Isogeometric Analysis for Scalar Convection-Diffusion Equations

Diplomarbeit
Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät II
Institut für Mathematik

Submitted by Liesel Schumacher
Born December 2, 1988 in Berlin
Supervisor Professor Volker John
Second supervisor Professor Carsten Carstensen
Berlin, May 8, 2013
# Contents

1. **Introduction**  

2. **Preliminaries**  
   2.1 Notation  
   2.2 Functional Analysis  

3. **Finite Element Method - FEM**  

4. **Non-Uniform Rational B-Splines - NURBS**  
   4.1 B-Splines  
      4.1.1 B-Spline Basis Functions  
      4.1.2 B-Spline Curves  
      4.1.3 B-Spline Surfaces  
   4.2 Non-Uniform Rational B-Splines - NURBS  
      4.2.1 NURBS Curves  
      4.2.2 NURBS in Homogeneous Coordinates  
      4.2.3 NURBS Surfaces  
   4.3 Refinement  
      4.3.1 Knot Insertion  
      4.3.2 Degree Elevation  
      4.3.3 k-Refinement  

5. **Isogeometric Analysis (IGA) with NURBS**  
   5.1 Isogeometric Concept  
   5.2 A-priori Error Estimates  
      5.2.1 Definitions  
      5.2.2 Approximation Error Bounds  
      5.2.3 Inverse Inequality  

6. **Application to Convection-Diffusion Equations**  
   6.1 Convection-Diffusion Equation  
   6.2 SUPG Stabilization  
   6.3 Example I: Skewed Advection on the Unit Square  
   6.4 Example II: Sine Hill on a Circle  
   6.5 Example III: Hemker Problem  

7. **Experiments**  
   7.1 Example I: Skewed Advection on the Unit Square  
   7.2 Example II: Sine Hill on a Circle  
   7.3 Example III: Hemker Problem  

8. **Implementation**  
   8.1 General Structure of the Code  

---

**Page Numbers:**  
- Introduction: 1  
- Preliminaries: 3  
- Finite Element Method - FEM: 7  
- Non-Uniform Rational B-Splines - NURBS: 10  
- Isogeometric Analysis (IGA) with NURBS: 37  
- Application to Convection-Diffusion Equations: 61  
- Experiments: 78  
- Implementation: 93
8.2 test_of_nurbs_convection_hrefine.m ........................................ 93
8.3 test_of_nurbs_convection_krefine.m and test_of_nurbs_convection_2parts.m 97
8.4 solve_FEM_on_NURBS.m ....................................................... 99
8.5 laplace.m and gaussparameters.m ........................................... 105
8.6 load_geometry_convection.m .................................................. 105
8.7 boundary_points.m, boundfunc_conv.m, convection_field.m, h_function.m 106
8.8 nrbplotcontrol.m and nrbplotmesh.m ...................................... 109
8.9 nrbonebasisfun.m, nrbdersonebasisfun.m,
nrbsdersbasisfuns.m, nrbsurfsourcederivs.m and nrbratsurfsourcederivs.m . . 110

9 Summary and Outlook 111
1 Introduction

Convection-diffusion problems describe the transportation and diffusion of temperature or particles in a fluid. Hence they are important, for example, in the modeling of mass transportation in ground water. The physical processes are described mathematically by partial differential equations (PDEs). Another example of a convection-diffusion equation is the so-called Hemker problem see Figure 1. A hot cylinder is inserted in a cold fluid that flows with constant speed. The temperature of the cylinder is transported by the fluid.

![Figure 1: Numerical solution of the Hemker problem.](image)

In general an analytical representation of the solution to the resulting PDEs is not known. Numerical computations aim at finding approximations to the unknown exact solutions. Nowadays, finite element methods (FEM) are the mean of choice, see for example [ACF+11].

FEMs have one major disadvantage. They compute solutions on polygonal meshes and hence only on polygonal bounded domains. Thus the circular hole in the domain of the Hemker problem cannot be represented exactly and errors at the boundary occur. Moreover, each refinement step of the mesh needs to communicate with the actual design of the domain in order to provide a better approximation of the boundary.

In industrial domains, non-polygonal shapes occur frequently. Computer aided design (CAD) often uses non-uniform rational B-splines (NURBS) to describe these shapes.
Hence NURBS are an appealing choice as basis functions for a parametrization of these domains. The approach of Hughes, Bazilevs et al. is to utilize them as basis functions for the approximation of a solution to PDEs as well, see for example [HCB05]. Hughes et al. use the concept of isogeometric analysis (IGA) in which they take the same functions as basis functions for the solution space as for the parametrization.

The goal of this thesis is to understand and reproduce the work of Hughes and Bazilevs and to apply it to a more challenging test problem than studied so far. Further it adds details to the proofs of some error estimates concerning the approximation with NURBS and some a-priori estimates in [BBdVC+06]. In addition, an IGA solver for scalar convection-diffusion equations at the basis of NURBS is implemented. This realization of the concept of IGA is used to solve the Hemker problem and compare the results with [ACF+11].

The thesis starts in Section 2 with the definition of the employed notation and a summary of important results of functional analysis.

It continues in Section 3 with a short introduction into finite element methods, illustrated by the Poisson model problem.

This is followed by the introduction of NURBS in Section 4. To begin, B-Splines are defined and their basic properties are proven. Afterwards they are generalized to non-uniform rational B-splines. The section ends with the definition of three refinement procedures for NURBS. These are possibilities to enlarge the vector space spanned by the NURBS basis functions. They are necessary in order to improve the approximation of a solution for a PDE with NURBS.

In Section 5 the concept of IGA is established. The necessary definitions are made and the approximation errors with NURBS are studied. Furthermore a local and a global a-priori error estimate are derived. The section ends with an equivalent to the inverse inequalities known for standard FEMs.

Convection-diffusion problems and their numerical solutions are discussed in Section 6. To this end the streamline-upwind-Petrov-Galerkin stabilization is introduced. The section concludes with three model problems for convection-diffusion equations.

The results of the numerical studies with these programs are presented in Section 7. The three model problems for scalar convection-diffusion equalities from Section 6 are solved. The obtained results are compared with results known from [HCB05] and [ACF+11].

Section 8 deals with the implementation of the described methods. It gives an insight in the code written in the course of this thesis.

The thesis finishes with a summary of the results and some conclusions in Section 9.
2 Preliminaries

2.1 Notation

This non-exhaustive list presents abbreviations and notations used throughout this diploma thesis.

\( \mathbb{N} \) Natural numbers, \( \mathbb{N} = \{0, 1, 2, \ldots\} \)
\( \mathbb{R} \) Real numbers
\( d \) Dimension
\( \Omega \) Open subset of \( \mathbb{R}^d \)
\( \text{diam} \Omega \) Diameter of \( \Omega \),
\[
\text{diam} \Omega = \sup_{x,y \in \Omega} |x - y|
\]
\( n \) Outer normal
\( \nabla \) Gradient operator, \( \nabla : = \left( \frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n} \right) \)
\( \nabla^k \) Partial derivation operator of order \( k \),
\[
\nabla^k = \frac{\partial^k}{\partial x_1^k} + \cdots + \frac{\partial^k}{\partial x_n^k}
\]
\( C^k(\Omega) \) Space of \( k \)-times continuously differentiable functions on \( \Omega \)
\( C_0^\infty(\Omega) \) Space of test functions on \( \Omega \)
\( \int_\Omega f \) Integral mean, \( \int_\Omega f \, dx = \frac{1}{|\Omega|} \int_\Omega f \, dx \)
\( [f]_x \) Jump of the function \( f \) in the point \( x \),
\[
[f]_x := \lim_{y \to x, y > x} f(y) - \lim_{y \to x, y < x} f(y)
\]
\( \text{im} f \) Image of the function \( f \)

2.2 Functional Analysis

**Definition 2.1 \((L^p\)-Spaces):**
Let \((X, \|\|)\) be a Banach space and \((\Omega, \mathcal{F}, \mu)\) be a measure space. For \( p \in \mathbb{R}, 1 \leq p < \infty \) the \( L^p \)-norm of a measurable function \( f : \Omega \to X \) is defined by

\[
\|f\|_{L^p} = \|f\|_p := \left( \int_\Omega |f(x)|^p \, dx \right)^{1/p}.
\]

For \( p = \infty \) it is defined by

\[
\|f\|_{L^\infty} = \|f\|_\infty := \inf_{N \in \mathcal{F}} \sup_{\mu(N) = 0, t \in \Omega \setminus N} |f(t)|.
\]

These norms define the quotient spaces via

\[
L^p(\Omega; X) := \left\{ f : \Omega \to X | f \text{ is measurable and } \|f\|_p < \infty \} / \{ \|f\|_p = 0 \} \right\}.
\]
Remark 2.2: The space \( L^p (\Omega, \mathbb{R}) \) is denoted by \( L^p (\Omega) \). For \( p \in [1, \infty] \), \( L^p (\Omega, X) \) is a Banach space [Alt06]. If \((X, \langle \cdot, \cdot \rangle)\) is a Hilbert space and \( p = 2 \), \( L^2 (\Omega, X) \) is a Hilbert space as well, endowed with the scalar product
\[
\langle f, g \rangle_{L^2(\Omega,X)} := \int_{\Omega} \langle f(x), g(x) \rangle \, dx.
\]

Lemma 2.3 (Hölder Inequality): Let \( f \in L^p (\Omega) \) and \( g \in L^q (\Omega) \) it holds \( f \cdot g \in L^1 (\Omega) \) and
\[
\|f \cdot g\|_1 \leq \|f\|_p \|g\|_q.
\]

Lemma 2.4 (Young’s Inequality): For \( p, q > 1 \) with \( \frac{1}{p} + \frac{1}{q} = 1 \) and \( a, b > 0 \) it holds
\[
ab < \frac{a^p}{p} + \frac{b^q}{q}.
\]

Definition 2.5 (Sobolev Spaces): Let \( f : \Omega \to \mathbb{R}^d \) be a \( k \)-times weakly differentiable function with weak derivative \( D^\alpha f \) for any multi-index \( \alpha \) with \( |\alpha| \leq k \). For \( p \in [1, \infty] \) the Sobolev norm \( \|f\|_{W^{k,p}(\Omega, \mathbb{R}^d)} \) and the seminorm \( |f|_{W^{k,p}(\Omega, \mathbb{R}^d)} \) are defined by
\[
\|f\|_{W^{k,p}(\Omega, \mathbb{R}^d)} = \sum_{|\alpha| \leq k} \|D^\alpha f\|_{L^p(\Omega, \mathbb{R}^d)} \quad \text{and} \quad |f|_{W^{k,p}(\Omega, \mathbb{R}^d)} = \sum_{|\alpha| = k} \|D^\alpha f\|_{L^p(\Omega, \mathbb{R}^d)}.
\]

The Sobolev norm defines the Sobolev space as follows
\[
W^{k,p} (\Omega; \mathbb{R}^d) := \left\{ f \in L^p (\Omega; \mathbb{R}^d) \mid f \text{ is } k \text{-times weakly differentiable and } \|f\|_{W^{k,p}(\Omega, \mathbb{R}^d)} < \infty \right\}.
\]

Remark 2.6 ([Ada09]): The spaces \( W^{k,p} (\Omega; \mathbb{R}^d), \|\cdot\|_{W^{k,p}(\Omega, \mathbb{R}^d)} \) are Banach spaces. The space \( H^k (\Omega; \mathbb{R}^d) := W^{k,2} (\Omega; \mathbb{R}^d) \) is a Hilbert space endowed with the scalar product
\[
\langle f, g \rangle_{H^k(\Omega, \mathbb{R}^d)} := \sum_{|\alpha| \leq k} \langle D^\alpha f, D^\alpha g \rangle_{L^2(\Omega, \mathbb{R}^d)} \, dx.
\]

If \( k = 0 \) the Sobolev space becomes \( W^{0,p} (\Omega; \mathbb{R}^d) \) is a Banach space. The Sobolev spaces can be defined as the closure of \( C^\infty (\Omega, \mathbb{R}^d) \) with respect to \( \|\cdot\|_{W^{k,p}(\Omega, \mathbb{R}^d)} \). The space \( H^k_0 (\Omega; \mathbb{R}^d) \) denotes the closure of \( C^\infty_0 (\Omega, \mathbb{R}^d) \) with respect to \( \|\cdot\|_{W^{k,p}(\Omega, \mathbb{R}^d)} \). For sake of simplicity throughout this thesis we will denote \( W^{k,p} (\Omega) = W^{k,p} (\Omega; \mathbb{R}^d) \) and \( H^k (\Omega) = H^k (\Omega; \mathbb{R}^d) \).
Theorem 2.7 (Rellich-Kondrachov Theorem [Ada09]):
Let $0 \leq k_2 < k_1$ and $1 \leq p_2 < p_1 < \infty$. Further let $\Omega \subset \mathbb{R}^d$ be bounded with Lipschitz boundary $\partial \Omega$. Then the embedding

$$W^{k_1,p_1}(\Omega;\mathbb{R}^d) \hookrightarrow W^{k_2,p_2}(\Omega;\mathbb{R}^d)$$

is continuous and compact.

Theorem 2.8 (Trace Theorem [Ada09]):
Let $1 < p < \infty$ and $\Omega \subset \mathbb{R}^d$ be a bounded domain with Lipschitz boundary $\partial \Omega$. Then for every $u \in W^{1,p}(\Omega;\mathbb{R}^d)$ there exists $Tu \in L^p(\partial \Omega;\mathbb{R}^d)$ such that

$$Tu = u|_{\partial \Omega},$$

for all $u \in W^{1,p}(\Omega;\mathbb{R}^d) \cap C(\overline{\Omega})$.

The constant $C$ does not depend on $u$.

Remark 2.9: $Tu$ is called the trace of $u$ and is interpreted as the restriction of $u$ onto the boundary $\partial \Omega$.

Lemma 2.10 (Faà di Bruno’s Formula [FdB55]):
Let $f$ and $g$ be sufficiently smooth and $y = g(x)$. Then it holds for the derivative of $h = f \circ g$

$$D^n h(x) = \sum \frac{n!}{k_1! \cdots k_n!} D^k f(y) \prod_{i=1}^n \left( \frac{D^{|i_1|} g(x)}{i_1!} \cdots \frac{D^{|i_n|} g(x)}{i_n!} \right)^{k_i},$$

with $k = k_1 + \cdots + k_n$, and the summation is over all $k_1, \ldots, k_n$ with $k_1 + 2k_2 + \cdots + nk_n = n$.

For a proof see [Rom80].

Remark 2.11: If $g$ is an affine function, then (2.1) simplifies to $D^n h(x) = D^n f(y) (Dg(x))^n$.

With Lemma 2.10 the norm of the derivatives of concatenated functions can be estimated.

Lemma 2.12:
The functions $f$, $g$, $h$ from Lemma 2.10 satisfy

$$\|\nabla^n h(x)\| \leq C \sum_{k=1}^n \|D^k f(y)\| \sum_{i \in I(k,n)} \|\nabla g(\alpha)\|^{i_1} \|\nabla^2 g(\alpha)\|^{i_2} \cdots \|\nabla^n g(\alpha)\|^{i_n},$$

where the constant $C$ depends only on $n$, $y = g(x)$, and

$$I(k,n) := \{i = (i_1, i_2, \ldots, i_n) \in \mathbb{N} | i_1 + i_2 + \cdots + i_n = k, i_1 + 2i_2 + \cdots + ni_n = n \}.$$
**Theorem 2.13** (Riesz Representation Theorem [Alt06]):

Let $(V, \langle \cdot, \cdot \rangle_V)$ be a Hilbert space. Then for every continuous, linear functional $L$ on $V$ there exists a unique $u \in V$ such that

$$L(v) = \langle u, v \rangle_V,$$

for all $v \in V$.

