**, M. Möller, J. H. Den Besten, A. Mantzaflaris & M.L. Kaminski, "Stretch-Based Hyperelastic Material Formulations for Isogeometric Kirchhoff–Love Shells with Application to Wrinkling", *Computer-Aided Design* **139**, 103075 (2021)

Journal Articles (not part of this dissertation)

5. L. Leonetti, **H. M. Verhelst**, "Effective isogeometric solid-shell model for general hyperelastic materials", under revision for *Computers & Structures*

4. A. Farahat, **H. M. Verhelst**, J. Kiendl & M. Kapl, "Isogeometric analysis for multi-patch structured Kirchhoff–Love shells ", *Computer Methods in Applied Mechanics and Engineering* **411**, 116060(2023)

3. **H. M. Verhelst**, A.W. Stannat & G. Mecacci, "Machine Learning Against Terrorism: How Big Data Collection and Analysis Influences the Privacy-Security Dilemma", *Science and Engineering Ethics* **26**, 2975-2984 (2020)

Conference Proceedings

2. **H. M. Verhelst**, M. Möller, J. H. Den Besten, F. J. Vermolen & M.L. Kaminski, "Equilibrium Path Analysis Including Bifurcations with an Arc-Length Method Avoiding A Priori Perturbations", Numerical Mathematics and Advanced Applications, ENUMATH 2019: European Conference, Egmond aan Zee, The Netherlands, September 30-October 4, 1109-1117 (2019)

Book Chapters

1. Y. Ji, M. Möller & **H. M. Verhelst**, "Design Through Analysis", (accepted for publication)

Acknowledgements

The PhD defence ceremony marks the end of an important step in my career and, most importantly, in my personal development. As a Bachelor student, I got in touch with **em. prof. dr. ir. Mirek Kaminski**, inspiring me to pursue an academic career and mentoring me throughout my studies and PhD to pursue this dream. Mirek, thanks for your trust and support in all these years. You undoubtedly made a very important contribution to reaching this milestone. I started working under the supervision of my co-promotors, **dr. Matthias Möller** and **dr. ir. Henk Den Besten** from my master's thesis onwards. Not only did they provide me with their point of view on my work from different angles, they also made me confident about my work, and they motivated me to pursue my dreams, both professionally and privately. Matthias and Henk, I look back on a beautiful journey together, and I am very grateful for the confidence and support you provided throughout. Soon after starting my PhD, I visited **dr. Angelos Mantzaflaris** for some months. We started our intensive collaboration on the development of the *Geometry + Simulation Modules* (G+Smo). I am super proud of the current state of our work in G+Smo , and I am looking forward to more years of collaboration. Angelos, thanks for all your time, for your patience, and for taking part in my committee. We are not finished yet, and I am very happy with that. I would like to extend my sincere gratitude to the other members of the committee for being there not only for my defence but also at other stages in my education: **prof. dr. Josef Kiendl** and **prof. dr. Alessandro Reali** for the discussions related to isogeometric analysis; **prof. dr. Dominic Vella** for the discussions on structural stability and wrinkling; and **prof. dr. ir. Fred van Keulen** and **prof. dr. ir. Kees Vuik** for educating and for facilitating and managing ecosystems that inspired me throughout several phases of my education.

Throughout my PhD research, I was happily of two great departments within two great faculties at TU Delft. The M&TT department within the (3)ME faculty is where it all started for me as a Bachelor's student in 2013. I would like to thank all my friends from my Bachelor's and Master's at (3)ME, in particular **Hadewich**, **Maarten**, **Maurits**, **Mike**, **Andreas**, **Jurjen**, **Jesper**, and **Egbert** for being by my side through my Bachelor, Master, and PhD. Also the friends from abroad that joined us later, especially **David**, **Luigi**, **Giacomo**. I would also like to express my pride in my friend **Peter-Paul**, whose passion and talent for entrepreneurship were always inspiring to me as an academic. Besides my study friends, I enjoyed the company of my beloved colleagues at M&TT. Many left and many joined in the last year, and I want to thank in particular **Reinier**, **Gabriele**, **Marco**, **Xiaobo**, **Anna**, **Vasu**, **Sietske**, **Wei Jun**, **Annabel** and my great office mates **Peng Peng**, **Fillipo**, **Jacob** and last but definitely not least, **Martin** for all the support, coffee breaks, and distraction. From the other departments in (3)ME, I want to thank all my friends and colleagues for being around, but in particular **Vishal** for all conversations about life, politics and culture.

