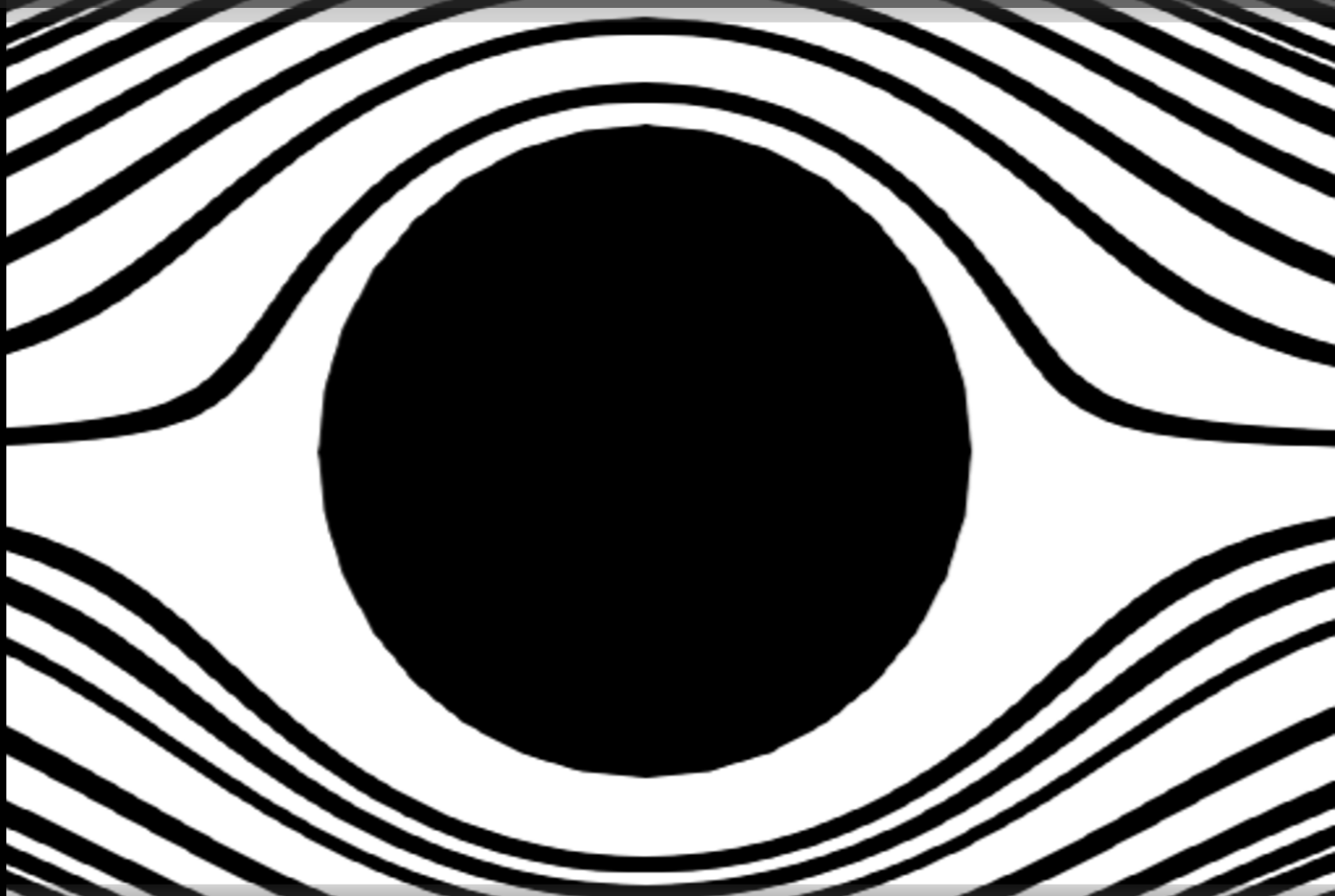
# Immersed Divergence-Conforming Finite Element Spaces

Master Thesis

Ferhat Sindy



Delft University of Technology

TU Delft

# Immersed Divergence-Conforming Finite Element Spaces

## Master Thesis

by

Ferhat Sindy

Studentnumber: 4751957

| | |
|---|---|
| Supervisor: | prof. dr. ir. C. Vuik |
| Daily Supervisor: | dr. ir. D. Toshniwal |
| Project Duration: | October, 2022 - September, 2023 |
| Faculty: | Faculty of Electrical Engineering, Mathematics and Computer Science |

Cover: Stream lines of fluid flow around a cylinder (Modified)

TU Delft

# Preface

This thesis marks the end of my time at university. I have had an amazing five years at TUDelft and a standout year at TUBerlin. None of this would have been possible without the support of others.

I want to begin by thanking my supervisors, Prof. Dr. Kees Vuik and Dr. Deepesh Toshniwal. Specifically, I'm grateful to Deepesh for giving me the opportunity to work on this topic and for our insightful biweekly meetings. I also enjoyed the boulder sessions we had outside of our meetings.

Over these six years, I have made some wonderful friends starting from my first year till now, and I want to express my thanks to all of them. In particular, I would like to thank Uki, Karan, Haoyang, Rohit, George, Ferran, Vova, Henry, Gijs, and Dewwret.

Lastly, but most importantly, I want to show my gratitude to my parents, my brothers (especially Hemin, with whom I shared a studio in Delft), and my little sister.

*Ferhat Sindy*
*Delft, August 2023*

# Summary

Computational Fluid Dynamics (CFD) offers numerous benefits, notably the ability to study flows that are challenging or costly to investigate using experiments. A central challenge in CFD lies in simulating fluid flow around complex geometries. Additionally, the governing equations follow conservation laws. This thesis aims to establish the foundation for constructing Immersed divergence-conforming finite element spaces that address both challenges. To tackle the first issue, an immersed method is considered. Instead of generating a mesh that conforms to the object where the flow occurs, the approach involves placing the object within a predefined mesh. However, this introduces new challenges, the most significant of which is the ill-conditioning of the associated linear system. The condition number depends on the location of the immersed object and can lead to an extremely large condition number. The second challenge is resolved by discretizing the problem in a way that preserves essential topological and homological structures at a discrete level, utilizing the principles of finite element exterior calculus. Within this thesis, a structure-preserving subcomplex is developed for the de Rham complex in 1D, and its accuracy is validated through numerical experimentation. Optimal convergence is achieved, the discrete inf-sup test is satisfied, and the resulting linear system's conditioning remains unaffected by the placement of the immersed object. However, the constructed structure-preserving subcomplex for the de Rham complex in 2D, known as the immersed divergence-conforming finite element spaces, does not exhibit convergence. For future research, I recommend focusing on simpler vertical and skewed cuts before exploring more intricate immersed shapes. These simpler cuts involve straightforward choices for edge basis functions/ 1-form basis functions that are relevant. Additionally, if the outcomes remain unsatisfactory, considering a global approach instead of a local approach could be worthwhile.

# Contents

# List of Figures

# 1

# Introduction

Computational Fluid Dynamics (CFD) was developed in the early 1970s as an interdisciplinary field that combined mathematics, physics, and computer science to simulate fluid flow [1]. CFD offers many advantages, such as the ability to study flows that would be difficult or costly to examine through experimentation. Moreover it is now widely used in a variety of scientific and engineering areas, such as aircraft and car design, meteorology, oceanography, astrophysics, oil recovery, and architecture . In some cases, CFD is even the only practical way to investigate unknown fields, such as the flow and pressure of a space vehicle during re-entry into Earth's atmosphere. As an example, Figure 1.1 shows a three-dimensional CFD model that was used to predict the pressure and velocity of the re-entry of the Apollo spacecraft, which was developed in the 1960s and consisted of three parts designed for landing astronauts on the moon and returning them safely to Earth. To simulate this, the commercial software package ANSYS was used together with a mesh comprising of around 2.45 million elements.

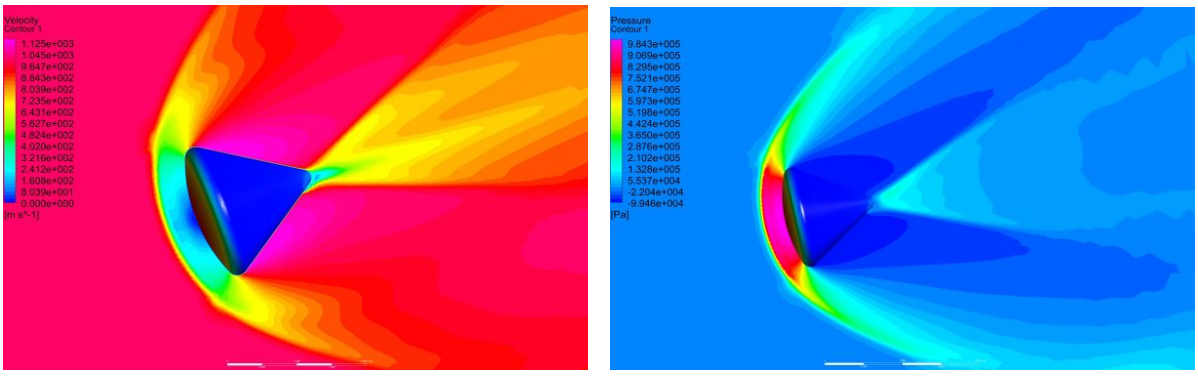



**Figure 1.1:** CFD model of supersonic flow around the Apollo Spacecraft [2][3].

A primary challenge in CFD is the fluid flow simulation around complex geometries. Prior to simulating the fluid flow, the surfaces of all boundaries are meshed and a boundary-fitted volume mesh inside the flow domain is generated. Note that the meshing of the boundaries is an approximation of the initial geometry designed with Computer Aided Design (CAD). There are several meshing options: a structured mesh, an unstructured mesh (sometimes also referred to as mixed mesh) and multi-block versions of both preceding methods. The structured mesh consists entirely of grid cells that are quadrilaterals in 2-D and hexahedra in 3-D and the unstructured method of a mix of quadrilaterals and triangles in 2D and of hexahedra, tetrahedra, prisms and pyramids in 3D. The multi-block methods divides the physical space into a number of topologically simpler parts -blocks - which can be more easily meshed using a structured or unstructured mesh [1]. When we have a geometry that is regular (e.g. rectangle), the mesh is simple to generate: the grid lines follow the coordinate directions. On the other hand for complex geometries the choice is not trivial. Unstructured meshes are the rule rather than the exception when considering complex geometries. However the generation of unstructured meshes is non-trivial,

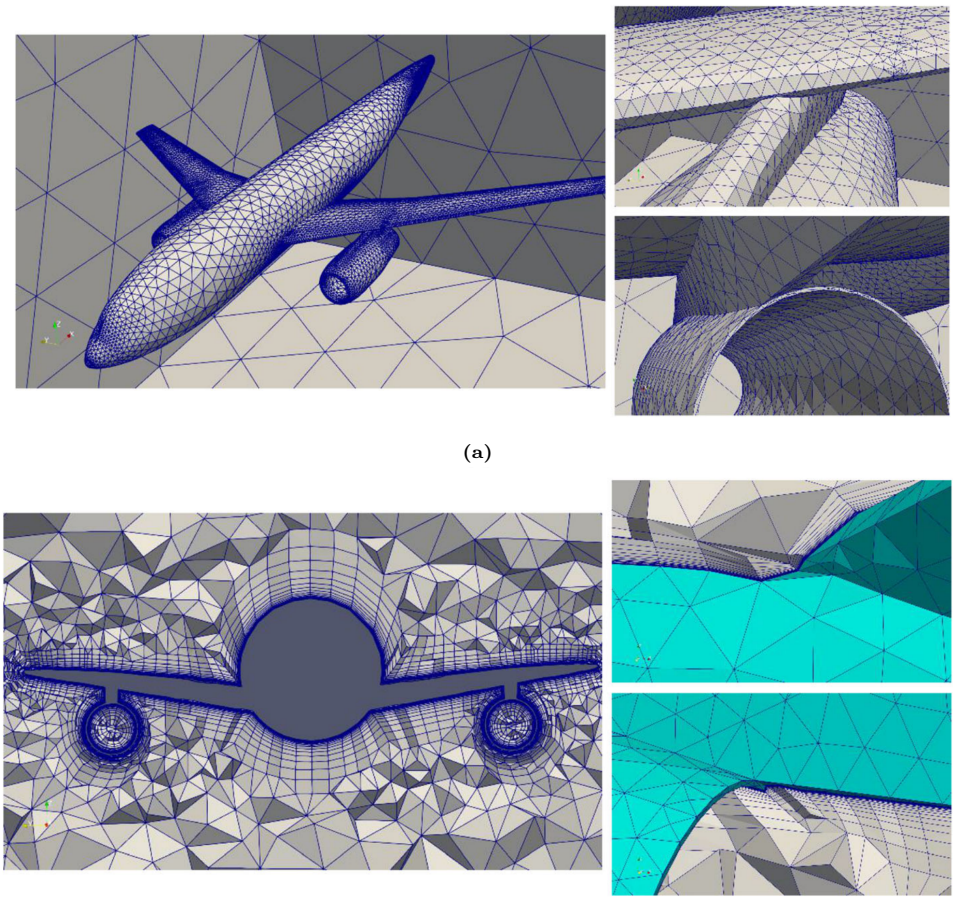


**Figure 1.2:** Mixed mesh of the DLR F6 aircraft model. (a) the surface mesh of the F6 aircraft model, it contains 14,866 nodes and 29,732 triangle elements. The close up shows the complex concave regions and corner nodes. (b) presents a cut view of the hybrid mesh for exterior flow simulations, in which 689,281 prisms, 160,452 tetrahedron and 9441 pyramids are contained. The closeup shows the mesh details around two complex corners. The Figures are adopted from [5].

computationally expensive and it usually consumes the largest amount of the simulation time by far. This is due to the significant manual intervention (e.g. when improving mesh quality, which is assessed by several metrics such as skewness, Smoothness, shapes and sizes of elements but also the optimal balance between the computational cost and the level of fineness etc.). It is normal for a designer to spend several weeks on generating a single grid [4]. As an example, in Figure 1.2 a mixed mesh for the DLR F6 aircraft model is shown. In the close ups several complex corner nodes are shown that create a challenge for the grid generation.

The governing equations that are to be solved are conservation laws (e.g. mass conservation and momentum conservation) therefore the numerical scheme should also satisfy these laws on a local and global basis. This is yet another challenge in CFD. These numerical schemes are referred to as conservative schemes. They ensures that the conservation laws are satisfied at a discrete level so that the error can only be improperly distributed over the solution domain. Non-conservative schemes create artificial sources and sinks, and therefore violating the conservation laws but can still be consistent and stable and consequently give the correct solution in the limit of a very fine grid. The errors due to conservation are only observable on coarse grids. The problem is that it is difficult to know on which grid these errors are small, therefore conservative schemes are preferred [4].
In this master thesis we focus on conservative numerical methods which are able to satisfy physical constraints on a discrete level. More specifically we are interested in flows with constant fluid density (or gases with a Mach number below 0.3), these are incompressible flows. For these flows the mass

conservation equation reduces to the simple expression:

$$\nabla \cdot \mathbf{v} = 0,$$

where $\mathbf{v}$ is the velocity of the fluid.

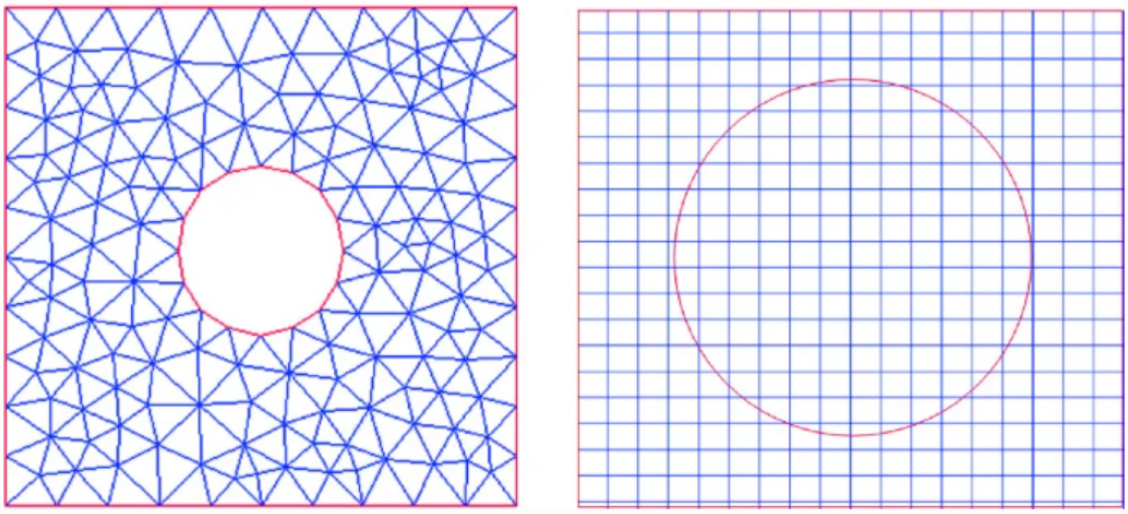


**Figure 1.3:** On the left a bodyfitted mesh is shown that conforms to the boundary of the geometry and on the right an unfitted mesh is shown with an immersed geometry.

To overcome the first challenge the complex geometry is immersed in the mesh rather than the mesh being fitted to the geometry as shown in Figure 1.3. This makes the time consuming and computationally expensive meshing step redundant when simulating fluid flow. Examples of methods that solve the governing equations by using an unfitted mesh are the Finite cell method [6] the Fictitious domain FEM [7][8] and Cut-FEM [9]. The basic idea of these methods is to extend the domain of computation up to the boundaries of an embedding domain, which can be meshed more easily. All of the methods are in principle Galerkin methods, which is also the method of choice in this master thesis. More specifically we choose spline based trial and test spaces. The idea of using spline based spaces originates from the concept of Isogeometric analysis which was introduced by Hughes et al. in [10]. Spline spaces favour higher order inter element continuity compared to standard FEM spaces (such as Lagrange and Hermite polynomial spaces). Moreover they attain the same order of convergence with far less degrees of freedom making them far more efficient. Our method of choice shares the same problems as all the other unfitted methods/immersed methods.

To start with immersed methods require special quadrature rules on the grid cells intersected by the immersed geometry (trimmed elements). There are several possible methods to integrate on such trimmed element. The first one being NURBS-enhanced integration scheme, it requires the intersection points between the mesh and a NURBS representation of the boundary of the immersed geometry. The trimmed element is partitioned into disjoint triangles and NURBS curved sided triangles. For the former standard Quadrature rules are applied and for the latter a special mapping is introduced to distribute the quadrature point appropriately [11]. Moreover the introduced mappings include the functions used for the representation of the geometry, consequently it uses the exact geometrical representation. Another possibility is quadrature free integration scheme. The idea is to successively apply the divergence theorem to transform integrals over boundary representations into line integrals with polynomial integrands. The resulting integrals are solved analytically with machine precision accuracy [12]. Lastly we have the Quadtree/ Octree methods. The general idea is to capture the geometry of a trimmed element by recursively bisecting trimmed sub elements that intersect with the boundary of the domain. At every level of this recursion, sub-cells that are completely inside the domain are preserved, while sub-cells that are completely outside of the domain are discarded [13]. The latter option might be cumbersome since every point needs to be evaluated whether it lies within or outside the immersed geometry. In addition under mesh refinement and when utilizing higher order basis functions a lot of recursive levels

are necessary and consequently a lot of quadrature points are necessary to retain optimal convergence rates. From an implementation point of view we opt for the NURBS-enhanced integration scheme since it reuses the routines implemented for the untrimmed elements.

Another challenge is imposing essential boundary conditions. The boundary of the geometry is not fitted by the mesh therefore it is not clear how to enforce essential boundary conditions. A possible way of enforcing these boundary conditions is by rewriting it as a robin boundary condition with a penalty parameter. By doing this the boundary equation are enforced weakly. This way of enforcing the boundary condition is referred to as the penalty method [14]. The drawback of this method is that the choice of the penalty parameter is challenging. It needs to be a large value to properly enforce the boundary condition and consequently making the condition number of the Galerkin matrix very large. Another possibility is the Lagrange multiplier technique [15], here the boundary conditions are treated as constraints in the minimization problem by introducing a Lagrange multiplier function. The main disadvantage of the Lagrange Multiplier technique is that it requires the user to introduce an appropriate space for the unknown Lagrange function, which must be carefully selected. In this Literature review we use Nitsche's method [16] to enforce the boundary conditions on the immersed geometry. Nitsche's method is similar to the penalty method, it enforces the essential boundary conditions weakly by altering the weak form and introducing a parameter. In contrast to the penalty method the stability parameters are evaluated by solving a local generalized eigenvalue problem at the essential boundary condition thereby allowing for a systematic way of choosing the parameter[17][18].

Moreover all of the mentioned immersed methods have conditioning problems. Standard fitted galerkin methods impose conditions on the shape and sizes of the elements in the considered meshes. For the immersed methods one has no control over the shape and size of the trimmed elements. Therefore trimmed elements can have arbitrarily small intersections with the immersed geometry. This yields ill-conditioned linear systems because basis functions can have an extremely small support on an element in the physical domain[19]. This was already noted by Klaus and Höllig [20][21], they introduced the Weighted Extended B-spline (WEB-spline) spaces. By adjoining the basis functions with small support with the stable basis functions, the condition number of the linear systems become independent of the location of the immersed geometry. However this requires the use of weight functions which for general domains have to be constructed numerically. Since we intend to enforce the essential boundary conditions weakly via the Nitsche's method the weight functions are omitted and the WEB-spline spaces are reduced to Extended B-spline spaces as described in [22].

Finally, we consider the second challenge, that is satisfying the conservation laws on the discrete level. For incompressible fluid flow, divergence-conforming B-splines spaces are a favorable candidate for the trial and test spaces, as introduced by Buffa et al. [23] in the context of Stokes flow in 2D. These spaces produce pointwise divergence-free velocity fields, thereby satisfying the incompressibility constraint. Furthermore, the divergence-conforming B-spline spaces conserve linear and angular momentum, energy, and vorticity. They are a generalization of other well-known finite element spaces, such as the Taylor-Hood, Nédélec, and Raviart-Thomas pairs of finite element spaces. These specific B-spline spaces were derived for three-dimensional vector spaces in [24]. They were inspired by the theory of finite element exterior calculus (FEEC), introduced by Arnold et al. [25], [26], which preserves essential topological and homological structure at a discrete level, thereby producing accurate and stable finite element spaces. However, the divergence-conforming B-spline spaces suffer from ill-conditioning in the context of immersed methods. The research on immersed methods that overcome the conditioning problems and simultaneously satisfy the incompressibility constraints is ongoing. A recent publication is given in the context of cutFEM applied to Darcy flow problems [27] (pre-print), where the Raviart-Thomas space, the Brezzi-Douglas-Marini spaces are employed to approximate the velocity field and a discontinuous space to approximate the pressure. Furthermore, the ghost penalty stabilization method is employed to control the condition number of the resulting linear system matrix, but it destroys the conservative property, which is rectified by introducing yet another stabilization. Another recent approach is in the context of the finite cell method for Stokes problem [28], where again a specific type of finite element spaces (Taylor–Hood, Sub-grid, Raviart–Thomas, and Nédélec elements) are used, but the conditioning problems are left unaddressed and only the convergence properties are evaluated. Lastly, in the context of fictitious domain method applied to Stokes flow [29], linear finite element spaces are used

together with a stabilization. All of the aforementioned attain the correct convergence order and for the first and latter, the condition number of the resulting linear system is independent of the location of the immersed geometry. However, one thing that is shared between the latter methods is that they use a stabilization in combination with specific finite element spaces (e.g. Taylor-Hood, Nédélec, and Raviart-Thomas lower order finite element spaces) as their trial and test spaces.

The goal of this Master thesis is to construct Immersed Divergence-Conforming Finite Element Spaces, such that the resulting condition number of the associated Galerkin matrix is independent of the location of the trimming curve. We do this by answering the following research questions:

> *RQ 1: What extended B-spline spaces form a structure preserving subcomplex of the B-spline spaces satisfying the de Rham complex in one and two dimensional domains?*

and if constructed

> *RQ 2: Is the condition number of the Galerkin matrix associated to the constructed structure preserving finite element spaces irrespective of the location of the trimming curve?*

> *RQ 3: Do the constructed structure preserving finite element spaces satisfy the standard error estimates for elliptic problems?*

To begin with, the theoretical background is introduced in Chapter 2, which concisely introduces B-splines, NURBS, the Immersed FEM, and Structure preserving discretization. Afterward, A structure preserving complex is constructed in for one and two dimensional finite element spaces in chapter 3 and assessed trough numerical experiments in the subsequent Chapter 4. Finally, in Chapter 5 a conclusion is drawn together with a recommendation for future work.

# 2

# Background Theory

In this chapter the theoretical background is concisely introduced. We start with B-splines, where we introduce B-spline basis functions and B-spline curves which form the basis for the next two sections, NURBS and the immersed Finite element method. NURBS are used to define the trimming curve which forms the boundary of the immersed body. Hitherto the geometrical representation of the immersed body is concluded and the immersed finite element method is introduced extensively together with finite element bases utilizing B-spline basis functions and extended B-spline basis functions. To conclude a brief section is devoted to structure preserving discretization.

## 2.1. B-splines

As mentioned before the B-splines form the basis of the next two section. We start by defining the B-spline basis functions together with their properties and derivatives. Thereafter the B-splines curves are defined, the B-spline curves are constructed by linear combinations of B-spline basis functions. For an extensive explanation we refer to the books by Piegl and Tiller [30] and De Boor [31]. To define B-spline basis functions a knotvector, $\Xi$, and an order $p \in \mathbb{N}$, must be given.

**Definition 2.1.1.** *A knotvector is a non-decreasing set of real numbers*

$$\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\},$$

*where $\xi_i \in \mathbb{R}$ is the $i$-th knot and $i$ is the index of the knot, $p$ is the order of the polynomial basis functions and $n$ the number of basis functions.*

Knotvectors are uniform if the knotvalues are equally spaced and non-uniform otherwise. Furthermore Knot values may be repeated, the number of times a knotvalue appears in a knotvector is defined as the multiplicity of the knotvalue, $m_i$. This has implications for the basis functions. Additionally a knotvector is said to be open if the the last and first knotvalue have multiplicity $p+1$.

Given a knotvector $\Xi$ and an order $p \in \mathbb{N}$. The $i$-th basisfunction ($1 \leq i \leq n$) of degree $p$, $N_{i,p}(\xi)$, is defined recursively due to De Boor, Cox, and Mansfield [32] [33] [31]:

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise} \end{cases}, \tag{2.1}$$

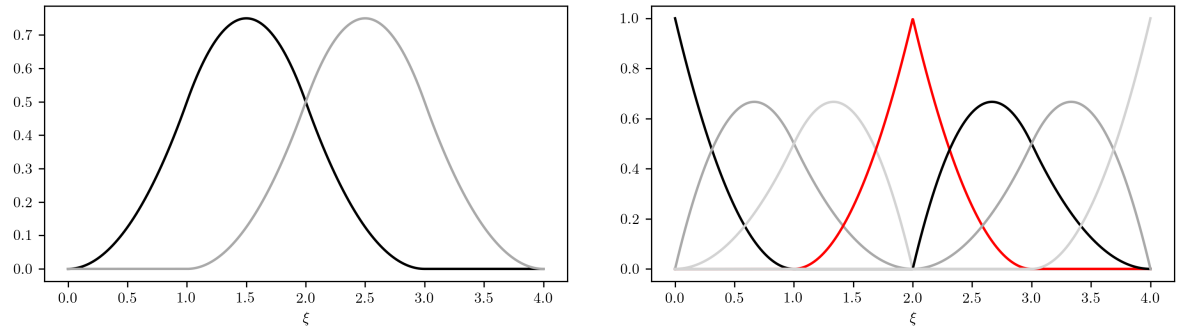
$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi), \quad p = 1, 2, \ldots. \tag{2.2}$$

The B-spline basis functions with uniform knotvector of order $p$ have the following important properties:

1. Homogeneous, The basis functions have the same shape but are translated.
2. Partition of union, $\sum_{i=1}^{n} N_{i,p}(\xi) = 1, \quad \forall \xi \in [\xi_1, \xi_{n+p+1}]$.

3. Non-negativity, $N_{i,p}(\xi) \geq 0, \quad \forall \xi \in [\xi_1, \xi_{n+p+1}]$.
4. $N_{i,p}$ has $p-1$ continuous derivatives across the knot values. (Note: all derivatives exist on the interior of each knot interval.)
5. Local support property, the support of $N_{i,p}$ is $p+1$ knot spans. More specifically it is nonzero on $[\xi_i, \xi_{i+p+1}]$.
6. Linear independence. Due to this property every piecewise polynomial $f_{p,\Xi}$ of degree p over a knot sequence $\Xi$ can be uniquely described by a linear combinations of $N_{i,p}$. Therefore they form a basis on the spline space $\mathbb{S}_{p,\Xi}$, collecting all functions of the form $\mathbb{S}_{p,\Xi} = \sum_{i=1}^{n} N_{i,p} c_i, \quad c_i \in \mathbb{R}$ [34].

The proofs of the first five properties can be found in [30, p. 55]. It is important to note that the first property does not hold for non-uniform knot vectors. Additionally, knot values may be repeated in non-uniform knot vectors, which reduces the number of continuous derivatives across knot values. In general, basis functions of order $p$ have $p - m_i$ continuous derivatives across the knot value $\xi_i$. When $m_i = p$, the basis function becomes interpolatory at the knot value $\xi_i$, and when $m_i = p+1$, the basis function becomes discontinuous. Figure 2.1 shows the basis functions for a uniform and non-uniform knot vector. An important difference between them is the number of basis functions and the continuity across the knot values. It is evident that for the non-uniform knot vector, the nonzero basis function at knot value 2 is $C^0$.




**(a)** **(b)**

**Figure 2.1:** Comparison between B-spline basis functions of order $p = 2$ with different type of knotvectors. (a) B-spline basis functions with the uniform knotvector $\Xi = \{0, 1, 2, 3, 4\}$, (b) B-spline basis functions with the non-uniform knotvector $\Xi = \{0, 0, 0, 1, 2, 2, 3, 4, 4, 4\}$.

Another important remark is that it seems that the local support of some basis functions has reduced for the non-uniform knotvector, but note that the local support is given by $[\xi_i, \xi_{i+p+1})$ and that knot values may be repeated. The derivatives of a B-spline basis function are given by equation (2.3), which can be derived by induction from equation (2.1) (see [30, p. 59]).

$$\frac{d^k}{d^k\xi} N_{i,p}(\xi) = \frac{p}{\xi_{i+p} - \xi_i} \left( \frac{d^{k-1}}{d^{k-1}\xi} N_{i,p-1}(\xi) \right) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} \left( \frac{d^{k-1}}{d^{k-1}\xi} N_{i+1,p-1}(\xi) \right). \tag{2.3}$$

Note that the derivative is zero for $k > p$. Finally we want to make a remark regarding the evaluation of the B-spline basis functions. The evaluation of a basisfunction $N_{i,p}(\xi)$ or its derivative at a point $\tilde{\xi}$ using the recursive equations (2.1) and (2.3) is computationally time inefficient [35], since it requires the evaluation of a truncated triangular table. A lot of these evaluations are redundant due to the local support property of the basis functions. Fortunately there are more time efficient algorithms exploiting this property to evaluate the basisfunction and its derivatives, for instance Algorithms A2.2 [30, p. 70] and A2.3 [30, p. 72]). The B-splines basis functions can be used to define curves.