In addition it holds

$$\|L\|_{V'} = \|u\|_V.$$

**Lemma 2.14** (Poincaré Inequality [Eva10]):

Let $\Omega \subset \mathbb{R}^d$ be bounded and connected with $C^1$ boundary $\partial \Omega$ and let $1 \leq p \leq \infty$. Then

$$\left\| u - \frac{1}{|\Omega|} \int_{\Omega} u \right\|_{L^p(\Omega \times \mathbb{R}^d)} \leq C \|Du\|_{L^p(\Omega \times \mathbb{R}^d)},$$

where $C$ only depends on $d$, $p$, and $\Omega$.

**Lemma 2.15** (Bramble-Hilbert Lemma [BS08]):

Let $B$ be a ball in $\Omega$ such that $\Omega$ is star-shaped with respect to $B$ and such that its radius $\rho > (1/2) \rho_{\text{max}}$. Then there exists a polynomial $p$ of order less or equal $(m - 1)$ such that for $k = 0, 1, \ldots, m$

$$\|u - p\|_{W^k,p(\Omega)} \leq C_{m,h,d} d^{m-k} \|u\|_{W^m,p(\Omega)},$$

where $d = \text{diam} (\Omega)$.

**Lemma 2.16** (Inverse Inequality for Polynomials [BS08]):

Let $\rho h \leq \text{diam} \Omega \leq h$, where $0 < h \leq 1$, and $\mathcal{P}$ be a finite dimensional subspace of $W^{k,p}(\Omega) \cup W^{l,q}(\Omega)$, where $1 \leq p, q \leq \infty$ and $0 \leq l \leq k$. Then there exists a constant $C$, independent of $h$, such that for all $v \in \mathcal{P}$

$$\|v\|_{W^{k,p}(\Omega)} \leq Ch^{l-k+n/p-n/q} \|v\|_{W^{l,q}(\Omega)},$$

**Remark 2.17**: The proof in [BS08] shows that

$$\|v\|_{W^{j,p}(\Omega)} \leq Ch^{-j+n/p-n/q} \|v\|_{L^p(\Omega)}$$

holds for $0 \leq j \leq k$ even if $h > 1$. 
3 Finite Element Method - FEM

Finite element methods are well-known methods for computing numerical solutions for partial differential equations. In this chapter the main concept of FEM is explained and illustrated by the Poisson model problem with homogeneous Dirichlet boundary condition. For a more detailed survey see Braess [Bra07] or Brenner, Scott [BS08].

Definition 3.1 (Poisson Problem):
Let $\Omega \subset \mathbb{R}^d$ be a domain with Lipschitz boundary $\partial \Omega$ and $f \in C(\Omega)$. Then the Poisson problem with homogeneous Dirichlet boundary condition seeks $u \in C^2(\Omega)$ such that $u$ satisfies

\begin{align*}
-\Delta u &= f \text{ in } \Omega, \\
u &= 0 \text{ on } \partial \Omega.
\end{align*}

(3.1) (3.2)

This is the strong formulation of the problem. To solve it numerically with FEM, the weak formulation of the problem is defined. To this end multiply the equation by some test function and integrate. A suitable Hilbert space $V$ on $\Omega$ is introduced. $V$ should be chosen such that the integrals are well-defined. If $u$ satisfies equation (3.1) it also fulfills

$$-\int_{\Omega} \Delta u \cdot v \, dx = \int_{\Omega} f \cdot v \, dx \text{ for all } v \in V.$$  

Integration by parts leads to

$$\int_{\Omega} \nabla u \cdot \nabla v \, dx = \int_{\Omega} f \cdot v \, dx + \int_{\partial \Omega} v \frac{\partial u}{\partial n} \, ds \text{ for all } v \in V,$$

where $n$ denotes the outer normal unit vector. The test function space $V$ has to satisfy the boundary condition and hence the space $V = H^1_0(\Omega)$ is chosen. The boundary integral then vanishes. Additionally this space already contains the boundary condition for $u$ by definition. Thus we finally obtain:

Definition 3.2 (Weak Formulation of the Poisson Problem):
Let $\Omega$ be as in Definition 3.1, $f \in H^{-1}(\Omega)$ and let $V = H^1_0(\Omega)$. Then $u \in V$ is called weak solution to the Poisson problem with homogeneous Dirichlet boundary condition if and only if it satisfies

$$\int_{\Omega} \nabla u \cdot \nabla v \, dx = \int_{\Omega} f \cdot v \, dx \text{ for all } v \in V.$$  

(3.3)

More generally, a weak or variational formulation of a PDE makes use of a bilinear form $a : V \times V \to \mathbb{R}$ and a linear functional $l : V \to \mathbb{R}$. The resulting weak problem then seeks $u \in V$ such that

$$a(u, v) = l(v) \text{ for all } v \in V.$$  

(3.4)
The energy norm is given by
\[ \|v\| := \sqrt{a(v, v)} \text{ for all } v \in V. \]

For the Poisson problem, the bilinear form is defined by
\[ a(u, v) := \int_{\Omega} \nabla u \cdot \nabla v \, dx \]
and the right-hand side (RHS) by
\[ l(v) = \int_{\Omega} f \cdot v \, dx. \]

The existence of a weak solution for the general problem (3.4) is guaranteed by the Riesz Representation Theorem 2.13.

To solve problem (3.4) numerically, we use the Galerkin method. The Hilbert space \( V \) is approximated by closed, finite-dimensional, linear subspaces \( V_h \subset V \). The approximation leads to the finite element problem.

**Definition 3.3 (Finite Element Problem):**
Let \( \Omega \subset \mathbb{R}^d \) be a domain with polyhedral Lipschitz boundary \( \partial \Omega \). Then the discrete finite element problem seeks \( u_h \in V_h \) such that
\[ a(u_h, v_h) = l(v_h) \text{ for all } v_h \in V_h. \] (3.5)

In the subspaces \( V_h \) a finite-dimensional basis \( \{\Psi_i\}_{i=0}^n \) is chosen. In terms of these basis functions \( u_h \) is given by
\[ u_h = \sum_{i=0}^n c_i \Psi_i, \]
for some coefficients \( c_i \in \mathbb{R}, i = 0, \ldots, n \).

It is sufficient to test equation (3.5) only with the basis functions.

**Definition 3.4 (Discrete Problem):**
Find coefficients \( \{c_i\}_{i=0}^n \subset \mathbb{R}^{n+1} \) such that
\[ \sum_{i=0}^n c_i \cdot a(\Psi_i, \Psi_j) = l(\Psi_j) \text{ for all } j = 0, \ldots, n. \]

It only remains to solve a system of linear equations \( A \cdot x = b \) with
\[ A = (a_{ij})_{i,j=0}^n = a(\Psi_j, \Psi_i) \]
and \( b = (b_i)_{i=0}^n = l(\Psi_i). \)
The matrix $A$ is called the stiffness matrix and $b$ stands for the right hand side of the problem. In the special case of the Poisson problem, $A$ and $b$ are given by

$$
A = (a_{ij})_{i,j=0}^{n,n} = \int_{\Omega} \nabla \Psi_i \cdot \nabla \Psi_j \, dx \quad \text{and} \quad b = (b_j)_{j=0}^{n} = \int_{\Omega} f \cdot \Psi_j \, dx.
$$

In order to improve the approximation $u_h$ of $u$, the number of basis functions of $V_h$ is increased. There are two common methods for this purpose.

The first method is called h-refinement. For the definition of the $\Psi_i$, the domain $\Omega$ is divided into so called elements. The $\Psi_i$ are then usually nonzero on an element and its neighboring elements, and zero otherwise. The h-refinement method divides the elements into a finite number of smaller elements. Thus the new basis functions have smaller support and the amount of basis functions increases with the number of elements.

The second method is called p-refinement. It increases the degree $p$ of the basis functions $\Psi_i$. For more details concerning refinement procedures see [BG92]. The treatment of other boundary conditions is explained, e.g., in [Bra07].
4 Non-Uniform Rational B-Splines - NURBS

This chapter gives an introduction to non-uniform rational B-splines (NURBS). It mainly follows Piegl and Tiller [PT97]. For a more descriptive treatment see Rogers [Rog01]. The first section deals with B-splines followed by a section about their generalization to NURBS. The chapter is concluded with a description of refinement procedures for splines.

4.1 B-Splines

4.1.1 B-Spline Basis Functions

Definition 4.1 (Knot Vector):
Let $\xi_i \in \mathbb{R}$ for $i = 0, \ldots, m$ with $\xi_i \leq \xi_{i+1}$ for $0 \leq i \leq m - 1$. The set $\Xi = \{\xi_0, \ldots, \xi_m\}$ is called a knot vector.

Remark 4.2: Throughout this thesis the division $0/0$ is defined to be $0$.

Definition 4.3 (B-Spline Basis Function):
The $i$-th B-spline basis function $N_{i,0}$ of degree $0$ on the knot vector $\Xi$, is defined by

$$N_{i,0}(\xi) := \begin{cases} 1 & \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise}, \end{cases} \quad 0 \leq i \leq m - 1.$$

For $p > 0$ and $0 \leq i \leq n := m - (p + 1)$ the $i$-th basis function of degree $p$, $N_{i,p}$, is defined recursively by

$$N_{i,p}(\xi) := \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi).$$

Remark 4.4: At a multiple knot $\xi_i = \xi_{i+1}$ there exists no $\xi$ with $\xi_i \leq \xi < \xi_{i+1}$ and hence

$$N_{i,0}(\xi) \equiv 0.$$

With the simple affine transformation $\xi \rightarrow \frac{\xi - \xi_0}{\xi_m - \xi_0}$ the interval $[\xi_0, \xi_m]$ can be mapped onto $[0, 1]$. Therefore we restrict ourselves to $\xi_0 := 0$ and $\xi_m := 1$ throughout this thesis.

Proposition 4.5 (Linear Independence of the $N_{i,p}$ [PT97]):
Let the knot sequence $0 = \xi_0, \ldots, \xi_m = 1$ be strictly increasing, i.e. $\xi_i < \xi_{i+1}$ for $0 \leq i \leq m - 1$. Then for every degree $p$ the basis functions $N_{i,p}$ are linearly independent.
Proof.
In order to prove linear independence assume
\[ 0 \equiv \sum_{i=0}^{n} c_i N_{i,p}(\xi) \quad (4.1) \]
for some coefficients \( c_i \in \mathbb{R} \). Hence it needs to be shown that \( c_i = 0 \) for all \( 0 \leq i \leq n \). The assertion is proven by mathematical induction.

\( p = 0 \) Since \( N_{i,p}(\xi_j) = 0 \) for \( i \neq j \) and the interval \([\xi_i, \xi_{i+1}]\) is nonempty by assumption, the insertion of the value \( \xi = \xi_j \) into the sum reduces it to
\[ 0 \equiv \sum_{i=0}^{n} c_i N_{i,p}(\xi_j) = c_j \cdot N_{i,p}(\xi_j) = c_j \cdot 1 = c_j. \]
The iterative insertion of all the knots into (4.1) yields \( c_i = 0 \) for \( i = 0 \ldots n \).

\( p \rightarrow p - 1 \) The recursive definition of the basis functions leads to
\[ 0 \equiv \sum_{i=0}^{n} c_i \cdot N_{i,p}(\xi) \]
\[ = \sum_{i=0}^{n} c_i \left( \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \cdot N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_i} \cdot N_{i+1,p-1}(\xi) \right), \]
\[ = \sum_{i=0}^{n} \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \cdot c_i \cdot N_{i,p-1}(\xi) + \sum_{i=1}^{n+1} \frac{\xi_{i+p} - \xi}{\xi_{i+p} - \xi_i} \cdot c_{i-1} \cdot N_{i,p-1}(\xi), \]
\[ = \sum_{i=0}^{n^*} c_i^* N_{i,p-1}(\xi). \]

Here \( n^* = m - (p - 1) - 1 = m - p = n + 1 \) denotes the number of basis functions of degree \( p - 1 \) and the coefficients are
\[ c_0^* = \frac{\xi - \xi_0}{\xi_p - \xi_0} \cdot c_0, \]
\[ c_n^* = \frac{\xi_{n+p} - \xi}{\xi_{n+p} - \xi_n} \cdot c_n^*, \]
\[ c_i^* = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \cdot c_i + \frac{\xi_{i+p} - \xi}{\xi_{i+p} - \xi_i} \cdot c_{i-1} \text{ for all } 0 < i < n^*. \]

Since the basis functions of degree \( p - 1 \) are linearly independent, the induction hypothesis yields that the coefficients \( c_i^* \) must be zero. The fractions defining these new coefficients depend on \( \xi \) and are not constant.
Hence iteratively it follows that

\[ 0 = c_0^* \implies c_0 = 0, \]
\[ 0 = c_0 \text{ and } 0 = c_1^* \implies c_1 = 0, \]
\[ \vdots \]
\[ 0 = c_n^* - 2 \text{ and } 0 = c_n^* - 1 \implies c_n^* - 1 = c_n = 0. \]

Hence all the coefficients are zero and therefore the basis functions are linearly independent. □

Proposition 4.5 assures that it is legitimate to refer to \( \{N_{i,p}\}_{i=0}^n \) as basis functions. The next theorem characterizes the space that is spanned by the B-spline basis functions.

**Definition 4.6 (Spline Space):**

Let \( S = S(\Xi, p) \) be the space of all functions \( f \) such that \( f|_{\xi_i, \xi_{i+1}} \) is polynomial of degree less then or equal to \( p \) for each knot span \((\xi_i, \xi_{i+1})\) and at a knot of multiplicity \( k \) the function \( f \) is \((p-k)\) times continuously differentiable. This space is called the spline space on \( \Xi \) of order \( p \).

**Theorem 4.7** (Curry and Schoenberg [CS66]):

The B-spline basis functions \( N_{i,p} \) corresponding to the knot vector \( \Xi \) form a basis of \( S(\Xi, p) \).

For a proof of Theorem 4.7 see [CS66].

**Proposition 4.8** (Properties of B-Spline Basis Functions [PT97]):

Frequently used properties of the B-spline basis functions are

- **P1** \( \text{supp}(N_{i,p}) \subseteq [\xi_i, \xi_{i+p+1}] \),
- **P2** \( N_{i,p}(\xi) = 0 \) for \( \xi \in [\xi_i, \xi_{i+1}) \) and \( j < i - p \) or \( j > i \),
- **P3** \( N_{i,p}(\xi) \geq 0 \) for \( i = 0, \ldots, n, \xi \in [0,1], \) and \( p \geq 0 \),
- **P4** \( \sum_{j=0}^{n} N_{j,p}(\xi) = \sum_{j=i-p}^{i} N_{j,p}(\xi) = 1 \), for \( \xi \in [\xi_i, \xi_{i+1}] \),
- **P5** \( N_{i,p}(\xi) \in C^\infty \) for \( \xi \in (\xi_j, \xi_{j+1}) \) and \( j = 0, \ldots, m, N_{i,p}(\xi_{j0}) \in C^{p-k} \) at a knot \( \xi_{j0} \) with multiplicity \( k \leq p \),
- **P6** For \( p > 0 \) \( N_{i,p} \) has exactly one maximum if the multiplicity \( k \) of \( \xi_i \) satisfies \( k \leq p \).

**Remark 4.9:** A set of functions satisfying **P3** and **P4** is called a partition of unity.

**Proof of Proposition 4.8.**

**Proof of P1** \( N_{i,0}(\xi) = 0 \) for \( \xi \notin [\xi_i, \xi_{i+1}) \) and \( N_{i,p}(\xi) \) only depends on \( N_{i,p-1}(\xi) \) and \( N_{i+1,p-1}(\xi) \). Hence the following triangle of functions illustrates the basis functions on which \( N_{i,p}(\xi) \) depends.
Therefore \( N_{i,p}(\xi) \neq 0 \) if and only if one of the \( N_{j,0}(\xi), \ j = i, \ldots, i+p \) occurring in the first column of the triangle is non-zero. The application of the definition of \( N_{j,0}(\xi) \) proves the statement.