My second office was at the Numerical Analysis section at DIAM in the EEMCS building of TU Delft. DIAM is a beautiful group, led by **Kees**, for whom I have developed a deep respect in the last years. I want to thank everyone within the NA section for the lovely time, and in particular **Roel** for the political coffee breaks, **Marieke**, **Deepesh**, **Vandana** and **Jochen** for the discussions about life, **Alice**, **Anne**, **Merel** and **Merel** for the nice dinners and game nights, and our new team mates **Jingya**, **Artur**, **Calin**, **Mengyun** and **Gijs** for the nice lunch breaks while finishing this dissertation. Last but not least, thanks to **Ye** and **Leila** for the amazing (yoga) time in the office over the last two years.

Delft is known for its vibrant association culture. Fortunately, this did not stop after obtaining my MSc. degrees. First and foremost, I want to thank **José**, **Jette**, **Iris**, **Pieter** and all other members of the **3mE PhD Council** for every single wonderful meeting in the last three years. Together with **Paul** and **Mascha**, you are doing an incredible job for your colleagues. Furthermore, I want to thank the **Maritime Technology Young Professionals (MTYP)** board for continuing the efforts made to create cohesion within the M&TT department after the COVID-19 crisis, and in particular **Sietske** for initiating this with me and **Bendiks** for believing in it. In addition, I want to thank my fellow board members of the **SIAM Student Chapter Delft**, **Elisa**, and **Kishan** for running this board with me through the COVID-19 crisis. I also had the opportunity to work together with a few master students. **Eline**, **Sam**, **Cas**, **Douwe**: Thanks for the trust you gave me to supervise your master theses. I learned a lot from you; both personally and professionally.

Beyond Delft, I have had the opportunity to work with many colleagues and friends on our projects. First and foremost, I had a great time in Sophia-Antipolis, France, together with **Lorenzo**, **Michelangelo**, **Christos**, **Rima**, **Tobias**, **Pablo**, **Bernard** and **Laurent** and the friends from **Balfolk Antibes**. Furthermore, my visit to Linz, Austria, was the beginning of good collaborations and friendships with **Pascal**, **Andrea**, **Thomas**, and **Mario**. Furthermore, I am grateful to all the other friends and colleagues around the world: **Carlotta** and **Cesare** in Florence, **Leonardo** in Calabria, **Michael** and **Daobo** in München, **Clemens**, **Harald** and **Robin** in Eindhoven, and all the others.

I am grateful for the support I have received from my friends in my hometown, Wijchen. **Joris**, you have been a big pillar of support for me in the last years. **Jeroen**, **Jeroen**, **Martijn**, and **Joris**, thanks for the great spontaneous nights out all over the country.

Ook ben ik dankbaar voor de oneindige steun van mijn familie: Van mijn geweldige opa's, oma's, ooms, tantes, neven, nichten en andere familieleden en -vrienden, maar bovenal wil ik mijn ouders **Bas** en **Liduine** bedanken. Voor de steun, toeverlaat en vrijheden die jullie mij en geweldige, talentvolle broers **Lucas** en **Kamiel** hebben gegeven, en het fijne gezin wat wij kunnen zijn samen met hun partners **Elise** en **Sammy**. Allemaal op jullie eigen manier zijn jullie als een rots in de branding geweest, waar ik mijn frustratie, enthousiasme en alles ertussenin kon luchten. Dit is bewust en onbewust een ongelofelijk belangrijke factor geweest in de totstandkoming van dit proefschift en de rest van mijn carrière. Vele malen dank hiervoor.

Ultima ma non per questo meno importante: **Sofia**, sono incredibilmente felice di averti al mio fianco in questo viaggio. Il nostro amore mi ha regalato stabilità, sostegno, una famiglia meravigliosa e amici affettuosi, conversazioni interessanti, molte ore trascorse insieme in treno, e cibi deliziosi oltre ogni immaginazione. Sono estremamente orgoglioso di te e non vedo l'ora di condividere tutto il tempo futuro che avremo insieme.