**Definition 2.1.2.** *A p-th degree B-spline curve is defined by*

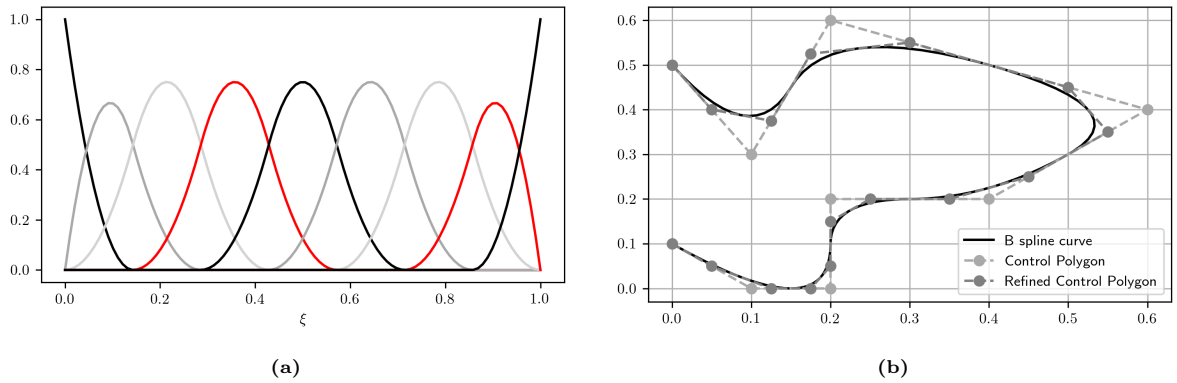
$$\mathbf{C}(u) = \sum_{i=1}^{n} N_{i,p}(u) \mathbf{P}_i, \quad a \leq u \leq b,$$

*where* $\mathbf{P}_i$ *are the control points, and* $N_{i,p}$ *B-spline basis functions of order* $p$ *with open knotvector* $U = \{\underbrace{a, \ldots, a}_{p+1}, u_{p+1}, \ldots, u_n, \underbrace{b, \ldots, b}_{p+1}\}$

The polygon formed by the control points $\{\mathbf{P}_i\}$ is called the control polygon. The properties of the B-spline curve are derived from the properties of the B-spline basis functions. The most important properties are:

1. if $n = p$ and $U = \{a, \ldots, a, b, \ldots, b\}$ then $\mathbf{C}(u)$ is a Bézier curve (which is a different way of representing curves and a special case of B-spline curves).
2. $\mathbf{C}(u)$ is a piecewise polynomial curve, since $N_{i,p}(u)$ are piecewise polynomials.
3. Endpoint interpolation: $\mathbf{C}(\mathbf{0}) = \mathbf{P}_0$ and $\mathbf{C}(1) = \mathbf{P}_n.$, because the knotvector is assumed to be open.
4. Strong convex hull property: the curve is contained in the convex hull of its control polygon. More specifically, if $u \in [u_i, u_{i+1})$ such that $p \leq i < n - p$ then $C(u)$ is in the convex hull of the control points $\mathbf{P_{i-p}}, \ldots, \mathbf{P_i}$.
5. The control polygon represents a piecewise linear approximation of the curve. Additionally the approximation is improved by knotrefinement (see Figure 2.2).
6. The continuity and differentiability of the curve follows from the basis functions, therefore the curve is infinitely often differentiable on the interior of the knots and at least $p - m_i$ times differentiable at the i-th knotvalue, where $m_i$ is the multiplicity.

In Figure 2.2 a B-spline curve with its control polygon and basis functions is shown. In addition the control polygon for a knot refinement is shown and shows an improved linear approximation to the curve.




**(a)** **(b)**

**Figure 2.2:** A B-spline curve with its corresponding basis functions. (a) The basis functions with knotvector $U = \{0, 0, 0, 1/7, 2/7, \ldots, 6/7, 1, 1, 1\}$ of order $p = 2$. (b) The B-spline curve with control points $\mathbf{P}_1$ = [0,0.1], $\mathbf{P}_2$ = [0.1,0], $\mathbf{P}_3$ = [0.2,0], $\mathbf{P}_4$ = [0.2,0.2], $\mathbf{P}_5$ = [0.4,0.2], $\mathbf{P}_6$ = [0.6,0.4], $\mathbf{P}_7$ = [0.2,0.6], $\mathbf{P}_8$ = [0.1,0.3], $\mathbf{P}_9$ = [0,0.5]. In addition the control polygon of a knotrefinement is shown to illustrate the piecewise approximation.

The derivatives of a B-spline curve are defined straightforwardly since it is a linear combination of control points and B-spline basis functions. The k-th derivative of a B-spline curve is determined by computing the k-th derivative of the B-spline basis functions:

**Definition 2.1.3.** *Let* $\mathbf{C^{(k)}}(u)$ *denote the k-th derivative of* $\mathbf{C}(u)$ *then*

$$\mathbf{C^{(k)}}(u) = \sum_{i=1}^{n} N_{i,p}^{(k)}(u)\mathbf{P}_i, \quad a \leq u \leq b.$$

To compute a point on the B-spline curve or its derivatives the B-spline basis functions need to be evaluated . As mentioned earlier in this subsection evaluating the B-spline basis function is computationally inefficient, therefore computational efficient algortihm exist to evaluate a point on the B-spline curve and its derivatives, see for instance Algorithm A3.1 [30, p. 82] and Algorithm A3.2 [30, p. 92].

## 2.2. NURBS

NURBS are an acronym for Non-Uniform Rational B-Splines, they are the industry standard for representation, design and data exchange of geometric information processed by computers. NURBS are successful because of the following reasons:

1. NURBS algorithms are fast and numerically stable.
2. NURBS can represent a variety of things: analytic shapes, such as conic sections and quadratic surfaces, as well as free-form entities, such as car bodies and ship hulls.
3. NURBS curves and surfaces are invarient under common geometric transformations.

All of these reasons have made NURBS popular in CAD [30]. We start this section by defining NURBS and give their properties together with their derivatives and conclude by illustrating an important application of knot insertion. Again we refer to the book by Piegl and Tiller [30] for a more detailed explanation.

**Definition 2.2.1.** *A p-th degree NURBS curve is defined by*

$$\mathbf{C}(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) w_i \mathbf{P_i}}{\sum_{i=0}^{n} N_{i,p}(u) w_i}, \quad a \leq u \leq b,$$

*where* $\mathbf{P}_i$ *are the control points,* $\{w_i > 0\}$ *the weights, and* $N_{i,p}(u)$ *the B-spline basis functions of order* $p$ *with open knotvector* $U = \{\underbrace{a, \ldots, a}_{p+1}, u_{p+1}, \ldots, u_n, \underbrace{b, \ldots, b}_{p+1}\}$

Next we define the rational basis functions:

$$R_{i,p} = \frac{N_{i,p}(u) w_i}{\sum_{j=1}^{n} N_{j,p}(u) w_j} \tag{2.4}$$

this allows for a compact notation of a NURBS curve

$$\mathbf{C}(u) = \sum_{i=1}^{n} R_{i,p}(u) P_i \tag{2.5}$$

The rational basis functions, $R_{i,p}$ have the following important properties, which are mostly derived from the properties of the B-spline basis functions:

1. Nonnegativity, $R_{i,p}(u) \geq 0, \forall i \in [1, \ldots, n], p$ and $u \in [u_1, u_{n+p+1}]$
2. Partition of union, $\forall u \in \sum_{i=1}^{n} R_{i,p}(u) = 1$.
3. Local support property, the support of $R_{i,p}(u)$ is $p+1$ knot spans. More specifically it is nonzero on $[u_i, u_{i+p+1}]$.
4. All derivatives of $R_{i,p}$ exist on the interior of each knot interval, where the rational function has a nonzero denominator and $p - m_i$ continuous derivatives across the knots.
5. if $w_i = 1 \quad \forall i \in 1, \ldots, n$ then $R_{i,p} = N_{i,p}, \quad \forall i \in 1, \ldots, n$.

These properties of the rational basis functions give the following geometrical properties for the NURBS curves:

1. Endpoint interpolation: $\mathbf{C}(\mathbf{0}) = \mathbf{P}_0$ and $\mathbf{C}(1) = \mathbf{P}_n$., because the knotvector is assumed to be open.
2. Strong convex hull property: the curve is contained in the convex hull of its control polygon. More specifically, if $u \in [u_i, u_{i+1})$ such that $p \leq i < n - p$ then $C(u)$ is in the convex hull of the control points $\mathbf{P_{i-p}}, \ldots, \mathbf{P_i}$.
3. The control polygon represents a piecewise linear approximation of the curve.
4. The continuity and differentiability of the curve follows from the basis functions, therefore the curve is infinitely often differentiable on the interior of the knots, where the rational basis functions are nonzero and at least $p - m_i$ times differentiable at the i-th knotvalue, where $m_i$ is the multiplicity.

5. A NURBS curve without interior knots is a rational Bézier curve.

The derivatives of a NURBS curve are more complicated than than the derivatives of a B-spline curves, since we have to work with rational functions. Therefore we want to express the derivatives of the NURBS curves as the derivatives of the B-spline curves. To this end, we define the weighted control points $P_i^w = (w_i x_i, w_i y_i, w_i z_i, w_i)$, where $x_i, y_i, z_i$ are the spatial components of the coordinates of the control points. Then define the non-rational B-spline curve in four dimensional space:

$$\mathbf{C^w}(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{P_i^w}$$

Furthermore the map $H$ is introduced:

$$\mathbf{P} = H\left\{\mathbf{P}^w\right\} = H\{(X,Y,Z,W)\} = \begin{cases} \left(\frac{X}{W}, \frac{Y}{W}, \frac{Z}{W}\right) & \text{if } W \neq 0 \\ \text{direction } (X,Y,Z) & \text{if } W = 0 \end{cases} \tag{2.6}$$

The map H applied to $\mathbf{C^w}(u)$ gives the NURBS curve $\mathbf{C}(u)$.

**Definition 2.2.2.** *Let* $\mathbf{C}(u)^{(k)}$ *denote the k-th derivative of* $\mathbf{C}(u)$ *then*

$$\mathbf{C}^{(k)}(u) = \frac{\mathbf{A}^{(k)}(u) - \sum_{i=1}^{k} \binom{k}{i} w^{(i)}(u)\mathbf{C}^{(k-i)}(u)}{w(u)},$$

*where* $\mathbf{A}(u)$ *is the vector-valued function whose coordinates are the first three coordinates of* $\mathbf{C^w}(u)$ *and* $w(u)$ *the last coordinate of* $\mathbf{C^w}(u)$.

The k-th derivative of the NURBS curve $\mathbf{C}^{(k)}(u)$ is expressed in terms of $\mathbf{A}^{(k)}(u)$ and $k-1$ derivatives of $w(u)$, which are determined by taking the derivative of $\mathbf{C^w}(u)$ using Definition 2.1.3, since $\mathbf{C^w}(u)$ is a non rational B-spline curve. The derivatives allow for the evaluations of the normal and tangent vectors. In Figure 2.3 a circle is approximated using a NURBS curve. The circle is constructed using four 90° arcs in each quadrant, this is apparent from the knotvector which is given by $U = \{0,0,0,1/4,1/4,1/2,1/2,3/4,3/4,1,1,1\}$ with order $p = 2$. At each double knot the basis functions are $C^0$ continuous and hence the curve as well. The points that are $C^0$ continuous are called cusps. That happens at every point where the 90° arcs meet.

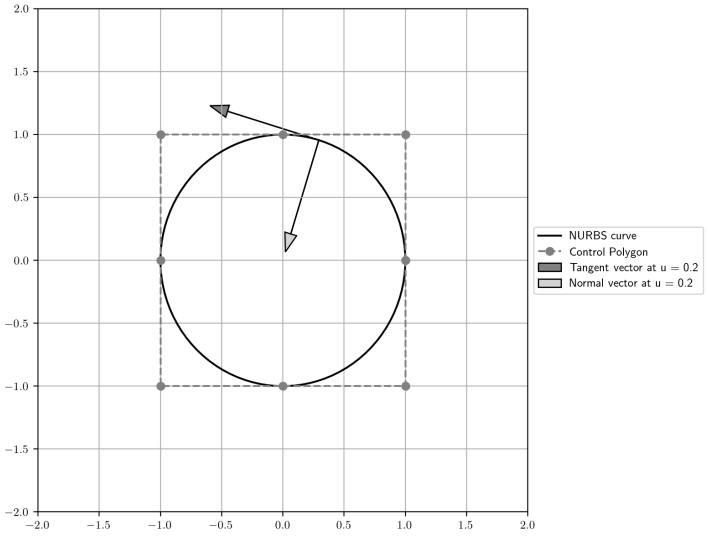



**Figure 2.3:** A NURBS curve representing a circle with knotvector $U = \{0,0,0,1/4,1/4,1/2,1/2,3/4,3/4,1,1,1\}$ of order $p = 2$, control points $\{\mathbf{P}_1\} = \{[1,0],[1,1],[0,1],[-1,1],[-1,0],[-1,-1],[0,-1],[1,-1],[1,0]\}$ and weights $\{\mathbf{w_i}\} = \{1,\sqrt{2}/2,1,\sqrt{2}/2,1,\sqrt{2}/2,1\sqrt{2}/2,1\}$. In addition the normal and tangential vectors at the point $u = 0.2$ are shown by evaluating the derivative of the NURBS curve.

In addition the tangential vector is evaluated at $u = 0.2$, which is equal to $\mathbf{C}^{(1)}(\mathbf{0.2})$, and normal vector are shown.

A fundamental tool when considering NURBS curves ( and B-spline curves ) is knot insertion. The problem statement is as follows: Let $\mathbf{C^w}(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{P_i^w}$ be a NURBS curve with knotvector $U = \{u_0, \ldots, u_m\}$. Let $\bar{u} \in [u_k, u_{k+1})$, and insert $\bar{u}$ in $U$ to form a new knotvector $\bar{U} = \{\bar{u}_0 = u_0, \ldots, \bar{u}_k = u_k, \bar{u}_{k+1} = \bar{u}, \bar{u}_{k+2} = u_{k+1}, \ldots, \bar{u}_{m+1} = u_m\}$. The curve $\mathbf{C^w}(u)$ is given by

$$\mathbf{C^w}(u) = \sum_{i=1}^{n+1} \bar{N}_{i,p}(u)\mathbf{Q_i^w},$$

where the basis functions $\bar{N}_{i,p}(u)$ are known and determined by equation (2.1) using the known order and knotvector $\bar{U}$, therefore knot insertion boils down to determining the controlpoints $\{\mathbf{Q_i^w}\}$. To determine these controlpoints we refer to Algorithm A5.1 in [30, p.151]. An important application of knot insertion is curve splitting. Consider the curve in Figure 2.3, where $p = 2$ and $U = \{0, 0, 0, 1/4, 1/4, 1/2, 1/2, 3/4, 3/4, 1, 1, 1\}$. We insert the knot $\bar{u} = 1/3$ and $\bar{u} = 2/3$ three times. This splits the curve at the both points and as a result a new splitted NURBS curve is obtained as shown in Figure 2.4. In general suppose $\bar{u} \in [u_k, u_{k+1})$ initially has multiplicity $s$ then the curve is splitted if $p - s + 1$ knots are inserted.

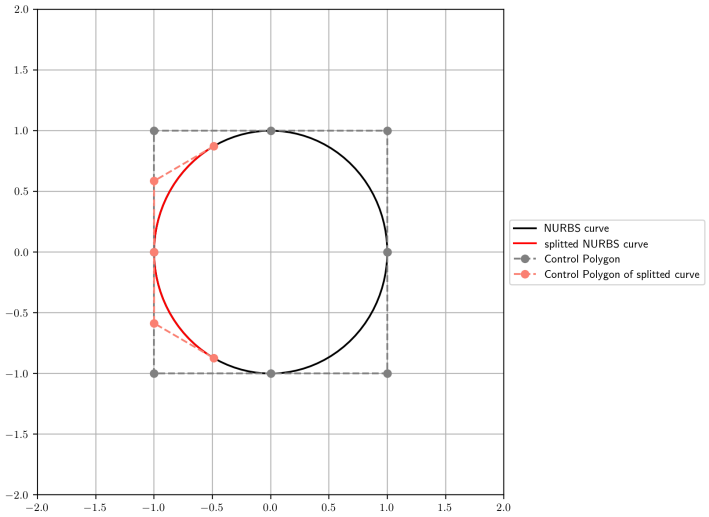



**Figure 2.4:** The NURBS curve representing a circle and a splitted NURBS curve obtained by $p + 1$ knot insertions at $u = 1/3$ and $u = 2/3$.

## 2.3. Immersed Finite Element Method

The finite element method (FEM) is a widely used numerical method to solve partial differential equations that emerge in engineering and physics problems. There exist several free and commercial software packages like FEniCS, COMSOL Multiphysics, ANSYS and many more. The main idea of the finite element method is to divide the domain in smaller subdomains, on which we define functions to expand the exact solution with. Moreover before the problem is actually solved the domain is meshed as a pre-processing step. This is the most time consuming part and is actually a piecewise polynomial approximation of the domain. In this section the geometry is immersed in the mesh rather than the mesh being fitted to the geometry as shown in Figure 1.3. This makes the time consuming and computationally expensive meshing step redundant when simulating. Additionally we consider B-spline basis functions as introduced in section 2.1 instead of the standard the FEM-approximation such as the Lagrange ($C^0$-continuous) and Hermite polynomials ($C^1$-continuous), which have a low inter-element continuity. B-spline basis functions favour higher inter-element continuity. Moreover they attain the same order of convergence with far less degrees of freedom making them far more efficient. In addition we restrict to a unit square domain, $\Omega = [0, 1] \times [0, 1]$. We start by introducing the Galerkin method to

reduce an infinite dimensional problem to a finite dimensional problem, thereafter we define a finite element bases for the immersed FEM. Moreover imposing essential boundary conditions on the introduced finite element bases require special attention which will be covered in the subsequent section and finally in the last section the numerical integration over trimmed and untrimmed elements is considered. For a more detailed treatment regarding theory and implementation of FEM and the use of B-spline basis functions for FEM we refer to the text-books [36] [10]. The Poisson equation is considered as example differential equation throughout this section:

Find $u : \bar{\Omega} \to \mathbb{R}$ such that

$$\begin{cases} -\Delta u = f & \text{in } \Omega, \\ u = g & \text{on } \Gamma_D, \\ \nabla u \cdot \mathbf{n} = h & \text{on } \Gamma_N, \end{cases} \tag{2.7}$$

where $\Gamma_D \cup \Gamma_N = \Gamma = \partial\Omega$, $\Gamma_D \cap \Gamma_N = \emptyset$ and $\mathbf{n}$ is the unit outward normal vector on $\partial\Omega$. The functions $f : \Omega \to \mathbb{R}$, $g : \Gamma_D \to \mathbb{R}$, $h : \Gamma_N \to \mathbb{R}$ are given. This form of the Poisson problem is referred to as the strong form of the boundary value problem.

### 2.3.1. Galerkin method

Before FEM became popular the terms "Galerkin method" and "FEM" were used interchangeably. While FEM has grown considerably the core of its approach is still in line with the Galerkin method. In short, The Galerkin method is a general strategy to convert a continuous (variational) problem into a finite dimensional system of linear equations. Before the Galerkin method can be applied the problem needs to be reformulated in its weak form. In equation (2.7) the strong form requires the solution $u \in C_0^2(\bar{\Omega})$, whereas the weak form allows the solution to be in a "larger" set making the solution "weaker". In order to rigorously introduce the weak form several definitions are introduced.

**Definition 2.3.1** (Smooth functions with compact support)**.** *Let $\Omega \subset \mathbb{R}^d$ be a domain and define the space of all smooth functions with compact support on $\Omega$*

$$C_c^\infty(\Omega) := \{h : \Omega \to \mathbb{R} \mid h \text{ is infinitely differentiable and } \operatorname{supp}(h) \subset \Omega \text{ is compact } \}$$

**Definition 2.3.2** (Square integrable function)**.** *Let $\Omega \subset \mathbb{R}^d$ be a domain and define the space of all square integrable functions on $\Omega$*

$$L^2(\Omega) = \left\{ v : \Omega \to \mathbb{R} | \int_\Omega |v(x)|^2 dx < \infty \right\},$$

*which is a vector space with the inner product $(u, v)_{L^2(\Omega)} = \int_\Omega u(x)v(x)dx$ and corresponding induced norm $\|u\|_{L^2(\Omega)} = \sqrt{(u,u)_{L^2(\Omega)}}$.*

**Definition 2.3.3** (Weak derivative)**.** *Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with a piecewise smooth boundary. Then $v \in L^2(\Omega)$ is the weak derivative of $u$ of order $\alpha$ if*

$$\int_\Omega v\varphi \mathrm{d}x = (-1)^{|\alpha|} \int_\Omega u D^\alpha \varphi \mathrm{d}x \quad \forall \varphi \in C_c^\infty(\Omega), \text{ where } D^\alpha f = \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \dots \partial x_n^{\alpha_n}}.$$

**Definition 2.3.4** ($k$-th Sobolev space)**.**

$$H^k(\Omega) = \left\{ u : \Omega \to \mathbb{R} | u \in L^2(\Omega), D^\alpha u \in L^2(\Omega) \text{ for } |\alpha| \leq k \right\}.$$

Given the above definitions, let $H^1(\Omega)$ be called the trial space in which we search for a solution and let $H_0^1 = \{w \in H^1(\Omega) | w(x) = 0 \text{ on } \Gamma_D\}$ be called the test space, the space of test functions. Next the Strong problem is multiplied with a test function and integrated over the domain $\Omega$.

$$-\int_\Omega \Delta u \cdot v \ \mathrm{d}x = \int_\Omega f v \ \mathrm{d}x \quad \forall v \in H_0^1.$$

Applying the product rule, $\nabla \cdot (v \nabla u) = \nabla u \cdot \nabla v + v \underbrace{\nabla \cdot (\nabla u)}_{=\Delta u}$, gives

$$\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}x - \int_\Omega \nabla \cdot (v \nabla u) \mathrm{d}x = \int_\Omega f v \, \mathrm{d}x \quad \forall v \in H_0^1.$$

Next we apply the theorem of Gauss to arrive at.

$$\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}x - \int_\Gamma \mathbf{n} \cdot (v \nabla u) \, \mathrm{d}x = \int_\Omega f v \, \mathrm{d}x \quad \forall v \in H_0^1,$$

where $\mathbf{n}$ denotes the outward normal vector. The integral over the boundary can be rewritten as a separate integral over the disjoint boundaries $\Gamma_D$ and $\Gamma_N$. Furthermore note that on $\Gamma_D$, the test functions are zero and therefore the integral on $\Gamma_D$ as well. On $\Gamma_N$, $\nabla u \cdot \mathbf{n} = h$ and the test functions are nonzero resulting in the following weak form

$$\text{Find } u \in H^1(\Omega) \text{ with } u = g \text{ on } \Gamma_D \text{ such that } \underbrace{\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}x}_{:=a(u,v) \text{ bilinear form}} = \underbrace{\int_{\Gamma_N} h v \, \mathrm{d}x + \int_\Omega f v \, \mathrm{d}x}_{:=F(v) \text{ linear form}} \quad \forall v \in H_0^1. \tag{2.8}$$

The Dirichlet boundary conditions are called essential boundary conditions for this problem, since they affect the choice of the underlying space $H_0^1$. The Neumann boundary conditions are called natural boundary conditions, since they do not affect the space $H_0^1$.

Having derived the weak formulation, the Galerkin method can be applied. The method can be summarized as follows:

1. **Get the weak formulation**: Let $V$, $W$ two vector spaces. Then the weak formulation is:

   $$\text{Find } u \in V \text{ such that } a(u, v) = F(v) \quad \forall v \in W.$$

   $V$ is the trial space and $W$ is the test space. In general $W$ and $V$ coincide
2. **Perform a Galerkin dimension reduction**: For each $n \in \mathbb{N}$ let $V_n \subset V$ be a subspace of V with $\dim(V_n) = n$. Then the Galerkin equation is given by:

   $$\text{Find } u_n \in V_n \text{ such that } a\left(u_n, v_n\right) = F\left(v_n\right) \quad \forall v_n \in V_n, \tag{2.9}$$

   which formulates the problem projected on $V_n$.
3. **Derive a linear system of equations**: $V_n$ is a finite dimensional space, therefore there exist a basis $\{\phi_1, \ldots, \phi_n\}$ of $V_n$. Consequently the solution can be written as $u_n = \sum_{i=1}^n \alpha_i \phi_i$. Using the bilinear property of $a(\cdot, \cdot)$ and substituting $u_n$ in the Galerkin equation gives:

   $$\text{Find } \alpha_j \in \mathbb{R} \quad \forall j \in \{1, \ldots, n\} \text{ such that } \sum_{j=1}^n \alpha_j a\left(\phi_j, \phi_i\right) = F\left(\phi_i\right) \quad \forall i \in \{1, \ldots, n\}.$$

   This translates to a matrix vector equation given by:

   $$A_n \alpha = \begin{bmatrix} a\left(\phi_1, \phi_1\right) & \cdots & a\left(\phi_n, \phi_1\right) \\ \vdots & \ddots & \vdots \\ a\left(\phi_1, \phi_n\right) & \cdots & a\left(\phi_n, \phi_n\right) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} F\left(\phi_1\right) \\ \vdots \\ F\left(\phi_n\right) \end{bmatrix} =: f_n \in \mathbb{R}^n.$$

   The matrix $A_n$ is the Galerkin matrix associated to the galerkin problem.

Note that if the discrete test space does not coincide with the discrete trial space, the method is called a Petrov-Galerkin-method. In conclusion a variational problem on infinite dimensional spaces is reduced to solving a linear system. Next the concept of Galerkin orthogonality is introduced. Let $u$ the exact solution and $u_n \in V_n$ the solution of equation (2.9). $V_n \subset V$ therefore

$$a(u_n, v_n) = F(v_n) = a(u, v_n).$$

This gives

$$a(u_n - u, v_n) = 0, \quad \forall v_n \in V_n.$$

That is if $a$ admits an inner product on $V$ then the error is orthogonal to $V_n$ with respect to the inner product induced by a. This means that $u_n$ gives the best approximation of an exact solution $u$ in term of the following norm:

$$\|v\|_a = \sqrt{a(v,v)}.$$

Thus the best approximation property is given by:

$$\|u - u_n\|_a = \inf_{v_n \in V_n} \|u - v_n\|_a .$$

This result allows us to state the standard error estimate for elliptic boundary value problems (for a proof and derivation see [36, p.185])

**Theorem 2.3.1.** *if $u \in H^{k+1}(\Omega)$ then*

$$\|u - u_h\|_{H^s} \leq ch^{\beta}\|u\|_{H^{k+1}},$$

*where $c$ is a constant, independent of $u$ and $h$ (= mesh parameter, it can be chosen as diameter of the largest element), $\beta = \min\{k+1-s, 2(k+1-m)\}$, $k$ is the degree of polynomial of the element shape functions and $m$ is the order of the highest derivatives appearing in the billinear form $a(\cdot,\cdot)$. Note that $H^0(\Omega) = L^2(\Omega)$.*

Lastly There is an estimate for the condition number of the Galerkin system arising from the model problem (2.7). For standard finite element subspaces the condition number with respect to the 2-norm ($\kappa_2(A) = \|A\|_2\|A^{-1}\|_2 \geq 1$) is bounded in terms of the mesh parameter $h$:

$$\kappa_2(A_h) = Ch^{-2}, \tag{2.10}$$

where $C$ is a constant independent of $h$. This concludes the chapter about the Galerkin method. In the next section appropriate finite element spaces are considered.

### 2.3.2. Finite element Bases

Let $\Omega = [0,1] \times [0,1] \subset \mathbb{R}^2$. We construct finite element basis functions on regular grids using the B-splines defined in section 2.1. We start by defining bi-variate B-spline basis functions and state their properties, which are derived from the uni-variate B-spline basis functions. Thereafter a closed trimming curve is introduced which trims the domain $\Omega$ and another finite element basis is constructed, by stabilizing the former truncated finite element bases. Let $p, q \in \mathbb{N}$ and let the knotvectors $\Xi$ and $\mathcal{H}$ be defined as

$$\begin{aligned} \Xi &= \{0 = \xi_1, \xi_2, \ldots, \xi_{n_1+p+1} = 1\}, \\ \mathcal{H} &= \{0 = \eta_1, \eta_2, \ldots, \eta_{n_2+q+1} = 1\}, \end{aligned} \tag{2.11}$$

and open. We assume that all knots have multiplicity $1 \leq r_1 \leq p+1$ in knotvector $\Xi$ and $1 \leq r_2 \leq q+1$ in knotvector $\mathcal{H}$. In addition we introduce the ordered knotvectors

$$\bar{\Xi} = \{\bar{\xi}_1, \ldots, \bar{\xi}_{l_1}\},$$

$$\bar{\mathcal{H}} = \{\bar{\eta}_1, \ldots, \bar{\eta}_{l_2}\},$$

which represents the vector of knots without repetitions, where $l_1 = (n_1 - p - 1)/(r_1 + 2)$ and $l_2 = (n_2 - q - 1)/(r_2 + 2)$. Associated to these knotvectors is a mesh $\Omega_h$ that partitions the domain into rectangles (Grid cells/ elements):

$$\Omega_h = \{Q = (\bar{\xi}_i, \bar{\xi}_{i+1}) \times (\bar{\eta}_j, \bar{\eta}_{j+1}) | 1 \leq i \leq l_1 - 1 \text{ and } 1 \leq i \leq l_2 - 1\}.$$

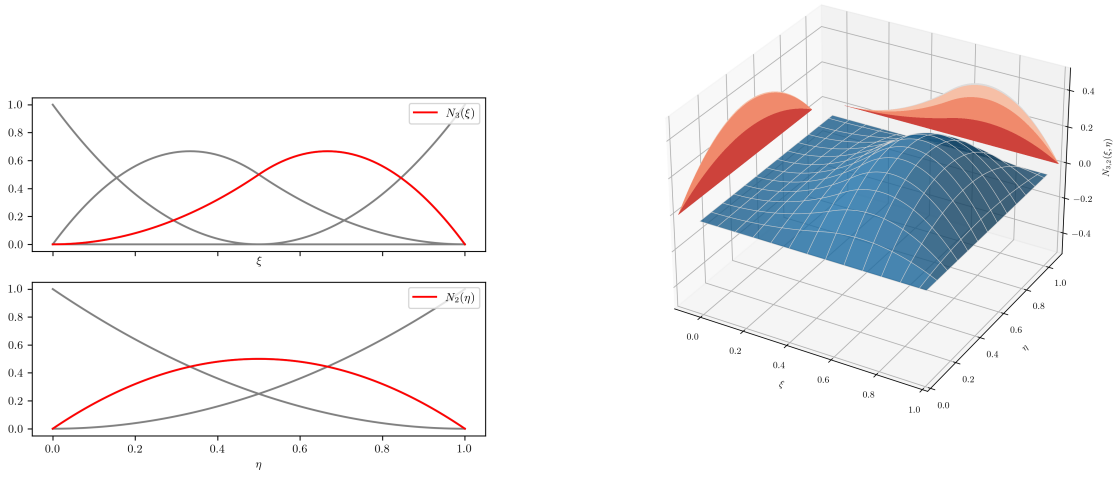
Given an element $Q \in \Omega_h$, $h_Q = \text{diam}(Q)$, and $h = \max\{h_Q, Q \in \Omega_h\}$. The knotvectors $\Xi$ and $\mathcal{H}$ of order $p$ and $q$ respectively give rise to the B-spline basis functions $\{N_i(\xi)\}_{i=1}^{n_1}$ and $\{N_i(\eta)\}_{i=1}^{n_2}$ with $\alpha_1 := p - r_1$ and $\alpha_2 := q - r_2$ continuous derivatives across the knotvalues. As mentioned in section 2.1 the B-spline basis functions are linear independent and therefore their span forms a spline space. We define the spline spaces as $\mathbb{S}^p_{\alpha_1}(\Omega_h) = \text{span}\{N_i(\xi)\}_{i=1}^{n_1}$ and $\mathbb{S}^q_{\alpha_2}(\Omega_h) = \text{span}\{N_i(\eta)\}_{i=1}^{n_2}$. This notation differs from the notation introduced in section 2.1, since we assume knotvectors of the form (2.11) therefore the spline spaces are fully characterized by their order, number of continuous derivatives accros knotvalues and mesh. We define the tensor-product B-spline basis functions as:

$$N_{ij}(\xi, \eta) := N_i(\xi) \otimes N_j(\eta), \quad i = 1, \ldots, n_1, \quad j = 1, \ldots, n_2.$$

Then, the tensor-product B-spline space is defined as the space spanned by these basis functions

$$\mathbb{S}^{p,q}_{\alpha_1,\alpha_2} = \mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h) := \text{span}\left\{N_{ij}\right\}_{i=1,j=1}^{n_1,n_2}.$$

All the properties of the univariate B-spline basis functions stated in section 2.1: positivity, parity of union, non-negativity, local support property and linear independence carry over due to the tensor product construction. Now we consider a specific example. Let $\Xi = \{0, 0, 0, 0.5, 1, 1, 1\}$ of order $p = 2$ and $\mathcal{H} = \{0, 0, 0, 1, 1, 1\}$ of order $q = 2$. Then in Figure 2.5a the uni-variate components of the tensor-product B-spline basis functions in each parametric direction are shown for the domain $\Omega = [0, 1] \times [0, 1]$. In Figure 2.5b the specific basisfunction $N_{3,2}(\xi, \eta)$ with support $[\xi_3, \xi_6] \times [\eta_2, \eta_5] = [0, 1] \times [0, 1]$ is shown. Moreover only two knotspans have positive measure. To clarify this, the concept of index space is introduced. The index space uniquely identifies each knot and discriminates knots with similar knot values. The index space for this specific example is shown in Figure 2.6. The two knotspans with nonzero measure are labeled $\Omega_1$ and $\Omega_2$. In addition the support of the two basis functions $N_{3,2}$ and $N_{1,1}$ are shown, their indices coincides with the lower left corner of their support in index space. Henceforth the coordinate of the lower left corner of the support of a basisfunction in index space is used as a label.