**Proof of P2** Similar to the triangle in the proof of P1 there is a triangle of basis functions that depend on \( N_{i,0}(\xi) \):

\[
N_{i,0} \rightarrow N_{i,1} \rightarrow \cdots \rightarrow N_{i,p}
\]

\[
N_{i+1,0} \rightarrow N_{i+1,1} \rightarrow \cdots
\]

\[
\vdots \quad \vdots \quad \vdots \quad \vdots
\]

\[
N_{i+p,0} \rightarrow N_{i+p,1}
\]

For \( \xi \in [\xi_i, \xi_{i+1}] \) all basis functions \( N_{j,0}(\xi) \) for \( j \neq i \) are zero. Hence the functions \( N_{j,0}(\xi) \) for \( i-p \leq j \leq i \) are the only basis functions of degree \( p \) that can be non-zero in \( [\xi_i, \xi_{i+1}] \).

**Proof of P3** Proof by mathematical induction

\( p = 0 \) Since \( N_{i,0}(\xi) = 0 \) or \( N_{i,0}(\xi) = 1 \) for all \( \xi \in [0,1] \) it holds

\( N_{i,0}(\xi) \geq 0 \) for all \( \xi \in [0,1] \).

\( p-1 \rightarrow p \) Assume that \( N_{i,p-1}(\xi) \geq 0 \) holds for all \( i = 0, \ldots, n \) and \( \xi \in [0,1] \). The recursive definition yields

\[
N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi).
\]

By the induction hypothesis it holds \( N_{i,p-1}(\xi) \geq 0 \). Since \( \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \geq 0 \) for \( \xi \geq \xi_i \) and \( N_{i,p-1}(\xi) = 0 \) for \( \xi < \xi_i \) property P1 implies that the first summand is nonnegative. Similarly it holds \( N_{i+1,p-1}(\xi) \geq 0 \) and \( N_{i+1,p-1}(\xi) = 0 \) for \( \xi \geq \xi_{i+p+1} \), and \( \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} \geq 0 \) for \( \xi \leq \xi_{i+p+1} \). Hence the second summand is
nonnegative as well. The combination of these observations concludes the proof.

**Proof of P4** The equality of the sums follows directly from P2, \( \sum_{j=i-p}^{i} N_{jp}(\xi) = 1 \) is proven by mathematical induction:

- **p = 0** Let \( \xi \in [\xi_i, \xi_{i+1}) \). Then the definition of basis functions of degree \( p = 0 \) implies
  \[
  \sum_{j=0}^{n} N_{j0}(\xi) = N_{i0}(\xi) = 1.
  \]

- **p − 1 → p** The application of the induction hypothesis, \( \sum_{j=0}^{n} N_{jp-1}(\xi) = 1 \), to the recursive formula leads to
  \[
  \sum_{j=0}^{n} N_{jp}(\xi) = \sum_{j=i-p}^{i} N_{jp}(\xi)
  \]
  \[
  = \sum_{j=i-p}^{i} \left( \frac{\xi - \xi_j}{\xi_{j+p} - \xi_j} N_{jp-1}(\xi) + \frac{\xi_{j+p-1} - \xi}{\xi_{j+p} - \xi_j} N_{j+1,p-1}(\xi) \right)
  \]
  \[
  = \sum_{j=i-p}^{i} \left( \frac{\xi - \xi_j}{\xi_{j+p} - \xi_j} N_{jp-1}(\xi) \right) + \sum_{j=i-p+1}^{i+1} \left( \frac{\xi_{j+p} - \xi}{\xi_{j+p} - \xi_j} N_{jp-1}(\xi) \right)
  \]
  \[
  = \frac{\xi - \xi_{i-p}}{\xi_{i} - \xi_{i-p}} N_{i-p,p-1}(\xi)
  \]
  \[
  + \sum_{j=i-p+1}^{i+1} \left( \frac{\xi - \xi_j}{\xi_{j+p} - \xi_j} N_{jp-1}(\xi) + \frac{\xi_{j+p} - \xi}{\xi_{j+p} - \xi_j} N_{jp-1}(\xi) \right)
  \]
  \[
  + \frac{\xi_{i+1+p} - \xi}{\xi_{i+1+p} - \xi_{i+1}} N_{i+1,p-1}(\xi).
  \]

P2 implies \( N_{i-p,p-1}(\xi) = 0 \) and \( N_{i+1,p-1}(\xi) = 0 \). Hence

\[
\sum_{j=0}^{n} N_{jp}(\xi) = \sum_{j=i-p+1}^{i} N_{jp-1}(\xi) = 1.
\]

**Proof of P5** [Flo07] In the interior \( (\xi_j, \xi_{j+1}) \) of a knot span, the B-spline basis functions \( N_{ij} \) are polynomials of degree \( p \) and therefore smooth. Hence it only remains to show that the left and right derivative of \( N_{ij}^{(l)} \) at a knot \( \xi_j \) of multiplicity \( k \) are equal for \( 0 \leq l \leq p - k \).
The first step is to show that for a knot $\xi_j$ of multiplicity $k \leq p$ the B-spline basis functions are continuous at $\xi_j$. Therefore let

$$\left[ N_{i,p} \right]_{x} := \lim_{y \to x} N_{i,p}(y) - \lim_{y \to x} N_{i,p}(y)$$

denote the jump of the basis function $N_{i,p}$ at a point $x$. Thus it suffices to show for all $i$ and $p$ that $\left[ N_{i,p} \right]_{\xi_j} = 0$ for all knots $\xi_j$ with multiplicity $k \leq p$. This is done by mathematical induction over $p$:

- **p = 1** In this case the condition $k \leq p$ allows only single knots. Hence it holds

$$N_{i,p}(\xi) = \begin{cases} \frac{\xi - \xi_i}{\xi_{i+1} - \xi_i}, & 0 < \xi < \xi_{i+1}, \\ \frac{\xi_{i+1} - \xi_i}{\xi_{i+2} - \xi_i}, & \xi_{i+1} \leq \xi < \xi_{i+2}, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore the jump of $N_{i,p}$ at a knot $\xi_j$ satisfies

$$\left[ N_{i,p} \right]_{\xi_j} = \begin{cases} \frac{\xi_j - \xi_i}{\xi_{i+1} - \xi_i} - \frac{\xi_{i+1} - \xi_i}{\xi_{i+2} - \xi_i}, & j = i, \\ \frac{\xi_{i+1} - \xi_i}{\xi_{i+2} - \xi_i}, & j = i + 1, \\ 0 - \frac{\xi_{i+2} - \xi_{i+1}}{\xi_{i+3} - \xi_{i+1}}, & j = i + 2, \\ 0, & \text{otherwise} \end{cases} = \begin{cases} \frac{0}{\xi_{i+1} - \xi_i} - \frac{1}{\xi_{i+2} - \xi_i}, & j = i, \\ \frac{1}{\xi_{i+2} - \xi_i} - \frac{0}{\xi_{i+3} - \xi_{i+1}}, & j = i + 1, \\ 0, & j = i + 2, \\ 0, & \text{otherwise} \end{cases} = 0.$$ 

- **p → p + 1** Since $(\xi - \xi_i) / (\xi_{i+p} - \xi_i)$ and $(\xi_{i+p+1} - \xi_i) / (\xi_{i+p+1} - \xi_{i+1})$ are continuous, the recursive definition of the basis functions implies

$$\left[ N_{i,p} \right]_{\xi_j} = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \left[ N_{i,p-1} \right]_{\xi_j} + \frac{\xi_{i+p+1} - \xi_i}{\xi_{i+p+1} - \xi_{i+1}} \left[ N_{i+1,p-1} \right]_{\xi_j}. \tag{4.2}$$

In the trivial case where all knots have a multiplicity $k \leq p - 1$ the induction hypothesis can be applied to $\left[ N_{i,p-1} \right]_{\xi_j}$ and $\left[ N_{i+1,p-1} \right]_{\xi_j}$ and yields $\left[ N_{i,p} \right]_{\xi_j} = 0$ for all knots $\xi_j$.

Assume now, that the knot $\xi_j$ has multiplicity $k = p$. In this case the induction hypothesis is not satisfied and an analysis considering different cases is necessary.
case 1: $\xi_j < \xi_i$ or $\xi_j > \xi_{i+p+1}$
Since $\text{supp}(N_{i,p}) \subseteq [\xi_i, \xi_{i+p+1}]$ it holds $[N_{i,p}]_{\xi_j} = 0 - 0 = 0$.

case 2: $\xi_j = \xi_i$
Under this assumption it holds $(\xi_j - \xi_i)/(\xi_{i+p} - \xi_i) = 0$. In addition $\text{supp}(N_{i+1,p-1}) \subseteq [\xi_{i+1}, \xi_{i+p+1}]$ yields $[N_{i+1,p-1}]_{\xi_j} = 0$. Hence (4.2) leads to $[N_{i,p}]_{\xi_j} = 0$.

case 3: $\xi_i < \xi_j = \xi_{i+1} = \ldots = \xi_{i+p} < \xi_{i+p+1}$
In this case the basis functions satisfy $N_{i+1,0} = N_{i+2,0} = \ldots = N_{i+p-1,0}$.
Hence the recursive definition of $N_{i,p-1}$ yields

$$N_{i,p-1}(\xi_j) = \prod_{k=1}^{p-1} \frac{\xi_j - \xi_i}{\xi_{i+k} - \xi_i} \cdot N_{i,0}(\xi_j)$$
$$= \prod_{k=1}^{p-1} \frac{\xi_{i+k} - \xi_j}{\xi_{i+k} - \xi_i} \cdot N_{i,0}(\xi_{i+k}) = N_{i,0}(\xi_{i+k}) .$$

Analogously it holds

$$N_{i+1,p-1}(\xi_j) = \prod_{k=1}^{p-1} \frac{\xi_{i+p+1} - \xi_j}{\xi_{i+p+1} - \xi_{i+1+k}} \cdot N_{i+p,0}(\xi_j)$$
$$= \prod_{k=1}^{p-1} \frac{\xi_{i+p+1} - \xi_{i+1+k}}{\xi_{i+p+1} - \xi_{i+1+k}} \cdot N_{i+p,0}(\xi_{i+p}) = N_{i+p,0}(\xi_{i+p}) .$$

Obviously the jumps of the basis functions with degree $p = 0$ satisfy $[N_{i,0}]_{\xi=1} = -1$ and $[N_{i+p,0}]_{\xi=1} = 1$. Finally inserting these relations into (4.2) yields

$$[N_{i,p}]_{\xi_j} = \frac{\xi_j - \xi_i}{\xi_{i+p} - \xi_i} \cdot [N_{i,p-1}]_{\xi_j} + \frac{\xi_{i+p+1} - \xi_j}{\xi_{i+p+1} - \xi_{i+1}} \cdot [N_{i+1,p-1}]_{\xi_j}$$
$$= \frac{\xi_{i+p} - \xi_i}{\xi_{i+p} - \xi_i} \cdot [N_{i,0}]_{\xi=1} + \frac{\xi_{i+p+1} - \xi_{i+1}}{\xi_{i+p+1} - \xi_{i+1}} \cdot [N_{i+p,0}]_{\xi=1}$$
$$= 1 \cdot (-1) + 1 \cdot 1 = 0 .$$

case 4: $\xi_j = \xi_{i+p+1}$
Similar to case 2 $(\xi_{i+p+1} - \xi_j)/(\xi_{i+p+1} - \xi_{i+1}) = 0$ and $[N_{i,p-1}]_{\xi_j} = 0$ together with (4.2) imply $[N_{i,p}]_{\xi_j} = 0$.

Hence for $k \leq p$ all basis functions are continuous at $\xi_j$.  

16
As it is already proven that \( N_{i,p} \) is \( l \leq p - k \) times continuously differentiable in the interior of a knot span, the recursive formula for the derivatives (4.4), which is proven below in Lemma 4.10, can be applied for \( \xi \in (\xi_j, \xi_{j+1}) \). Thus considering the jump of the \( l \)-th derivative at \( \xi_j \) and observing that \( l - 1 \leq p - k \iff l \leq (p+1) - k \) leads to a proof of the continuity of \( N_{i,p}^{(l)} \) that is similar to the proof for \( l = 0 \).

**Proof of P6** A proof of this statement can be found in [CS66, Theorem 1] for single knots. For multiple knots see [CS66, remark before Section 4]. □

**Lemma 4.10** (Derivatives of B-Spline Basis Functions [PT97]):

*If the first derivative of the \( i \)-th basis function exists, it is given by*

\[
N'_{i,p} (\xi) = \frac{p}{\xi_{i+p} - \xi_i} N_{i,p-1} (\xi) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1} (\xi) .
\]  

(4.3)

**Remark 4.11:** A generalization to the higher derivatives for \( k \leq p \), if existent, is given by

\[
N_{i,p}^{(k)} = p \left( \frac{N_{i,p-1}^{(k-1)} (\xi) - N_{i+1,p-1}^{(k-1)} (\xi)}{\xi_{i+p} - \xi_i} - \frac{N_{i+1,p}^{(k-1)} (\xi)}{\xi_{i+p+1} - \xi_{i+1}} \right)
\]

\[
= \frac{p!}{(p - k)!} \sum_{j=0}^{k} a_{k,j} N_{i+j,p-k} (\xi) ,
\]

with

\[
a_{0,0} = 1,
\]

\[
a_{k,0} = \frac{a_{k-1,0}}{\xi_{i+p+k+1} - \xi_i},
\]

\[
a_{k,j} = \frac{a_{k-1,j} - a_{k-1,j-1}}{\xi_{i+p+k-j+1} - \xi_{i+j}} \text{ for } j = 1, \ldots, k - 1,
\]

\[
a_{k,k} = \frac{a_{k-1,k-1}}{\xi_{i+p+k} - \xi_{i+k}} .
\]

**Proof.**

Only the formula for the first derivative is proven. The generalization for higher derivatives follows by deriving the equation for \( N_{i,p}^{(l)} (\xi) \) an additional \( k - 1 \) times. The explicit formulation in terms of the \( N_{i+j,p-k} (\xi) \) is proven by mathematical induction over \( k \).

The statement for the first derivative is proven by induction over \( p \).

**p = 1** By definition it holds

\[
N_{i,1} (\xi) = \frac{\xi - \xi_i}{\xi_{i+1} - \xi_i} N_{i,0} (\xi) + \frac{\xi_{i+2} - \xi}{\xi_{i+2} - \xi_{i+1}} N_{i+1,0} (\xi) .
\]
Since $N_{i,0}$ and $N_{i+1,0}$ are piecewise constant, the first derivative is given by

$$N'_i (\xi) = \frac{1}{\xi_{i+1} - \xi_i} N_{i,0} (\xi) - \frac{1}{\xi_{i+2} - \xi_{i+1}} N_{i+1,0} (\xi).$$

$p - 1 \to p$ Assume that formula (4.3) holds for degree $p - 1$. Then it has to be proven that (4.3) also holds for degree $p$. The definition of $N_{i,p}$ and the Leibniz rule lead to

$$N'_{i,p} (\xi) = \left( \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1} (\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1} (\xi) \right)'$$

$$= \frac{1}{\xi_{i+p} - \xi_i} N_{i,p-1} (\xi) + \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N'_{i,p-1}$$

$$- \frac{1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1} (\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N'_{i+1,p-1} (\xi)$$

$$= I + II - III + IV.$$

Inserting the induction hypothesis into the second term, abbreviating $N_{i,p} (\xi)$ by $N_{i,p}$, together with the definition of $N_{i,p-1}$ leads to

$$II = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \left( \frac{p - 1}{\xi_{i+p-1} - \xi_i} N_{i,p-2} - \frac{p - 1}{\xi_{i+p} - \xi_{i+1}} N_{i+1,p-2} \right)$$

$$= \frac{p - 1}{\xi_{i+p} - \xi_i} \left( \frac{\xi - \xi_i}{\xi_{i+p-1} - \xi_i} N_{i,p-2} + \frac{\xi_{i+p} - \xi}{\xi_{i+p} - \xi_{i+1}} \frac{\xi_{i+p} - \xi_{i+1}}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-2} \right)$$

$$= \frac{p - 1}{\xi_{i+p} - \xi_i} N_{i,p-1} - \frac{p - 1}{\xi_{i+p} - \xi_i} \frac{\xi_{i+p} - \xi_{i+1}}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-2}$$

$$= \frac{p - 1}{\xi_{i+p} - \xi_i} N_{i,p-1} - \frac{p - 1}{\xi_{i+p} - \xi_{i+1}} N_{i+1,p-2}.$$

Similarly the fourth summand results in

$$IV = \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} \left( \frac{p - 1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-2} - \frac{p - 1}{\xi_{i+p+1} - \xi_{i+2}} N_{i+2,p-2} \right)$$

$$= -\frac{p - 1}{\xi_{i+p+1} - \xi_{i+1}} \left( \frac{\xi - \xi_{i+1}}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-2} + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+2}} N_{i+2,p-2} \right)$$

$$= -\frac{p - 1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1} + \frac{p - 1}{\xi_{i+p} - \xi_{i+1}} N_{i+1,p-2}.$$
Inserting these two expressions finally leads to

\[ N'_{i,p} = \frac{1}{\xi_{i+p} - \xi_i} N_{i,p-1} + \frac{p - 1}{\xi_{i+p} - \xi_i} N_{i,p-1} - \frac{p - 1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-2} \]

\[ - \frac{1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1} - \frac{p - 1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1} + \frac{p - 1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-2} \]

\[ = \frac{1}{\xi_{i+p} - \xi_i} N_{i,p-1} - \frac{1}{\xi_{i+p+1} - \xi_{i+1}} N_{i+p+1,p-1} \]

\[ = \frac{p}{\xi_{i+p} - \xi_i} N_{i,p-1} - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} N_{i+p+1,p-1}. \]

\[ \square \]

### 4.1.2 B-Spline Curves

An element of the spline space \( S(\Xi, p) \) is called a B-spline curve.

**Definition 4.12 (B-Spline Curve):**

Let

\[ \Xi = \{ \xi_0 = 0, \ldots, 0, \xi_{p+1}, \ldots, \xi_{m-p-1}, 1, \ldots, 1 = \xi_m \} \]

The knot vector \( \Xi \) is called open since the first \( p+1 \) knots are equal as well as the last \( p+1 \) ones, respectively. Further let \( N_{i,p} \) be the B-spline basis functions of degree \( p \) defined on \( \Xi \).

Now let \( P_i \in \mathbb{R}^d \) for \( i = 0, \ldots, n \) be a set of so-called control points. The polygon spanned by the \( P_i \)s is called control polygon. The control points define a B-spline curve of degree \( p \) on \( \Xi \):

\[ C(\xi) := \sum_{i=0}^{n} N_{i,p}(\xi) P_i. \]

**Proposition 4.13 (Properties of B-Spline Curves [PT97]):**

The properties of the basis functions imply some important properties of the B-spline curves:

**P1** \( C(\xi) \) is piecewise polynomial of degree \( p \). If the curve is defined on a knot vector containing \( m+1 \) knots with \( n+1 \) control points, the parameters satisfy the equation \( m = n + p + 1 \).

**P2** The endpoints of a B-spline curve are interpolated exactly, namely by \( C(0) = P_0 \) and \( C(1) = P_n \).

**P3** For an affine transformation \( \Phi : \mathbb{R}^d \to \mathbb{R}^d \) it holds

\[ \Phi \left( \sum_{i=0}^{n} N_{i,p}(\xi) P_i \right) = \sum_{i=0}^{n} N_{i,p}(\xi) \Phi(P_i). \]

This property is called affine invariance.
A B-spline curve fulfills the strong convex hull property, that is the curve lies in the convex hull of the control polygon. In addition if $\xi \in [\xi_i, \xi_{i+1})$ and $p \leq i < m - p - 1$, property P2 yields that $C(\xi)$ lies in the convex hull of the control points $P_{i-p}, \ldots, P_i$.

A B-spline curve has the so-called variation diminishing property. That is every plane intersects the curve at most as often as the plane intersects the control polygon.

In the interior of a knot span the curve is infinitely often continuously differentiable, at a knot of multiplicity $k$ it is at least $(p - k)$ times continuously differentiable.

The derivative of a B-spline curve of degree $p$ is given by

$$C'(\xi) = \sum_{i=0}^{n} N'_{i,p}(\xi) P_i.$$  \hspace{1cm} (4.5)

It is itself a B-spline curve of degree $p - 1$, defined on the new knot vector

$$\Xi' = \left\{ 0, \ldots, 0, \xi_{p+1}, \ldots, \xi_{m-p-1}, 1, \ldots, 1 \right\}$$  \hspace{1cm} (4.6)

by

$$C'(\xi) = \sum_{i=0}^{n-1} N_{i,p-1}(\xi) Q_i,$$

with $Q_i := p \cdot \frac{P_{i+1} - P_i}{\xi_{i+p+1} - \xi_{i+1}}$.  \hspace{1cm} (4.7)

Proof.

Proof of P1 $C(\xi)$ is a polynomial of degree $p$ on each knot span $(\xi_i, \xi_{i+1})$ by definition. The relation between $m, n$ and $p$ follows directly from the definition of $n$.

Proof of P2 Due to the local support property of the B-spline basis functions it holds

$$C(0) = \sum_{i=0}^{n} N_{i,p}(0) P_i = N_{0,p}(0) P_0 = P_0.$$

The last equality holds as $N_{0,p} = \sum_{i=0}^{n} N_{i,p} = 1$. Analogously it holds $C(1) = P_m$.

Proof of P3 Since $\Phi$ is an affine transformation it has a representation of the form $\Phi(x) = Ax + b$ where $A$ is a square matrix and $b$ a vector.
Hence, with \( \sum_{i=0}^{n} N_{i,p}(\xi) = 1 \), the fact that \( N_{i,p}(\xi) \) is a scalar implies

\[
\Phi \left( \sum_{i=0}^{n} N_{i,p}(\xi) P_i \right) = A \left( \sum_{i=0}^{n} N_{i,p}(\xi) P_i \right) + b
\]

\[
= \sum_{i=0}^{n} A \left( N_{i,p}(\xi) P_i \right) + \left( \sum_{i=0}^{n} N_{i,p}(\xi) \right) b
\]

\[
= \sum_{i=0}^{n} N_{i,p}(\xi) (A \cdot P_i + b)
\]

\[
= \sum_{i=0}^{n} N_{i,p}(\xi) \Phi (P_i).
\]

**Proof of P4** The convex hull property of B-splines follows from the partition of unity property of the basis functions proven in Proposition 4.8 P3 and P4.

**Proof of P5** A geometric proof of the variation diminishing property can be found in [LR83]. It is beyond the scope of this thesis.

**Proof of P6** This property follows canonically from the corresponding property of the basis functions.

**Proof of P7** Since the \( P_i \)s do not depend on \( \xi \), equation (4.5) is trivial. Applying (4.4) to the derivatives of the basis functions leads to

\[
C'(\xi) = \sum_{i=0}^{n} N'_{i,p}(\xi) P_i
\]

\[
= \sum_{i=0}^{n} \left( \frac{p}{\xi_{i+1} - \xi_i} N_{i,p-1}(\xi) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \right) P_i
\]

\[
= p \left( \frac{N_{0,p-1}(\xi) P_0}{u_p - u_0} + \sum_{i=0}^{n-1} \left( N_{i+1,p-1}(\xi) \frac{P_{i+1} - P_i}{\xi_{i+p+1} - \xi_{i+1}} - \frac{N_{n+1,p-1}(\xi) P_n}{\xi_{n+p+1} - \xi_n} \right) \right)
\]

\[
= p \sum_{i=0}^{n-1} \left( N_{i+1,p-1}(\xi) \frac{P_{i+1} - P_i}{\xi_{i+p+1} - \xi_{i+1}} \right).
\]

The last equality holds as 0/0 is defined to be 0.

A basis function \( N_{i+1,p-1} \) defined on \( \Xi \) is equivalent to \( N_{i,p-1} \) defined on \( \Xi' \).
Hence it holds
\[ C' (\xi) = \sum_{i=0}^{n-1} N_{i,p-1} (\xi) Q_i \]
with \( Q_i := p \cdot \frac{P_{i+1} - P_i}{\xi_{i+p+1} - \xi_{i+1}} \) and \( N_{i,p-1} \) defined on \( \Xi' \).

\[ \square \]

### 4.1.3 B-Spline Surfaces

A B-spline surface is the tensor product of two one-dimensional B-spline curves of degree \( p^{(1)} \) and \( p^{(2)} \), respectively. Given two knot vectors

\[ \Xi^{(1)} = \{ 0, \ldots, 0, \xi^{(1)}_{p^{(1)}+1}, \ldots, \xi^{(1)}_{m^{(1)}-p^{(1)}-1}, 1, \ldots, 1 \} \]

and

\[ \Xi^{(2)} = \{ 0, \ldots, 0, \xi^{(2)}_{p^{(2)}+1}, \ldots, \xi^{(2)}_{m^{(2)}-p^{(2)}-1}, 1, \ldots, 1 \} \]

the \( n^{(1)} + 1 \) (resp. \( n^{(2)} + 1 \)) B-spline basis functions \( N_{i,j,p^{(1)}} (\xi^{(1)}) \) and \( N_{i,j,p^{(2)}} (\xi^{(2)}) \) are defined. With the help of a two-dimensional control net formed by the points \( P_{i,j} \in \mathbb{R}^d \) for \( i = 0, \ldots, n^{(1)} \) and \( j = 0, \ldots, n^{(2)} \), the B-spline surface is defined.

**Definition 4.14 (B-Spline Surface):**

A two dimensional B-spline basis function is defined as the tensor product of one-dimensional basis functions

\[ N_{ij} (\xi^{(1)}, \xi^{(2)}) = N_{i,p^{(1)}} (\xi^{(1)}) \otimes N_{j,p^{(2)}} (\xi^{(2)}) = N_{i,p^{(1)}} (\xi^{(1)}) \cdot N_{j,p^{(2)}} (\xi^{(2)}) . \]

The span of these functions form the space of two-dimensional splines, denoted by

\[ S \equiv S (\Xi^{(1)}, \Xi^{(2)}, p^{(1)}, p^{(2)}) := \text{span} \{ N_{ij} \}_{i=0,j=0}^{n^{(1)},n^{(2)}} . \]

An element of this space is called a B-spline surface and given by

\[ S (\xi^{(1)}, \xi^{(2)}) := \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{ij} (\xi^{(1)}, \xi^{(2)}) P_{i,j} \]

\[ = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,p^{(1)}} (\xi^{(1)}) N_{j,p^{(2)}} (\xi^{(2)}) P_{i,j} . \]

**Proposition 4.15 (Properties of B-Spline Surfaces [PT97]):**

Most of the properties of the one-dimensional B-spline basis functions and curves can be easily adapted to the two-dimensional case.
**P1** The two-dimensional B-spline basis functions satisfy

\[ \text{supp}(N_{ij}) = \text{supp}(N_{i,p(1)} \cdot N_{j,p(2)}) \subseteq \left[ \xi_i^{(1)}, \xi_{i+p(1)+1}^{(1)} \right] \times \left[ \xi_j^{(2)}, \xi_{j+p(2)+1}^{(2)} \right]. \]

**P2** For \( \xi^{(1)} \in \left[ \xi_{i_0}^{(1)}, \xi_{i_0+1}^{(1)} \right] \) and \( \xi^{(2)} \in \left[ \xi_{j_0}^{(2)}, \xi_{j_0+1}^{(2)} \right] \) it holds \( N_{ij}(\xi^{(1)}, \xi^{(2)}) = 0 \) if either \( i < i_0 - p^{(1)} \), \( i > i_0 \), \( j < j_0 - p^{(2)} \) or \( j > j_0 \).

**P3** For \( i = 0, \ldots, n^{(1)}, j = 0, \ldots, n^{(2)} \), \( (\xi^{(1)}, \xi^{(2)}) \in [0, 1] \times [0, 1] \), and \( p^{(1)}, p^{(2)} \geq 0 \) it holds

\[ N_{ij}(\xi^{(1)}, \xi^{(2)}) = N_{i,p^{(1)}}(\xi^{(1)}) \cdot N_{j,p^{(2)}}(\xi^{(2)}) \geq 0. \]

**P4** For \( (\xi^{(1)}, \xi^{(2)}) \in \left[ \xi_{i_0}^{(1)}, \xi_{i_0+1}^{(1)} \right] \times \left[ \xi_{j_0}^{(2)}, \xi_{j_0+1}^{(2)} \right] \) it holds

\[ \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{ij}(\xi^{(1)}, \xi^{(2)}) = \sum_{i=0}^{i_0} \sum_{j=0}^{j_0} N_{i,p^{(1)}}(\xi^{(1)}) \cdot N_{j,p^{(2)}}(\xi^{(2)}) = 1. \]

**P5** The B-spline surface basis function \( N_{ij}(\xi^{(1)}, \xi^{(2)}) = N_{i,p^{(1)}}(\xi^{(1)}) \cdot N_{j,p^{(2)}}(\xi^{(2)}) \) is smooth for all \( \xi^{(1)} \in \left( \xi_{i_0}^{(1)}, \xi_{i_0+1}^{(1)} \right), \xi^{(2)} \in \left( \xi_{j_0}^{(2)}, \xi_{j_0+1}^{(2)} \right) \) and \( i_0 = 0, \ldots, n^{(1)}, j_0 = 0, \ldots, n^{(2)} \). At a knot \( (\xi_{b_i}^{(1)}, \xi_{j_0}^{(2)}) \) with multiplicity \( k \) (resp. \( l \)) it holds that \( N_{ij}(\xi^{(1)}, \xi^{(2)}) \) is \( (p^{(1)} - k) \) times continuously differentiable in \( \xi^{(1)} \)-direction and \( (p^{(2)} - l) \) times continuously differentiable in \( \xi^{(2)} \)-direction.

**P6** For \( p^{(1)} > 0 \) and \( p^{(2)} > 0 \) each function \( N_{ij} = N_{i,p^{(1)}} N_{j,p^{(2)}} \) has exactly one maximum if the multiplicities \( k \) (resp. \( l \)) of the knots \( \xi_i^{(1)} \) (resp. \( \xi_j^{(2)} \)) satisfy \( k \leq p^{(1)} \) (resp. \( l \leq p^{(2)} \)).

**P7** The endpoints are interpolated exactly, \( S(0, 0) = P_{0,0}, S(0, 1) = P_{0,n^{(2)}}, S(1, 0) = P_{n^{(1)}, 0} \) and \( S(1, 1) = P_{n^{(1)}, n^{(2)}} \).

**P8** For an affine transformation \( \Phi : \mathbb{R}^d \rightarrow \mathbb{R}^d \) B-spline surfaces fulfill the so-called affine invariance property

\[ \Phi \left( \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{ij}(\xi^{(1)}, \xi^{(2)}) P_{i,j} \right) = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{ij}(\xi^{(1)}, \xi^{(2)}) \Phi(P_{i,j}). \]

**P9** A B-spline surface fulfills the strong convex hull property:
For \( (\xi^{(1)}, \xi^{(2)}) \in \left[ \xi_{i_0}^{(1)}, \xi_{i_0+1}^{(1)} \right] \times \left[ \xi_{j_0}^{(2)}, \xi_{j_0+1}^{(2)} \right] \) \( S(\xi^{(1)}, \xi^{(2)}) \) lies in the convex hull of the \( P_{i,j} \) with indices \( i_0 - p^{(1)} \leq i \leq i_0 \) and \( j_0 - p^{(2)} \leq j \leq j_0 \)
In the interior of a knot span the surface is infinitely often continuously differentiable in both directions, at a knot of multiplicity \( k \) (resp. \( l \)) it is at least \( (p^{(1)} - k) \) (resp. \( (p^{(2)} - l) \)) times continuously differentiable in the \( \xi^{(1)} \)- (resp. \( \xi^{(2)} \))-direction.