**(a)** **(b)**

**Figure 2.5:** A tensor product B-spline basis for the domain $\Omega = [0, 1] \times [0, 1]$. (a) The basis functions related to the knotvectors $\Xi = \{0, 0, 0, 0.5, 1, 1, 1\}$ of order $p = 2$ and $\mathcal{H} = \{0, 0, 0, 1, 1, 1\}$ of order $q = 2$. The basis functions in red are the uni variate basis functions of the tensor product basis function shown in sub figure (b). (b) The Bi-variate basis function $N_{3,2}(\xi, \eta)$ with its projections on the respective direction $\eta$ and $\xi$.

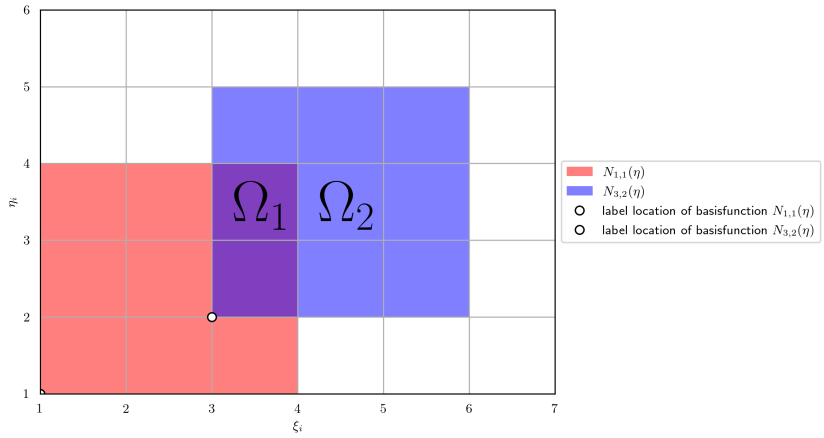



**Figure 2.6:** Index space of the bi-variate basis functions on the knotvectors $\Xi = \{0, 0, 0, 0.5, 1, 1, 1\}$ of order $p = 2$ and $\mathcal{H} = \{0, 0, 0, 1, 1, 1\}$ of order $q = 2$. Additionally the support of the basis functions $N_{1,1}$ and $N_{3,2}$ in respectively red and blue is shown together with their labels in the lower left corner of the support. Moreover the two knot spans with nonzero measure are labeled $\Omega_1$, $\Omega_2$.

Next we introduce a trimming curve $\mathbf{C}(u)$ on the domain $\Omega$ with a mesh $\Omega_h$ induced by the knotvectors $\Xi$ and $\mathcal{H}$ with $r_1 = \alpha_1$, $r_2 = \alpha_2$ and order $p$ and $q$. The trimming curve $\mathbf{C}(u)$, is defined by NURBS as in definition 2.2.1 and we denote the order, knotvector, weights and control points of the trimming curve by $l$, $U$, $w_i$ and $\mathbf{P_i}$ respectively. The trimming curve represents the boundary of a subdomain $D \subset \Omega$ that trims the domain $\Omega$. The model problem 2.7 is adjusted by adding an extra boundary condition on the trimming curve:

$$\begin{cases} \Delta u = f & \text{in } \Omega, \\ u = g & \text{on } \Gamma_D, \\ u = k & \text{on } \partial D, \\ \nabla u \cdot \mathbf{n} = h & \text{on } \Gamma_N, \end{cases} \tag{2.12}$$

Note that now $\Gamma = \Gamma_{D_1} \cup \Gamma_{D_2} \cup \Gamma_N$ and that all boundaries are disjoint. The relevant basis functions of the spline space $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h)$ are those basisfunction which are nonzero for some $x \in \Omega \setminus D$.

**Definition 2.3.5.** *(Spline space on a trimmed domain) Given a spline space $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h)$ on the domain $\Omega$ and a subdomain $D \subset \Omega$ which trims the domain $\Omega$. The spline space on the trimmed domain is defined as*

$$\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h \setminus D) = \{N_{i,j} \in S^{p,q}_{\alpha_1,\alpha_2}(\Omega_h) : N_{i,j}(\tilde{\xi}, \tilde{\eta}) \neq 0 \text{ for some } (\tilde{\xi}, \tilde{\eta}) \in \Omega \setminus D\}$$

*For convenience, let the two dimensional index array $K$ contains all the relevant basis functions. Then $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h \setminus D) = span\{N_k\}_{k \in K}$.*

The space on the trimmed domain can be seen as a restriction of the original space on the untrimmed space. In Figure 2.7 the mesh induced by the space $\mathbb{S}^{2,2}_{1,1}(\Omega_h)$ is shown with equally spaced knot values in both parametric directions together with circles labeling the basis functions. Furthermore the trimming curve from Figure 2.3 is used (after translating and rescaling) to represent the boundary of the trimmed domain $D$, which is a centered circle with radius 0.25. All the relevant basis functions are labeled with a white dot and are in the spline space $\mathbb{S}^{2,2}_{1,1}(\Omega_h \setminus D)$ while the redundant ones are labeled with a black dot. In addition the support of a relevant basis function is shown in red. Its support in $\Omega \setminus D$ is very small, this leads to excessively large condition numbers of the Galerkin matrix (and the estimate 2.10 not being valid anymore). In [19], a specific example shows that the condition number is inversely proportional to the size of the smallest boundary grid cell (more specifically to the power of a function depending on the basis functions' order and the domain's dimension). This leads to an excessively large condition number, increasing the solution's sensitivity to changes in the right-hand side and reducing the accuracy of numerical solutions. Additionally, it causes extremely slow convergence of iterative methods. Note that other factors, such as initial guesses and eigenvalue distribution, also influence

iterative method convergence. Consequently, the restrictive B-spline space is not uniformly stable with respect to the mesh parameter $h$ [20][21].

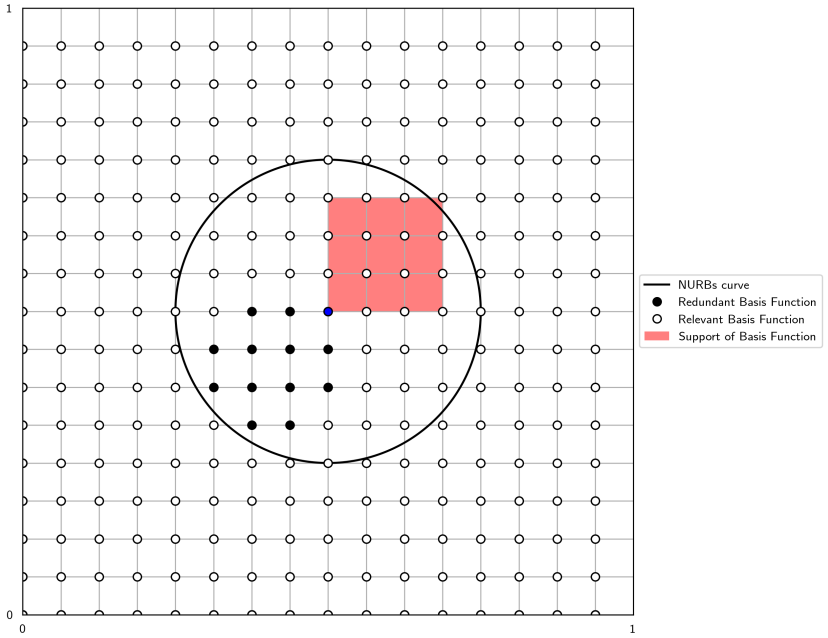



**Figure 2.7:** The domain $\Omega$ with its mesh $\Omega_h$ and a trimming curve defined by a NURBS curve, more specifically the NURBS curve defined in Figure 2.3. The white dots represent the relevant basis functions and the black dots the redundant ones.

To overcome this problem the basis functions with small support are adjoined suitably with the stable basis functions of $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega \setminus D)$. This idea was introduced by Klaus Höllig et al. in [21] and is named extended B-splines. To distinguish between the stable and unstable B-splines the mesh $\Omega_h$ is partitioned into interior, boundary and exterior gridcells and the basis functions are labeled as either inner or outer basis functions depending on the gridcells in their support.

**Definition 2.3.6.** *Given a spline space on a trimmed domain $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h \setminus D)$. A gridcell $Q \in \Omega_h$ is classified as interior, boundary or exterior depending on whether $Q \subset \bar{\Omega} \setminus D$, the interior of $Q$ intersects $\partial D$ or $Q \cap (\Omega \setminus D) = \emptyset$. Then the relevant basisfunction of the space $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h \setminus D)$ on the trimmed domain $\Omega \setminus D$*

$$N_k, \quad k \in K,$$

*are called inner basis functions*

$$N_i, \quad i \in I,$$

*if at least one interior cell is in their support, and outer basisfunction*

$$N_j, \quad j \in J = K \setminus I,$$

*if the support of the basis function consist entirely of boundary and exterior grid cells.*

In Figure 2.8 a continuation of the previous example is considered. The boundary, interior and exterior gridcells are colored dark grey, lights grey and white respectively. In addition the inner basis functions are labeled with a white dot and the outer basis functions with a black dot. The outline of the support of two specific basis functions are shown in red and blue. The red outline contains one inner grid cell and consequently the respective basis function is an inner basis functions. Unlike the blue outline, that only contains boundary and outer gridcells, the corresponding basis functions is an outer basis function.

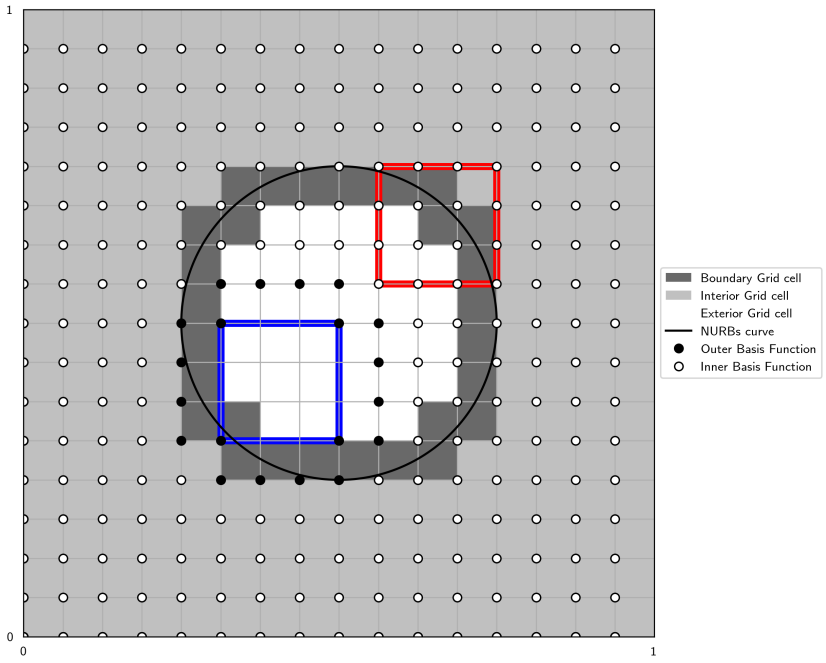



**Figure 2.8:** Gridcell classification. The boundary, interior and exterior gridcells are colored dark grey, light grey and white respectively. In addition the relevant basisfunction of the spline space on a trimmed domain are partitioned into inner (white dot) and outer (black dot) basisfunction. Moreover the support of a specific inner and outer basisfunction is shown in blue and red respectively.

To stabilize the space $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h \setminus D)$, the outer b-splines are connected to the inner b-splines in a suitable way. It is important that all polynomials of coordinate degree $\leq p$ and $\leq q$ remain an element of the resulting stabilized subspace to preserve the approximation power of the space. We consider the Marsden identity

**Theorem 2.3.2** (Marsden Identity)**.** *For a bi-infinite knot sequence $\xi$, any polynomial of degree $\leq n$ can be represented as a linear combination of B-splines. In particular, for any $y \in \mathbb{R}$,*

$$(x-y)^n = \sum_{k\in\mathbb{Z}} \psi^n_{k,\xi}(y) N^n_{k,\xi}(x), x \in \mathbb{R},$$

*where $\psi^n_{k,\xi}(y) = (\xi_{k+1} - y)\cdots(\xi_{k+n} - y)$. (Note the change of notation of the B-spline basis function($N^n_{k,\xi}(x)$). This is to emphasize the order of the basisfunction, which is $n$ and the dependence on the knotvector $\xi$.)*

The Marsden's identity shows how polynomials are represented as linear combinations of B-splines. This identity is formulated for the whole real line but we only consider finite knotvectors. Restricting the Marsden identity to a bounded domain gives the following quantitative result:

**Theorem 2.3.3.** *Any multivariate polynomial $f$ can be represented on the domain $D$ as a linear combination*

$$\sum_{k\in K} g(k) N_k(x), \quad x \in D,$$

*where $g$ is a multivariate polynomial of degree $\leq n$ in each variable $k_v$.*

Lets consider a polynomial $f(\xi,\eta)$ then by the Marsden identity we arrive at:

$$f(\xi,\eta) = \sum_{i\in I} g(i) N_i(\xi,\eta) + \sum_{j\in J} g(j) N_j(\xi,\eta), \quad (\xi,\eta) \in \Omega \setminus D. \tag{2.13}$$

Since $g$ is also a polynomial of coordinate degree $\leq p$ and $\leq q$, we can determine $g(j)$ using any $(p+1)(q+1)$ inner indices $I(j) \subset I$, if the polynomial interpolation problem at the points $I(j)$ is uniquely solvable. We denote the Lagrange polynomials of order $n$ related to $I(j)$ by $l_{j,i}$, then:

$$l_{j,i}(k) = \delta_{i,k}, \quad i,k \in I(j)$$

and for fixed $j$ we define:

$$\begin{cases} e_{i,j} = l_{j,i}(j), & i \in I(j), \\ e_{i,j} = 0, & i \notin I(j). \end{cases}$$

So that $g(j)$ can be written as

$$g(j) = \sum_{i \in I(j)} e_{i,j} g(i)$$

Substituting this in equation 2.13 and interchanging the sum yields

$$f = \sum_{i \in I} g(i) \left[ N_i(x) + \sum_{j \in J(i)} e_{i,j} N_j(x) \right],$$

where $J(i) = \{j \in J : i \in I(j)\}$. The term in the brackets is the correct combination of inner and outer B-splines. Using this linear combination all polynomials of coordinate degree $\leq p$ and $\leq q$ can be represented without referring to the outer b-splines explicitly.

**Definition 2.3.7** (Extended B-splines)**.** *For an outer index $j \in J$ let $I(j) = (\ell_1, \ell_2) + \{0, \dots, p\} \times \{0, \dots, q\} \subset I$ be a 2-dimensional index array of inner indices closest to $j$, assuming that $h$ is small enough so that such an array exists. Moreover, denote by*

$$e_{i,j} = \prod_{\nu=1}^{2} \prod_{\substack{\mu=0 \\ \ell_\nu + \mu \neq i_\nu}}^{n_\nu} \frac{j_\nu - \ell_\nu - \mu}{i_\nu - \ell_\nu - \mu}, \quad \text{where } n_\nu = \begin{cases} p & \text{if } \nu = 1, \\ q & \text{if } \nu = 2. \end{cases},$$

*the values of the Lagrange polynomials associateed with $I(j)$ and $J(i) = \{j \in J : i \in I(j)\}$. Then, the extended B-splines*

$$B_i = N_i + \sum_{j \in J(i)} e_{i,j} N_j, \quad i \in I$$

*form a basis for the extended b-spline space ${}^e\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}$.*

Note that this construction of the extended B-spline space requires uniform knot vectors. This way the construction solely depends on the indices of the B-splines involved. For non-uniform knotvectors the dual functional are required [31] and the simple structure is lost [37] [34]. Note that we are actually not working with nonuniform knotvectors, the knotvalues at the boundary of the domain have a multiplicity bigger than one in both coordinate direction but this does not pose a problem as long as the mesh is fine enough and the trimming curve is not intersecting the boundary of the unit square domain that we are considering.

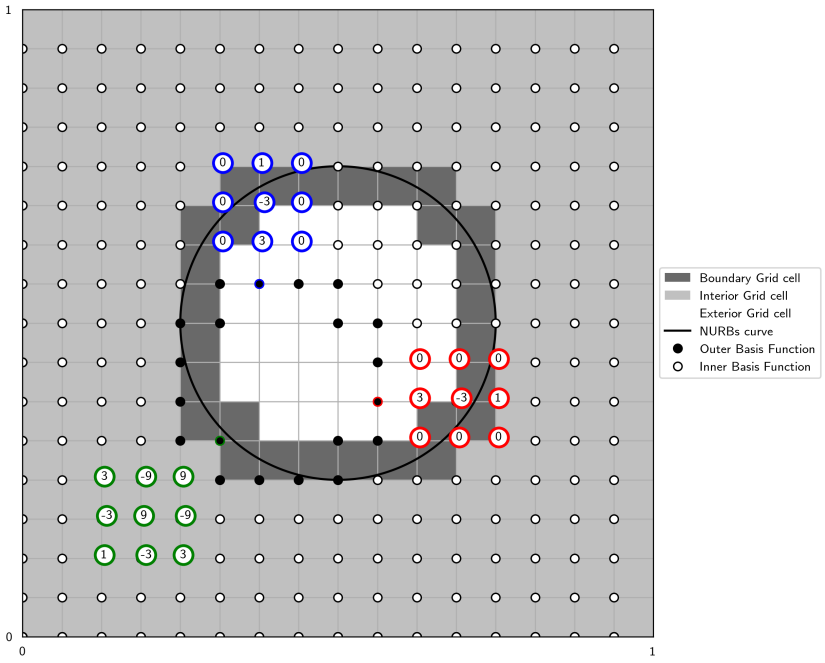



**Figure 2.9:** For several outer basis functions with index $j$ the index set $I(j)$ with the values of $e_{i,j}$ are shown. Specifically, we focus on the outer basis functions with red, green, and blue trims. For each of these basis functions, we show the index set $I(j)$ of stable basis functions labeled with a white circle and their corresponding trim color. Additionally, the extension coefficients $e_{ij}$ are provided within the circles.

In Figure 2.9 the index set $I(j)$ for several outer basis functions $j$ is shown together with the values $e_{i,j}$ of the Lagrange polynomial. For example lets consider the outer basis function with a green trim and look at $i = j - (3, 3)$

$$e_{i,j} = \left(\frac{3-1}{0-1}\frac{3-2}{0-2}\right)\left(\frac{3-1}{0-1}\frac{3-2}{0-2}\right) = 1,$$

Since $j - \ell = (3, 3)$ and $i - \ell = (0, 0)$, in the other two examples, many zeros appear due to the fact that the Lagrange polynomials vanish whenever $j_\nu \in (\ell_\nu + 0, \dots, n) \setminus i_\nu$ for some $\nu$. Conversely, in Figure 2.10, the index set $J(i)$ for several inner basis functions $j$ is shown (Note that only the outer basis functions with $e_{i,j} \neq 0$ are shown). Moreover, the number of outer basis functions with nonzero $e_{i,j}$ is different for each inner basis function. For instance, the red trimmed inner basis function has three outer basis functions with nonzero $e_{i,j}$ in its index set $J(i)$, whereas the green and blue trimmed inner basis functions have only one. Additionally, the extended B-splines that are far away from the trimming curve are just regular B-splines, i.e., $J(i) = \emptyset$. The extended B-splines inherit all the properties of B-splines except for non-negativity, since the extension coefficients $e_{i,j}$ can be negative. Compared to regular B-splines, extended B-splines have the property that the condition number of their associated Galerkin matrix is independent of the location of the trimming curve since B-splines with small support are substituted.

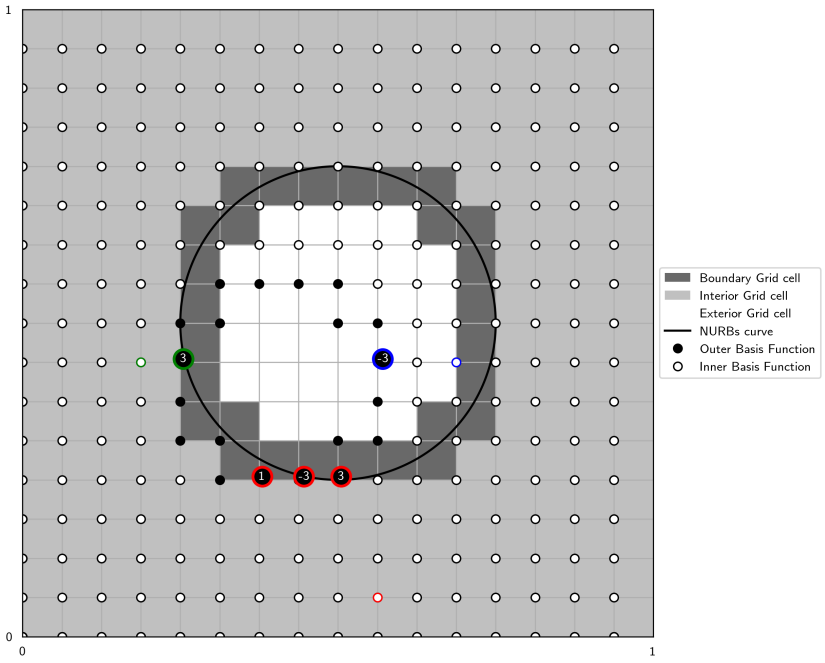



**Figure 2.10:** For several inner basis functions with index $i$ the index set $J(i)$ with the values of $e_{i,j}$ are shown. The considered inner basis functions have a red, green, and blue trims. For each of these basis functions, we show the index set $I(j)$ of unstable basis functions labeled with a black circle and their corresponding trim color. Additionally, the extension coefficient $e_{ij}$ is provided within the circle.

We conclude this section with a final remark concerning the assembly of the Galerkin matrix using extended B-spline trial and test spaces in both global and local approaches. In the global approach, we start by transforming the Galerkin matrix from the standard B-spline space given by.

$$A_h u = f, \tag{2.14}$$

where $A_h \in \mathbb{R}^{n\times n}$ is the Galerkin matrix for the B-spline spaces, $u \in \mathbb{R}^n$ and $f \in \mathbb{R}^n$. For the extended B-spline spaces, we introduce an extension matrix $E \in \mathbb{R}^{m\times n}$, where $m < n$ and

$$e_{i,k} = \begin{cases} 1 & \text{for } k = i, \\ e_{i,j} & \text{for } k = j \in J(i), \\ 0 & \text{otherwise,} \end{cases} \tag{2.15}$$

to modify the linear system as follows

$$EA_hE^T{}^e u = Ef \implies {}^eA_h{}^e u = {}^e f, \tag{2.16}$$

with ${}^eA_h \in \mathbb{R}^{m\times m}$ representing the Galerkin matrix for the extended B-spline spaces, ${}^ef \in \mathbb{R}^m$ the right hand side and ${}^eu \in \mathbb{R}^m$ the vector containing the weight of each basis function in the extended B-spline space. The global approach is particularly useful when there is an existing assembly procedure implemented for the standard B-spline spaces. On the other hand, a more local approach is also possible. In this case for each boundary grid cell, a local extension matrix is constructed to transform the element matrices and vectors of the standard B-spline spaces to the extended B-spline spaces. This local approach is discussed further in section 3.2.

### 2.3.3. Boundary conditions

The spline spaces defined in the previous section seem infeasible to enforce essential boundary conditions since they are not interpolatory. Lets first consider the mother problem 2.7 on the untrimmed domain $\Omega = [0,1] \times [0,1]$ together with knotvectors defined in the previous section for illustration. Classical finite elements generally interpolate $g$ on the boundary. If $g$ is a constant the same applies to the spline space, but for a general function $g$ this is not possible. Instead the boundary will only be enforced approximatley $g_h|_{\Gamma_D} \approx g$. More specifically an $L^2$ projection is considered [10]. The $L^2$ projection

projects an arbritrary function $g \in L^2(\Omega)$ into a finite element space $V_h \subset L^2(\Omega)$. Mathematically. the following minimisation problem needs to be solved:

$$\text{Find } u_h \in V_h \text{ such that } u_h = \min \frac{1}{2} \|u_h - g\|^2_{L^2(\Omega)}.$$

This minimization problem is convex and therefore has a global minimum. Applying the Fréchet derivative the problem reduces to

$$\text{Find } u_h \in V_h \text{ such that } (u_h, v_h)_{L^2(\Omega)} = (g, v_h)_{L^2(\Omega)}, \quad \forall v_h \in V_h.$$

Applying Galerkin method from section 2.3.1 a linear system is obtained: $M\alpha = b$. Solving this system gives the weights to expand the function $g$ on the spline space. In this way the Dirichlet boundary conditions are enforced on the boundaries of the untrimmed domain $\Omega$. Note that the mesh is fitted to the boundary meaning that the boundary of the mesh coincides with the the boundary of the physical domain. In contrast the boundary $\partial D$ is not fitted by the mesh. It is not clear how to enforce essential boundary conditions. This problem occurs in meshfree and embedded finite element methods where Nitsche's method [16] is used to enforce Dirichlet boundary conditions weakly [17] [18]. Other possible method exist like the the penalty method [14] and Lagrange multiplier technique [15], but the former increases the condition number of the Galerkin matrix considerably and the latter requires the introduction of an additional unknown function, therefore we consider Nitsche's method. Nitsche's method enforces the Dirichlet boundary conditions weakly consequently the finite element spaces in the weak formulations are not enforced to be zero at the boundary. Hence the weak formulation is altered. For the mother problem on a trimmed domain 2.12 the weak formulation is now given by

$$\begin{aligned}\text{Find } u \in H^1(\Omega \setminus D) \text{ with } u = g \text{ on } \Gamma_D \text{ such that } & \int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\Omega - \int_{\partial D} v(\nabla u \cdot \mathbf{n}) + u(\nabla v \cdot \mathbf{n}) \, \mathrm{d}\Gamma \\ + \alpha \int_{\partial D} vu \, \mathrm{d}\Gamma = \int_{\Gamma_N} hv \, \mathrm{d}\Omega + \int_\Omega fv \, \mathrm{d}x - \int_{\partial D} k(\nabla v \cdot \mathbf{n}) \mathrm{d}\Gamma + \alpha \int_{\partial D} vk \, \mathrm{d}\Gamma & \quad \forall v \in H^1_0,\end{aligned}$$

where $H^1_0 = \{h \in H^1(\Omega \setminus D) : h|_{\Gamma_D} = 0\}$ and $\alpha \in \mathbb{R}$ a stabilization parameter. The choice of $\alpha$ is discussed in [38]. The stabilization parameter is chosen such that the bilinear form is coercive. This is done by solving an eigenvalue problem. This can be done either locally or globally, where the former is adopted for its efficiency and simplicity of implementation. For each boundary gridcell $\Omega_e$ with a trimming curve segment $\Gamma_e$ (inside the boundary gridcell) the following generalized eigenvalue problem is constructed:

$$\begin{aligned}&\mathbf{Ax} = \Lambda \mathbf{Bx}, \quad \text{where} \\ &[\mathbf{A}]_{ij} = \int_{\Gamma_e} (\nabla N_i \cdot \mathbf{n}) (\nabla N_j \cdot \mathbf{n}) \, \mathrm{d}\Gamma, \\ &[\mathbf{B}]_{ij} = \int_{\Omega_e} \nabla N_i \cdot \nabla N_j \, \mathrm{d}\Omega,\end{aligned}$$

where $N_k$ can be in either $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h \setminus D)$ or $\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h \setminus D)$ and are the basis functions with support on the considered boundary gridcell $\Omega_e$. Note that the domain integrals representing the entries in $\mathbf{B}$ are evaluated on the entire domain irrespective of where the trimming curve intersects. Choosing $\alpha$ bigger than the maximum eigenvalue obtained from the generalized eigenvalue problem ensures coercitiviy of the bilinear form, hence we choose $\alpha = 2\max(\Lambda)$.

### 2.3.4. Numerical integration

The Assembly of the Galerkin matrix requires the evaluation of integrals over trimmed and non trimmed elements, where the latter is a rectangle. Lets first consider a non trimmed element $\Omega_e = [\xi_{k+1}, \xi_k] \times [\eta_{l+1}, \eta_l]$ and let $f : \Omega_e \subset \mathbb{R}^2 \to \mathbb{R}$ be a given function that is smooth and integrable. We want to evaluate

$$\int_{\Omega_e} f(\mathbf{x})\ \mathrm{d}\Omega = \int_{\xi_k}^{\xi_{k+1}} \int_{\eta_k}^{\eta_{k+1}} f(\xi,\eta)\ \mathrm{d}\xi\ \mathrm{d}\eta \tag{2.17}$$

First the integral is mapped to a reference element $\Omega_{\text{ref}} = [-1,1] \times [-1,1]$ by applying the linear transformations

$$X(x) = \frac{1}{2}(\xi_{k+1} - \xi_k)x + \frac{1}{2}(\xi_{k+1} + \xi_k), \quad Y(y) = \frac{1}{2}(\eta_{k+1} - \eta_k)y + \frac{1}{2}(\eta_{k+1} + \eta_k) \tag{2.18}$$

yielding

$$\int_{-1}^{1}\int_{-1}^{1} f(X(x),Y(y)) \left|\frac{1}{2}(\xi_{k+1} - \xi_k)\right| \left|\frac{1}{2}(\eta_{k+1} - \eta_k)\right|\ \mathrm{d}x\ \mathrm{d}y. \tag{2.19}$$

We approximate this integral by Gaussian Quadrature in each dimension.

$$\sum_{k=1}^{n_{in}^{(1)}} \sum_{l=1}^{n_{\text{in}}^{(2)}} f\left(X(\tilde{x}_l), Y(\tilde{y}_k)\right) w_l w_k \left|\frac{1}{2}(\xi_{k+1} - \xi_k)\right| \left|\frac{1}{2}(\eta_{k+1} - \eta_k)\right|, \tag{2.20}$$

where the values for $w_k/w_l$ and $\tilde{x}_l/\ \tilde{y}_k$ are tabulated in [39]. In one dimension the Gaussian Quadrature is optimal and an accuracy of order $2n_{\text{in}}$ is achieved by $n_{\text{in}}$ points [36]. This simple approach breaks down when a trimmed element is considered, since the element is not a rectangle anymore. There are several options to integrate on such a trimmed element:

1. NURBS-enhanced integration scheme: This requires the intersection points between the mesh $\Omega_h$ and NURBS trimming curve $C(\mathbf{u})$. The trimmed element is partitioned in disjoint triangles and NURBS curved side triangles. For the former standard Quadrature rules can be applied and for the later a mapping is introduced [11].
2. Quadrature free integration scheme: By successively applying the divergence theorem, integrals over boundary representations are transformed into line integrals with polynomial integrands and solved analytically with machine precision accuracy [12].
3. Quadtree methods: The general idea of quadtree integration is to capture the geometry of a cut-cell by recursively bisecting sub-cells that intersect with the boundary of the domain. At every level of this recursion, sub-cells that are completely inside the domain are preserved, while sub-cells that are completely outside of the domain are discarded [13]

We opt for the NURBS-enhanced integration scheme, since it reuses the routines from the untrimmed elements. As mentioned this method requires the intersection points between the mesh and the trimming curve a discussion on how to obtain these intersection points can be found in Appendix B. Moreover a sub-curve needs to be extracted from the trimming curve for each trimmed element, as discussed in section 2.2 by means of knot insertion. The trimmed elements are decomposed into triangles and NURBS curve sided triangles. There are several possible decompositions depending on how the trimming curve intersect the grid cell. In Figure 2.11 the different possibilities are distinguished by the number of vertices of the polyhedron, due to the trimming process. For the first case we have five vertices and split the trimmed grid cell into two triangles and one NURBS curve sided triangle. For the next case we have four vertices and divide the trimmed grid cell into one triangle and one NURBS curve sided triangle. For the last case with only three vertices no decomposition is necessary and the trimmed cell is a NURBS curve triangle itself. We have several special cases, for instance a NURBS curve with a high curvature. This should not pose a problem since a sufficiently fine grid is assumed to exclude this situation. Another special case is that the trimming curve $\mathbf{C}(u)$, which is defined using NURBS, has a knot inside the grid cell. At such a knot the properties of the NURBS curve may change (e.g. continuity), hence the curve sided triangle is split into two NURBS curve sided triangles.

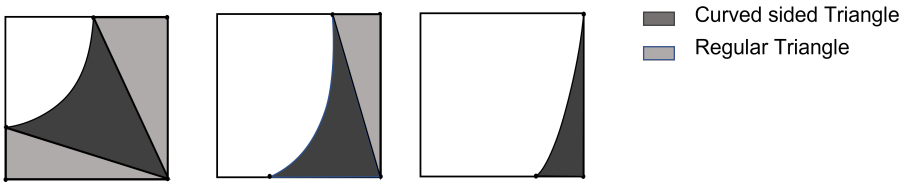



**Figure 2.11:** Possible decomposition's of the trimmed grid cell depending on the number of vertices of the resulting polyhedron due to the trimming.