The \((k + l)\)-th derivative of a B-spline surface is given by

\[
\frac{\partial^{k+l}}{\partial (\xi^{(1)})^k \partial (\xi^{(2)})^l} S(\xi^{(1)}, \xi^{(2)}) = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N^{(k)}_{i,p^{(1)}}(\xi^{(1)}) N^{(l)}_{j,p^{(2)}}(\xi^{(2)}) P_{i,j}.
\]

The proof of Proposition 4.15 is omitted here since the properties follow directly from the one-dimensional case which was broadly discussed in Propositions 4.8 and 4.13.

4.2 Non-Uniform Rational B-Splines - NURBS

4.2.1 NURBS Curves

For the description of circles and other ‘nice’ geometrical structures B-splines are not sufficient. Instead we need to switch to non-uniform rational B-splines (NURBS). As before the B-spline basis functions \( N_{i,p}(\xi) \) of degree \( p \) are defined on the open knot vector

\[
\Xi = \left\{ 0, \ldots, 0, \xi_{p+1}, \ldots, \xi_{m-p-1}, 1, \ldots, 1 \right\}.
\]

Furthermore the control points \( \{P_i\}_{i=0}^n \) form the control polygon. In addition the corresponding set of weights \( \{\omega_i\}_{i=0}^n \) is defined.

**Definition 4.16** (NURBS Curve):

The NURBS curve of degree \( p \) on the knot vector \( \Xi \) with control points \( \{P_i\}_{i=0}^n \) and weights \( \{\omega_i\}_{i=0}^n \) is defined by

\[
C(\xi) := \frac{\sum_{i=0}^{n} N_{i,p}(\xi) \omega_i P_i}{\sum_{i=0}^{n} N_{i,p}(\xi) \omega_i} \text{ for all } 0 \leq \xi \leq 1.
\]

**Remark 4.17:** This definition is often written as

\[
C(\xi) = \sum_{i=0}^{n} R_{i,p}(\xi) P_i,
\]

where

\[
R_{i,p}(\xi) := \frac{N_{i,p}(\xi) \omega_i}{\sum_{j=0}^{n} N_{j,p}(\xi) \omega_j}.
\]
are the rational (or NURBS) basis functions. \( \omega \) denotes the weighting function given by

\[
\omega(\xi) = \sum_{j=0}^{n} N_{i,p}(\xi) \omega_j.
\]

The NURBS space formed by these basis functions is denoted by

\[
\mathcal{N} \equiv \mathcal{N}(\Xi, p, \omega) := \text{span}\{R_{i,p}\}_{i=0,...,n}.
\]

**Proposition 4.18 (Properties of NURBS Basis Functions [PT97]):**
NRBS and B-splines have similar properties.

**P1** \( R_{i,p}(\xi) \geq 0 \) for all \( i \) and \( p \), and for \( 0 \leq \xi \leq 1 \),

**P2** since \( R_{0,p}(0) = R_{n,p}(1) = 1 \) the endpoints are interpolated exactly, namely by \( C(0) = P_0 \) and \( C(1) = P_n \),

**P3** \( \sum_{i=0}^{n} R_{i,p}(\xi) = 1 \) for \( 0 \leq \xi \leq 1 \),

**P4** for \( p > 0 \) and if the multiplicity \( k \) of \( \xi_i \) satisfies \( k \leq p \) each function \( R_{i,p} \) has exactly one maximum in \([0,1]\),

**P5** \( \text{supp}(R_{i,p}) \subseteq [\xi_i, \xi_{i+p+1}] \) and for \( \xi \in [\xi_i, \xi_{i+1}] \) it holds \( R_{i,p} = 0 \) if \( j < i - p \) or \( j > i \).

**P6** A non-zero denominator yields that \( R_{i,p}(\xi) \in C^\infty \) for \( \xi \in (\xi_j, \xi_{j+1}) \), \( j = 0, \ldots, m \). At a knot \( \xi_j \) with multiplicity \( k \) it holds \( R_{i,p}(\xi_j) \in C^{p-k} \).

**P7** If \( \omega_i = a \) with \( a \neq 0 \) for all \( i \), the NURBS basis functions are equal to the B-spline basis functions \( R_{i,p}(\xi) = N_{i,p}(\xi) \).

**P8** The NURBS curve satisfies the affine invariance property, the strong convex hull property and the variation diminishing property.

**Proof of 4.18.**
The properties **P1** to **P6** and **P8** follow directly from the corresponding properties of the B-spline basis functions. For **P7** the insertion of \( \omega_i = a \) into the definition of \( R_{i,p}(\xi) \) and property **P4** of Proposition 4.8 yield

\[
R_{i,p}(\xi) = \frac{\sum_{j=0}^{n} N_{i,p}(\xi) \omega_j}{\sum_{j=0}^{n} N_{i,p}(\xi) \omega_j} = \frac{N_{i,p}(\xi) \cdot a}{\sum_{j=0}^{n} N_{i,p}(\xi) \cdot a} = \frac{N_{i,p}(\xi) \cdot a}{a} = N_{i,p}(\xi).
\]

□
4.2.2 NURBS in Homogeneous Coordinates

For programming NURBS there is another, more suitable approach. For a d-dimensional control point \( P_i = (x_i)_j \) and the corresponding weight \( \omega_i \), let \( P_i^\omega = (x_i, \omega_i, x_i \omega_i, \ldots, x_i \omega_i, \omega_i) \). Then the NURBS curve \( C(\xi) \) can be considered as the homogeneous projection of the \((d + 1)\)-dimensional B-spline curve \( C^\omega(\xi) \)

\[
C(\xi) = H(C^\omega(\xi)) = H\left( \sum_{i=0}^{n} N_{i,p}(\xi) P_i^\omega \right)
= \sum_{i=0}^{n} N_{i,p}(\xi) \omega_i P_i
= \sum_{i=0}^{n} R_{i,p}(\xi) P_i.
\]

Here \( H \) denotes the projection from homogeneous coordinates in \( \mathbb{R}^{d+1} \) onto the \( \mathbb{R}^d \) defined by

\[
H(x_1, \ldots, x_{d+1}) = \begin{cases} 
(x_1, \ldots, x_d), & \text{if } x_{d+1} \neq 0, \\
\text{the direction } (x_1, \ldots, x_d), & \text{if } x_{d+1} = 0.
\end{cases}
\]

This approach leads to an easier way of calculating the derivatives of a NURBS curve. Let \( A(\xi) := \omega(\xi) C(\xi) \), hence

\[
C(\xi) = \frac{\omega(\xi) C(\xi)}{\omega(\xi)} = \frac{A(\xi)}{\omega(\xi)}.
\]

Then treating \( C^\omega(\xi) \) as a \((d + 1)\)-dimensional B-spline curve with control points \( P_i^\omega \) and computing its derivatives results in the derivatives of the \((d + 1)\)-dimensional pair of functions \((A(\xi), \omega(\xi))^T\). It only remains to find a relationship between these derivatives and \( C'(\xi) \). Applying the quotient rule yields

\[
C'(\xi) = \left( \frac{A(\xi)}{\omega(\xi)} \right)'
= \frac{\omega(\xi) A'(\xi) - \omega'(\xi) A(\xi)}{\omega(\xi)^2}
= \frac{A'(\xi) - \omega'(\xi) C(\xi)}{\omega(\xi)}.
\]
Applying Leibniz’ rule leads to a recursive formula for higher derivatives of \( C(\xi) \)

\[
A^{(k)}(\xi) = (\omega(\xi) C(\xi))^{(k)}
\]

\[
= \sum_{i=0}^{k} \binom{k}{i} \omega^{(i)}(\xi) C^{(k-i)}(\xi)
\]

\[
= \omega(\xi) C^{(k)}(\xi) + \sum_{i=1}^{k} \binom{k}{i} \omega^{(i)}(\xi) C^{(k-i)}(\xi)
\]

This leads to

\[
C^{(k)}(\xi) = \frac{A^{(k)}(\xi) - \sum_{i=1}^{k} \binom{k}{i} \omega^{(i)}(\xi) C^{(k-i)}(\xi)}{\omega(\xi)}
\]

### 4.2.3 NURBS Surfaces

A NURBS surface is defined as the tensor product of two one-dimensional NURBS curves.

**Definition 4.19 (NURBS Surface):**

Let \( N_{ir}(\xi^{(1)}) \) and \( N_{jq}(\xi^{(2)}) \) denote B-spline basis functions defined on

\[
\xi^{(1)} = \{0, \ldots, 0, \xi_{p+1}^{(1)}, \ldots, \xi_{r-p-1}^{(1)}, 1, \ldots, 1\}
\]

and

\[
\xi^{(2)} = \{0, \ldots, 0, \xi_{q+1}^{(2)}, \ldots, \xi_{s-q-1}^{(2)}, 1, \ldots, 1\}
\]

respectively.

Further let \( P_{ij} \in \mathbb{R}^d \) form a bidirectional control net with corresponding weights \( \omega_{ij} \). Then a NURBS surface is defined by

\[
S(\xi^{(1)}, \xi^{(2)}) := \frac{\sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{ir}(\xi^{(1)}) N_{jq}(\xi^{(2)}) \omega_{ij} P_{ij}}{\sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{ir}(\xi^{(1)}) N_{jq}(\xi^{(2)}) \omega_{ij}}
\]

\[
= \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} R_{ij}(\xi^{(1)}, \xi^{(2)}) P_{ij}
\]
for all \( 0 \leq \xi^{(1)}, \xi^{(2)} \leq 1 \), where
\[
R_{i,j} \left( \xi^{(1)}, \xi^{(2)} \right) = \frac{N_{i,p} \left( \xi^{(1)} \right) N_{j,q} \left( \xi^{(2)} \right) \omega_{i,j}}{\sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,p} \left( \xi^{(1)} \right) N_{j,q} \left( \xi^{(2)} \right) \omega_{i,j}}
\]
\[
= \frac{N_{i,j} \left( \xi^{(1)}, \xi^{(2)} \right) \omega_{i,j}}{\sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,j} \left( \xi^{(1)}, \xi^{(2)} \right) \omega_{i,j}}
\]
\[
= \frac{N_{i,j} \left( \xi^{(1)}, \xi^{(2)} \right) \omega_{i,j}}{\omega \left( \xi^{(1)}, \xi^{(2)} \right)}.
\]

The \( R_{i,j} \) are the two-dimensional NURBS basis functions and \( \omega \) denotes the weighting function,
\[
\omega \left( \xi^{(1)}, \xi^{(2)} \right) = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,j} \left( \xi^{(1)}, \xi^{(2)} \right) \omega_{i,j}.
\]

The space of NURBS in two dimensions is defined by
\[
\mathcal{N} \equiv \mathcal{N} \left( \Xi^{(1)}, \Xi^{(2)}; p, q, \omega \right) := \text{span} \left\{ R_{i,j} \right\}_{i=0,...,j=0}^{n^{(1)},n^{(2)}}.
\]

The NURBS surfaces have properties similar to the B-spline surfaces which follow directly from the properties of the NURBS curves. Again it is more convenient to consider NURBS surfaces as the homogeneous projection of B-spline surfaces. With \( H \) as above and the homogeneous coordinates \( P_{i,j}^{w} = \left( x_{1,i} \omega_{i,j}, x_{2,i} \omega_{i,j}, \ldots, x_{d,i} \omega_{i,j}, \omega_{i,j} \right) \) the following representation of NURBS surfaces holds
\[
S \left( \xi^{(1)}, \xi^{(2)} \right) = H \left( S^{w} \left( \xi^{(1)}, \xi^{(2)} \right) \right)
\]
\[
= H \left( \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,p} \left( \xi^{(1)} \right) N_{j,q} \left( \xi^{(2)} \right) P_{i,j}^{w} \right)
\]
\[
= \frac{\sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,p} \left( \xi^{(1)} \right) N_{j,q} \left( \xi^{(2)} \right) \omega_{i,j} P_{i,j}}{\sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,p} \left( \xi^{(1)} \right) N_{j,q} \left( \xi^{(2)} \right) \omega_{i,j}}
\]
\[
= \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} R_{i,p} \left( \xi^{(1)} \right) R_{j,q} \left( \xi^{(2)} \right) P_{i,j}.
\]
4.3 Refinement

There are two elementary ways to refine a NURBS curve, namely the insertion of new knots and the elevation of the degree \( p \) of the curve. The difficulty in doing so is to maintain the original curve. This means, neither its geometry nor its parametrization should be changed during the refinement process.

4.3.1 Knot Insertion

Knot insertion is the equivalent to classical h-refinement as it refines the control net. Let \( \xi \in [\xi_k, \xi_{k+1}) \) be inserted into the knot vector

\[
\Xi = \{\xi_0, \ldots, \xi_m\},
\]

thus creating a new knot vector

\[
\Xi^* = \{\xi_0 = \xi_0, \ldots, \xi_k = \xi_k, \xi_{k+1} = \xi, \xi_{k+2} = \xi_{k+1}, \ldots, \xi_{m+1} = \xi_m\}.
\]

On this knot vector a new set of basis functions \( \{\bar{N}_i|0 \leq i \leq n+1\} \) is defined. A curve \( C(\xi) = \sum_{i=0}^{n} P_iN_{i,p}(\xi) \) should remain geometrically and parametrically unchanged under the change of basis from \( \{N_i|0 \leq i \leq n\} \) to \( \{\bar{N}_i|0 \leq i \leq n+1\} \). Therefore new control points \( Q_i \) satisfying

\[
C(\xi) = \sum_{i=0}^{n} P_iN_{i,p}(\xi) = \sum_{i=0}^{n+1} Q_i\bar{N}_{i,p}(\xi)
\]

are sought.

Remark 4.20: The support property of the basis functions implies that the \( N_{i,p} \) remain unchanged for \( i \leq k-p-1 \) and for \( i \geq k+1 \) they are given by \( \bar{N}_{i+1,p} = N_{i,p} \). Therefore the new control points are given by \( Q_i = P_i \) (resp. \( Q_{i+1} = P_i \)) for \( i \leq k-p-1 \) (resp. \( i \geq k+1 \)).

Proposition 4.21 (New Control Points for Knot Insertion [PT97]):

For the new control points

\[
Q_i = \alpha_i P_i + (1-\alpha_i) P_{i-1},
\]

where \( \alpha_i = \begin{cases} 1, & \text{if } i \leq k-p, \\ \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i}, & \text{if } k-p+1 \leq i \leq k, \\ 0, & \text{if } i \geq k+1. \end{cases} \)

the curve \( C(\xi) \) stays geometrically and parametrically unchanged.

An outline of the proof of this statement can be found in [PT97, pp. 142]. Although it is very technical it is given here with some additional details since h-refinement is
fundamental for isogeometric analysis. In order to prove Proposition 4.21 a relation between the basis functions on the knot vectors $\Xi$ and $\Xi$ is essential.