Now we discuss how to perform numerical quadrature on these decomposed trimmed elements. First Gaussian quadrature on the regular triangles are discussed. In Figure 2.12 the quadrature rules defined on a reference element $\Omega_{\text{ref}} = [-1, 1] \times [-1, 1]$ are mapped to a regular triangle.

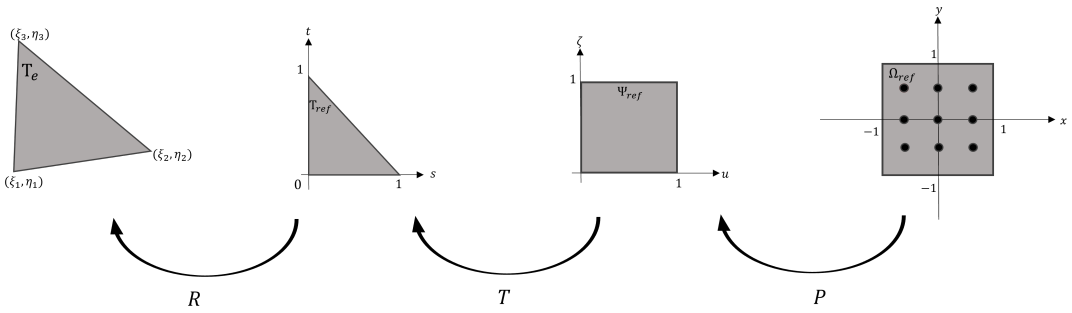



**Figure 2.12:** Mapping of quadrature points from a reference square to a triangle.

We start from the reference element $\Omega_{\text{ref}} = [-1, 1] \times [-1, 1]$ and rescale it to a unit square by applying the map $P$.

$$\mathbf{P} : \{x, y\} \to \{u, \zeta\}, \quad \begin{cases} u = \dfrac{\xi}{2} + \dfrac{1}{2}, \\ \zeta = \dfrac{\eta}{2} + \dfrac{1}{2}. \end{cases} \tag{2.21}$$

Thereafter the unit square is mapped to a reference triangle by collapsing one edge of the unit square with the map $T$.

$$\mathbf{Q} : \{u, \zeta\} \to \{s, t\}, \quad \begin{cases} s = u, \\ t = (1 - u)\zeta. \end{cases} \tag{2.22}$$

Lastly the reference triangle is mapped to the triangle element in the actual domain by applying the map $R$.

$$\mathbf{R} : \{s, t\} \to \{\xi, \eta\}, \quad \begin{cases} \xi = \xi_1 s + (1 - s - t)\xi_2 + \xi_3 s, \\ \eta = \eta_1 t + (1 - s - t)\eta_2 + \eta_3 s. \end{cases} \tag{2.23}$$

In conclusion, to map the quadrature point from the reference element to the triangle element $T_e$ the above mappings are consecutively applied. To this end we define the composition of these mappings as:

$$\Psi_{\text{regular}} = R \circ T \circ P. \tag{2.24}$$

Now we consider Gaussian quadrature on the NURBS curve sided triangle. In Figure 2.13 the quadrature rules defined on a reference element $\Omega_{\text{ref}} = [-1, 1] \times [-1, 1]$ are mapped to a NURBS curve sided triangle.

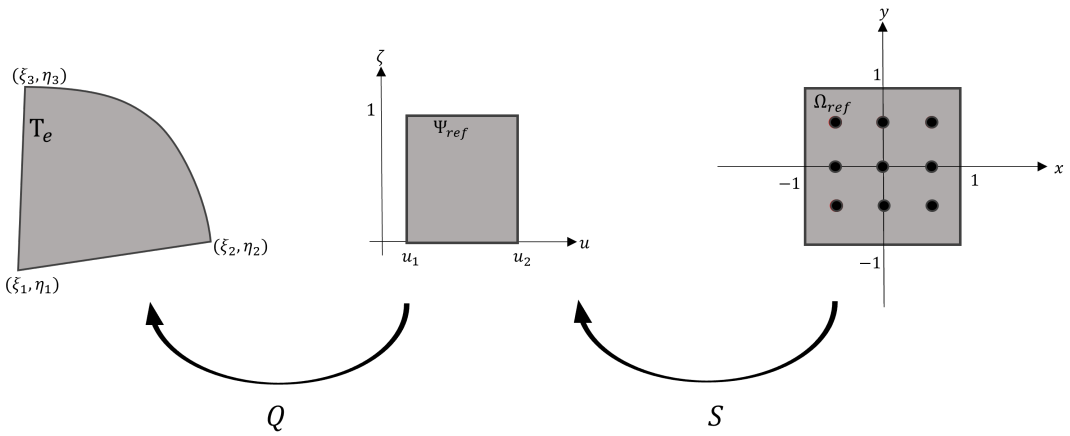



**Figure 2.13:** Mapping of quadrature points from a reference square to a curved triangle.

In contrast to the regular triangle, the reference square is re-scaled by applying the map S.

$$\mathbf{S} : \{x, y\} \rightarrow \{u, \zeta\}, \quad \begin{cases} u = \dfrac{\xi}{2}(u_2 - u_1) + \dfrac{1}{2}(u_2 + u_1), \\ \zeta = \dfrac{\eta}{2} + \dfrac{1}{2}, \end{cases} \tag{2.25}$$

where $u_1$ and $u_2$ are NURBS curve coordinates of the points $(\xi_2, \eta_2)$ and $(\xi_3, \eta_3)$ respectively, thereafter the rectangle is mapped to the actual NURBS curve sided triangle by applying the map Q.

$$\mathbf{Q} : \{u, \zeta\} \rightarrow \{\xi, \eta\}, \quad (1 - \zeta)\mathbf{C}(u) + \zeta \begin{bmatrix} \xi_1 \\ \eta_1 \end{bmatrix} \tag{2.26}$$

Note that the vertices of the NURBS curve sided triangle need to be chosen carefully. The vertex that is not on the NURBS curve is labeled as $(\xi_1, \eta_1)$. To this end we define the composition of these mappings as

$$\Psi_{\text{curved}} = S \circ Q \tag{2.27}$$

The defined mappings allow us to evaluate equation (2.17) over a trimmed gridcell $\Omega_{\text{trim}}$. The trimmed gridcell is decomposed in regular and curved triangles

$$\int_{\Omega_e} f(\mathbf{x})\, \mathrm{d}\Omega = \sum_{i=1}^{N_{\text{regular}}} \int_{T_{e_i}} f(\mathbf{x})\, \mathrm{d}\Omega + \sum_{i=1}^{N_{\text{curved}}} \int_{T_{e_i}} f(\mathbf{x})\, \mathrm{d}\Omega, \tag{2.28}$$

where $N_{\text{regular}}$ and $N_{\text{curved}}$ is the number of regular and curved triangles respectively. The integrals are evaluated with quadrature after applying the mappings

$$\begin{aligned} \sum_{i=1}^{N_{\text{regular}}} \int_{T_{e_i}} f(\mathbf{x})\, \mathrm{d}\Omega + \sum_{i=1}^{N_{\text{curved}}} \int_{T_{e_i}} f(\mathbf{x})\, \mathrm{d}\Omega = & \sum_{i=1}^{N_{\text{regular}}} \sum_{k=1}^{n_{in}^{(1)}} \sum_{l=1}^{n_{\text{in}}^{(2)}} f\left(\Psi_{\text{regular},i}(\tilde{x}_l, \tilde{y}_k)\right) w_l w_k |J_{\Psi_{\text{regular},i}}(\tilde{x}_l, \tilde{y}_k)| \\ & + \sum_{i=1}^{N_{\text{curved}}} \sum_{k=1}^{n_{in}^{(1)}} \sum_{l=1}^{n_{\text{in}}^{(2)}} f\left(\Psi_{\text{curved},i}(\tilde{x}_l, \tilde{y}_k)\right) w_l w_k |J_{\Psi_{\text{curved},i}}(\tilde{x}_l, \tilde{y}_k)|, \end{aligned} \tag{2.29}$$

where $J_g$ is defined as the jacobian of the mapping $g$. Lastly we consider Gaussian quadrature for a line integral on a segment of the NURBS curve $\Gamma_e = \mathbf{C}([u_1, u_2])$ with integrand $f(\xi, \eta) : \Gamma_e \in \mathrm{R} \rightarrow \mathrm{R}$:

$$\int_{\Gamma_e} f(\xi,\eta)d\ell = \int_{u_1}^{u_2} f(\boldsymbol{C}(u))\,\|\mathbf{C}'(u)\|\;\mathrm{d}u = \int_{-1}^{1} f(\boldsymbol{C}(X(x)))\,\|\mathbf{C}'(X(x))\|\left|\frac{1}{2}(u_2-u_1)\right|\;\mathrm{d}x$$
$$= \sum_{k=1}^{n_{in}} f\left(\mathbf{C}(X(\tilde{x}_k))\right) w_k \,\|\mathbf{C}'(X(\tilde{x}_k)))\|\left|\frac{1}{2}(u_2-u_1)\right|,$$

.

this type of integral occurs in enforcing natural boundary conditions and essential boundary conditions which are imposed weakly.

## 2.4. Structure preserving discretization

Hitherto we have discussed problems in two forms: the strong form (as seen in equation (2.7)) and the (primal) weak form (as seen in equation (2.8)). However, there exists a third form known as the mixed weak form. This formulation allows us to impose constraints exactly, like the incompressibility constraint $\nabla \cdot \boldsymbol{u} = 0$. It also makes it easier to access certain variables, such as the derivative of the numerical solution, without having to differentiate the solution itself. Additionally, it helps avoid difficulties that arise when discretizing spaces of regular functions, like $H_0^2$, which appear in fourth-order problems. Approximating such spaces requires ensuring continuity of derivatives at element interfaces, which can be more challenging than approximating spaces like $H^1$ or $H_0^1$, which appear in the mixed weak form of fourth-order problems [40]. To arrive at the mixed weak formulation for the mother problem in equation (2.7), we introduce an extra variable $\sigma$, the flux, to rewrite the strong form as follows:

$$\begin{cases} \boldsymbol{\sigma} - \nabla u = 0 & \text{in } \Omega, \\ \nabla \cdot \boldsymbol{\sigma} = -f & \text{in } \Omega, \\ u = g & \text{on } \Gamma_D, \\ \boldsymbol{\sigma} \cdot n = h & \text{on } \Gamma_N. \end{cases} \tag{2.30}$$

Next both equations are multiplied with test functions $\boldsymbol{\tau} \in \Sigma$ and $v \in V$ respectively, integrated over the domain $\Omega$ and the product rule is applied to arrive at

$$\int_\Omega (\boldsymbol{\sigma}\cdot\boldsymbol{\tau} + \nabla\cdot\boldsymbol{\tau} u)\mathrm{d}\Omega = \int_\Gamma \boldsymbol{\tau}\cdot\boldsymbol{n} g\;\mathrm{d}s \quad \forall \boldsymbol{\tau}\in\Sigma,$$
$$\int_\Omega \nabla\cdot\boldsymbol{\sigma} v\;\mathrm{d}\Omega = -\int_\Omega f v\;\mathrm{d}\Omega \quad \forall v \in V.$$

Note that the boundary condition for the flux is essential and the boundary condition for $u$ is natural. The space $V$ is set to be equal to $L^2(\Omega)$ and the space $\Sigma = H(\mathrm{div}) := \{\mathbf{u} \in L^2(\Omega) | \nabla\cdot\mathbf{u} \in L^2(\Omega)\}$. The mixed weak form is now given by:

Find $(u,\boldsymbol{\sigma}) \in L^2(\Omega)\times H(\mathrm{div};\Omega)$ with $\boldsymbol{\sigma}\cdot\boldsymbol{n} = h$ on $\Gamma_N$ such that

$$\int_\Omega \boldsymbol{\sigma}\cdot\boldsymbol{\tau} + \nabla\cdot\boldsymbol{\tau} u\mathrm{d}\Omega = \int_{\Gamma_D} \boldsymbol{\tau}\cdot\mathbf{n} g\;\mathrm{d}s \qquad \forall\boldsymbol{\tau}\in H_0(\mathrm{div}),$$
$$\int_\Omega \nabla\cdot\boldsymbol{\sigma} v\;\mathrm{d}\Omega = -\int_\Omega f v\;\mathrm{d}\Omega \qquad \forall v\in L^2(\Omega),$$

where $H_0(\mathrm{div}) = \{\mathbf{u}\in H(\mathrm{div}) | \mathbf{u}\cdot\mathbf{n} = 0\}$. As usual, to solve the problem Galerkin method is applied and appropriate discrete spaces $V_h \subset V$ and $\Sigma_h \subset \Sigma$ are chosen to arrive at a linear system. It is not straightforward on how to choose $V_h$ and $\Sigma_h$. Certain combinations of discrete spaces can lead to a non-singular Galerkin matrix and/or instabilities. For instance, if we choose $\boldsymbol{\tau} = \boldsymbol{\sigma_h}$ and $v = u_h$ and add both equations, it results in $\sigma_h = 0$ when $f = 0$, but it doesn't necessarily mean that $u_h$ is equal to zero, therefore the linear system can be singular. Even if we choose finite element spaces in such a way that the linear system becomes non-singular, there are still other issues related to stability (see for instance to Figure 2.2. [25, p.295]). To ensure stability, we must consider discrete spaces that satisfy the inf-sup stability condition, which for the mixed Poisson problem is given by

$$\inf_{v\in V_h}\sup_{\boldsymbol{\tau}\in\Sigma_h}\frac{\int_\Omega(\nabla\cdot\boldsymbol{\tau}v)\mathrm{d}\Omega}{\|v\|_{L^2}\|\boldsymbol{\tau}\|_{H^1}}\geq\gamma>0, \tag{2.31}$$

In most cases, proving this condition analytically for a specific pair of finite element spaces is out of reach. Therefore, we assess the discrete inf-sup constant, denoted as $\gamma_h$, as introduced by [41]. A numerical evaluation is very useful, but by no means an analytical proof, but from the experience it is that when the numerical test is passed, in fact, the inf-sup condition is satisfied. The test involves the calculation of $\gamma_h$ for a sequence of meshes. If $\gamma_h$ for these discretizations do not show a decrease towards zero, the test is passed. Next we outline how the discrete inf-sup constant is defined for the mixed Poisson problem. Let's assume that the finite element spaces are represented as $V_h=\mathrm{span}\{N_i^u\}_{i=1}^{n_u}$ and $\Sigma_h=\mathrm{span}\{\boldsymbol{N}_i^\tau\}_{i=1}^{n_\tau}$, where $N_i^\cdot$ represents spline basis functions as introduced in subsection 2.3.2. By expressing the solution as a linear combination of the basis functions with degrees of freedom $u_i$ and $p_i$:

$$u_h(x)=\sum_{i=1}^{n_u}N_i^u(x)u_i,\quad \boldsymbol{\sigma}_h(x)=\sum_{i=1}^{n_{\boldsymbol{\sigma}}}N_i^\sigma(x)\sigma_i \tag{2.32}$$

we obtain a linear system of the following form

$$\begin{bmatrix} A & B^T\\ B & 0\end{bmatrix}\begin{bmatrix}\hat{u}\\ \hat{p}\end{bmatrix}=\begin{bmatrix} f_1\\ f_2\end{bmatrix}, \tag{2.33}$$

where

$$\begin{aligned} A_{ij}&=\int_\Omega \boldsymbol{N}_i^{\boldsymbol{\sigma}}\cdot\boldsymbol{N}_i^{\boldsymbol{\sigma}}\mathrm{d}\Omega,\quad \hat{u}\quad =(u_1,\dots,u_{n_u})^T,\quad f_1=\int_\Gamma \boldsymbol{N}_i^{\boldsymbol{\sigma}}\cdot\boldsymbol{n}g\ \mathrm{d}s,\\ B_{ij}&=\int_\Omega N_i^u\nabla\cdot\boldsymbol{N}_i^{\boldsymbol{\sigma}}\mathrm{d}\Omega,\quad \hat{\sigma}\quad =(\sigma_1,\dots,\sigma n_\sigma)^T,\quad f_2=-\int_\Omega fN_i^u\ \mathrm{d}\Omega. \end{aligned} \tag{2.34}$$

To this end, the discrete inf-sup constant $\gamma_h$ is the square root of the smallest non-zero eigenvalue of the following generalized eigenvalue problem

$$BM_{uu}^{-1}B^T\tau=(\gamma_h)^2M_{\boldsymbol{\sigma}\boldsymbol{\sigma}}\tau, \tag{2.35}$$

where $M_uu$ and $M_{\boldsymbol{\sigma}\boldsymbol{\sigma}}$ are Gramian matrices associated to the inner products of $V_h$ and $\Sigma_h$, i.e.

$$(M_{uu})_{ij}=<N_i^u,N_j^u>_{V_h},\quad (M_{\sigma\sigma})_{ij}=<\boldsymbol{N}_i^u,\boldsymbol{N}_j^u>_{V_h}. \tag{2.36}$$

To construct finite element spaces that ensure accuracy and stability of the mixed finite element method the concept of finite element exterior calculus method (FEEC) [26] is adopted. FEEC preserves essential topological and homological structures at the discrete level. FEEC requires knowledge from homological algebra, algebraic topology, and functional analysis. For compactness, we will only introduce necessary definitions from homological algebra, where relevant for a more detailed explanation we refer to [26]. We start with the definition of a graded vector space.

**Definition 2.4.1** (Graded vector space)**.** *A graded vector space is a vector space $V$ expressed as a direct sum of subspaces $V_k$ indexed by integers:*

$$V=\bigoplus_{k=-\infty}^{\infty}V_k.$$

*It is non negative if $V_k=0$ for $k<0$ and finite if finitely many of the summands are nonzero.*

Next we can also define a mapping from a graded vector space to another graded vector space.

**Definition 2.4.2** (Graded map)**.** *A linear map $f:V\to W$ between two graded vector spaces is called a graded map of degree $p$ if $f(V_k)\subset W_{k+p}$ for $k$.*

With the latter two definitions the Chain complex is introduced.

**Definition 2.4.3** (Chain complex)**.** *A chain complex is a graded vector space $V$ with a graded linear map $\partial : V \to V$ of degree $-1$ which satisfies $\partial^2 = 0$. the sequence of operators $\partial_k = \partial|_{V_k}$ is called the differential of the chain complex. We write the chain complex as:*

$$\cdots \to V_{k+1} \xrightarrow{\partial_{k+1}} V_k \xrightarrow{\partial_k} V_{k-1} \to \cdots,$$

*or in short $(V, \partial)$. The elements of $V_k$ are called $k$-chains*

Note that $\partial_k : V_k \to V_{k-1}, k \in \mathbb{Z}$, with the property that $\partial_k \circ \partial_{k+1} = 0$. Under certain condition a subspace of a graded vector space forms a chain complex itself which is called a subcomplex of the chain complex.

**Definition 2.4.4** (subcomplex)**.** *A subcomplex of $(V, \partial)$ is a chain complex $(S, \partial)$ such that the subspace $S \subset V$ decomposes as a direct sum of subspaces $S_k \subset V_k$ and $\partial S_k \subset S_{k-1}$.*

Given a chain complex $(V, \partial)$. The null space $\mathcal{E}$ and the range space $\mathcal{B}$ equipped with the differential operator $\partial$ are each subcomplexes. The elements of $\mathcal{E}_k$ are called $k$-cycles and the elements of the range $\mathcal{B}_k$ of $\partial_{k-1}$ are called $k$-boundaries. Recall that $\partial^2 = 0$ implies $\mathcal{B}_k \subset \mathcal{E}_k$. The $k$-th Homology space is defined to be the quotient space $\mathcal{H}_k = \mathcal{E}_k \setminus \mathcal{B}_k$. The elements of the homology space are equivalence classes of $k$-cycles, where two cycles are equivalent if their difference is the $k$-boundary of a $(k+1)$-chain. If the boundary space is equal to the cycle space the homology space vanishes and the complex is said to be exact. Next we introduce chain maps which connect two chain complexes

**Definition 2.4.5.** *(chain maps) A chain map from a chain complex $(V, \partial)$ to a second chain complex $(W, \delta)$ is a sequence of linear maps $f_k : V_k \to W_k$ such that the following diagram commutes*

$$\begin{array}{ccccccccc}
\ldots \to & V_{k+1} & \xrightarrow{\partial_{k+1}} & V_k & \xrightarrow{\partial_k} & V_{k-1} & \to \ldots \\
& \downarrow{\scriptstyle f_{k+1}} & & \downarrow{\scriptstyle f_k} & & \downarrow{\scriptstyle f_{k-1}} & \\
\ldots \to & W_{k+1} & \xrightarrow{\delta_{k+1}} & W_k & \xrightarrow{\delta_k} & W_{k-1} & \to \ldots
\end{array}$$

*More specifically a chain map is a linear map of degree 0 which commutes with the differentials $f \circ \partial = \delta \circ f$*

the chain map maps the $k$-boundaries and cycles of $V$ to the $k$-boundaries and cycles of $W$. In addition the chain map induces a linear map $\bar{f} : \mathcal{H}_k(V) \to \mathcal{H}_k(W)$. A special case is when the complex $W$ is a subcomplex of $V$ then the chain map is called a chain projection since the maps $f_k$ are projections of $V_k$ onto $W_k$. The last concept that we will introduce is a cochain complex, which is the same as a chain complex except with the indexing reversed. The cochain differential have degree +1 instead of -1. From a notation point of view the subscripts are replaced by superscripts.

$$\cdots \to V^{k-1} \xrightarrow{d^{k-1}} V^k \xrightarrow{d^k} V^{k+1} \to \cdots.$$

All the definitions carry over with a prefix of co-: coboundary $\mathcal{B}^k$, cocycles $\mathcal{E}^k$ and cohomology $\mathcal{H}^k$. Many finite element spaces are defined as discrete subspaces $(V_h^j)$ of continuous spaces $(V^j)$ by applying Galerkin projection. These continuous spaces and finite element spaces are connected respectively in a cochain complex and subcomplex via differential operators $(d^j)$. The subcomplex is again connected to the chain complex via projections $(\Pi_h^j)$:

$$\begin{array}{ccccccccc}
\ldots \to & V^{k-1} & \xrightarrow{d^{k-1}} & V^k & \xrightarrow{d^k} & V_{k+1} & \to \ldots \\
& \downarrow{\scriptstyle \Pi_h^{k-1}} & & \downarrow{\scriptstyle \Pi_h^k} & & \downarrow{\scriptstyle \Pi_h^{k+1}} & \\
\ldots \to & V_h^{k-1} & \xrightarrow{d^{k-1}} & V_h^k & \xrightarrow{d^k} & V_h^{k+1} & \to \ldots
\end{array}$$

The Galerkin method is consistent and stable and converges with the rate of the best approximation if the Galerkin subspaces satisfy three conditions:

1. The subspaces afford a good approximation
2. The subspaces form a subcomplex
3. The subspaces admit a bounded cochain projection, i.e. bounded linear projections $V^j \to V_h^j$ which commute with $d^j$

We revisit the mixed weak formulation of the poisson problem and use spline test and trial spaces and assume $\Omega \subset \mathbb{R}^2$. The chain complex associated with the spaces appearing in the weak form of the problem is known as the De Rham complex. De Rham complex is connected to the spline spaces via Galerkin projections ($\Pi_h^j$):

$$\begin{array}{ccccccccc}
0 \to & H^1(\Omega) & \xrightarrow{\nabla\times} & H(\mathrm{div};\Omega) & \xrightarrow{\nabla\cdot} & L^2(\Omega) & \to \mathbb{R} \\
 & \Pi_h^{k+1}\downarrow & & \Pi_h^{k}\downarrow & & \Pi_h^{k-1}\downarrow & \\
0 \to & \mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_h) & \xrightarrow{\nabla\times} & \mathbb{S}^{p,q-1}_{\alpha_1,\alpha_2-1}(\Omega_h) \times \mathbb{S}^{p-1,q}_{\alpha_1-1,\alpha_2}(\Omega_h) & \xrightarrow{\nabla\cdot} & \mathbb{S}^{p-1,q-1}_{\alpha_1-1,\alpha_2-1}(\Omega_h) & \to \mathbb{R}
\end{array}$$

, where $\nabla \times u = \begin{bmatrix} -\frac{\partial u}{\partial y} \\ \frac{\partial u}{\partial x} \end{bmatrix}$ for $u \in H^1(\Omega)$ .The diagram is constructed by A.Buffa et al. [24] for $\Omega \subset \mathbb{R}^3$ and satisfies all of the three bulletpoints resulting in consistent and stable method that converges with the rate of best approximation. A final remark regarding the terminology of the basis functions within the spline subcomplex of the chain map. The basis functions in the first column are referred to as the 0-form basis functions, and those in the second column as the 1-form basis functions and so forth. Another commonly used terminology is vertex basis functions, edge basis functions, and face basis functions.

# 3

# Construction of a structure preserving subcomplex

In this chapter, our objective is to expand the chain map presented in Section 2.4, which consists of the De Rham complex and a structure-preserving subcomplex comprised of spline spaces. We introduce an additional subcomplex. This additional subcomplex is constructed in a sequential manner, starting from the 0-form. More specifically, the 0-form is defined as the extended B-spline space derived from the 0-form spline space in the predefined subcomplex. Moving forward, each subsequent form is constructed in a progressive fashion by exploiting the commutative property of the diagram in a similar fashion as [42]. To provide a comprehensive overview, we also consider and introduce the one dimensional De Rham complex in Section 3.1. Furthermore, in the subsequent Section 3.2, we delve into the examination of the two dimensional De Rham complex.

## 3.1. One dimensional domain

The De Rham complex in 1D is given by:

$$0 \to H^1(\Omega) \xrightarrow{\frac{d}{dx}} L^2(\Omega) \to \mathbb{R}.$$

Next De Rham complex is connected to a spline subcomplex as constructed by A.Buffa et al. [24] via Galerkin projections $\Pi_h^2$ and $\Pi_h^1$

$$\begin{array}{ccccccc} 0 \to & H^1(\Omega) & \xrightarrow{\frac{d}{dx}} & L^2(\Omega) & \to \mathbb{R} \\ & \Big\downarrow \Pi_h^{k+1} & & \Big\downarrow \Pi_h^{k} & \\ 0 \to & \mathbb{S}_{\alpha}^{p}(\Omega_h) & \xrightarrow{\frac{d}{dx}} & \mathbb{S}_{\alpha-1}^{p-1}(\Omega_h) & \to \mathbb{R} \end{array}$$

Now we want to introduce an additional subcomplex. The 0-form of this additional subcomplex is the extended B-spline space of the 0-form spline space of the predefined subcomplex. The extended B-spline basis function are related to the B-spline basis function by an extension matrix $E^0$ as discussed in subsection 2.3.2. To this end we have the following chain map

$$
\begin{array}{ccccccc}
0 \to & H^1(\Omega) & \xrightarrow{\frac{d}{dx}} & L^2(\Omega) & \to \mathbb{R} \\
 & \Pi_h^{k+1} \downarrow & & \Pi_h^{k} \downarrow & \\
0 \to & \mathbb{S}_\alpha^p(\Omega_h) & \xrightarrow{\frac{d}{dx}} & \mathbb{S}_{\alpha-1}^{p-1}(\Omega_h) & \to \mathbb{R} \\
 & E^0 \downarrow & & E^1 =? \downarrow & \\
0 \to & {}^e\mathbb{S}_\alpha^p(\Omega_h) & \xrightarrow{\frac{d}{dx}} & ? & \to \mathbb{R}
\end{array}
$$

where $E^1$ is left to be determined by exploiting the commutative property. To this end we assume $\Omega := [0,1] \setminus (t_{\text{trim}}, 1] = [0, t_{\text{trim}}]$, where $0 < t_{\text{trim}} < 1$ is the trimming location, $h$ fixed in $\mathbb{R}$, $p$ fixed in $\mathbb{N}_{>0}$ and $\alpha \in \{0, \dots, p-1\}$. Let $g \in^e \mathbb{S}_\alpha^p(\Omega_h)$ then

$$
g = \sum_{i=1}^{m} g_i \left[ N_i^p + \sum_{j \in J(i)} e_{i,j} N_j^p \right] = \begin{bmatrix} N_1^p & \dots & N_n^p \end{bmatrix} E^0 \begin{bmatrix} g_1 \\ \vdots \\ g_m \end{bmatrix}, \tag{3.1}
$$

where $m$ is the number of stable basis functions, $n$ is the number of basis functions whit nonzero support on $\Omega$, $J(i)$ is the set of unstable basis functions $j \in J(i)$ that use stable basis function $i$ to stabilise with, $E^0 \in \mathbb{R}^{n \times m}$ is the extension matrix of the 0-form with entries the extension coefficients $e_{i,j}$. We define $\beta_i = \sum_{j=1}^{m} g_j e_{ij}$ for $i \in \{1, \dots, n\}$ then:

$$
g = \begin{bmatrix} N_1^p & \dots & N_n^p \end{bmatrix} \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_n \end{bmatrix}. \tag{3.2}
$$

Taking the derivative of $g$ gives

$$
\begin{aligned}
\frac{dg}{dx} = \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} \begin{bmatrix} \beta_2 - \beta_1 \\ \vdots \\ \beta_n - \beta_{n-1} \end{bmatrix} &= \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} \begin{bmatrix} -1 & 1 & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \vdots \\ \vdots \\ \vdots \\ \beta_n \end{bmatrix} \\
&:= \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} D \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_n \end{bmatrix} = \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} D E^0 \begin{bmatrix} g_1 \\ \vdots \\ g_m \end{bmatrix},
\end{aligned} \tag{3.3}
$$

where $D \in \mathbb{R}^{(n-1) \times n}$. To this end we try to recover an expression of the following form:

$$
\begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} E^1 \begin{bmatrix} g_2 - g_1 \\ \vdots \\ g_m - g_{m-1} \end{bmatrix} \tag{3.4}
$$

and determine $E^1$. We exploit the partition of union property of the extended b-splines and the fact that applying the matrix $D$ to a constant vector gives "0" to arrive at:

$$\frac{dg}{dx} = \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} DE^0 \begin{bmatrix} g_1 \\ \vdots \\ g_m \end{bmatrix} = \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} DE^0 \left( \begin{bmatrix} g_1 \\ \vdots \\ g_m \end{bmatrix} - g_1 \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \right)$$

$$= \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} DE^0 \begin{bmatrix} 0 \\ g_2 - g_1 \\ \vdots \\ g_m - g_1 \end{bmatrix} = \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} DE^0 \begin{bmatrix} 0 & \dots & 0 \\ 1 & \ddots & \vdots \\ \vdots & \ddots & 0 \\ 1 & \dots & 1 \end{bmatrix} \begin{bmatrix} g_2 - g_1 \\ \vdots \\ g_m - g_{m-1} \end{bmatrix}$$

$$= \begin{bmatrix} N_1^{p-1} & \dots & N_{n-1}^{p-1} \end{bmatrix} DE^0 I \begin{bmatrix} g_2 - g_1 \\ \vdots \\ g_m - g_{m-1} \end{bmatrix}, \quad (3.5)$$

where $I \in \mathbb{R}^{m \times (m-1)}$, and also referred to as the integration matrix and to be confused with the identity matrix. We conclude that $E^1 = DE^0 I \in \mathbb{R}^{(n-1)\times(m-1)}$. Lastly we have several remarks:

- The choice to subtract the constant vector with value $g_1$ is done because of convenience, since the first row of matrix $I$ will be a row of zeros and the remaining rows form a lower triangular matrix. One can choose any of the values $g_l$ with $l \in \{1, \dots, m\}$.
- We observe from several numerical examples that the span of the derived basis functions, i.e. the unknown space "?", coincides with the extended B-spline space ${}^e\mathbb{S}_\alpha^{p-1}$. A proof is given in Appendix C. We conclude that the extended chain map is given by:

$$\begin{array}{ccccccc}
0 \to & H^1(\Omega) & \xrightarrow{\frac{d}{dx}} & L^2(\Omega) & \to \mathbb{R} \\
& \Pi_h^{k+1} \downarrow & & \Pi_h^k \downarrow & \\
0 \to & \mathbb{S}_\alpha^p(\Omega_h) & \xrightarrow{\frac{d}{dx}} & \mathbb{S}_{\alpha-1}^{p-1}(\Omega_h) & \to \mathbb{R} \\
& E^0 \downarrow & & E^1 \downarrow & \\
0 \to & {}^e\mathbb{S}_\alpha^p(\Omega_h) & \xrightarrow{\frac{d}{dx}} & {}^e\mathbb{S}_{\alpha-1}^{p-1}(\Omega_h) & \to \mathbb{R}
\end{array}$$

- Note that we have assumed only one trimming location, nevertheless the approach can be extended to multiple trimming locations straightforwardly.
- To numerically verify the convergence and stability the constructed structure preserving subcomplex we consider the mixed Poisson problem:

$$\begin{cases} \sigma = -u', & \text{in } \Omega \\ \sigma' = f, & \text{in } \Omega \\ u = 0, & \text{on } \partial\Omega \end{cases} \quad (3.6)$$

with the following weak form

$$\begin{aligned} &\text{Find } (u, \sigma) \in L^2(\Omega) \times H^1(\Omega) \text{ such that} \\ &\int_\Omega \sigma\tau dx - \int_\Omega u\tau' dx = 0, \qquad \tau \in H^1(\Omega), \\ &\int_\Omega \sigma' v dx = \int_\Omega f v dx, \qquad\quad v \in L^2(\Omega). \end{aligned}$$

which has trial and test spaces $H^1(\Omega)$ and $L^2(\Omega)$.