**Lemma 4.22** (Transformation of the Basis Functions):
Let $\Xi, \Xi, N_{i,p}$ and $\bar{N}_{i,p}$ be defined as above. For $k − p ≤ i ≤ k$ the following relation between the basis functions holds

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} \cdot N_{i,p}(\xi) + \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} \cdot \bar{N}_{i+1,p}(\xi).$$

**Proof of Lemma 4.22.**
The proof is carried out by induction over $p$.

$p = 0$ If $p = 0$ only the formula for $k − 0 ≤ i ≤ k$ i.e. $i = k$ needs to be proven. By definition the basis functions satisfy

$$N_{i,0}(\xi) = \begin{cases} 1, & \text{if } \xi_k ≤ \xi ≤ \xi_{k+1}, \\ 0, & \text{otherwise,} \end{cases}$$

$$\bar{N}_{i,0}(\xi) = \begin{cases} 1, & \text{if } \xi_k ≤ \xi ≤ \xi_k, \\ 0, & \text{otherwise,} \end{cases}$$

$$\bar{N}_{i+1,0}(\xi) = \begin{cases} 1, & \text{if } \xi_k ≤ \xi ≤ \xi_{k+1}, \\ 0, & \text{otherwise.} \end{cases}$$

These definitions and the relations between the knots, $\xi_k = \xi_k$, $\xi_{k+1} = \xi$ and $\xi_{k+2} = \xi_{k+1}$ yield

$$\frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} \cdot N_{i,p}(\xi) + \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} \cdot \bar{N}_{i+1,p}(\xi)$$

$$= \frac{\xi - \xi_k}{\xi - \xi_k} \cdot N_{k,0}(\xi) + \frac{\xi_{k+1} - \xi}{\xi_{k+1} - \xi} \cdot \bar{N}_{k+1,0}(\xi)$$

$$= \begin{cases} 1, & \text{if } \xi_k ≤ \xi < \xi_{k+1}, \\ 1, & \text{if } \xi_k ≤ \xi < \xi_{k+2}, \\ 0, & \text{otherwise,} \end{cases}$$

$$= \begin{cases} 1, & \text{if } \xi_k ≤ \xi < \xi_{k+2}, \\ 0, & \text{otherwise,} \end{cases}$$

$$= N_{k,0}(\xi)$$

$$= N_{i,p}(\xi).$$
Assume that the formula holds for degree $p$ and $i = k - p, \ldots, k$. First consider the case $k - p - 1 < i < k$. For sake of simplicity we write $N_{i,p} = N_{i,p}(\xi)$. Then we can apply the definition of $N_{i,p+1}$ and the induction hypothesis to obtain

$$N_{i,p+1} = \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} N_{i,p} + \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} N_{i+1,p}$$

$$= \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} \cdot \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} N_{i,p} + \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} \cdot \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} N_{i+1,p}$$

$$+ \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} \cdot \frac{\xi}{\xi_{i+p+2} - \xi_{i+1}} N_{i+2,p}$$

$$+ \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} \cdot \frac{\xi_{i+p+3} - \xi}{\xi_{i+p+3} - \xi_{i+2}} N_{i+2,p}$$

$$= I + II + III + IV.$$

Recall that $i \le k - 1 < k$ and $i + 1 \le k$, thus $\xi_i = \xi_i$ and $\xi_{i+1} = \xi_{i+1}$. On the other hand $k + 1 \le i + p + 1$ yields $\xi_{i+p+2} = \xi_{i+p+1}$ and $\xi_{i+p+3} = \xi_{i+p+2}$. The application of these relations and the permutation of the denominators converts the first and last summand to the following form:

$$I = \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} \cdot \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} N_{i,p} = \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} \cdot \frac{\xi - \xi_i}{\xi_{i+p+1} - \xi_i} N_{i,p}$$

$$IV = \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} \cdot \frac{\xi_{i+p+3} - \xi}{\xi_{i+p+3} - \xi_{i+2}} N_{i+2,p} = \frac{\xi_{i+p+3} - \xi}{\xi_{i+p+3} - \xi_{i+2}} \cdot \frac{\xi_{i+p+3} - \xi}{\xi_{i+p+3} - \xi_{i+1}} N_{i+2,p}.$$

The other two terms are more complicated to handle. Let CD denote the common denominator, the application of the knot relations then implies
II + III

\[
\frac{\xi_i - \xi_i}{\xi_{i+p+1} - \xi_i} \cdot \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} + \frac{\xi_{i+p+2} - \xi}{\xi_{i+p+2} - \xi_{i+1}} \cdot \frac{\xi - \xi_{i+1}}{\xi_{i+p+2} - \xi_{i+1}} \times \text{N}_{i+1,p}
\]

\[
= \frac{\text{N}_{i+1,p}}{ \text{CD} } \left( \frac{\xi_{i+p+2} - \xi_i}{\xi_{i+p+2} - \xi_i} \cdot \frac{\xi_{i+p+3} - \xi_{i+1}}{\xi_{i+p+2} - \xi_{i+1}} \times \left( \xi_i \cdot (\xi_{i+p+2} - \xi) \cdot \text{N}_{i+1,p} - \xi_i \cdot (\xi_{i+p+3} - \xi_{i+1}) \right) \right)
\]

\[
= \frac{\text{N}_{i+1,p}}{ \text{CD} } \left( \frac{\xi_{i+p+2} - \xi_i}{\xi_{i+p+2} - \xi_i} \cdot \frac{\xi_{i+p+3} - \xi_{i+1}}{\xi_{i+p+2} - \xi_{i+1}} \times \left( \xi_i \cdot (\xi_{i+p+2} - \xi) \cdot \text{N}_{i+1,p} - \xi_i \cdot (\xi_{i+p+3} - \xi_{i+1}) \right) \right)
\]
The summation of these new representations of I-IV and the application of the definition of $\overline{N}_{i,p}$ and $\overline{N}_{i+1,p}$ finally lead to

$$N_{i,p+1} = I + II + III + IV$$

$$= \frac{\xi - \xi_i}{\xi_i - \xi_{i+1}} \cdot \frac{\xi - \xi_i}{\xi_i - \xi_{i+1}} \cdot N_{i,p} + \frac{\xi - \xi_i}{\xi_i - \xi_{i+1}} \cdot \overline{N}_{i+1,p}$$

$$+ \frac{\xi - \xi_{i+1}}{\xi_{i+1} - \xi_i} \cdot \frac{\xi - \xi_{i+1}}{\xi_{i+1} - \xi_i} \cdot N_{i,p+1} + \frac{\xi - \xi_{i+1}}{\xi_{i+1} - \xi_i} \cdot \overline{N}_{i+2,p}$$

$$= \overline{N}_{i+1,p} = \overline{N}_{i+1,p}$$

Let us now consider the case $i = k - p - 1$. Here we have to be careful as we cannot apply the induction hypothesis for $N_{i,p} = N_{k-p-1}$. As described in Remark 4.20 it holds $\overline{N}_{k-p-1} = N_{k-p-1}$. Hence in this case it holds

$$N_{i,p+1} = \frac{\xi - \xi_{k-1}}{\xi_k - \xi_{k-1}} \cdot N_{k-1,p} + \frac{\xi_{k+1} - \xi}{\xi_{k+1} - \xi_k} \cdot N_{k,p}$$

$$= \frac{\xi - \xi_{k-1}}{\xi_k - \xi_{k-1}} \cdot N_{k-1,p}$$

$$+ \frac{\xi_{k+1} - \xi}{\xi_{k+1} - \xi_k} \cdot \left( \frac{\xi - \xi_k}{\xi_k - \xi_{k-1}} \cdot N_{k-1,p} + \frac{\xi - \xi_{k+2}}{\xi_{k+2} - \xi_{k+1}} \cdot N_{k-1,p} \right)$$

$$= \frac{\xi - \xi_{k-1}}{\xi_k - \xi_{k-1}} \cdot N_{k-1,p} + \frac{\xi_{k+1} - \xi}{\xi_{k+1} - \xi_k} \cdot \frac{\xi - \xi_k}{\xi_k - \xi_{k-1}} \cdot N_{k-1,p}$$

$$+ \frac{\xi_{k+1} - \xi}{\xi_{k+1} - \xi_k} \cdot \frac{\xi - \xi_{k+2}}{\xi_{k+2} - \xi_{k+1}} \cdot N_{k-1,p}$$

$$= I + II + III.$$
The second summand can be rearranged to

$$ II = \frac{\xi_{k+1}\xi - \xi_{k+1}\xi_{k-p} - \xi \xi + \xi \xi_{k-p}}{(\xi_{k+2} - \xi_{k-p}) \cdot (\xi_{k+1} - \xi_{k-p})} N_{k-p,p} $$

$$ = \frac{\xi \xi_{k+2} - \xi \xi_{k+2} \xi_{k-p} + \xi_{k-p} \xi}{(\xi_{k+2} - \xi_{k-p}) \cdot (\xi_{k+1} - \xi_{k-p})} N_{k-p,p} + \frac{\xi_{k+2} - \xi_{k+2} \xi - \xi \xi_{k-p} + \xi_{k-p} \xi}{(\xi_{k+2} - \xi_{k-p}) \cdot (\xi_{k+1} - \xi_{k-p})} N_{k-p,p} $$

$$ = \frac{(\xi_{k+2} - \xi_{k-p}) \cdot (\xi - \xi_{k-p})}{(\xi_{k+2} - \xi_{k-p}) \cdot (\xi_{k+1} - \xi_{k-p})} N_{k-p,p} + \frac{(\xi_{k+2} - \xi_{k-p}) \cdot (\xi_{k+1} - \xi_{k-p})}{(\xi_{k+2} - \xi_{k-p}) \cdot (\xi_{k+1} - \xi_{k-p})} N_{k-p,p} $$

$$ = \frac{\xi_{k+1} - \xi}{\xi_{k+1} - \xi_{k-p}} N_{k-p,p} + \frac{\xi_{k+2} - \xi}{\xi_{k+2} - \xi_{k-p}} \cdot \frac{\xi - \xi_{k-p}}{\xi_{k+1} - \xi_{k-p}} N_{k-p,p}. $$

Insertion of (4.8) finally leads to

$$ N_{i,p+1}(\xi) = I + II + III $$

$$ = \frac{\xi - \xi_{k-p-1}}{\xi_{k} - \xi_{k-1}} \cdot N_{k-p-1,p} + \frac{\xi_{k+1} - \xi}{\xi_{k+1} - \xi_{k-p}} \cdot N_{k-p,p} $$

$$ + \frac{\xi_{k+2} - \xi}{\xi_{k+2} - \xi_{k-p}} \cdot \frac{\xi - \xi_{k-p}}{\xi_{k+1} - \xi_{k-p}} \cdot N_{k-p,p} + \frac{\xi_{k+2} - \xi}{\xi_{k+2} - \xi_{k-p}} \cdot \frac{\xi_{k+2} - \xi}{\xi_{k+2} - \xi_{k-p+1}} \cdot N_{k-p+1,p} $$

$$ = N_{k-p-1,p+1} + \frac{\xi_{k+2} - \xi}{\xi_{k+2} - \xi_{k-p}} \cdot N_{k-p,p+1} $$

$$ = \frac{\xi - \xi_{k-p-1}}{\xi_{k+1} - \xi_{k-p-1}} \cdot N_{k-p-1,p+1} + \frac{\xi_{k+2} - \xi}{\xi_{k+2} - \xi_{k-p}} \cdot N_{k-p,p+1}. $$

The last equality holds as $\xi = \xi_{k+1}$.

In the case $i = k$ it holds $\xi_{i+1} = \xi_{k+1} = \xi \neq \xi_{k+1}$. Hence the induction hypothesis cannot be applied for $N_{k+1,p}$. Instead it holds $N_{k+1,p} = \overline{N}_{k+2,p}$. Keeping this detail in mind the proof works similarly to the other cases. \( \square \)
Proof of Proposition 4.21.
Having proven Lemma 4.22 we can insert the relations between the new and old basis functions in the definition of the curve \( C(\xi) \):

\[
\sum_{i=0}^{n+1} Q_i \overline{N}_{i,p}(\xi) = \sum_{i=0}^{n} P_i N_{i,p}(\xi)
\]

\[
= \sum_{i=0}^{k-p-1} P_i \overline{N}_{i,p}(\xi) + \sum_{i=k+1}^{n} P_i \overline{N}_{i-1,p}(\xi)
\]

\[
+ \sum_{i=k-p}^{k} P_i \left( \frac{\overline{\xi} - \overline{\xi}_i}{\overline{\xi}_{i+p+1} - \overline{\xi}_i} \overline{N}_{i,p}(\xi) + \frac{\overline{\xi}_{i+p+2} - \overline{\xi}}{\overline{\xi}_{i+p+2} - \overline{\xi}_{i+1}} \overline{N}_{i+1,p}(\xi) \right)
\]

\[
= \sum_{i=0}^{k-p-1} P_i \overline{N}_{i,p}(\xi) + \sum_{i=k+1}^{n} P_i \overline{N}_{i-1,p}(\xi)
\]

\[
+ P_{k-p} \frac{\overline{\xi} - \overline{\xi}_{k-p}}{\overline{\xi}_{k+1} - \overline{\xi}_{k-p}} \overline{N}_{k,p}(\xi) + P_{k+1} \frac{\overline{\xi}_{k+1} - \overline{\xi}}{\overline{\xi}_{k+1} - \overline{\xi}_{k+2}} \overline{N}_{k+1,p}(\xi)
\]

\[
+ \sum_{i=k-p+1}^{k} \left( P_i \left( \frac{\overline{\xi} - \overline{\xi}_i}{\overline{\xi}_{i+p+1} - \overline{\xi}_i} \overline{N}_{i,p}(\xi) + P_{i-1} \left( \frac{\overline{\xi}_{i+p+1} - \overline{\xi}}{\overline{\xi}_{i+p+1} - \overline{\xi}_{i+1}} \overline{N}_{i,p}(\xi) \right) \right) \right)
\]

The last equality holds as \( \overline{\xi} = \overline{\xi}_{k+1} \).
Recall that \( \overline{\xi}_{i+p+1} = \overline{\xi}_{i+p} \), then the comparison of the coefficients in front of each basis function \( \overline{N}_{i,p} \) leads to

\[
Q_i = \alpha_i P_i + (1 - \alpha_i) P_{i-1}, \text{ with } \alpha_i = \begin{cases} 
1, & \text{if } i \leq k - p, \\
\frac{\overline{\xi} - \overline{\xi}_i}{\overline{\xi}_{i+p} - \overline{\xi}_i}, & \text{if } k - p + 1 \leq i \leq k, \\
0, & \text{if } i \geq k + 1.
\end{cases}
\]

\[\square\]

35
4.3.2 Degree Elevation

Degree elevation is the equivalent to classical p-refinement. When elevating the degree of a curve from \( p \) to \( p + 1 \) the discontinuities in the \( p \)-th derivative should be preserved. Let the curve \( C(\xi) \) be defined on the knot vector 
\[
\Xi = \left\{ 0, \ldots, 0, \xi_i, \ldots, \xi_i, \ldots, 1, \ldots, 1 \right\}.
\]

Since the curve is \((p - m_i)\)-times continuously differentiable at a knot \( \xi_i \) of multiplicity \( m_i \), when elevating the degree from \( p \) to \( p + 1 \) the multiplicity must be elevated as well. Hence every knot of \( \Xi \) must be inserted once to create a new knot vector of the form 
\[
\Xi' = \left\{ 0, \ldots, 0, \xi_i, \ldots, \xi_i, \ldots, 1, \ldots, 1 \right\}_{(p+1)+1}.
\]

In addition to that the curve should remain geometrically and parametrically unchanged. Hence new control points \( Q_i \) which satisfy 
\[
C_p(\xi) = \sum_{i=0}^{n} N_{i,p}(\xi) P_i = \sum_{i=0}^{n} N_{i,p+1}(\xi) Q_i = C_{p+1}(\xi)
\]
are sought.

To this end the curve is divided into its polynomial segments by inserting points. That is to get the segment corresponding to \([\xi_i, \xi_{i+1}]\), \( \xi_i \) and \( \xi_{i+1} \) are inserted in sufficient number, such that the curve interpolates these points exactly. Afterwards the degree of the curve is elevated in each segment, the segments are combined and redundant knots are erased. The detailed description of this proof is out of the scope of this thesis. It can be found in [PT97].

4.3.3 k-Refinement

In contrast to the classical h- and p-refinements, knot insertion and degree elevation do not commute. To show this consider the continuity of a NURBS curve \( C(\xi) \). After an insertion of a knot \( \xi \) the curve is \((p - 1)\) times continuously differentiable at this knot. During an elevation of the degree of the curve to \( q \), the continuity is preserved. That is at \( \xi \) the curve still is \((p - 1)\) times continuously differentiable. On the other hand, elevating the degree of the curve to \( q \) first and inserting a new knot \( \xi \) afterwards leads to \((q - 1)\) continuous derivatives at this point.

This non-commutativity is used by [HCB05] to define the new k-refinement. It elevates the degree of the curve to degree \( q \) first and then proceeds with knot insertions. This way the basis functions have better continuity than the other way round. Additional details are out of the scope of this thesis and can be found in [HCB05] and [Baz06].
5 Isogeometric Analysis (IGA) with NURBS

This chapter starts with an introduction of the isogeometric concept followed by the corresponding analysis.

5.1 Isogeometric Concept

In classical FEM the basis functions of $V_h$ are usually piecewise polynomial functions of degree $p$ on a given mesh with polygonal boundary. The approach of this diploma thesis is to use NURBS functions instead.

The main issue of the isogeometric concept is to describe the geometry of the domain $\Omega$ and the solution space $V$ with the same basis functions. Since NURBS are used in computer aided design (CAD) to describe exact shapes, it is an appealing choice to utilize them as basis functions.

Let us assume that we have a two-dimensional domain $\Omega$ given as the image of the unit square under a transformation $S$,

$$\Omega = \{(x, y) | (x, y) = S(\xi^{(1)}, \xi^{(2)}) \text{ with } (\xi^{(1)}, \xi^{(2)}) \in [0, 1]^2 \}.$$ 

The transformation $S$ is given as a NURBS, that is we have two knot vectors $\Xi^{(1)}$ and $\Xi^{(2)}$, a set of control points $P_{i,j} \in \mathbb{R}^d$, and a set of weights $\omega_{i,j} \in \mathbb{R}$. The homogeneous control points $P_{i,w}^{w}$ are defined as in Chapter 4.2. Thus $S(\xi^{(1)}, \xi^{(2)})$ is described via its representation in homogeneous coordinates:

$$S(\xi^{(1)}, \xi^{(2)}) = H(S^{w}(\xi^{(1)}, \xi^{(2)})) = \sum_{i=0}^{\mu^{(1)}} \sum_{j=0}^{\mu^{(2)}} N_{i,j}(\xi^{(1)}, \xi^{(2)}) P_{i,j}^{w}. $$

The same basis functions $N_{i,j}$ should be used for the solution space. That is, some coefficients $c_{i,j}$ are computed such that the numerical solution is given by

$$u = \sum_{i=0}^{\mu^{(1)}} \sum_{j=0}^{\mu^{(2)}} N_{i,j}(\xi^{(1)}, \xi^{(2)}) c_{i,j}. $$

These coefficients can simply be inserted as a third dimension into the control points $P_{i,j}$. This approach guarantees the isogeometric concept of a common basis for geometry and solution.

The main advantage of the isogeometric concept is that it preserves the exact geometry of shapes designed in CAD. In this way geometrical errors are prevented. Furthermore the exact geometry of the domain is already established in the coarsest mesh. Hence it is no longer necessary to communicate with the exact design in every refinement step.
5.2 A-priori Error Estimates

This section discusses the approximation properties of splines. It justifies that NURBS are well chosen basis functions. Hence it shows that the IGA solutions given by Definition 3.3 converge to the solution of the weak problem defined in Definition 3.2.

5.2.1 Definitions

This first part introduces some notation utilized in the following inequalities. It basically follows Bazilevs et al. [BBdVC+06] with some minor changes.

Definition 5.1 (d-Dimensional Tensor Product Space of B-Splines):
Let \( \Xi^{(1)}, \ldots, \Xi^{(d)} \) be \( d \) open knot vectors, \( p^{(1)}, \ldots, p^{(d)} \) the corresponding degrees and
\[
\Xi^{(\alpha)} = \{ \xi^{(\alpha)}_0 = 0, \ldots, \xi^{(\alpha)}_{p^{(\alpha)}+1}, \ldots, \xi^{(\alpha)}_{m^{(\alpha)}-p^{(\alpha)}-1} = \xi^{(\alpha)}_{m^{(\alpha)}} \} \text{ for } \alpha = 1, \ldots, d.
\]
Then we define \( p := \min_{1 \leq \alpha \leq d} \{ p^{(\alpha)} \} \).

Further let \( N_{i_1, \ldots, i_d}^{(\alpha)} \) for \( i_\alpha = 0, \ldots, n^{(\alpha)} \) and \( \alpha = 1, \ldots, d \) be the B-spline basis functions of degree \( p^{(\alpha)} \) defined on \( \Xi^{(\alpha)} \). Then a \( d \)-dimensional B-spline basis function is defined by
\[
N_{i_1, \ldots, i_d} := N_{i_1, p^{(1)}} \otimes \cdots \otimes N_{i_d, p^{(d)}} = \bigotimes_{\alpha=1}^d N_{i_\alpha, p^{(\alpha)}}.
\]
The \( d \)-dimensional space of B-splines is then
\[
S \equiv S(\Xi^{(1)}, \ldots, \Xi^{(d)}; p^{(1)}, \ldots, p^{(d)}) := \bigotimes_{\alpha=1}^d S(\Xi^{(\alpha)}; p^{(\alpha)}) = \text{span}\{ N_{i_1, \ldots, i_d}^{(\alpha)} \}_{i_1=0, \ldots, i_d=0}^{n^{(\alpha)}},
\]
Similarly we introduce the space of NURBS:

Definition 5.2 (NURBS Space):
Let knot vectors, degrees, and B-Spline basis functions be defined as in Definition 5.1. Further let \( \omega_{i_1, \ldots, i_d} > 0 \) denote the weight which corresponds to \( N_{i_1, \ldots, i_d}^{(\alpha)} \) and
\[
\omega = \sum_{i_1=0, \ldots, i_d=0}^{n^{(1)}, \ldots, n^{(d)}} \omega_{i_1, \ldots, i_d} N_{i_1, \ldots, i_d}
\]
the weighting function. A \( d \)-dimensional NURBS basis function is defined by
\[
R_{i_1, \ldots, i_d} := \frac{\omega_{i_1, \ldots, i_d} N_{i_1, \ldots, i_d}}{\omega}.
\]
The \( d \)-dimensional space of NURBS is
\[
N \equiv N(\Xi^{(1)}, \ldots, \Xi^{(d)}; p^{(1)}, \ldots, p^{(d)}; \omega) := \text{span}\{ R_{i_1, \ldots, i_d}^{(\alpha)} \}_{i_1=0, \ldots, i_d=0}^{n^{(\alpha)}},
\]
In the one-dimensional case these spaces and basis functions are defined on the interval
[0,1] divided into subintervals \([\xi_i, \xi_{i+1}]\), the so-called knot spans. Let us now extend this concept to higher dimensions.

**Definition 5.3** (Meshes, Elements, and Support Extensions):
Let \(\Xi^{(1)}, \ldots, \Xi^{(d)}\) denote knot vectors as in Definition 5.1.

1. The knot vectors divide \((0,1)^d\) into \(d\)-dimensional knot spans also called elements. The set of these elements \(Q\) defines a mesh \(Q\):

\[
Q \equiv Q(\Xi^{(1)}, \ldots, \Xi^{(d)}) := \left\{ Q = \bigotimes_{\alpha=1}^{d} (\xi^{(\alpha)}_{i_{\alpha}}, \xi^{(\alpha)}_{i_{\alpha}+1}) | Q \neq \emptyset, p^{(\alpha)} + 1 \leq i_{\alpha} \leq n^{(\alpha)} \right\}.
\]

2. For an element \(Q \in Q\) with \(Q = \bigotimes_{\alpha=1}^{d} (\xi^{(\alpha)}_{i_{\alpha}}, \xi^{(\alpha)}_{i_{\alpha}+1})\), we define its support extension \(\hat{Q}\) as the union of all elements in the support of that basis functions \(N_{i_1, \ldots, i_d}\) whose support intersects \(Q\):

\[
\hat{Q} := \left\{ Q' \mid \exists N_{i_1, \ldots, i_d} \text{ with } Q \cap \text{supp} (N_{i_1, \ldots, i_d}) \neq \emptyset \text{ s.t. } Q' \cap \text{supp} (N_{i_1, \ldots, i_d}) \neq \emptyset \right\} = \bigotimes_{\alpha=1}^{d} (\xi^{(\alpha)}_{i_{\alpha}-p^{(\alpha)}}, \xi^{(\alpha)}_{i_{\alpha}+p^{(\alpha)}+1}).
\]

3. We define \(m_{Q_1, Q_2}\) to be the number of continuous derivatives of a function across the common border of two elements, \(\partial Q_1 \cap \partial Q_2\).

4. Let \(h_Q\) denote the diameter of an element \(Q \in Q\) and \(h = \max\{h_Q | Q \in Q_h\}\) the mesh size of a mesh \(Q_h\).

In the context of isogeometric analysis NURBS are used to parametrize some physical domain \(\Omega\). Therefore the NURBS geometrical map is defined.

**Definition 5.4** (Parametrization):
Let \(S \in \mathcal{N}\) be a NURBS. Then \(S\) is given by the control points \(P_{i_1, \ldots, i_d}\) and is a parametrization of the physical domain \(\Omega = \text{im} (S)\):

\[
S : (0,1)^d \rightarrow \Omega
\]

\[
S = \sum_{i_1=0, \ldots, i_d=0}^{n^{(1)}, \ldots, n^{(d)}} P_{i_1, \ldots, i_d} R_{i_1, \ldots, i_d}.
\]

For further calculations we always suppose that \(S\) is invertible and its inverse \(S^{-1}\) is smooth on each element \(Q\).

Via \(S\) each element \(Q \in Q\) is mapped onto an element \(K\) of the physical domain,

\[
K = S (Q) := \{ S (\xi) | \xi \in Q \},
\]

and the support extension \(\hat{Q}\) is mapped onto \(\hat{K} = S(\hat{Q})\).
The union of these elements in the physical domain form a mesh of the physical domain, namely
\[ K := \{ K = S(Q) | Q \in \mathcal{Q} \}. \]

Finally, to use the isogeometric concept, the space \( \mathcal{V} \) of NURBS on the physical domain is necessary,
\[ \mathcal{V} \equiv \mathcal{V}(p_1, \ldots, p^d) := \text{span}\left\{ R_{i_1, \ldots, i_d} \circ S^{-1} \right\}_{i_1=0, \ldots, i_d=0}, \]
the so-called push-forward of the space \( \mathcal{N} \) onto the parametric domain \((0, 1)^d\).

The goal of isogeometric analysis is to approximate the solution of a PDE. Therefore knot insertion, i.e. \( h \)-refinement, is applied, which leads to a family of refined meshes.

**Definition 5.5 (Refined Meshes):**
Let \( \{ Q_h \}_h \) be a family of meshes on the parametric domain \((0, 1)^d\) with global mesh size \( h \) as defined above. We assume that there is a coarsest mesh \( Q_{0h} \) such that all other meshes are refinements of \( Q_{0h} \) obtained via the refinement procedure described in Section 4.3.1. Further we assume the family of meshes to be shape regular, meaning that there is a uniform bound with respect to \( Q \) and \( h \) of the ratio between the diameter \( h_Q \) of an element \( Q \) and its smallest edge.

**Lemma 5.6:**
A shape regular mesh is locally quasi-uniform. That is the ratio \( h_{Q_1} / h_{Q_2} \) between the sizes of two neighboring elements \( Q_1 \) and \( Q_2 \) is uniformly bounded.

**Proof of Lemma 5.6.**
Let \( q_1 \) and \( q_2 \) denote the length of the smallest edge in \( Q_1 \) and \( Q_2 \), respectively, and let \( q \) be the length of the common edge of \( Q_1 \) and \( Q_2 \). By assumption there exists an upper bound \( N \in \mathbb{N} \) such that \( h_{Q_1} / q_1 \leq N \) and \( h_{Q_2} / q_2 \leq N \). As \( q \geq q_1 > 0 \) and \( q \geq q_2 > 0 \) this leads to
\[ \frac{h_{Q_1}}{q} \leq \frac{h_{Q_1}}{q_1} \leq N. \]
Multiplication by \( 0 < q / h_{Q_2} \leq 1 \) leads to
\[ \frac{h_{Q_1}}{h_{Q_2}} \leq N \cdot \frac{q}{h_{Q_2}} \leq N. \]
Analogously \( h_{Q_2} / h_{Q_1} \leq N \) is proven. Hence it holds \( h_{Q_1} \cdot 1/N \leq h_{Q_2} \leq h_{Q_1} \cdot N \) and the mesh is locally quasi uniform. \( \square \)

**Remark 5.7:** Since an element \( Q_i \in Q_h \) is a \( d \)-dimensional multi-interval, the area \( A_{Q_i} \) of \( Q_i \) satisfies for \( i = 1, 2 \)
\[ q_i^d \leq A_{Q_i} \leq h_{Q_i}^d, \]
Hence the ratio between the areas of neighboring elements is bounded as well

\[
\frac{A_{Q_1}}{A_{Q_2}} \leq \frac{h_{Q_1}}{q_{Q_2}} \leq \frac{h_{Q_2}}{q_{Q_2}} \leq N^{2d}.
\]

As before, starting from a mesh \( Q_h \) we can construct a mesh of the physical domain \( K_h \) and the function spaces \( S_h, N_h \) and \( V_h \). This construction leads to the families of meshes and function spaces \( \{K_h\}_h \), \( \{S_h\}_h \), \( \{N_h\}_h \) and \( \{V_h\}_h \).

As seen in Proposition 4.21 the geometrical map \( S \) and the weighting function \( \omega \) remain geometrically and parametrically unchanged during the refinement process. Thus they are fixed in \( S_{h_0} \) and \( N_{h_0} \) and are the same for every mesh size \( h \).

In the estimates proven in this section two constants appear. They might be different at each occurrence. \( C > 0 \) only depends on the dimension \( d \), the degrees \( p^{(a)} \), and the shape regularity of the mesh family \( \{Q_h\}_h \). \( C_s > 0 \) may in addition depend on the shape of \( \Omega \). Hence \( C_s \) is no longer independent of \( VS \) and the weighting function \( \omega \). But it must be independent of the size of \( \Omega \). Hence it is homogeneous of order 0 with respect to \( VS \) and \( \omega \). It is only allowed to depend on \( VS/\|VS\|_{L^\infty(\Omega)} \) and \( \omega/\|\omega\|_{L^\infty(\Omega)} \).

For the definition of a weak solution of a PDE we need a Hilbert space \( V \). An appropriate choice is the so-called bent Sobolev space introduced in [BBdVC+06].

**Definition 5.8 (Bent Sobolev Space):**

The bent Sobolev space of order \( m \in \mathbb{N} \) is defined by

\[
H^m := \left\{ v \in L^2 \left( (0,1)^d \right) \text{ such that } v|_Q \in H^m(Q), \text{ for all } Q \in \mathcal{Q}, \text{ and } \nabla^k (v|_Q) = \nabla^k (v|_{Q_2}) \text{ on } \partial Q_1 \cap \partial Q_2, \right. \\
\left. \text{ for all } k \in \mathbb{N} \text{ with } 0 \leq k \leq \min \{m_{Q_1,Q_2}, m - 1\} \right. \\
\left. \text{ for all } Q_1, Q_2 \text{ with } \partial Q_1 \cap \partial Q_2 \neq \emptyset \right\}.
\]

The restriction of \( H^m \) to a given support extension \( \tilde{Q} \) is defined by

\[
H^m(\tilde{Q}) := \left\{ v|_{\tilde{Q}} \mid v \in H^m \right\}.
\]

This space has the property that \( v \in H^l(\bar{K}) \) implies \( v \circ S \in H^l(\tilde{Q}) \). The important detail is, that \( \nabla^k (v \circ S_{Q_1}) = \nabla^k (v \circ S_{Q_2}) \) is only guaranteed for \( k \leq \min \{m_{Q_1,Q_2}, m - 1\} \). Hence the bent Sobolev space is a canonical choice.

As for the spline spaces we get an entire family \( \{H^m_h\}_h \) of bent Sobolev spaces corresponding to the family of meshes \( \{Q_h\}_h \).