- One can also consider a local approach instead of the global approach mentioned earlier. For example, let's consider the domain mentioned at the beginning of this section. We have a function $g$ belonging to $\mathbb{S}(\Omega_e)^2_1$, where $\Omega_e$ represents a boundary grid cell or element. Within the element $\Omega_e$, there are a total of 3 nonzero basis functions. The local extension matrix, denoted as $E^0 \in \mathbb{R}^{3\times 3}$, is defined as:
$$E^0 = \begin{bmatrix} 3 & -3 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix},$$
the first row corresponds to unstable basis functions, stabilized by 3 stable basis functions, while the last two rows correspond to the stable basis functions. Next we have the difference matrix, $D \in \mathbb{R}^{2\times 3}$, which is given by
$$D = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix},$$
and lastly a possible integration matrix, $I \in \mathbb{R}^{3\times 2}$, is given by
$$I = \begin{bmatrix} 0 & 0 \\ -1 & 0 \\ -1 & -1 \end{bmatrix},$$
This gives the following 1-form local extension matrix:
$$E^1 = \begin{bmatrix} 2 & -1 \\ 1 & 0 \end{bmatrix}$$
which is exactly the local extension matrix of a function $f \in \mathbb{S}^1_0(\Omega_e)$. This approach is especially useful in higher dimensions. Moreover if we consider an interior grid cell the 1-form extension matrix is the identity which is what we expect since we do not require stabilization for these basis functions:
$$E^1 = DE^0I = D \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} I = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ -1 & 0 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

## 3.2. Two dimensional domain

In the previous section, we discussed the one-dimensional case. Now, we shift our focus to the two-dimensional case. The approach is similar, but there's a difference: the extension matrices are now local for each element. Let's consider an element labeled as a boundary grid cell, denoted by $\Omega_e$. For the other grid cells, which are interior, the untouched chain complex is sufficient. To extend the chain map from section 2.4, we introduce another subcomplex with its 0-form, the extended b-spline space of the predefined subcomplex's 0-form.

$$\begin{array}{ccccccccc}
0 \to & \mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_e) & \xrightarrow{\nabla\times} & \mathbb{S}^{p,q-1}_{\alpha_1,\alpha_2-1}(\Omega_e) \times \mathbb{S}^{p-1,q}_{\alpha_1-1,\alpha_2}(\Omega_e) & \xrightarrow{\nabla\cdot} & \mathbb{S}^{p-1,q-1}_{\alpha_1-1,\alpha_2-1}(\Omega_e) & \to \mathbb{R} \\
& \Big\downarrow E^0 & & \Big\downarrow E^1 & & \Big\downarrow E^2 & \\
0 \to & {}^e\mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_e) & \xrightarrow{\nabla\times} & ? & \xrightarrow{\nabla\cdot} & ? & \to \mathbb{R}
\end{array}$$

where $E^1$ and $E^2$ are to be determined, and $E^0 \in \mathbb{R}^{n_{\text{loc}}\times m_{\text{loc}}}$, where $n_{\text{loc}}$ is the number of basis functions with nonzero support on the element $\Omega_e$, i.e. $n_{\text{loc}} = (p+1)(q+1)$, and $m_{\text{loc}}$ is the number of stable basis functions with nonzero support on $\Omega_e$ and the stable basis functions used to stabilize the unstable basis functions with nonzero support on $\Omega_e$. We begin with determining $E^1$. Suppose we have a function $g \in^e \mathbb{S}^{p,q}_{\alpha_1,\alpha_2}(\Omega_e)$

$$g = \begin{bmatrix} N^{p,q}_1 & \dots & N^{p,q}_{n_{\text{loc}}} \end{bmatrix} E^0 \begin{bmatrix} g_1 \\ \vdots \\ g_{m_{\text{loc}}} \end{bmatrix} \tag{3.7}$$

We define $\beta_i = \sum_{j=1}^{m_{\text{loc}}} g_j e_{ij}$ for $i \in \{1, \dots, n_{\text{loc}}\}$ then:

$$g = \begin{bmatrix} N_1^{p,q} & \dots & N_{n_{\text{loc}}}^{p,q} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_{n_{\text{loc}}} \end{bmatrix}. \tag{3.8}$$

Taking the derivative of $g$ with respect to both spatial coordinates gives

$$\begin{bmatrix} \frac{dg}{dx} & \frac{dg}{dy} \end{bmatrix} = \begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} D \begin{bmatrix} \beta_1 \\ \vdots \\ \vdots \\ \vdots \\ \beta_{n_{\text{loc}}} \end{bmatrix}, \tag{3.9}$$

where $D = \begin{bmatrix} D_{11} \otimes D_{12} \\ D_{21} \otimes D_{22} \end{bmatrix} \in \mathbb{R}^{[p(q+1)+q(p+1)] \times n_{\text{loc}}}$. Moreover $D_{11} = \begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{p \times (p+1)}$, $D_{11} = I \in \mathbb{R}^{(q+1)\times(q+1)}$, $D_{21} = I \in \mathbb{R}^{(p+1)\times(p+1)}$ and $D_{22} = \begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{q \times (q+1)}$. We expand the vector of $\beta_i$'s to arrive at.

$$\begin{aligned} \begin{bmatrix} \frac{dg}{dx} & \frac{dg}{dy} \end{bmatrix} &= \begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} D \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_{n_{\text{loc}}} \end{bmatrix}, \\ &= \begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} D E^0 \begin{bmatrix} g_1 \\ \vdots \\ g_{m_{\text{loc}}} \end{bmatrix}. \end{aligned} \tag{3.10}$$

To this end we try to recover an expression of the following form:

$$\begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} E^1 \tilde{D} \begin{bmatrix} g_1 \\ \vdots \\ g_{m_{\text{loc}}} \end{bmatrix}. \tag{3.11}$$

The matrix $\tilde{D} \in \mathbb{R}^{n_{\text{edges}} \times n_{\text{loc}}}$ is a difference matrix that contains $n_{\text{edges}} \in \mathbb{N}$ differences consisting of all the possible vertical and horizontal differences. Just as we did for the one-dimensional domain, we take advantage of the partition of union property of the extended b-splines and the fact that applying the matrix $D$ to a constant vector results in a vector of zeros to arrive at:

$$
\begin{aligned}
\begin{bmatrix} \frac{dg}{dx} & \frac{dg}{dy} \end{bmatrix} &= \begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} DE^0 \begin{bmatrix} g_1 \\ \vdots \\ g_{m_{\text{loc}}} \end{bmatrix} \\
&= \begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} DE^0 \left( \begin{bmatrix} g_1 \\ \vdots \\ g_{m_{\text{loc}}} \end{bmatrix} - g_1 \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \right) \\
&= \begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} DE^0 \begin{bmatrix} 0 \\ g_2 - g_1 \\ \vdots \\ g_{m_{\text{loc}}} - g_1 \end{bmatrix} \\
&= \begin{bmatrix} N_1^{p,q-1} & \dots & N_{q(p+1)}^{p,q-1} & N_1^{p-1,q} & \dots & N_{p(q+1)}^{p-1,q} \end{bmatrix} DE^0 I\tilde{D} \begin{bmatrix} g_1 \\ \vdots \\ g_{m_{\text{loc}}} \end{bmatrix},
\end{aligned}
\tag{3.12}
$$

where the matrix $I \in \mathbb{R}^{m_{\text{loc}} \times n_{\text{edges}}}$ is known as the integration matrix (not to be confused with the identity matrices used earlier). The integration matrix is constructed row by row. For each difference $g_i - g_1 \in 2, \dots, m_{\text{loc}}$, we determine a path from $g_i$ to $g_1$ through a path-finding algorithm that traverses through the edges. Then:

$$
I_{i,j} = \begin{cases} 0 & \text{if edge is not used} \\ -1 & \text{if edge is used in its correct orientation} \\ 1 & \text{if edge is used in its incorrect orientation} \end{cases}
\tag{3.13}
$$

By comparing the final results in equations 3.12 and 3.11, we can conclude that $E^1 = DE^0 I \in \mathbb{R}^{[p(q+1)+q(p+1)] \times n_{\text{edges}}}$. To illustrate this approach, let's consider the following example:

Let $g \in \mathbb{S}_{1,1}^{2,2}(\Omega_e)$, where $\Omega_e$ is the element highlighted in red in Figure 3.2. The nonzero basis functions (labeled with a circle in the lower left corner of their support and their stability marked as explained in section (2.3.2)) on this element are enclosed within the red square. In total, there are $n = 9$ nonzero basis functions, of which 3 are unstable (the ones with a black trim). To stabilize these unstable basis functions, we choose an index set. For this example and convenience, the same index set is chosen for all three unstable basis functions, with appropriate extension coefficients $e_{ij}$, indicated by a colored square corresponding to the color of each basis function. This gives us the following local extension matrix for the 0-form:

$$
E^0 = \begin{bmatrix}
0 & 9 & -9 & 3 & 0 & -9 & 9 & -3 & 3 & -3 & 1 \\
0 & 3 & 0 & 0 & 0 & -3 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 3 & -3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{bmatrix} \in \mathbb{R}^{9 \times 11}.
\tag{3.14}
$$

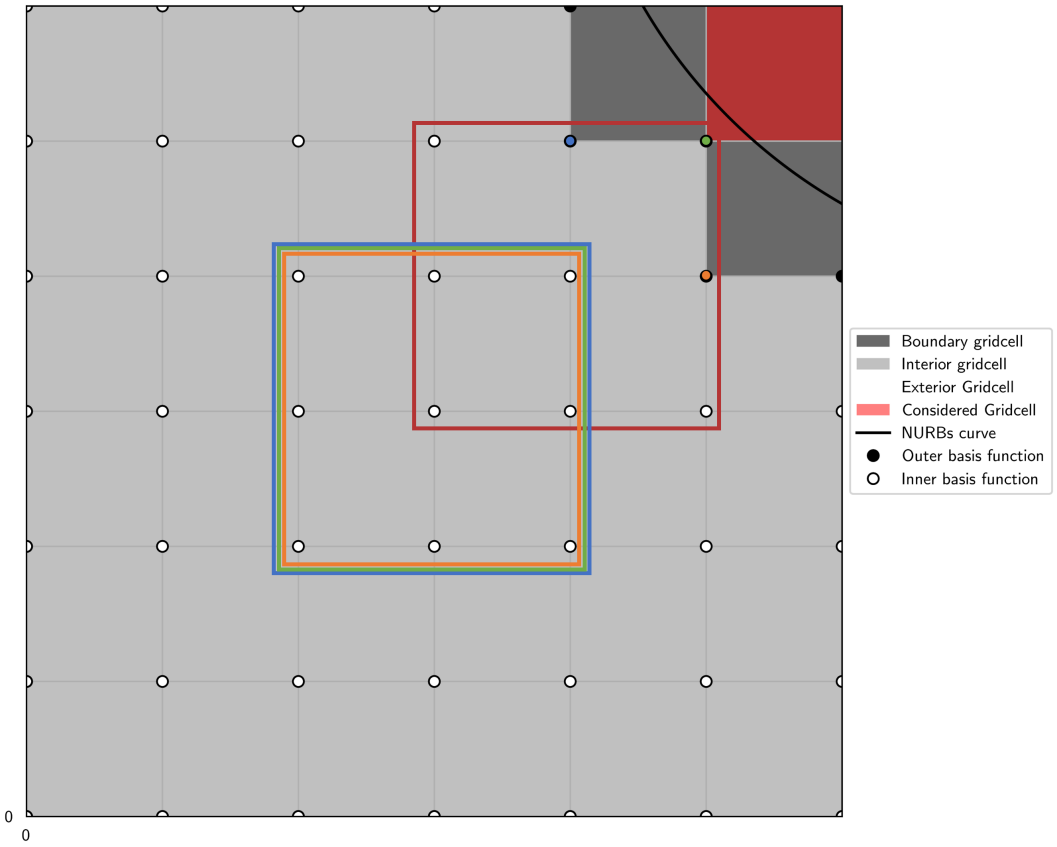



**Figure 3.1:** Illustration of a particular situation in the space $\mathbb{S}^{2,2}_{1,1}$ and its corresponding ${}^{e}\mathbb{S}^{2,2}_{1,1}$, where we focus on the element $\Omega_e$ highlighted in red. The basis basis functions with nonzero support are shown in a red square. In this scenario, there are three unstable basis functions, each assigned the same index set with a colored square that matches the color of the unstable basis function (please note that they have different corresponding extension coefficients $e_{ij}$).

After stabilizing the unstable basis functions, the number of basis functions with nonzero support on the element $\Omega_e$ increases to $n_{\text{loc}} = 11$, as shown in Figure 3.1 enclosed by a red rectangular curve. Additionally, in Figure 3.1, the 1-form basis functions are depicted as arrows, and we have only chosen the trivial differences, therefore $n_{\text{edges}} = 14$. Each basis function is locally numbered.

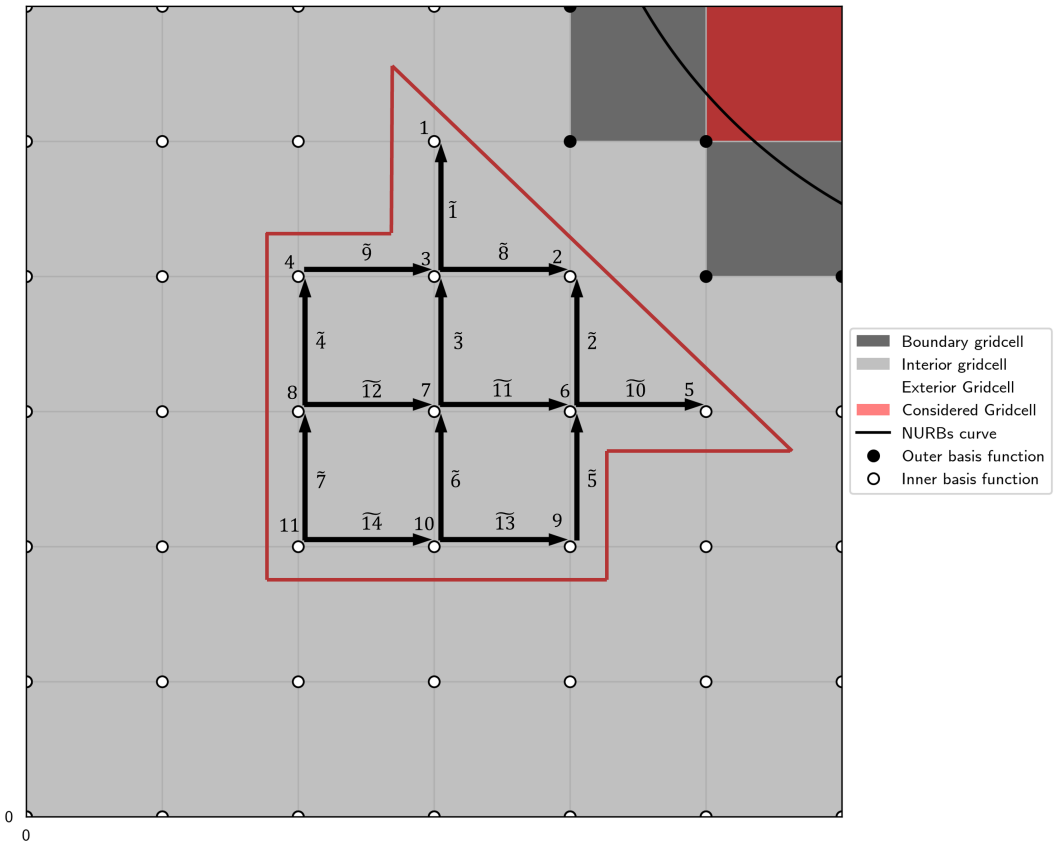



**Figure 3.2:** Illustration of a particular situation in the space ${}^{e}\mathbb{S}^{2,2}_{1,1}$, where we focus on the element $\Omega_e$ highlighted in red. The basis functions with nonzero support are shown in a red rectangular curve and represented as circles.Additionally, we use arrows to represent the basis functions associated with trivial differences (i.e., edge basis functions). Both types of basis functions are locally numbered, and the degree of freedom related to the edge basis functions is marked with a tilde.

For this specific example:

$$\tilde{D} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \in \mathbb{R}^{14\times 11}. \tag{3.15}$$

Furthermore, we choose $g_1$ as the value we subtract (as done in equation 3.12). Therefore, a possible integration matrix can be represented as follows:

$$I = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \in \mathbb{R}^{11\times 14}. \tag{3.16}$$

The difference $g_5 - g_1$ can be represented by transversing trough the edges: $\tilde{1}, \tilde{2}, \tilde{9}, \tilde{11}$, where the corresponding values $I_{5,1}$ and $I_{5,2}$ are $-1$ because we go with the direction of the difference and for $I_{5,9}$ and $I_{5,11}$ are 1 because we go against the direction of the difference. For this specific example the 1-form extension matrix is given by:

$$E^1 = DE^0 I = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & -3 & 0 & -6 & 3 & 2 & -1 \\ -1 & 0 & 2 & 0 & 0 & -1 & 0 & 3 & 0 & 0 & -3 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & -1 & 0 & 6 & -2 & 0 & -9 & 3 & 3 & -1 \\ 0 & 0 & 2 & 0 & 0 & -1 & 0 & 2 & 0 & 0 & -3 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 3 & -1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{12\times 14}. \tag{3.17}$$

Lastly we have several remarks:

- To find the paths from $g_i$ to $g_j$, one can use a pathfinding algorithm like Dijkstra's algorithm. In certain situations, evaluating a path may not be possible because the 0-form basis functions cannot be reached through the edge basis functions. In such cases, we include additional trivial differences that are not unstable. By incorporating weights into the Dijkstra algorithm, we can force it not to include the unstable edge basis functions in its path. This ensures that the condition number of the Galerkin matrix remains unaffected by the location of the trimming curve.
- It's important to note that we have some flexibility in selecting the differences or edge basis functions, and we have chosen only the trivial differences.
- To numerically verify the convergence and stability the constructed structure preserving subcomplex we consider the vector Laplacian problem:

$$\begin{cases} \nabla \times \sigma - \nabla\nabla \cdot \boldsymbol{u} = \boldsymbol{f} & (x,y) \in \Omega, \\ \sigma = \nabla \times \boldsymbol{u} & (x,y) \in \Omega, \\ \sigma = 0, & (x,y) \in \partial\Omega, \\ \boldsymbol{u} \cdot \boldsymbol{n} = 0, & (x,y) \in \partial\Omega, \end{cases}$$

with the following weak form

$$\begin{aligned} &\text{Find } (\boldsymbol{u}, \sigma) \in H_0(\mathrm{Div};\Omega) \times H_0^1(\Omega) \text{ such that} \\ &\int_{\Omega\setminus D} \sigma\tau - \boldsymbol{u} \cdot (\nabla \times \tau)\mathrm{d}x = 0 \qquad \forall \tau \in H_0^1(\Omega), \\ &\int_\Omega (\nabla \times \sigma) \cdot \boldsymbol{v} + \nabla \cdot \boldsymbol{v}\nabla \cdot \boldsymbol{u}\mathrm{d}x = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v}\mathrm{d}x \qquad \forall \boldsymbol{v} \in H_0(\mathrm{Div};\Omega), \end{aligned}$$

which has trial and test spaces $H^1(\Omega)$ and $H(\mathrm{Div};\Omega)$.

We are left to determine $E^2$. Nonetheless, because the outcomes of the constructed extension matrix $E^1$ in section 4.3 were unsatisfactory, the determination of $E^2$ is deferred for future investigations.

# 4

# Numerical Results

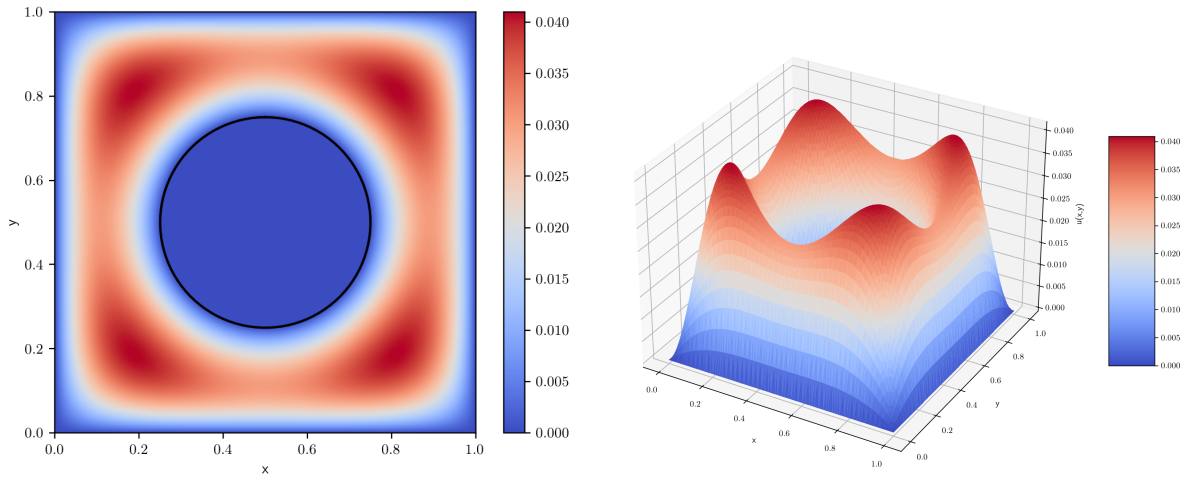
In this chapter the numerical result are presented for several test problems. In section 4.1 the immersed Poisson problem is considered, where the known results from the Chapter 2 are confirmed. Thereafter the constructed structure preserving spaces in Chapter 3 are numerically assessed in sections 4.2 and 4.3.

## 4.1. Immersed Poisson Problem

For the first test problem a standard Poisson problem (4.1) is solved on the domain $\Omega \setminus D$ where $\Omega$ is a unit square, $\Omega = [0,1] \times [0,1]$, and $D$ is the trimmed region. The boundary $\partial D$ is given by the trimming curve $\mathbf{C}(u)$ (defined by using NURBS) that describes a circle with radius 0.25 and midpoint $(0.5, 0.5)$.

$$\begin{cases} -\Delta u = 4\pi(0.5-x)\cos\pi x \sin\pi y + 4\pi(0.5-y)\sin\pi x\cos\pi y \\ \qquad\quad - 4\sin\pi x \sin\pi y, & (x,y) \in \Omega \setminus D, \\ u(x,y) = 0, & (x,y) \in \partial D, \\ u(x,y) = 0, & (x,y) \in \partial\Omega. \end{cases} \tag{4.1}$$

The exact solution is given by $\tilde{u} = \sin\pi x \sin\pi y((x-0.5)^2 + (y-0.5)^2 - (0.25)^2)$ and is shown in Figure 4.1.




**(a)** **(b)**

**Figure 4.1:** Exact solution of problem (4.1). In Figure (a) a 2D surface plot is shown, the trimming of the domain $\Omega$ by the trimming curve $\mathbf{C}(u)$, which is a circle, is represented by the black curve. In Figure (b) the exact solution is plotted as a 3D surface plot.

Problem (4.1) is solved using the theory discussed in section 2.3. In short the problem is reformulated in its weak form (see equation (2.8)) and the infinite dimensional space $H^1(\Omega \setminus D)$ is approximated by spline spaces $\mathbb{S}^{p,q}_{p-1,q-1}(\Omega_h)$ of order $p = q = 1, 2$ and 3. Moreover proper numerical integration on the trimmed elements, strong enforcement of the boundary condition on $\partial\Omega$ and weak enforcement of the boundary condition on $\partial D$ using Nitsche method is used to obtain the numerical solution $u_h$. The convergence order in the $L^2$ and $H^1$ norm is shown in Figure 4.2.

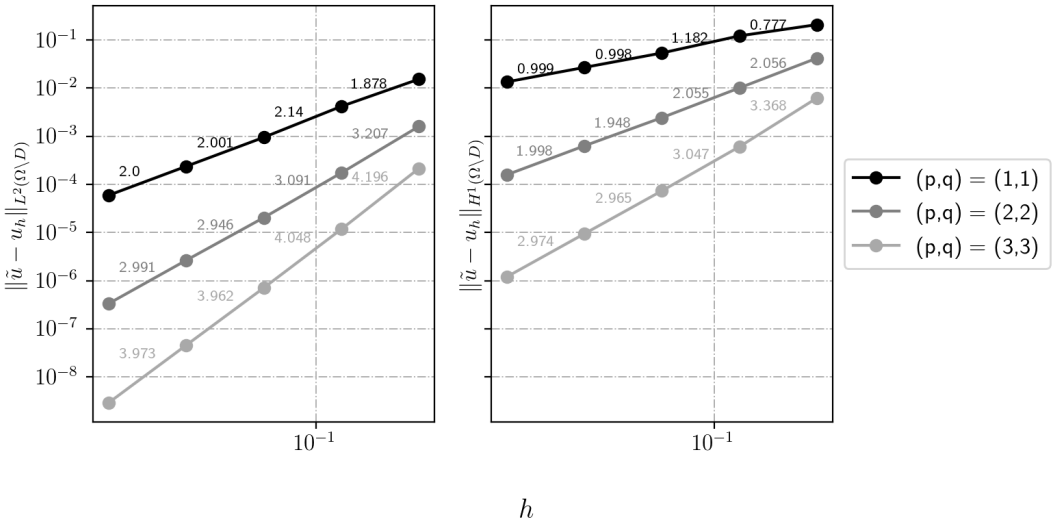



**Figure 4.2:** On the left the convergence order in the $L^2$ norm for B-spline space of order (p,q) is shown and on the right in the $H^1$ norm.

The convergence order coincide with the theoretical error estimate stated in section 2.3.1. Next we consider the condition number of the Galerkin matrix which is shown in Figure 4.3. The condition number is not in line with the theoretical estimate shown in equation (2.10). This is because the B-spline spaces are not stable when trimming is involved. More specifically there are basis functions with small support due to the trimming of the domain.

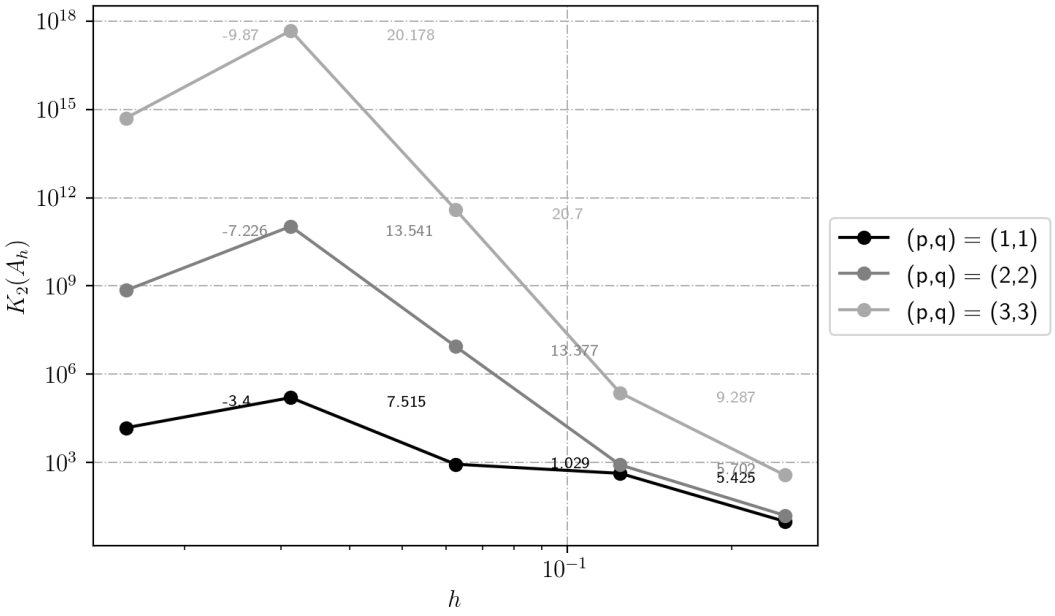



**Figure 4.3:** The condition number of the Galerkin matrix is calculated using the standard B-spline space for trial and test space approximation for various values of the mesh parameter $h$. Additionally, the slopes are displayed.

Next we are going to solve problem (4.1) again, but now the extended B-spline space ${}^{e}\mathbb{S}^{p,q}_{p-1,q-1}(\Omega_h)$ is used instead to approximate the infinite dimensional space $H^1(\Omega \setminus D)$. The corresponding convergence order in the $L^2$ and $H^1$ norm is shown in Figure 4.4.

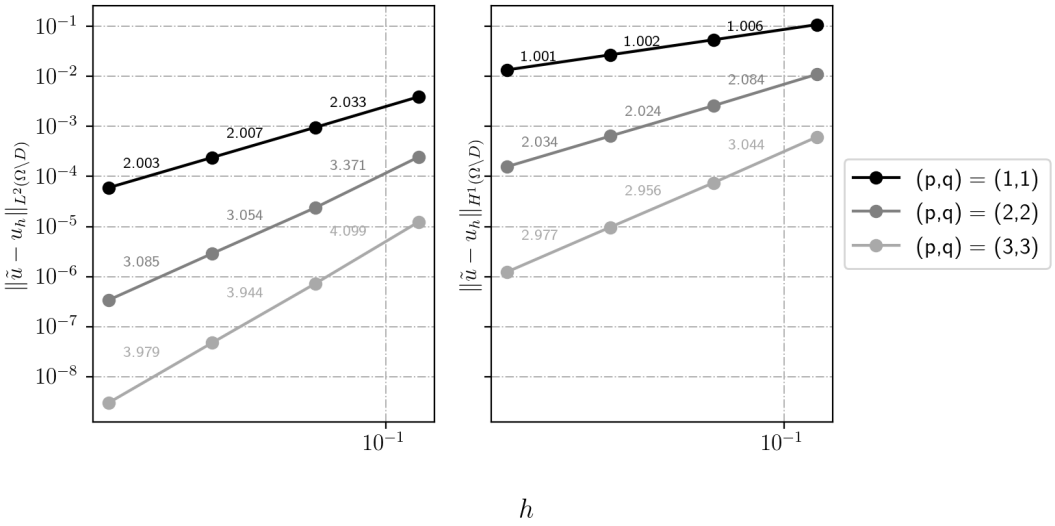



**Figure 4.4:** On the left the convergence order in the $L^2$ norm for the extended B-spline space of order (p,q) is shown and on the right in the $H^1$ norm.

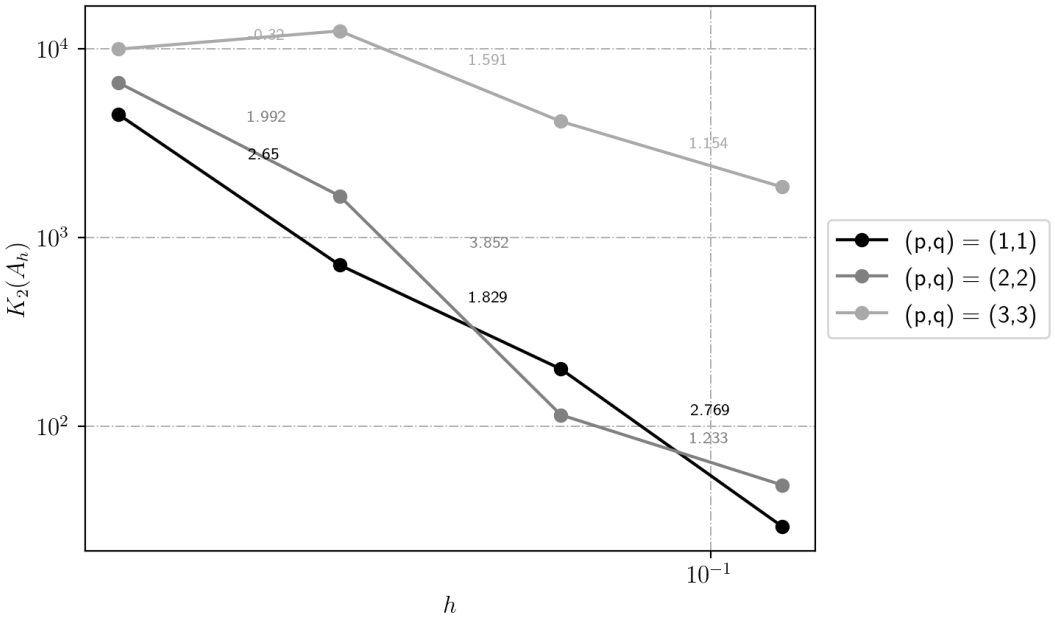



**Figure 4.5:** The condition number of the Galerkin matrix is calculated using the extended B-spline space for trial and test space approximation for various values of the mesh parameter $h$. Additionally, the slopes are displayed.

The convergence order aligns with the theoretical error estimate mentioned in section 2.3.1. However, when comparing it to the B-spline space, Figure 4.5 reveals that the condition number of the Galerkin matrix is significantly smaller. This difference arises from the characteristic of the extended B-splines, wherein the condition number of the resulting Galerkin matrix remains unaffected by the location of the Galerkin matrix. Figure 4.6 illustrates this property more clearly, here we solve the following test problem instead.

$$\begin{cases} -\Delta\tilde{u} = 2\pi^2 \sin(\pi x)\sin(\pi y), & (x,y) \in \Omega \setminus D, \\ \dfrac{du(x,y)}{dx} \cdot \mathbf{n} = \dfrac{d\tilde{u}(x,y)}{dx} \cdot \mathbf{n}, & (x,y) \in \partial D, \\ u(x,y) = 0, & (x,y) \in \partial\Omega. \end{cases} \tag{4.2}$$

where the exact solution is given by $\tilde{u} = \sin(\pi x)\sin(\pi y)$ and $\mathbf{n}$ is the outward pointing normal vector of the trimming curve. The mesh parameter is fixed ($h = \frac{1}{16}$) and the trimming curve is defined as before but shifted by varying the center of the circle with a value $\epsilon$, which is also represented on the horizontal axis in Figure 4.6. Moreover in Figure 4.7, problem 4.2 is solved for two different trimming curve for illustration:

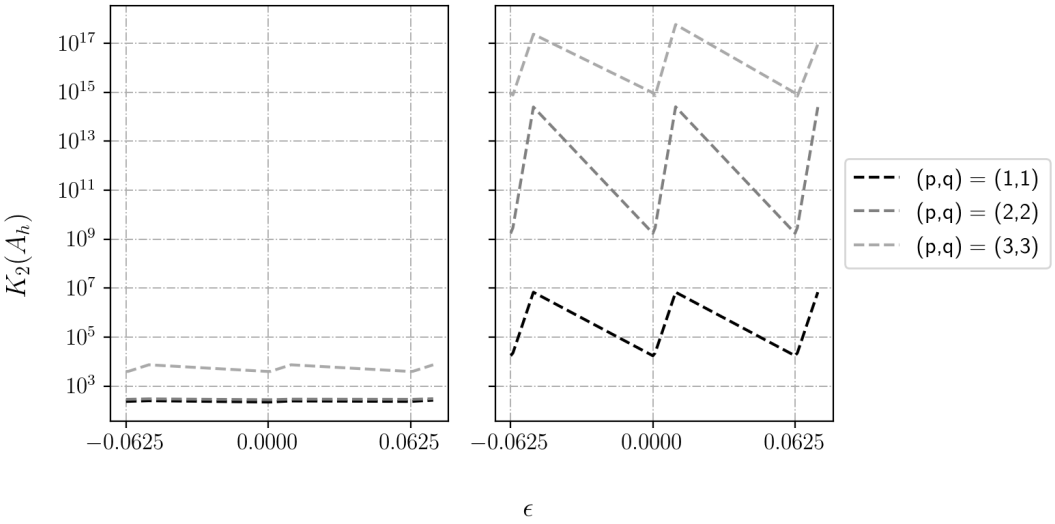
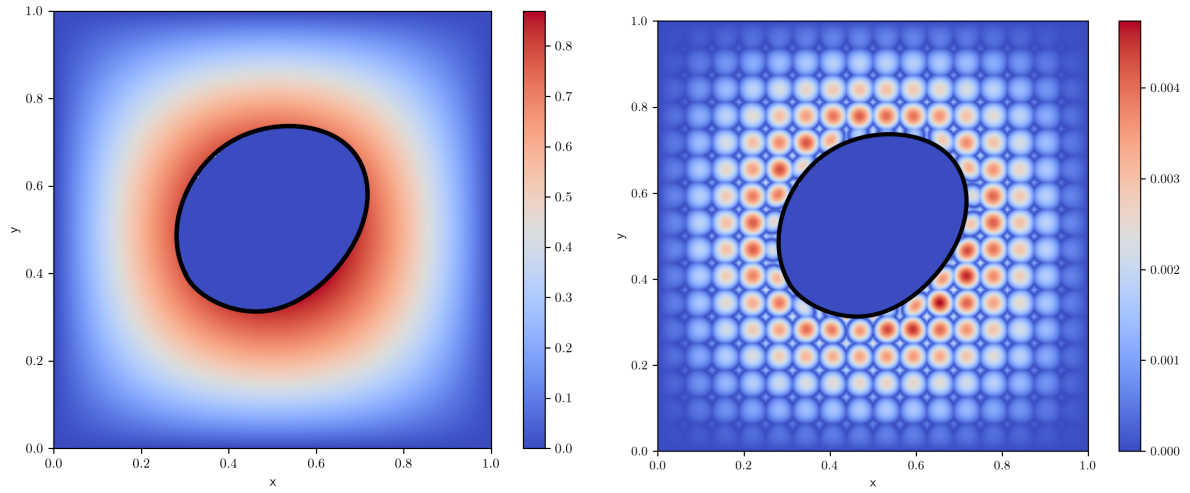
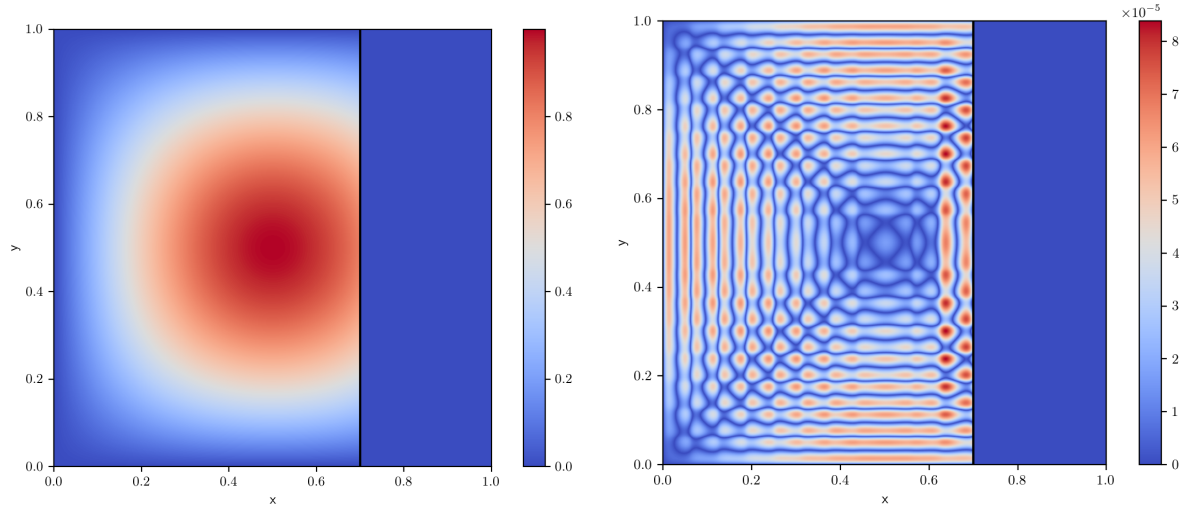



**Figure 4.6:** On the left the condition number is shown with respect to the parameter $\epsilon$, where the value $\epsilon$ shifts the center of the circle along the horizontal axis, for the standard B-spline space and on the right for the extended B-spline space.




**(a)** **(b)**

**Figure 4.7:** In Figure (a) the exact solution of problem (4.2) is shown, the domain $\Omega$ is trimmed by the $D$ where $\partial D$ is represented by the trimming curve $\mathbf{C}(u)$, with $U = \{0,0,0,0,1/5,2/5,3/5,4/5,1,1,1,1\}$ of order $l = 3$, control points $\{\mathbf{P}_1\} = \{[0.3,0.4],[0.32,0.35],[0.5,0.25],[0.75,0.5],[0.7,0.75],[0.35,0.75],[0.25,0.5],[0.3,0.4]\}$ and unit weights, and is represented by the black curve. In Figure (b) the absolute error is shown between the exact solutions and the Galerkin approximation using a standard spline space of order 1 and mesh parameter $h = 1/16$.




**(a)** **(b)**

**Figure 4.8:** In Figure (a) the exact solution of problem (4.2) is shown, the domain $\Omega$ is trimmed by the $D = [0.7 + 10^{-10}, 1] \times [0, 1]$ where $\partial D$ is represented by a vertical cut at $x = 0.7 + 10^{-10}$, and is represented by the black curve. In Figure (b) the absolute error is shown between the exact solutions and the Galerkin approximation using a extended B-spline space of order 2 and mesh parameter $h = 1/16$.

## 4.2. Immersed Mixed Poisson Problem - One dimensional domain

In this section the spaces constructed in Chapter 3 are numerically assessed. The one dimensional domain case is considered. For the two dimensional domain some auxiliary results are shown in Appendix D. In this section, we address a mixed Poisson problem defined on the domain $\Omega \setminus D$. Here, $\Omega$ represents the standard unit interval, and $D = (t_{\text{trim}}, 1]$ denotes the trimmed region. The value of $t_{\text{trim}}$ lies between 0 and 1, specifying the trimming point. We aim to solve the following specific boundary value problem:

$$\begin{cases} \sigma(x) = u'(x) & x \in [0, t_{\text{trim}}], \\ \sigma'(x) = -\left(\frac{\frac{1}{2}\pi}{t_{\text{trim}}}\right)^2 a \cos\left[\frac{1}{2}\pi\left(\frac{x}{t_{\text{trim}}}\right)\right] - \frac{b}{1-e^{t_{\text{trim}}}} e^x & x \in [0, t_{\text{trim}}], \\ u(0) = a, \\ u(t_{\text{trim}}) = b. \end{cases} \tag{4.3}$$

The exact solution is given by: $u = a \cos\left[\frac{1}{2}\pi \frac{x}{t_{\text{trim}}}\right] + \frac{b}{1-e^{t_{\text{trim}}}}(1 - e^x)$, and $\sigma = -\frac{\frac{1}{2}\pi}{t_{\text{trim}}} a \sin\left[\frac{1}{2}\pi \frac{x}{t_{\text{trim}}}\right] - \frac{b}{1-e^{t_{\text{trim}}}} e^x$. For completeness we state the weak formulation, which is given by:

Find $(u, \sigma) \in L^2([0, t_{\text{trim}}]) \times H^1([0, t_{\text{trim}}])$ such that

$$\int_0^{t_{\text{trim}}} \sigma\tau dx + \int_0^{t_{\text{trim}}} u\tau' dx = b\tau(t_{\text{trim}}) - a\tau(0), \qquad \tau \in H^1([0, t_{\text{trim}}]),$$

$$\int_0^{t_{\text{trim}}} \sigma' v dx = \int_0^{t_{\text{trim}}} \left( -\left(\frac{\frac{1}{2}\pi}{t_{trim}}\right)^2 a \cos\left[\frac{1}{2}\pi\left(\frac{x}{t_{trim}}\right)\right] - \frac{b}{1-e^{t_{trim}}} e^x \right) v dx, \qquad v \in L^2([0, t_{\text{trim}}]).$$

In order to have consistent and stable method, we choose the trial and test spaces for $H^1([0, t_{\text{trim}}])$ and $L^2([0, t_{\text{trim}}])$, that appear in the weak form based on diagram (3.1). Specifically, we utilize the sub spaces $\mathbb{S}^p_{p-1}$ and $\mathbb{S}^{p-1}_{p-2}$ of order $p = 1, 2, 3$. Additionally, we set $t_{\text{trim}} = 0.5 + 1e^{-13}$. The convergence order in the $L^2$ norm for both $u_h$ and $\sigma_h$, as well as the $H^1$ norm for $\sigma_h$, are depicted in Figure 4.9 and 4.10, respectively.

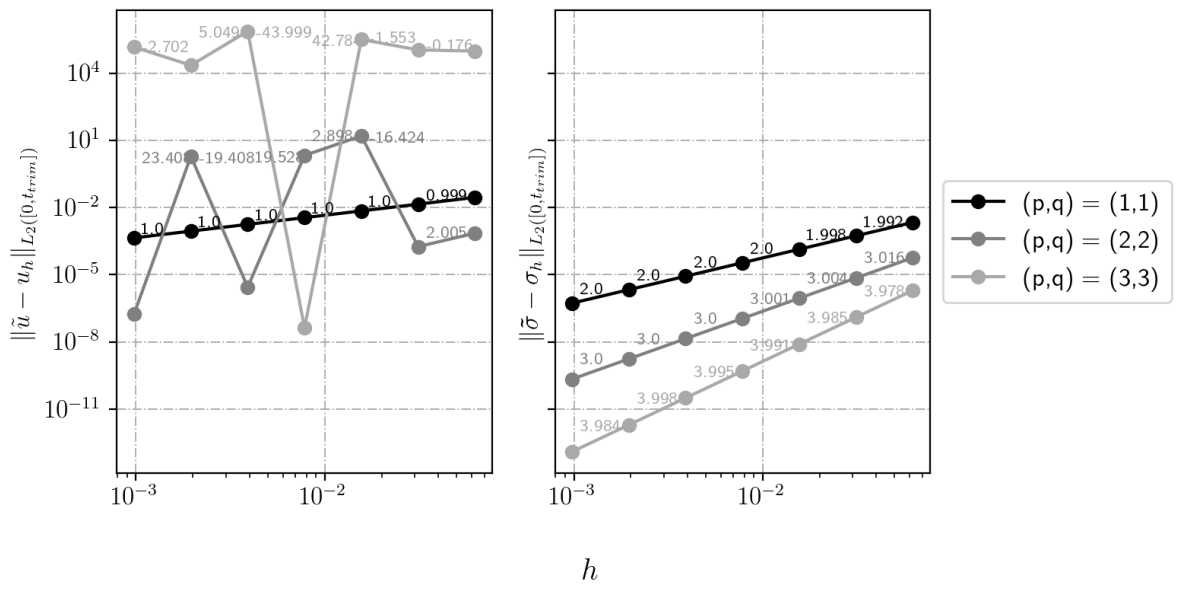



**Figure 4.9:** $L^2$ norm of the error for B-spline basis functions of order 1,2 and 3. On the left, the error of $u_h$ is represented in the $L^2$ norm, while on the right, the error of $\sigma$ is represented in the $L^2$ norm.

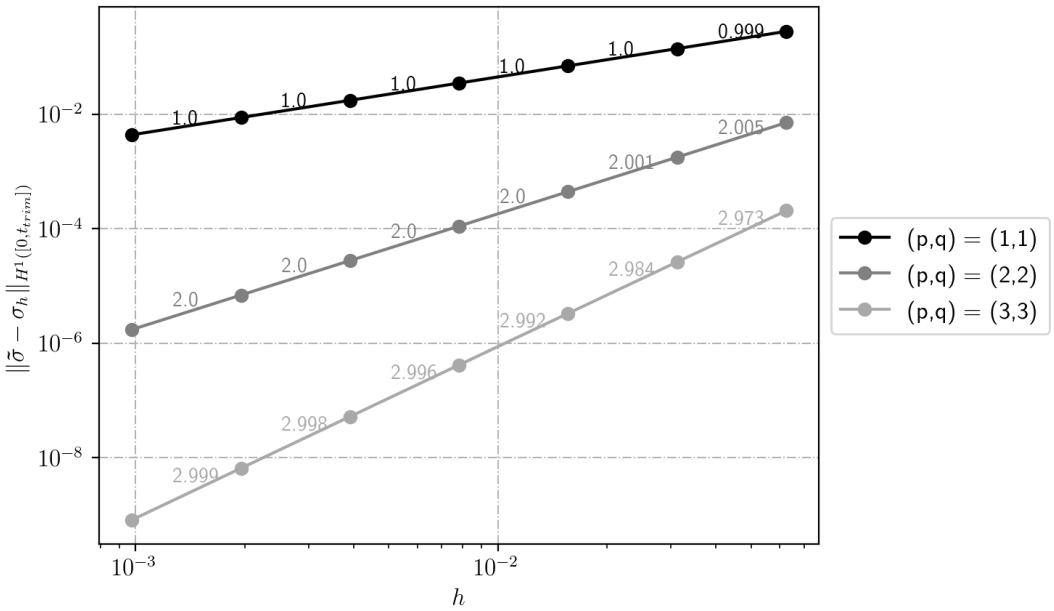



**Figure 4.10:** The error of $\sigma_h$ in the $H^1$ norm for B-spline basis functions of order 1,2 and 3.

We observe optimal convergence order for $\sigma_h$ for all values of $p$. However, for $u_h$, we observe optimal convergence order only when $p = 1$. This discrepancy arises from the condition number increasing as the mesh parameter is refined and the basis function order is raised. Figure 4.11 illustrates this phenomenon, which is a consequence of certain basis functions having small support within the domain. We observe convergence in the $H^1$ and $L^2$ norms for the $\sigma$ variable. To solve the system, we utilize LU-decomposition with pivoting. However, when solving the system again using the CG-method, both variables do not converge. On the other hand, the opposite occurs for the GMRES-method compared to the LU-decomposition method.

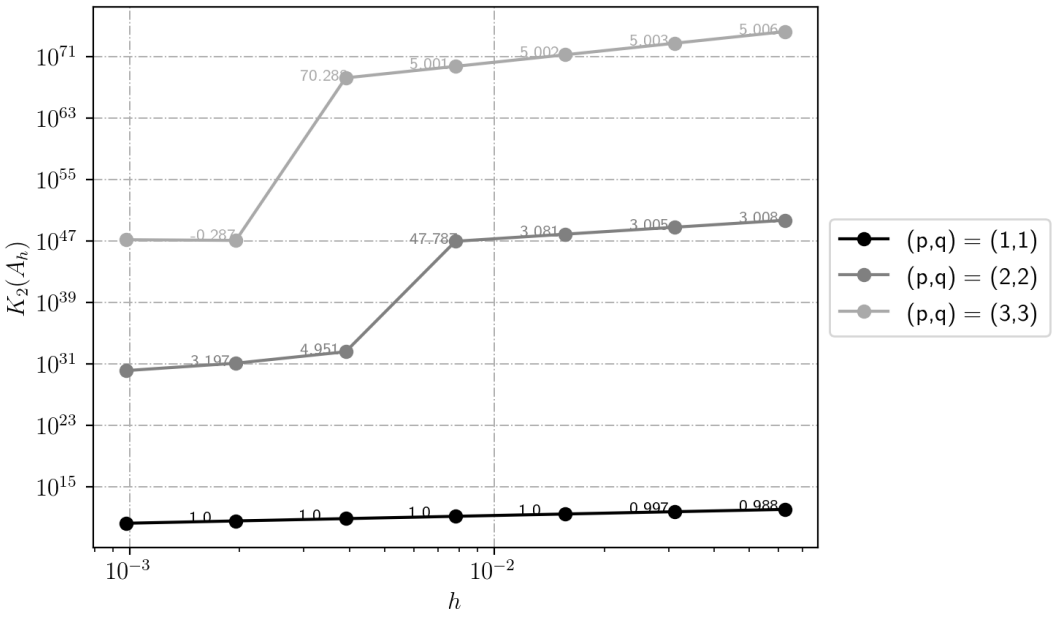



**Figure 4.11:** The condition number of the Galerkin matrix is calculated using the standard B-spline space for trial and test space approximation for various values of the mesh parameter $h$. Additionally, the slopes are displayed.

Next, we choose the test and trial spaces ${}^e\mathbb{S}^p_{p-1}$ and ${}^e\mathbb{S}^{p-1}_{p-2}$ for $H^1(\Omega \setminus D)$ and $L^2(\Omega \setminus D)$, respectively as constructed in section 3.1. The convergence order in the $L^2$ norm for both $u_h$ and $\sigma_h$, and $H^1$ norm for $\sigma_h$ are shown in Figure (4.12) and (4.13). We observe optimal convergence order.

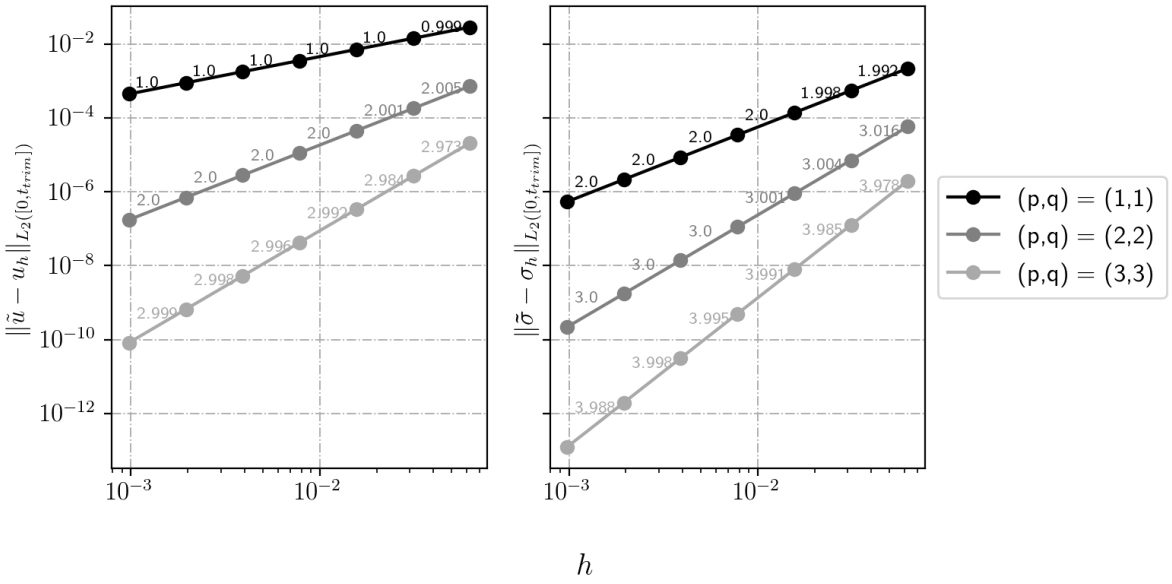



**Figure 4.12:** $L^2$ norm of the error for extended B-spline basis functions of order 1,2 and 3. On the left, the error of $u_h$ is represented in the $L^2$ norm, while on the right, the error of $\sigma$ is represented in the $L^2$ norm.

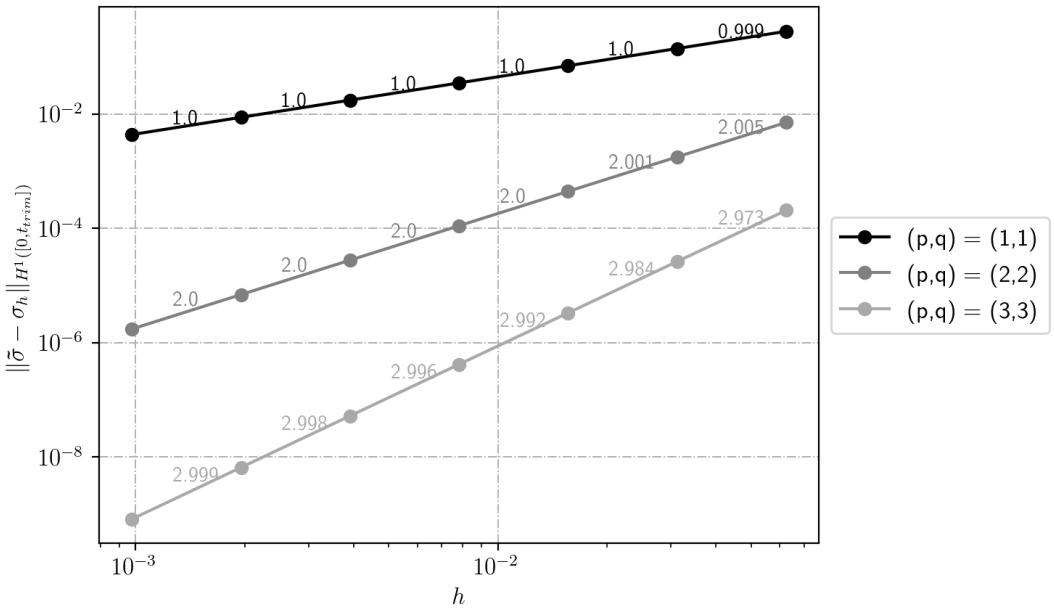



**Figure 4.13:** The error of $\sigma_h$ in the $H^1$ norm for extended B-spline basis functions of order 1,2 and 3.

In Figure 4.14, we observe a significant reduction in the condition number of the Galerkin matrix compared to the standard B-spline spaces. The condition number now follows an order of $\kappa_2(A_h) = \mathcal{O}(h^{-1})$. This is due to the property of the extended B-splines, that is the condition number of the resulting Galerkin matrix is irrespective of the location of the trimming point.

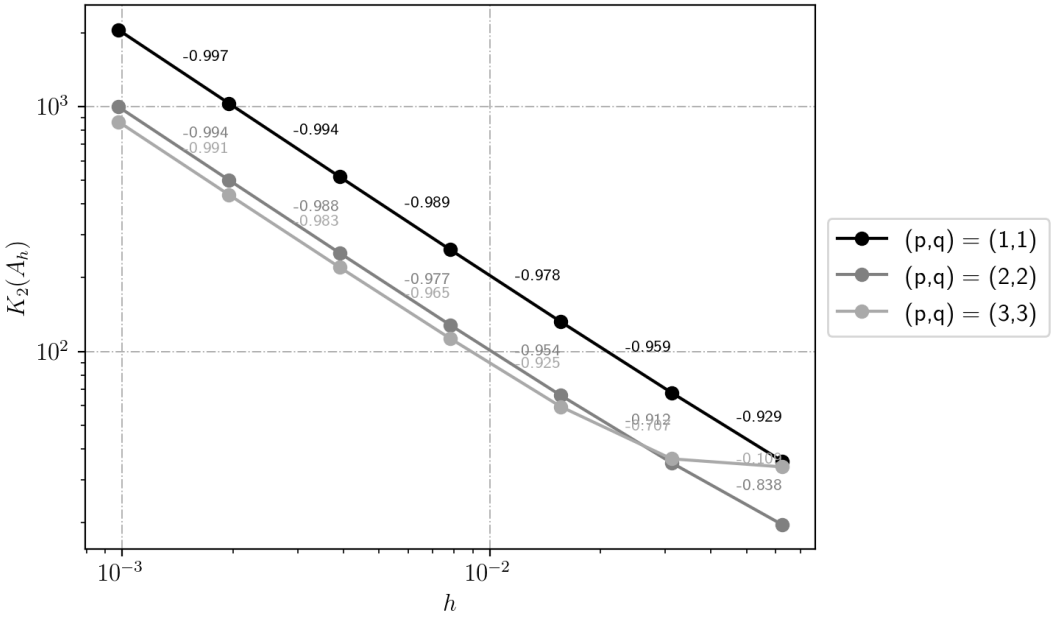



**Figure 4.14:** The condition number of the Galerkin matrix is calculated using the extended B-spline space for trial and test space approximation for various values of the mesh parameter $h$. Additionally, the slopes are displayed.

To highlight this property further, Figure 4.15 compares the condition number of the Galerkin matrix associated with the constructed extended B-spline space in section 3.1 with that of the standard B-spline spaces. The mesh parameter is fixed to $h = \frac{1}{8}$ while the trimming location is varied.

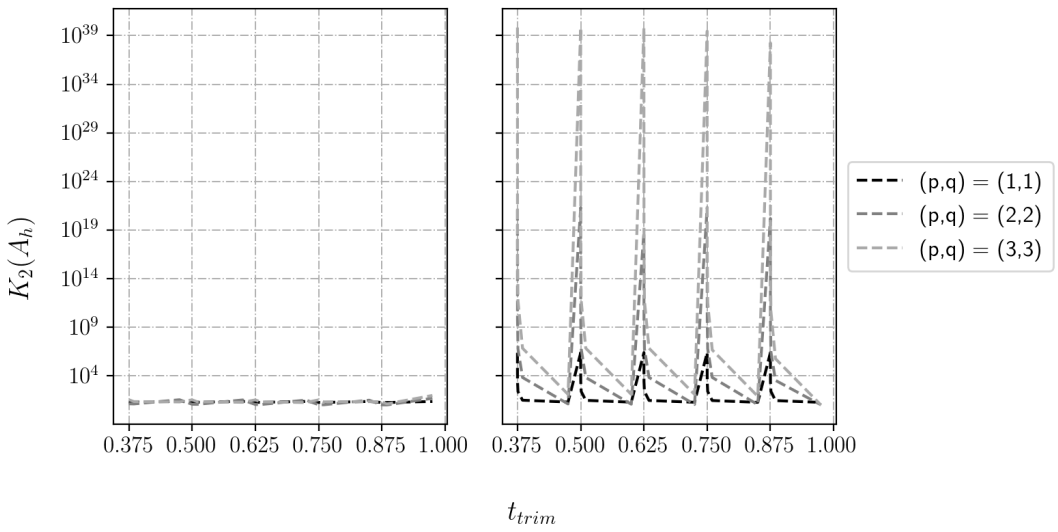



**Figure 4.15:** The condition number of the Galerkin matrix is computed using the extended B-spline space for trial and test space approximation. We evaluate it for various values of the trimming point $t_{\text{trim}}$ on the left and for the standard B-spline spaces on the right. The parameter $t_{\text{trim}}$ represents the trimming point.

Lastly we assess the stability of the method numerically as discussed in section 2.4. In Figure 4.16 the discrete inf-sup constant, $\gamma_h$ for both the standard B-spline spaces and extended B-spline spaces is shown for various mesh parameters $h$. We observe that the discrete inf-sup constant grows therefore the test is passed and we can say that the inf-sup condition is satisfied.

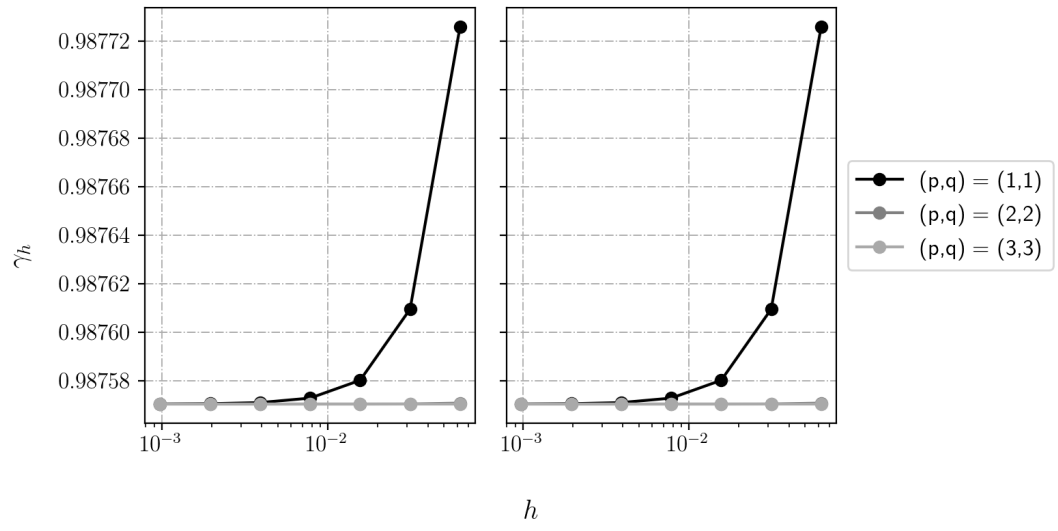



**Figure 4.16:** The condition number of the Galerkin matrix with respect to the mesh parameter $h$. In addition the slopes are shown.

## 4.3. Immersed Mixed Vector Laplacian Problem

In this section the convergence, conditioning and stability of the finite element spaces constructed in section 3.2 are assessed numerically by solving a vector Laplacian problem. First the problem is solved using the standard b-spline spaces and thereafter using the constructed spaces in subsection. We consider the same domain as mentioned in section 4.1. We considered the vector Laplacian with the following boundary conditions

$$
\begin{cases}
\nabla \times \sigma - \nabla\nabla \cdot \boldsymbol{u} = \begin{bmatrix} 4\pi^2 \cos \pi y \sin \pi x \\ 2\pi^2 \sin \pi y \cos \pi x \end{bmatrix} & (x, y) \in \Omega \setminus D, \\
\sigma = \nabla \times \boldsymbol{u} & (x, y) \in \Omega \setminus D, \\
\sigma = 0, & (x, y) \in \partial\Omega, \\
\boldsymbol{u} \cdot \boldsymbol{n} = 0, & (x, y) \in \partial\Omega, \\
\nabla \cdot \boldsymbol{u} = h := \nabla \cdot \tilde{\boldsymbol{u}}, & (x, y) \in \partial D, \\
\boldsymbol{u} \cdot \boldsymbol{s} = g := \tilde{\boldsymbol{u}} \cdot \boldsymbol{s}, & (x, y) \in \partial D,
\end{cases} \tag{4.4}
$$

where $\nabla \times \sigma = \begin{bmatrix} \sigma_y \\ \sigma_x \end{bmatrix}$, $\nabla \times \boldsymbol{u} = \frac{\partial u_2}{\partial x} - \frac{\partial u_1}{\partial y}$, $\mathbf{n}$ is the inward pointing normal vector of the curve $\boldsymbol{C}(u)$ (note that here the variable $u$ is the local coordinate of the curve) and $\boldsymbol{s}$ is the tangent vector of the curve $\boldsymbol{C}(u)$. The exact solution is given by $\tilde{\boldsymbol{u}} = \begin{bmatrix} 2 \cos \pi y \sin \pi y \\ \sin \pi y \cos \pi x \end{bmatrix}$ and $\sigma = \nabla \times \tilde{\boldsymbol{u}} = \pi \sin \pi y \sin \pi x$. For completeness we also state the weak formulation.

Find $(\boldsymbol{u}, \sigma) \in H_0(\mathrm{Div}; \Omega \setminus D) \times H_0^1(\Omega \setminus D)$ such that

$$
\begin{aligned}
\int_{\Omega \setminus D} \sigma\tau - \boldsymbol{u} \cdot (\nabla \times \tau) \mathrm{d}x &= \int_{\partial D} \tau g \, \mathrm{d}s && \forall \tau \in H_0^1(\Omega \setminus D), \\
\int_{\Omega \setminus D} (\nabla \times \sigma) \cdot \boldsymbol{v} + \nabla \cdot \boldsymbol{v} \nabla \cdot \boldsymbol{u} \mathrm{d}x &= \int_{\partial D} h \boldsymbol{v} \cdot \boldsymbol{n} \, \mathrm{d}s + \int_{\Omega \setminus D} \boldsymbol{f} \cdot \boldsymbol{v} \mathrm{d}x && \forall \boldsymbol{v} \in H_0(\mathrm{Div}; \Omega \setminus D),
\end{aligned} \tag{4.5}
$$

where $H_0^1(\Omega \setminus D) = \{u \in H^1(\Omega \setminus D) : u = 0 \text{ on } \partial\Omega\}$ and $H_0(\mathrm{Div}; \Omega \setminus D) = \{\boldsymbol{u} \in H(\mathrm{Div}; \Omega \setminus D) : \boldsymbol{u} \cdot \boldsymbol{n} = 0\}$. The Galerkin approximations for the test and trial spaces are chosen in a structure preserving manner as discussed in section 2.4. More specifically the spline space $\mathbb{S}^{p,q}_{p-1,q-1}$ and $\mathbb{S}^{p,q-1}_{p-1,q-2} \times \mathbb{S}^{p-1,q}_{p-2,q-1}$ of order $p = q = 1, 2$ and 3 are used as trial and test spaces for $H^1(\Omega \setminus D)$ and $H(\mathrm{Div}; \Omega \setminus D)$ respectively together with proper numerical integration on the trimmed elements and proper enforcement of the boundary conditions. The convergence order in the $L^2$ norm for both $\sigma_h$ and $\boldsymbol{u}_h$ is shown in Figure 4.17 and in the $H^1(\Omega \setminus D)$ and $H(\mathrm{Div}; \Omega \setminus D)$ norm for $\sigma_h$ and $\boldsymbol{u}_h$ respectively in Figure 4.18.

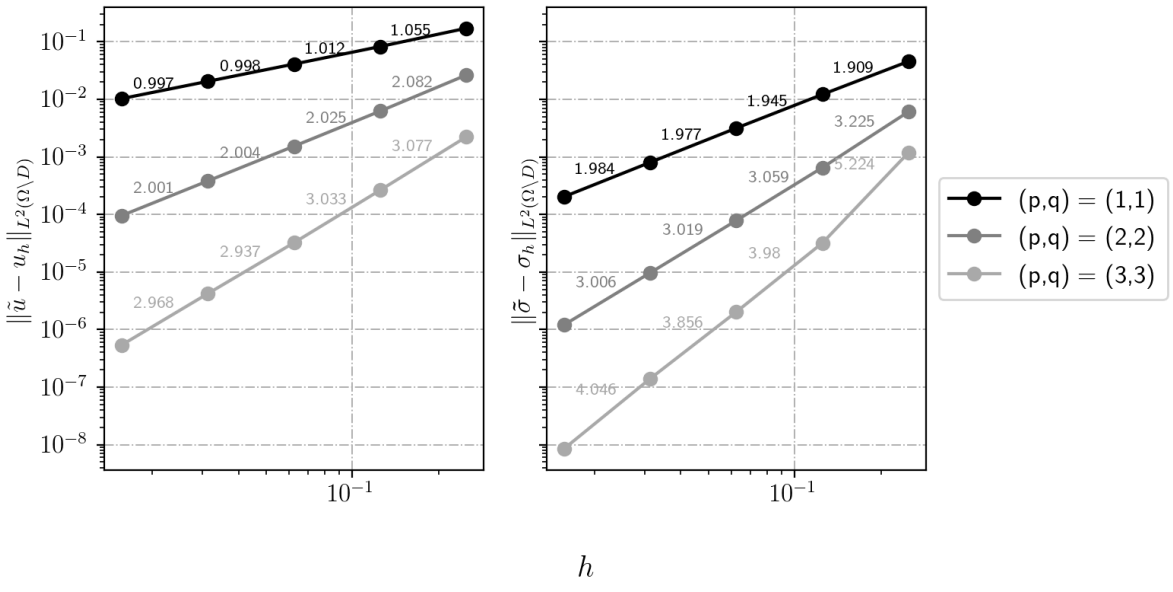



**Figure 4.17:** $L^2$ norm of the error for B-spline basis functions of order 1,2 and 3 in both coordinate directions. Left the $L^2$ norm of the error of $\boldsymbol{u}$ is shown and at the right for $\sigma$.

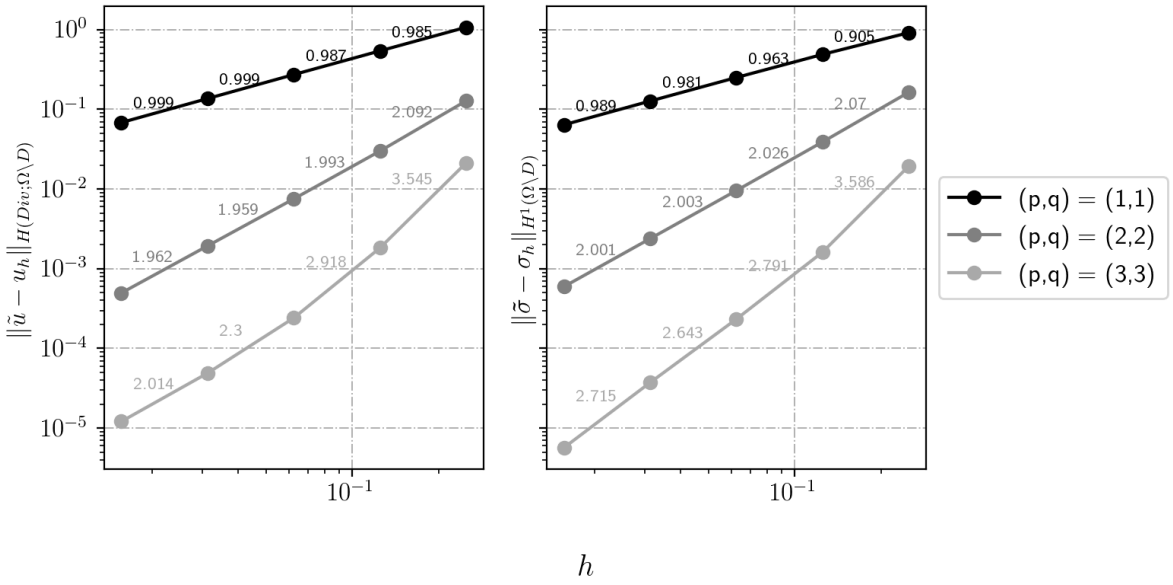



**Figure 4.18:** On the left, we display the $H(\text{Div})$ norm of the error for $\boldsymbol{u}$, while on the right, we show the $H^1$ norm of the error for $\sigma$. Both graphs show B-spline basis functions of order 1, 2, and 3 in both coordinate directions.

Optimal convergence order is observed when refining the mesh parameter $h$ for both $\boldsymbol{u}_h$ and $\sigma_h$ for $p = q = 1$ and 2. For $p = q = 3$ we do not have optimal convergence this is due to the condition number of the Galerkin matrix being of the order $10^{18}$ and as a result the round off error becomes larger and thereby reducing the numerical accuracy. Next we consider the condition number of the Galerkin matrix which is shown in Figure 4.19. The condition numbers grow as the mesh parameter is refined and order of the basis function is increased. This is because of the unstable space we have chosen. More specifically there are basis functions with small support due to the trimming of the domain.

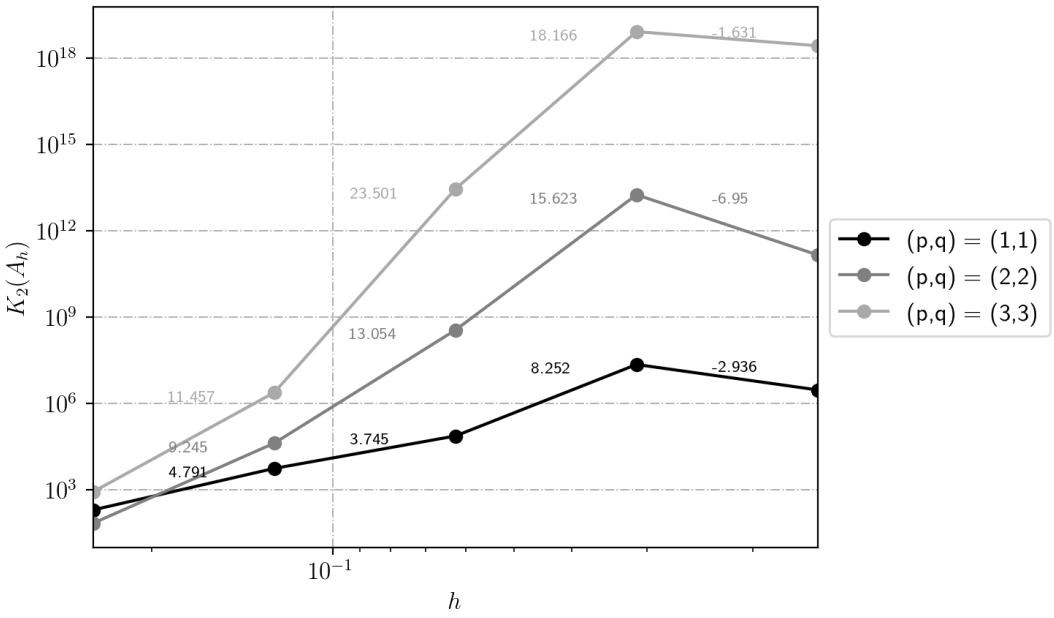



**Figure 4.19:** The condition number of the Galerkin matrix with respect to the mesh parameter $h$. In addition the slopes are shown.

Next, we choose the test and trial spaces as constructed in section 3.2. The convergence order in the $L^2$ norm for both $u_h$ and $\sigma_h$, and $H^1$ norm for $\sigma_h$ are shown in Figure (4.20) and (4.21) for basis functions

of order one. We do not observe optimal convergence order. For the order two and three the Galerkin matrix is singular. This could be due to multiple reasons. First of all this could be due to the local construction. Since there is a possibility that an extension coefficient, $e_{i,j}$, for some stable basis function with index $i$ and unstable basis function with index $j$ has a different values for different local extension matrices constructed. Another possible reason is that we have only included the trivial differences in the matrix $\tilde{D}$.

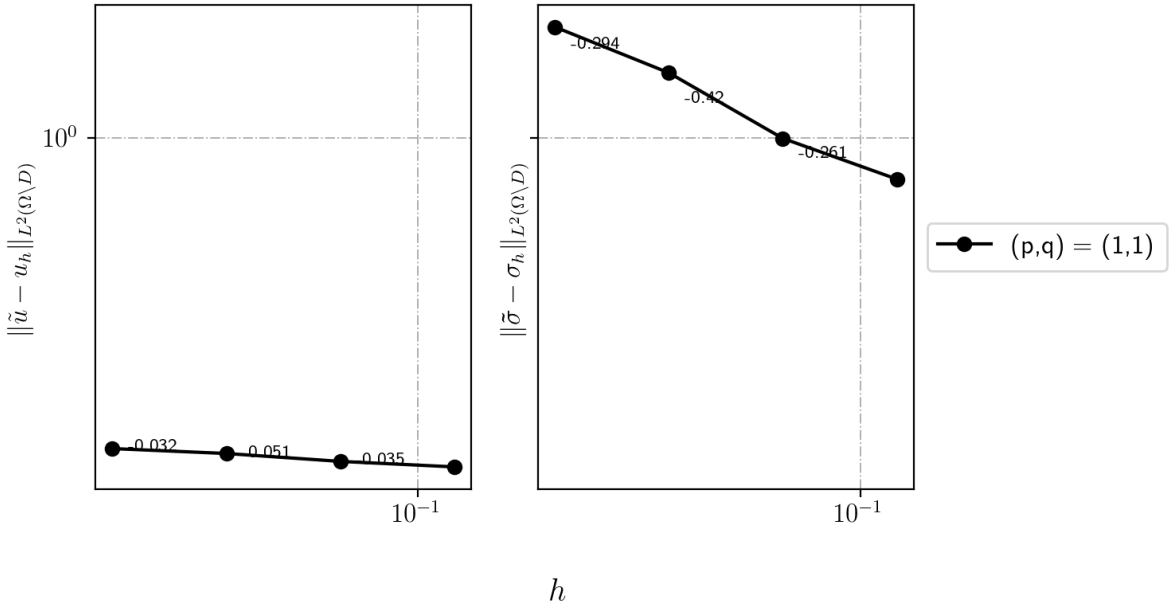



**Figure 4.20:** $L^2$ norm of the error for B-spline basis functions of order 1,2 and 3 in both coordinate directions. Left the $L^2$ norm of the error of $\boldsymbol{u}$ is shown and at the right for $\sigma$.

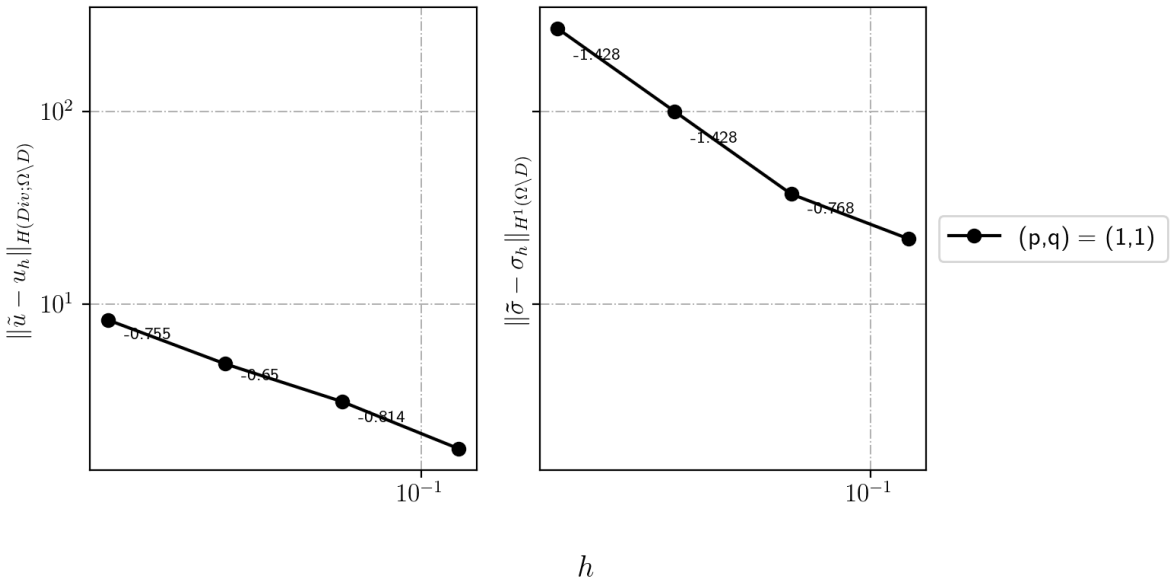



**Figure 4.21:** On the left, we display the $H^1$ norm of the error for $\sigma$, while on the right, we show the $H(\mathrm{Div})$ norm of the error for $\boldsymbol{u}$. Both graphs show B-spline basis functions of order 1, 2, and 3 in both coordinate directions.

For completeness, in Figure 4.22, the condition number of the associated Galerkin matrix is shown for various mesh parameters $h$. We observe that for a finer mesh the condition number becomes very large.

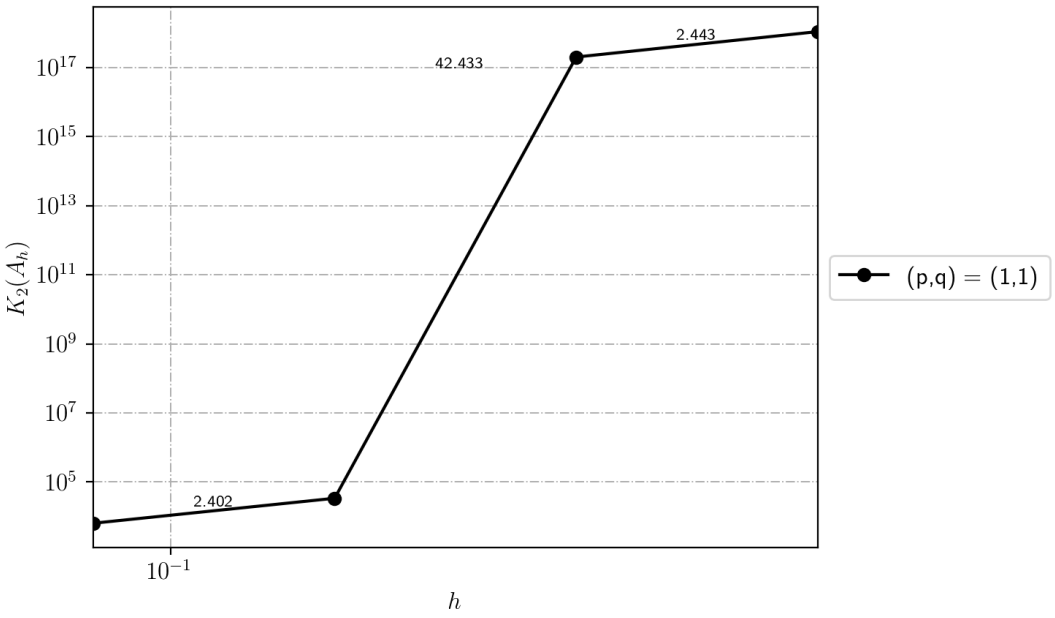



**Figure 4.22:** The condition number of the Galerkin matrix with respect to the mesh parameter $h$. In addition the slopes are shown.

# 5

# Conclusion and Recommendations

In this chapter the conclusion are first discussed in section 5.1 based on the results of chapter 4 and thereafter a recommendation is given in section 5.2. But first the research questions are promptly answered:

> *RQ 1: What extended B-spline spaces form a structure preserving subcomplex of the B-spline spaces satisfying the de Rham complex in one and two dimensional domains?*
>
> *A 1: For the one-dimensional domain a structure preserving subcomplex is constructed and for the two-dimensional domain a structure preserving subcomplex is constructed for the 0- and 1-form.*
>
> *RQ 2: Is the condition number of the Galerkin matrix associated to the constructed structure preserving finite element spaces irrespective of the location of the trimming curve?*
>
> *A 2: For the one-dimensional domain the structure preserving subcomplex is irrespective of the location of the trimming curve (numerically assessed). For the two-dimensional domain this is not the case.*
>
> *RQ 3: Do the constructed structure preserving finite element spaces satisfy the standard error estimates for elliptic problems?*
>
> *A 3: For the one-dimensional domain the structure preserving subcomplex satisfy the standard error estimates (numerically assessed). Moreover the spaces satisfy (problem related) the inf-sup condition (numerically assessed). For the two-dimensional domain the standard error estimates are not satisfied.*

## 5.1. Conclusion

In section 3.1, we construct a structure-preserving subcomplex for the de Rham complex in 1D, focusing on immersed problems. Subsection 4.2 contains the numerical assessment of this subcomplex. Our findings indicate optimal convergence rates in both the $L^2$ and $H^1$ norms, along with stability ensured by the discrete inf-sup constant being positive. Furthermore, the associated Galerkin matrix exhibits a condition number that remains independent of the trimming curve's location. Despite the impressive results, the immersed problems are somewhat redundant in 1D. An alternative approach involves adding the trimmed grid cell to its neighboring untrimmed grid cell, effectively avoiding conditioning problems. The 2D case is more interesting. In section 3.2, we construct a structure preserving subcomplex in 2D but only for the 1-form, due to the unsatisfactory numerical results. We do not observe convergence for basis functions of order 1. For basis functions of order 2 and 3 we obtain a linear system that is singular. This could be due to the local construction. There is a possibility that an extension coefficient, $e_{i,j}$ for some stable basis function with index $i$ and unstable basis function with index $j$ has different values for different the local extension matrices constructed.

## 5.2. Recommendations

As mentioned earlier, dealing with immersed problems in 1D is superfluous. However, there's still room to explore what extended hierarchical B-spline spaces form a structure-preserving subcomplex for the hierarchical B-splines that satisfy the de Rham complex. Moreover, it is worthwhile to evaluate how well things converge and remain stable across different problems. When it comes to immersed problems in 2D, it is prudent to first consider the local approach for vertical and skewed cuts. This should happen before looking into more complicated trimming curves like circles or irregularly shaped curves as done in this Master thesis. For these kinds of cuts, it is easier to determine the correct differences that appear in the matrix $\tilde{D}$ (as shown in section 3.2). Additionally, one must still formulate the local extension matrix for the 2-form. But for these specific cuts involving higher-order B-spline basis functions, we must use extended B-splines on non-uniform knot vectors. The simpler method based on the indices of the B-spline basis functions, which we discussed in subsection 2.3.2, is no longer applicable. Another possible approach is to construct the extension matrices globally. This way, we avoid the problem of having different extension coefficients for the same stabilization.

# References


[1] Jiri Blazek. *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, 2015.

[2] Atif Masood. *velocity plot Apollo Spacecraft CFD Modeling*. 2022. URL: https://s3-eu-west-1.amazonaws.com/fetchcfd/resized/file-1663110060046.jpg.

[3] Atif Masood. *pressure plot Apollo Spacecraft CFD Modeling*. 2022. URL: https://s3-eu-west-1.amazonaws.com/fetchcfd/resized/file-1663110055954.jpg.

[4] Joel H Ferziger, Milovan Peri'c, and Robert L Street. *Computational methods for fluid dynamics*. Vol. 3. Springer, 2002.

[5] Hongfei Ye et al. "Hybrid grid generation for viscous flow simulations in complex geometries". In: *Advances in Aerodynamics* 2 (2020), pp. 1–18.

[6] Jamshid Parvizian, Alexander Düster, and Ernst Rank. "Finite cell method: h-and p-extension for embedded domain problems in solid mechanics". In: *Computational Mechanics* 41.1 (2007), pp. 121–133.

[7] Erik Burman and Peter Hansbo. "Fictitious domain finite element methods using cut elements: I. A stabilized Lagrange multiplier method". In: *Computer Methods in Applied Mechanics and Engineering* 199.41-44 (2010), pp. 2680–2686.

[8] Erik Burman and Peter Hansbo. "Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method". In: *Applied Numerical Mathematics* 62.4 (2012), pp. 328–341.

[9] Erik Burman et al. "CutFEM: discretizing geometry and partial differential equations". In: *International Journal for Numerical Methods in Engineering* 104.7 (2015), pp. 472–501.

[10] J. Austin Cottrell, Thomas J. R. Hughes, and Yuri Bazilevs. *Isogeometric Analysis*. Wiley, Aug. 2009. ISBN: 9780470748732. DOI: 10.1002/9780470749081. URL: https://onlinelibrary.wiley.com/doi/book/10.1002/9780470749081.

[11] Ruben Sevilla, Sonia Fernández-Méndez, and Antonio Huerta. "NURBS-Enhanced Finite Element Method (NEFEM) A Seamless Bridge Between CAD and FEM". In: *Arch Comput Methods Eng* 18 (2011), pp. 441–484. DOI: 10.1007/s11831-011-9066-5.

[12] P. Antolin and T. Hirschler. "Quadrature-free immersed isogeometric analysis". In: *Engineering with Computers* (Oct. 2022). ISSN: 14355663. DOI: 10.1007/s00366-022-01644-3.

[13] Alireza Abedian et al. "The finite cell method for the J2 flow theory of plasticity". In: *Finite Elements in Analysis and Design* 69 (2013), pp. 37–47. ISSN: 0168-874X. DOI: https://doi.org/10.1016/j.finel.2013.01.006. URL: https://www.sciencedirect.com/science/article/pii/S0168874X13000152.

[14] Ivo Babuška. "The finite element method with penalty". In: *Mathematics of computation* 27.122 (1973), pp. 221–228.

[15] Ivo Babuška. "The finite element method with Lagrangian multipliers". In: *Numerische Mathematik* 20.3 (1973), pp. 179–192.

[16] Joachim Nitsche. "Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind". In: *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*. Vol. 36. 1. Springer. 1971, pp. 9–15.

[17] Martin Ruess et al. "Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method". In: *International Journal for Numerical Methods in Engineering* 95.10 (2013), pp. 811–846.

[18] Yujie Guo and Martin Ruess. "Weak Dirichlet boundary conditions for trimmed thin isogeometric shells". In: *Computers & Mathematics with Applications* 70.7 (2015), pp. 1425–1440.

[19] Frits de Prenter et al. “Stability and conditioning of immersed finite element methods: analysis and remedies”. In: (Aug. 2022). URL: http://arxiv.org/abs/2208.08538.

[20] Klaus Höllig. *Finite element methods with B-splines*. SIAM, 2003.

[21] Ulrich Reif, Joachim Wipper, and Siam J Numer Anal. *WEIGHTED EXTENDED B-SPLINE APPROXIMATION OF DIRICHLET PROBLEMS **. 2001, pp. 442–462. URL: http://www.siam.org/journals/sinum/39-2/37320.html.

[22] Benjamin Marussig, René Hiemstra, and Thomas J.R. Hughes. “Improved conditioning of isogeometric analysis matrices for trimmed geometries”. In: *Computer Methods in Applied Mechanics and Engineering* 334 (June 2018), pp. 79–110. ISSN: 0045-7825. DOI: 10.1016/J.CMA.2018.01.052.

[23] A. Buffa, C. de Falco, and G. Sangalli. “IsoGeometric Analysis: Stable elements for the 2D Stokes equation”. In: *International Journal for Numerical Methods in Fluids* 65 (11-12 Apr. 2011), pp. 1407–1422. ISSN: 02712091. DOI: 10.1002/fld.2337. URL: https://onlinelibrary.wiley.com/doi/10.1002/fld.2337.

[24] A. Buffa et al. “Isogeometric discrete differential forms in three dimensions”. In: *SIAM Journal on Numerical Analysis* 49 (2 2011), pp. 818–844. ISSN: 00361429. DOI: 10.1137/100786708.

[25] Douglas Arnold, Richard Falk, and Ragnar Winther. “Finite element exterior calculus: from Hodge theory to numerical stability”. In: *Bulletin of the American mathematical society* 47.2 (2010), pp. 281–354.

[26] Douglas N Arnold. *Finite element exterior calculus*. SIAM, 2018.

[27] Thomas Frachon et al. “A divergence preserving cut finite element method for Darcy flow”. In: *arXiv preprint arXiv:2205.12023* (2022).

[28] Tuong Hoang et al. “Mixed Isogeometric Finite Cell Methods for the Stokes problem”. In: *Computer Methods in Applied Mechanics and Engineering* 316 (Apr. 2017), pp. 400–423. ISSN: 00457825. DOI: 10.1016/j.cma.2016.07.027.

[29] Andre Massing et al. “A stabilized Nitsche fictitious domain method for the Stokes problem”. In: (June 2012). URL: http://arxiv.org/abs/1206.1933.

[30] Les Piegl and Wayne Tiller. *The NURBS book*. Springer Science & Business Media, 1996.

[31] Carl De Boor. *A practical guide to splines*. Vol. 27. springer-verlag New York, 1978.

[32] Maurice G Cox. “The numerical evaluation of B-splines”. In: *IMA Journal of Applied mathematics* 10.2 (1972), pp. 134–149.

[33] Carl De Boor. “On calculating with B-splines”. In: *Journal of Approximation theory* 6.1 (1972), pp. 50–62.

[34] Benjamin Marussig and Thomas J. R. Hughes. “A Review of Trimming in Isogeometric Analysis: Challenges, Data Exchange and Simulation Aspects”. In: *Archives of Computational Methods in Engineering* 25 (4 Nov. 2018), pp. 1059–1127. ISSN: 1134-3060. DOI: 10.1007/s11831-017-9220-9. URL: http://link.springer.com/10.1007/s11831-017-9220-9.

[35] Kestutis Jankauskas. “Time-Efficient NURBS Curve Evaluation Algorithms”. In: Apr. 2010.

[36] Thomas JR Hughes. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2000.

[37] Benjamin Marussig et al. “Stable isogeometric analysis of trimmed geometries”. In: *Computer Methods in Applied Mechanics and Engineering* 316 (Apr. 2017), pp. 497–521. ISSN: 00457825. DOI: 10.1016/j.cma.2016.07.040. URL: https://linkinghub.elsevier.com/retrieve/pii/S0045782516308222.

[38] Anand Embar, John Dolbow, and Isaac Harari. “Imposing dirichlet boundary conditions with Nitsche’s method and spline-based finite elements”. In: *International Journal for Numerical Methods in Engineering* 83 (7 Aug. 2010), pp. 877–898. ISSN: 00295981. DOI: 10.1002/nme.2863.

[39] Arthur H Stroud and Don Secrest. *Gaussian Quadrature Formulas: By AH Stroud and Don Secrest*. Prentice-Hall, 1966.

[40] Daniele Boffi and Lucia Gastaldi. “A finite element approach for the immersed boundary method”. In: *Computers & structures* 81.8-11 (2003), pp. 491–501.

[41] Dominique Chapelle and Klaus-Jürgen Bathe. “The inf-sup test”. In: *Computers & structures* 47.4-5 (1993), pp. 537–545.

[42] R. R. Hiemstra et al. “High order geometric methods with exact conservation properties”. In: *Journal of Computational Physics* 257 (PB Jan. 2014), pp. 1444–1471. ISSN: 10902716. DOI: 10.1016/j.jcp.2013.09.027.

[43] Hyun-Jung Kim, Yu-Deok Seo, and Sung-Kie Youn. “Isogeometric analysis for trimmed CAD surfaces”. In: *Computer Methods in Applied Mechanics and Engineering* 198 (37-40 Aug. 2009), pp. 2982–2995. ISSN: 00457825. DOI: 10.1016/j.cma.2009.05.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0045782509001856.

[44] Thomas W Sederberg and Tomoyuki Nishita. “Curve intersection using Bézier clipping”. In: *Computer-Aided Design* 22.9 (1990), pp. 538–549.

# A

# Grid classification

Before constructing the extended B-spline spaces, the grid cells must be classified in a preprocessing step. The grid cells are classified as inner, exterior, or boundary grid cells. There are different ways to approach this problem. If one has already determined the intersection points between the mesh and the trimming curve then this can be exploited to classify the grid cells in a straightforward manner. However we determine the intersection points after the grid classification and therefore consider an alternative approach. We classify the grid cells in two steps. First, the vertices of the grid cells are classified, and thereafter the grid cell is classified using the classification of its vertices. Each vertex is classified using the Raycasting algorithm (see Algorithm 1) using a polygonal approximation of the trimming curve through sampling. The Raycasting algorithm checks if a point (in this case, a vertex) is inside or outside the polygon given a point and a polygon (in this case, a polygon approximation of the trimming curve).

**Algorithm 1** Raycasting Algorithm

```
Require: Polygon, Point
  Count ← 0
  for Side in Polygon do
      if Ray_intersect_segment(Point,Side) then
          Count ← Count + 1
      end if
  end for
  if Is_odd(Count) then
      return Inner Point
  else
      return Outer Point
  end if
```

Figure A.1 shows the classification of the vertices for the trimming curve from Figure 2.3 (after translating and re-scaling), which is a centered circle with radius 0.25. Additionally, rays are drawn for three specific vertices colored green, blue, and red. The Raycasting algorithm counts the number of times the ray, which starts from the vertex, intersects the trimming curve, if it is a even number (or 0) the corresponding vertex is classified as an outer vertex and if it is an odd number the vertex is classified as an inner vertex. For instance the green ray intersects the trimming curve twice, therefore it is classified as an inner vertex. The blue ray intersects the trimming curve once therefore its corresponding vertex is classified as an outer vertex.

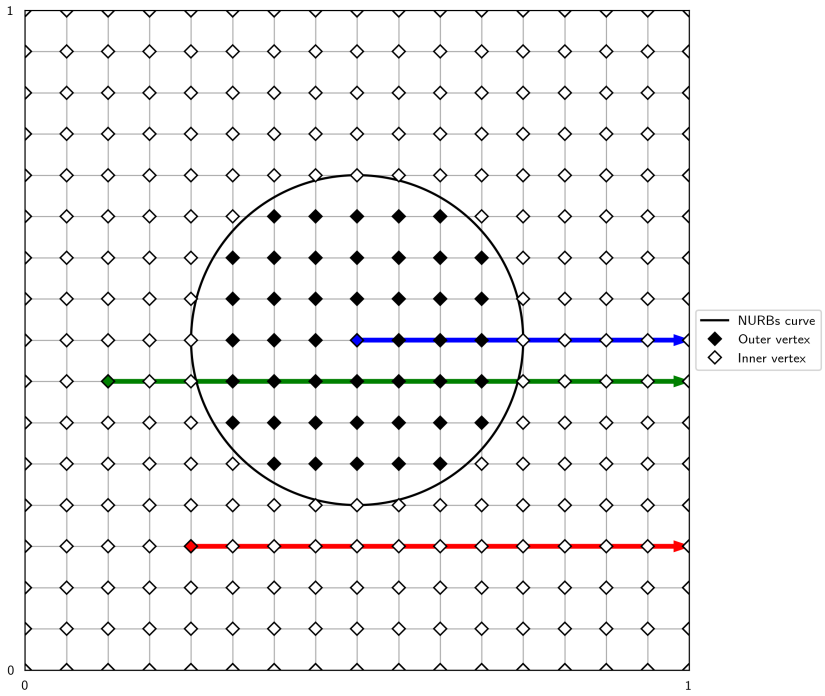



**Figure A.1:** Classification of the vertices of each grid cell. The vertices are classified as inner or outer, and colored white or black respectively. Additionally, a green, red, and blue ray is drawn for the three vertices of each color.

The accuracy of this approach depends on the number of sampling points used to approximate the trimming curve as a polygon. To address this problem, a hierarchical approach is used. The vertices are first classified by the Raycasting algorithm using a polygon generated by $s_0 \in \mathbb{N}$ sampling points. Then, the vertices are classified again using $s_1 > s_0$ sampling points. Next, the classification of the vertices at each level is compared. If the classification for each vertex is the same, then the classification process is stopped, and the classification of the vertices is returned. If not, this process is repeated until a classification is returned or the prescribed level is reached. Using the classification of the vertices the grid cells are classified as follows: if all the vertices of a grid cell are classified as inner vertices then the grid cell is classified as a interior grid cell, if all the vertices are classified outer vertices then the grid cell is classified as outer grid cell and if the vertices are classified as either inner or outer grid cells then the grid cell is classified as a boundary grid cell. In Figure A.2 the classification of the grid cell is shown for the vertex classification of Figure A.1.

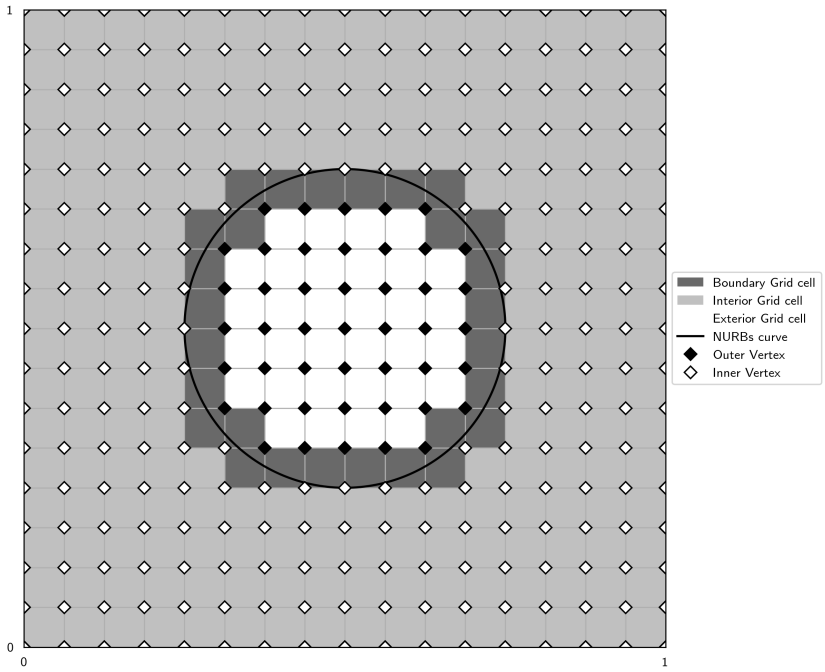



**Figure A.2:** Grid classification and vertex classification. The vertices are classified as inner or outer, and colored white or black respectively. The boundary, interior and exterior gridcells are colored dark grey, light grey and white respectively

# B

# Intersection points

To perform numerical integration over the trimming curve and trimmed elements, we require the intersection points between the mesh and trimming curve. Throughout this section, we assume that the grid cells are classified. We follow the approach proposed by [43]. Since the grid cells are classified, we know which specific grid cells contain the intersection points, i.e., the boundary grid cells. Figure B.1 shows a specific example of a trimmed grid cell with two intersection points.

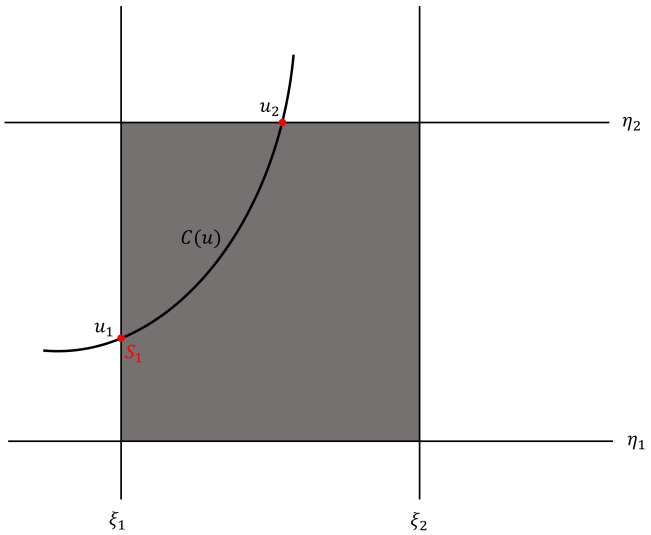



**Figure B.1:** Boundary grid cell.

We consider the intersection point $S_1$. At this intersection point $\xi = \xi_1$ is known and $u_1$ and $\eta$ are unknown. To find the unknowns we solve the following set of equations using Newton's method.

$$\begin{cases} \mathbf{C}_\xi(u) - \xi_1 = 0, \\ \mathbf{C}_\eta(u) - \eta = 0. \end{cases} \tag{B.1}$$

Since we are working with a closed trimming curve, equation (B.1) can have multiple solutions. To avoid this problem, we use a divide and conquer approach. This approach involves splitting the trimming curve, which is a NURBS curve, into several Bézier curves by curve splitting the NURBS curve at every knot value as mentioned in section 2.2. Afterward, the NURBS curve, $\mathbf{C}(u)$, in equation (B.1) is replaced by the closest Bézier curve. The closest Bézier curve is determined by calculating the distance between one of the vertices of the boundary grid cell and the Bézier curves evaluated at the midpoint of their knot vector. Figure B.2 shows the intersection points for the trimming curve from Figure 2.3 (after translating and rescaling) using the above approach. Another approach, which is interesting but

not used in this Master's thesis, is referred to as Bézier clipping [44]. Bézier clipping is a more robust way of finding the intersection points.

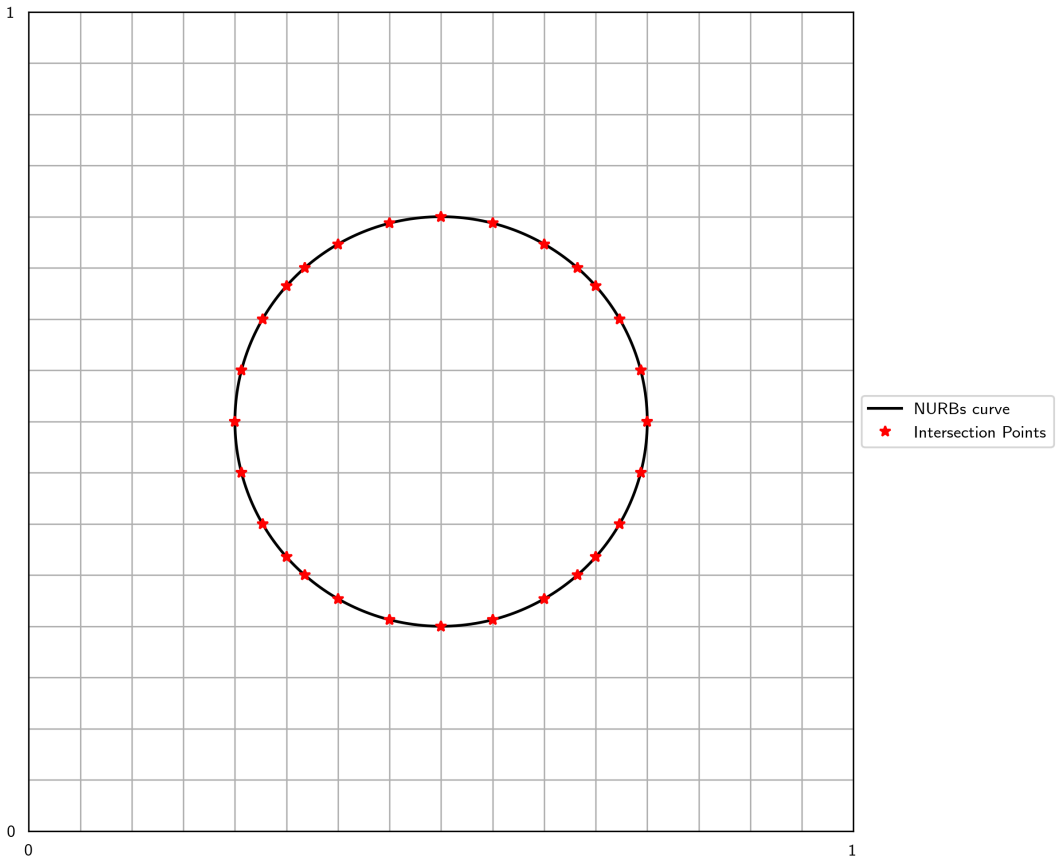



**Figure B.2:** Intersection points for a centered circle with radius 0.25.

# C Proof

We assume a domain $\Omega = [a, b]$ that is trimmed by the domain $D = (t_{trim}, b]$, where $t_{trim}$ is the trimming location. Furthermore we assume that $n + 1$ basis functions have nonzero support on $\Omega \setminus D$ and are of order $p$, of which $n$ are stable basis functions. The unstable basis functions are stabilized by choosing the closest $p + 1 << n$ stable basis functions, therefore $j - l = p + 1$ (, where $l$ is the index of the most left basisfunction in the index set $I(j)$). Since we assume trimming from the right this simplifies the expression for the extension coefficients. That is the extension coefficients:

$$e_{i,j}^{q} = \prod_{\substack{\mu=0 \\ \ell+\mu\neq i}}^{q} \frac{j-\ell-\mu}{i-\ell-\mu},$$

where $q$ is the order of the basis function and $i, j$ the indices of the stable and unstable basis functions respectively can be rewritten as:

$$e_{\alpha}^{q} = \prod_{\substack{\mu=0 \\ \alpha-\mu\neq 0}}^{q} \frac{q+1-\mu}{\alpha-\mu},$$

where $\alpha := i - l$ To Proof : $DE^0I = E^1$, where

$$D = \begin{bmatrix} -1 & 1 & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & -1 & 1 \end{bmatrix} \in \mathbb{R}^{n\times(n+1)}, \qquad E^0 = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & \ddots & \\ & & & & & 1 \\ 0 & \dots & 0 & e_0^p & \dots & e_p^p \end{bmatrix} \in \mathbb{R}^{(n+1)\times n},$$

$$I = \begin{bmatrix} 0 & \dots & \dots & \dots & 0 \\ 1 & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & 0 \\ 1 & \dots & \dots & \dots & 1 \end{bmatrix} \in \mathbb{R}^{n\times(n-1)}, \qquad E^1 = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & \ddots & \\ & & & & & 1 \\ 0 & \dots & 0 & e_0^{p-1} & \dots & e_{p-1}^{p-1} \end{bmatrix} \in \mathbb{R}^{n\times(n-1)}.$$

Proof:

$$DE^0 I = \begin{bmatrix} -1 & 1 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & \ddots & \ddots & & & & \\ & & & \ddots & \ddots & & & \\ & & & & \ddots & \ddots & & \\ & & & & & \ddots & \ddots & \\ & & & & & & -1 & 1 \\ 0 & \dots & 0 & e_0^p & e_1^p & \dots & e_{p-1}^p & e_p^p - 1 \end{bmatrix} I$$

$$= \begin{bmatrix} & 1 & & & & & & \\ & & \ddots & & & & & \\ & & & \ddots & & & & \\ & & & & \ddots & & & \\ & & & & & \ddots & & \\ & & & & & & \ddots & \\ & & & & & & & 1 \\ (\sum_{\alpha=0}^p e_\alpha^p) - 1 & \dots & (\sum_{\alpha=0}^p e_\alpha^p) - 1 & (\sum_{\alpha=0}^p e_\alpha^p) - 1 & (\sum_{\alpha=1}^p e_\alpha^p) - 1 & \dots & e_p^p - 1 \end{bmatrix}$$

Note that $(\sum_{\alpha=0}^p e_\alpha^p) = 1$. It remain to show that:

$$\begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \ddots & & & & \\ & & & \ddots & & & \\ & & & & \ddots & & \\ & & & & & 1 & \\ 0 & \dots & 0 & (\sum_{\alpha=1}^p e_\alpha^p) - 1 & \dots & e_p^p - 1 \end{bmatrix} \stackrel{?}{=} \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & 1 & \\ 0 & \dots & 0 & e_0^{p-1} & \dots & e_{p-1}^{p-1} \end{bmatrix}$$

We focus on the last row since the other rows coincide. That is we need to show that:

$$\begin{bmatrix} 0 & \dots & 0 & (\sum_{\alpha=1}^p e_\alpha^p) - 1 & \dots & e_p^p - 1 \end{bmatrix} \stackrel{?}{=} \begin{bmatrix} 0 & \dots & 0 & e_0^{p-1} & \dots & e_{p-1}^{p-1} \end{bmatrix}$$

Comparing each entry of the row gives:

$$\begin{aligned} e_p^p - 1 &= e_{p-1}^{p-1} \\ e_{p-1}^p + e_p^p - 1 = e_{p-2}^{p-1} &\implies e_{p-1}^p + e_{p-1}^{p-1} = e_{p-2}^{p-1} \\ e_{p-2}^p + e_{p-1}^p + e_p^p - 1 = e_{p-2}^{p-1} &\implies e_{p-2}^p + e_{p-2}^{p-1} = e_{p-3}^{p-1} \\ &\vdots \\ \left( \sum_{\alpha=1}^p e_\alpha^p \right) - 1 = e_0^{p-1} &\implies e_1^p + e_1^{p-1} = e_0^{p-1} \end{aligned}$$

This leaves us to show that:

$$\begin{aligned} e_p^p - 1 &= e_{p-1}^{p-1} \\ e_k^p + e_k^{p-1} &= e_{k-1}^{p-1}, \quad k \in \{1, \dots, p-1\} \end{aligned}$$

For the first expression we first show that $e_q^q = q + 1$:

$$
\begin{aligned}
e_q^q &= \prod_{\substack{\mu=0 \\ q-\mu\neq 0}}^{q} \frac{q+1-\mu}{q-\mu} \\
&= \prod_{\mu=0}^{q-1} \frac{q+1-\mu}{q-\mu} \\
&= \frac{\prod_{\mu=0}^{q-1}(q+1-\mu)}{\prod_{\mu=0}^{q-1}(q-\mu)} \\
&= \frac{\prod_{\mu=-1}^{q-2}(q-\mu)}{\prod_{\mu=0}^{q-1}(q-\mu)} \\
&= \frac{\prod_{\mu=0}^{q-1}(q-\mu)}{\prod_{\mu=0}^{q-1}(q-\mu)} \frac{p+1}{1} = q+1
\end{aligned}
$$

Using the above expression we have that:

$$
e_p^p - 1 = p + 1 - 1 = p = e_{p-1}^{p-1}
$$

To proof the second expression we first show that $e_k^p = \frac{p+1}{k} e_{k-1}^{p-1}, \quad k \in \{1, \ldots, p-1\}$ and that $e_k^{p-1} = \frac{k-p-1}{k} e_{k-1}^{p-1}, \quad k \in \{1, \ldots, p-1\}$

$$
\begin{aligned}
e_k^p &= \prod_{\substack{\mu=0 \\ k-\mu\neq 0}}^{p} \frac{q+1-\mu}{k-\mu} \\
&= \prod_{\substack{\mu=-1 \\ k-1-\mu\neq 0}}^{p-1} \frac{p+\mu}{k-1-\mu} \\
&= \frac{p+1}{k} \prod_{\substack{\mu=0 \\ k-1-\mu\neq 0}}^{p-1} \frac{p+\mu}{k-1-\mu} \\
&= \frac{p+1}{k} \prod_{\substack{\mu=0 \\ k-1-\mu\neq 0}}^{p-1} \frac{p+\mu}{k-1-\mu} = \frac{p+1}{k} e_{k-1}^{p-1}
\end{aligned}
$$

$$
\begin{aligned}
e_k^{p-1} &= \prod_{\substack{\mu=0 \\ k-\mu\neq 0}}^{p} \frac{p-\mu}{k-\mu} \\
&= \frac{\prod^{p}_{\substack{\mu=0 \\ k-\mu\neq 0}} (p-\mu)}{\prod^{p}_{\substack{\mu=0 \\ k-\mu\neq 0}} (k-\mu)} \\
&= \frac{\prod^{p}_{\substack{\mu=0 \\ k-\mu\neq 0}} (p-\mu)}{\prod^{p-2}_{\substack{\mu=-1 \\ k-1-\mu\neq 0}} (k-1-\mu)} \\
&= \frac{k-p}{k} \frac{\prod^{p}_{\substack{\mu=0 \\ k-\mu\neq 0}} (p-\mu)}{\prod^{p-1}_{\substack{\mu=0 \\ k-1-\mu\neq 0}} (k-1-\mu)}
\end{aligned}
$$

$$= \frac{k-p}{k} \frac{1}{p-k} \frac{\prod_{\mu=0}^{p} (p-\mu)}{\prod_{\substack{\mu=0 \\ k-1-\mu\neq 0}}^{p-1} (k-1-\mu)}$$

$$= \frac{k-p}{k} \frac{p-k+1}{p-k} \frac{\prod_{\substack{\mu=0 \\ k-1-\mu\neq 0}}^{p-1} (p-\mu)}{\prod_{\substack{\mu=0 \\ k-1-\mu\neq 0}}^{p-1} (k-1-\mu)}$$

$$= \frac{k-p-1}{k} \prod_{\substack{\mu=0 \\ k-1-\mu\neq 0}}^{p-1} \frac{p-\mu}{k-1-\mu} = \frac{k-p-1}{k} e_{k-1}^{p-1}$$

Using the above expression we have that:

$$e_k^p + e_k^{p-1} = \frac{p+1}{k} e_{k-1}^{p-1} + \frac{k-p-1}{k} e_{k-1}^{p-1} = \left( \frac{p+1}{k} + \frac{k-p-1}{k} \right) e_{k-1}^{p-1} = e_{k-1}^{p-1},$$

Which concludes that $DE^0 I = E^1$.

# D

# Auxiliary Results

## D.1. Immersed Mixed Poisson Problem - Two dimensional domain

In this section the convergence and conditioning for several finite sub-spaces are assessed by solving a mixed Poisson problem. First the problem is solved using the standard b-spline spaces and thereafter using the constructed spaces in subsection (3.2). We consider the same domain as mentioned in section 4.1. To asses convergence we use the method of manufactured solution with the following exact solution and right hand side

$$
\begin{aligned}
\tilde{u} &= \cos(2\pi(y-0.5)(x-0.5))((x-0.5)^2+(y-0.5)^2-0.25^2),\\
\tilde{\boldsymbol{\sigma}} &= \nabla\tilde{u},\\
\tilde{f} &= -\nabla\cdot\tilde{\boldsymbol{\sigma}}.
\end{aligned}
$$

Moreover we solve the mixed Poisson problem with the following prescribed boundary conditions

$$
\begin{cases}
\sigma - \nabla u = 0 & (x,y)\in\Omega\setminus D,\\
-\nabla\cdot\sigma = \tilde{f} & (x,y)\in\Omega\setminus D,\\
u(x,y) = 0, & (x,y)\in\partial D,\\
u(0,y) = \tilde{u}(0,y), & y\in[0,1],\\
u(1,y) = \tilde{u}(1,y), & y\in[0,1],\\
\boldsymbol{\sigma}(x,0)\cdot\boldsymbol{n}_{\text{down}} = \tilde{\boldsymbol{\sigma}}(x,0)\cdot\boldsymbol{n}_{\text{down}}, & x\in[0,1],\\
\boldsymbol{\sigma}(x,1)\cdot\boldsymbol{n}_{\text{up}} = \tilde{\boldsymbol{\sigma}}(x,1)\cdot\boldsymbol{n}_{\text{up}}, & x\in[0,1].
\end{cases}
\tag{D.1}
$$

where $\boldsymbol{n}_{\text{down}} = \begin{bmatrix}0\\-1\end{bmatrix}$ and $\boldsymbol{n}_{\text{up}} = \begin{bmatrix}0\\1\end{bmatrix}$. For completeness we state the weak form which is given by

$$
\begin{aligned}
&\text{Find } (u,\boldsymbol{\sigma})\in L^2(\Omega\setminus D)\times H(\text{Div};\Omega\setminus D) \text{ with } \boldsymbol{\sigma}(x,0)\cdot\boldsymbol{n}_{\text{down}} = \tilde{\boldsymbol{\sigma}}(x,0)\cdot\boldsymbol{n}_{\text{down}}\\
&\text{on } (x,y)\in[0,1]\times\{0\} \text{ and } \boldsymbol{\sigma}(x,1)\cdot\boldsymbol{n}_{\text{up}} = \tilde{\boldsymbol{\sigma}}(x,1)\cdot\boldsymbol{n}_{\text{up}}\\
&\text{on } (x,y)\in[0,1]\times\{1\}. \text{ such that}\\
&\int_\Omega \boldsymbol{\sigma}\cdot\tau + \nabla\cdot\boldsymbol{\tau}u\,\mathrm{d}x = \int_0^1 \tilde{u}(0,y)\boldsymbol{\tau}\cdot\begin{bmatrix}-1\\0\end{bmatrix}\,\mathrm{d}y + \int_0^1 \tilde{u}(1,y)\boldsymbol{\tau}\cdot\begin{bmatrix}1\\0\end{bmatrix}\,\mathrm{d}y \qquad \forall\boldsymbol{\tau}\in H_0(\text{Div};\Omega\setminus D),\\
&\int_\Omega \nabla\cdot\boldsymbol{\sigma}v\,\mathrm{d}x = -\int_\Omega fv\,\mathrm{d}x \qquad \forall v\in L^2(\Omega\setminus D),
\end{aligned}
\tag{D.2}
$$

where $H_0(\text{Div};\Omega\setminus) = \{\boldsymbol{f}\in H(\text{Div};\Omega\setminus D) : \boldsymbol{f}(0,y)\cdot\begin{bmatrix}-1\\0\end{bmatrix} \text{ and } \boldsymbol{f}(1,y)\cdot\begin{bmatrix}1\\0\end{bmatrix}\}$. The galerkin approximations for the test and trial spaces are chosen in a structure preserving manner as discussed in section 2.4. More specifically the spline space $\mathbb{S}^{p-1,q-1}_{p-2,q-2}$ and $\mathbb{S}^{p,q-1}_{p-1,q-2}\times\mathbb{S}^{p-1,q}_{p-2,q-1}$ of order $p=q=1,2$ and 3 are used as trial

and test spaces for $L^2(\Omega \setminus D)$ and $H(\mathrm{Div}; \Omega \setminus D)$ respectively together with proper numerical integration on the trimmed elements and proper enforcement of the boundary conditions. The convergence order in the $L^2$ norm for both $u_h$ and $\boldsymbol{\sigma}_h$ is shown in Figure D.1 and in the $H(\mathrm{Div}; \Omega \setminus D)$ norm for $\boldsymbol{\sigma}_h$ in Figure D.2.

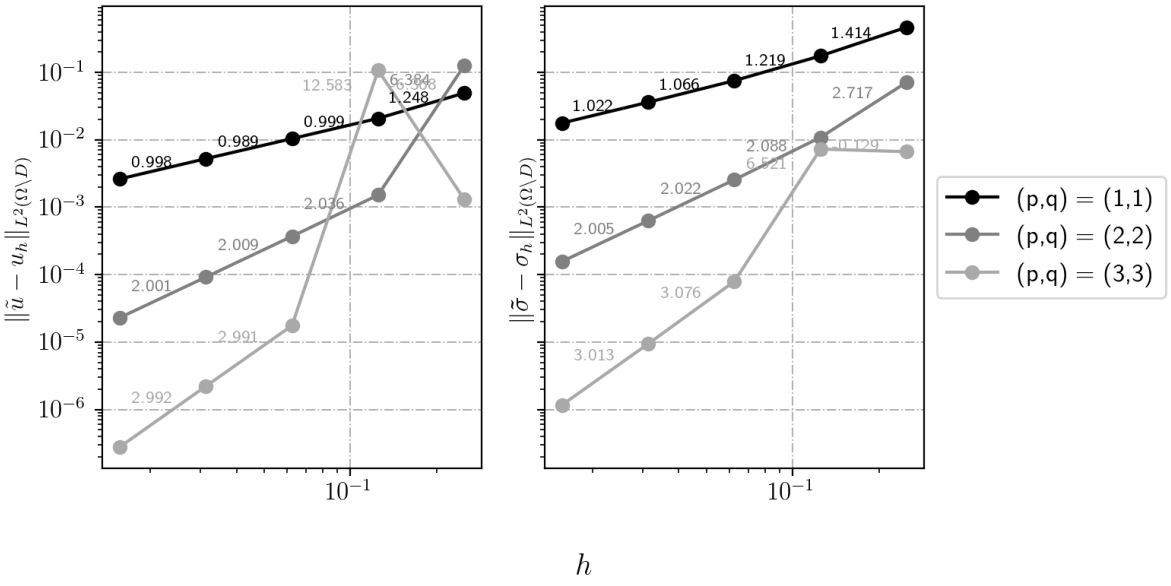



**Figure D.1:** $L^2$ norm of the error for B-spline basisfunctions of order 1,2 and 3 in both coordinate directions. Left the $L^2$ norm of the error of $u$ is shown and at the right for $\boldsymbol{\sigma}$.

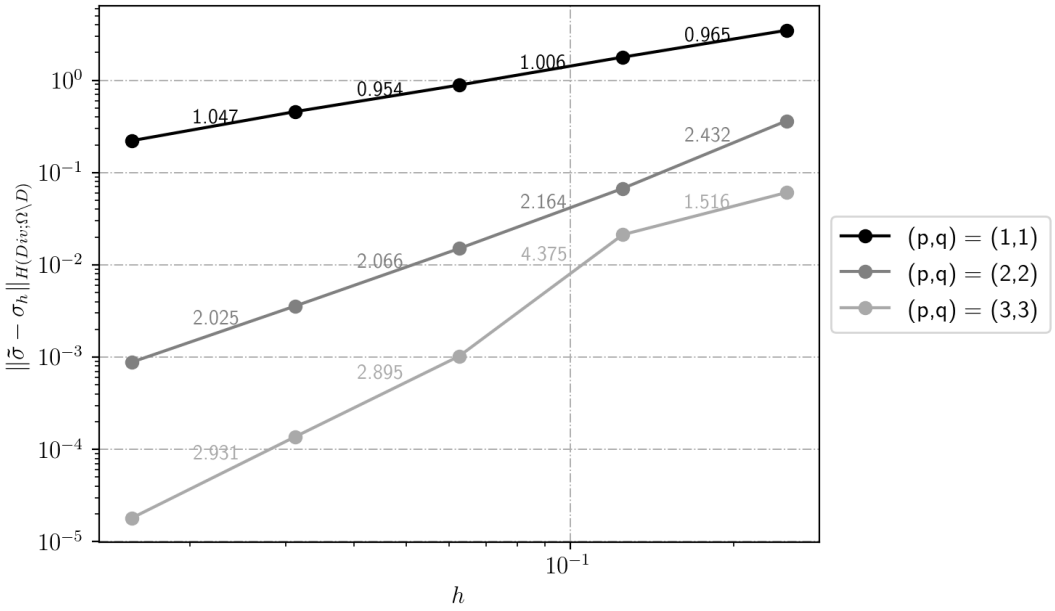



**Figure D.2:** $H(\mathrm{Div}; \Omega \setminus D)$ norm of the error of $\boldsymbol{\sigma}_h$ for B-spline basis functions of order 1, 2 and 3 in both coordinate directions.

Optimal convergence order is observed when refining the mesh parameter $h$ for both $u_h$ and $\boldsymbol{\sigma}_h$. Next we consider the condition number of the Galerkin matrix which is shown in Figure D.3. The condition numbers grow as the mesh parameter is refined and order of the basisfunction is increased. This is because of the unstable space we have chosen. More specifically there are basis functions with small support due to the trimming of the domain.

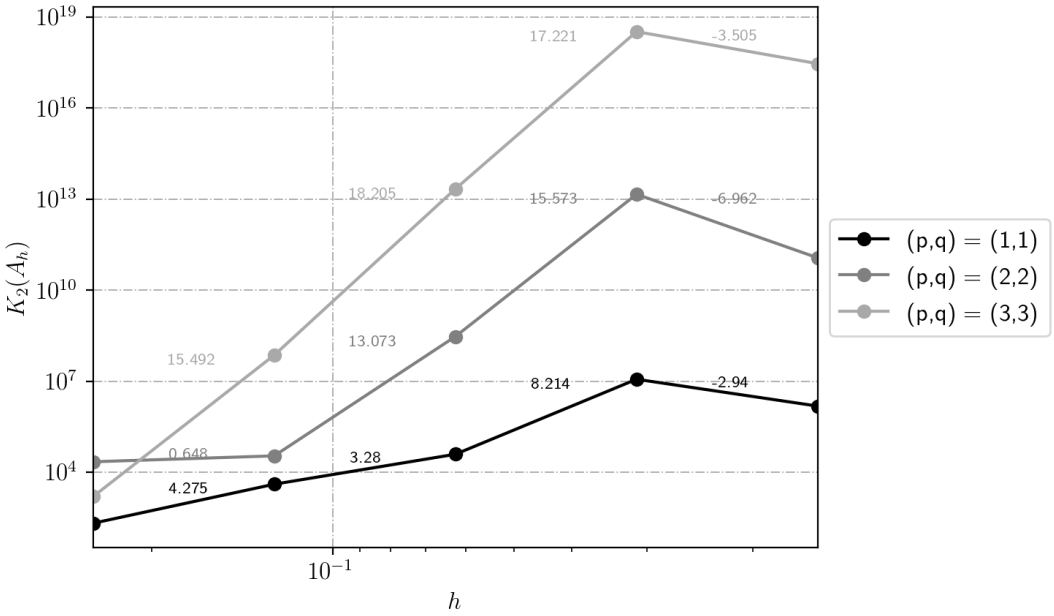



**Figure D.3:** The condition number of the Galerkin matrix with respect to the mesh parameter $h$. In addition the slopes are shown.

# E

# Source Code

The code used to obtain the results presented in this literature review was written by the author of this paper using Python. The code is available upon request. Please contact the author at **F.Sindy@outlook.com** to obtain the details regarding the code.