**Lemma 5.9 (Properties of Bent Sobolev Spaces):**

The trace of \( \nabla^k \) with \( 0 \leq k \leq \min \{m_{Q_1,Q_2}, m - 1\} \) is well-defined on the common boundary \( \partial Q_1 \cap \partial Q_2 \) of two adjacent elements.
Endowed with the seminorm and norm
\[ |v|_{\mathcal{H}^i}^2 := \sum_{Q \in Q_i} |v|_{H^i(Q)}^2, \]
\[ 0 \leq i \leq m, \text{ and } \|v\|_{\mathcal{H}^m}^2 := \sum_{i=0}^m |v|_{\mathcal{H}^i}^2, \]
the bent Sobolev spaces are well-defined Hilbert spaces.
The restrictions of norm and seminorm to \( \mathcal{H}^m(\tilde{Q}) \) are given by
\[ |v|_{\mathcal{H}^i(\tilde{Q})}^2 := \sum_{Q \in Q_i} |v|_{H^i(Q')}^2 \]
\[ 0 \leq i \leq m, \text{ and } \|v\|_{\mathcal{H}^m(\tilde{Q})}^2 := \sum_{i=0}^m |v|_{\mathcal{H}^i(\tilde{Q})}^2. \]

Proof of Lemma 5.9.
First we prove that the trace is well-defined for functions in \( \mathcal{H}^m \).
Therefore let \( v \in \mathcal{H}^m \). Then, by definition, for two elements \( Q_1, Q_2 \in Q \) with common boundary \( \partial Q_1 \cap \partial Q_2 \) it holds that \( v|_{Q_1} \in \mathcal{H}^m(Q_1) \) and \( v|_{Q_2} \in \mathcal{H}^m(Q_2) \). Hence the trace
Theorem 2.8 ensures that \( \nabla^k (v|_{Q_1}) \) and \( \nabla^k (v|_{Q_2}) \) both exist and are well-defined for
\( 0 \leq k \leq m - 1 \). Finally the equality \( \nabla^k (v|_{Q_1}) = \nabla^k (v|_{Q_2}) \) holds for \( k \leq m_{Q_1,Q_2} \) by definition
of \( m_{Q_1,Q_2} \).
Let us now prove that \( \mathcal{H}^m \) is a Hilbert space and the seminorm and norm are well-defined. The corresponding scalar product for the norm is
\[ \langle \cdot, \cdot \rangle_{\mathcal{H}^m} : \mathcal{H}^m \times \mathcal{H}^m \rightarrow \mathbb{R} \]
\[ \langle v, w \rangle_{\mathcal{H}^m} \mapsto \sum_{i=0}^m \sum_{Q \in Q_i} \langle v, w \rangle_{H^i(Q)}, \]
where \( \langle \cdot, \cdot \rangle_{H^i(Q)} \) denotes the scalar product in \( H^m(Q) \), see Definition 2.5. The properties
of a scalar product are inherited from those of \( H^m \).

Positive definiteness
\[ \langle v, v \rangle_{\mathcal{H}^m} = \sum_{i=0}^m \sum_{Q \in Q_i} \langle v, v \rangle_{H^i(Q)} \geq 0 \]
and \( \langle v, v \rangle_{\mathcal{H}^m} = 0 \iff \langle v, v \rangle_{H^i(Q)} = 0 \text{ for all } Q \in Q_i \text{ for all } i = 0, \ldots, m \)
\[ \iff v|_Q \equiv 0 \in H^i(Q) \text{ for all } Q \in Q_i \text{ for all } i = 0, \ldots, m \]
\[ \iff v \equiv 0 \in \mathcal{H}^m \]

Symmetry
\[ \langle v, w \rangle_{\mathcal{H}^m} = \sum_{i=0}^m \sum_{Q \in Q_i} \langle v, w \rangle_{H^i(Q)} = \sum_{i=0}^m \sum_{Q \in Q_i} \langle w, v \rangle_{H^i(Q)} = \langle w, v \rangle_{\mathcal{H}^m} \]
Linearity

\[
\langle av_1 + bv_2, w \rangle_{H^w} = \sum_{i=0}^{m} \sum_{Q \in Q} \langle av_1 + bv_2, w \rangle_{H(Q)} = \alpha \sum_{i=0}^{m} \sum_{Q \in Q} \langle v_1, w \rangle_{H(Q)} + \beta \sum_{i=0}^{m} \sum_{Q \in Q} \langle v_2, w \rangle_{H(Q)}
\]

\[
\|v\|_{H^w} = \sqrt{\langle v, v \rangle_{H^w}} \text{ is the induced norm and in particular a norm. Similarly it can be proven that } \|v\|_{H^i} \text{ is a seminorm using the properties of } \|v\|_{H(Q)} \text{ and the Hölder inequality Lemma 2.3. It remains to show the completeness of } H^m \text{ with respect to } \|\cdot\|_{H^w}.
\]

Let \(\{v_n\}_{n \in \mathbb{N}} \subset H^m\) be a Cauchy sequence with respect to \(\|\cdot\|_{H^w}\). Then

\[
\|v_n - v_l\|_{H^w} \xrightarrow{n,l \to \infty} 0
\]

\[
\implies \forall \epsilon > 0 \exists n(\epsilon) \text{ s.t. } \|v_n - v_l\|_{H^w} < \epsilon \quad \text{for all } n, l \geq n(\epsilon)
\]

\[
\implies \forall \epsilon > 0 \exists n(\epsilon) \text{ s.t. } \sum_{i=0}^{m} |v_n - v_l|^2_{H^i} < \epsilon \quad \text{for all } n, l \geq n(\epsilon)
\]

\[
\implies \forall \epsilon > 0 \exists n(\epsilon) \text{ s.t. } |v_n - v_l|^2_{H(Q)} < \epsilon \quad \text{for all } n, l \geq n(\epsilon) \text{ and } i = 0, \ldots, m
\]

\[
\implies \forall \epsilon > 0 \exists n(\epsilon) \text{ s.t. } \sum_{Q \in Q} |v_n - v_l|^2_{H(Q)} < \epsilon \quad \text{for all } n, l \geq n(\epsilon) \text{ and } i = 0, \ldots, m
\]

\[
\implies \forall \epsilon > 0 \exists n(\epsilon) \text{ s.t. } |v_n - v_l|^2_{H^w} < \epsilon \quad \text{for all } n, l \geq n(\epsilon), i = 0, \ldots, m, Q \in Q.
\]

Hence \(\{v_n\}_{n \in \mathbb{N}}\) is a Cauchy sequence in \(H^i(Q)\) for all \(i = 0, \ldots, m\) and \(Q \in Q\). Since \(H^i(Q)\) is a Hilbert space, the \(v_n\) converge to an element \(v^Q \in H^i(Q)\) for each \(Q \in Q\). Therefore we define \(v\) by \(v_Q = v^Q\). It follows

\[
|v_n - v_Q|_{H^i(Q)} \to 0 \quad \text{for all } i = 0, \ldots, m \text{ and for all } Q \in Q
\]

\[
\implies \|v_n - v_Q\|_{H^w} \to 0.
\]

It remains to show that \(v \in H^m\). Clearly \(v \in L^2((0,1)^d)\) and \(v_Q \in H^m(Q)\) for all \(Q \in Q\) hold. Additionally,

\[
\left\| V^k(v_Q) - V^k(v_{Q_1}) \right\|_{L^2} \leq \left\| V^k(v_Q) - V^k(v_{Q_1}) \right\|_{L^2} + \left\| V^k(v_{Q_1}) - V^k(v_{Q_2}) \right\|_{L^2} + \left\| V^k(v_{Q_2}) - V^k(v_Q) \right\|_{L^2} \to 0
\]
as for the first and third term \( \| \nabla^k (v_1|Q) - \nabla^k (v_2|Q) \|_{L^2(Q)} = \| v_1|Q - v_2|Q \|_{H^k(Q)} \to 0 \). Furthermore \( v_n \in \mathcal{H}^m \) i.e. the second term is zero for \( k \leq m \). Thus the third constraint for elements of \( \mathcal{H}^m \) is fulfilled by \( v \) as well.

Altogether, \( v \in \mathcal{H}^m \) and \( \mathcal{H}^m \) is a well defined Hilbert space. □

5.2.2 Approximation Error Bounds

This section proves some upper bounds for the error of the approximation with NURBS. Further it derives an a-priori error estimate for IGA. These main results are given in Theorems 5.15, 5.16, and 5.17.

The section starts with some lemmata that are essential for the proofs of the main theorems. The first inequality needed is an extension of the Bramble-Hilbert Lemma 2.15 in classical FEM. It is proven in [BBdVC+06]. The proof below follows [BBdVC+06] but some details are added.

Lemma 5.10:

Let \( k, l \in \mathbb{N} \) with \( 0 \leq k \leq l \leq p + 1 \). Then for any given \( Q \in \mathcal{Q}_h \), any support extension \( \tilde{Q} \) as in Definition 5.3, and any \( v \in \mathcal{H}^l h(\tilde{Q}) \) there exists an \( s \in S_h \) such that

\[
|v - s|_{\mathcal{H}^k h(\tilde{Q})} \leq C h^{-k} |v|_{\mathcal{H}^l h(\tilde{Q})}.
\]

(5.1)

Proof. The proof of Lemma 5.10 follows from step 1 to 6 below.

**Step 1** It is sufficient to show inequality (5.1) for a particular \( \tilde{Q} \).

For a nonempty element \( Q \in \mathcal{Q}_h \) \( Q = \bigotimes_{\alpha=1}^d (\xi^{(\alpha)}_{i_{\alpha}}, \xi^{(\alpha)}_{i_{\alpha}+1}) \), its support extension is given by \( \tilde{Q} = \bigotimes_{\alpha=1}^d (\xi^{(\alpha)}_{i_{\alpha}-p^{(\alpha)}}, \xi^{(\alpha)}_{i_{\alpha}+p^{(\alpha)}+1}) \). Hence there is a maximal number of elements \( Q' \neq Q \in \tilde{Q} \), namely \( \prod_{\alpha=1}^d (2p^{(\alpha)}) - 1 \). This number may be reduced by multiple knots as these result in empty knot spans. Nevertheless the number of possible patterns of elements \( Q' \) in the support extension is finite and only depends on \( p^{(\alpha)} \) and \( d \). Thus it is sufficient to show (5.1) for a particular \( \tilde{Q} \) with a constant not depending on the shape of the \( Q' \)’s.

**Step 2** Pullback of \( \mathcal{H}^m h \) to Hypercubes.

We define a piecewise affine map \( G : \tilde{Q} \to Q \) such that each element \( Q' \) is the image of a hypercube \( G^{-1}(Q') \) with unit edge length. Under this mapping, an internal boundary \( e \) is the image of an internal boundary \( \hat{e} \) in \( \tilde{Q} \). Let \( n_e \) (resp. \( n_{\hat{e}} \)) be unit normal vectors on \( e \) (resp. \( \hat{e} \)). Recall that for a function \( v \in \mathcal{H}^m h(\tilde{Q}) \) the relation

\[
\frac{\partial^j}{\partial n^j_e} (v|Q_1) = \frac{\partial^j}{\partial n^j_{\hat{e}}} (v|Q_2)
\]
holds along $e = \partial Q_1 \cap \partial Q_2$ for $0 \leq i \leq \min\{m_{Q_1, Q_2}, m - 1\}$. Hence for a function $\hat{v} := v \circ G$ the chain rule yields
\[
\frac{\partial}{\partial n_e}(v \circ G)\bigg|_{G^{-1}(Q_1)} = \frac{\partial}{\partial n_e}(v(G(x)))\bigg|_{x \in G^{-1}(Q_1)} = \frac{\partial}{\partial n_e}(v(y))\bigg|_{y \in Q_1} \cdot \frac{\partial}{\partial x}(G(x))\bigg|_{x \in G^{-1}(Q_1)}.
\]
Since $G$ is affine on $G^{-1}(Q_1)$, its derivative is constant. By construction of $G$, its derivative equals the area of $Q_1$, denoted by $A_{Q_1}$. For $i > 1$ Faà di Bruno’s formula, Lemma 2.10, yields
\[
\frac{\partial^i}{\partial n_e^i}(v \circ G)\bigg|_{G^{-1}(Q_1)} = \left(\frac{A_{Q_1}}{A_{Q_2}}\right)^i \frac{\partial^i}{\partial n_e^i}(v \circ G)\bigg|_{G^{-1}(Q_2)} = (c_e)^i \frac{\partial^i}{\partial n_e^i}(v \circ G)\bigg|_{G^{-1}(Q_2)}.
\]
Performing the same calculations for $Q_2$ results in
\[
\frac{\partial^i}{\partial n_e^i}(v \circ G)\bigg|_{G^{-1}(Q_2)} = \left(\frac{A_{Q_1}}{A_{Q_2}}\right)^i \frac{\partial^i}{\partial n_e^i}(v \circ G)\bigg|_{G^{-1}(Q_2)} = (c_e)^i \frac{\partial^i}{\partial n_e^i}(v \circ G)\bigg|_{G^{-1}(Q_2)}.
\]
Since the mesh $Q_\tilde{h}$ is locally quasi-uniform, the ratio $c_e = \frac{A_{Q_1}}{A_{Q_2}}$ is uniformly bounded as well as its reciprocal. Thus all these $c_e$ belong to a compact set.

Assume that the internal boundaries $e$ have an order, then the coefficients $c_e$ form a vector $c$. Depending on $c$ let
\[
\hat{H}_m^c := \{\hat{v} \circ G, v \in \hat{H}_m^c(Q)\}
\]
be the pullback of $\hat{H}_m^c(Q)$ through $G$.

**Step 3** An Inequality for the broken Sobolev space.

In addition to the pullback of the bent Sobolev space, $\hat{H}_m^c$, one can introduce the broken Sobolev space of order $m$ without restrictions on the derivatives at internal boundaries,
\[
\hat{H}_m := \{\hat{v} \circ G, v|_{Q'} \in H^m(Q') \text{ for all } Q' \text{ with } Q' \cap Q \neq \emptyset\}
\]
as done in [ABCM02]. Clearly $\hat{H}_m^c \subset \hat{H}_m$ holds for any vector $c$. 

45
The broken Sobolev space is endowed with the seminorm and norm

\[ |\hat{\mathcal{V}}|_{H^i}^2 := \sum_{Q' \in \mathcal{Q}_h} |\hat{\mathcal{V}}|^2_{H^i(G^{-1}(Q'))}, \quad 0 \leq i \leq m \quad \text{and} \quad ||\hat{\mathcal{V}}||_{H^m}^2 := \sum_{i=0}^{m} |\hat{\mathcal{V}}|_{H^i}^2.\]

Let \( \hat{\mathcal{V}} \in \mathcal{H}^m \) be arbitrary. As the integral mean \( \int_{G^{-1}(Q)} \hat{\mathcal{V}} \, dx \) is constant, the seminorm \( |\int_{G^{-1}(Q)} \hat{\mathcal{V}} \, dx|_{H^i(G^{-1}(Q))} \) is zero for \( m > i \geq 1 \). Hence, due to the triangle inequality and the Poincaré inequality Lemma 2.14, the seminorm of \( \hat{\mathcal{V}} \) satisfies for \( m > i \geq 1 \)

\[ |\hat{\mathcal{V}}|_{H^i(G^{-1}(Q))} = |\hat{\mathcal{V}}|_{H^i(G^{-1}(Q))} - \int_{G^{-1}(Q)} \hat{\mathcal{V}} \, dx|_{H^i(G^{-1}(Q))} \]
\[ \leq \hat{\mathcal{V}} - \int_{G^{-1}(Q)} \hat{\mathcal{V}} \, dx|_{H^i(G^{-1}(Q))} \]
\[ \leq C \cdot H^i(G^{-1}(Q)) = C \cdot |\hat{\mathcal{V}}|_{H^i(G^{-1}(Q))}.\]

Recursive application of this argument for \( m \geq 1 \) leads to

\[ ||\hat{\mathcal{V}}||_{H^m}^2 = \sum_{i=0}^{m} |\hat{\mathcal{V}}|_{H^i}^2 = |\hat{\mathcal{V}}|_{H^m}^2 + \sum_{i=1}^{m-1} |\hat{\mathcal{V}}|_{H^i}^2 + |\hat{\mathcal{V}}|_{H^m}^2 \]
\[ = \sum_{Q' \in \mathcal{Q}_h} |\hat{\mathcal{V}}|_{H^i}^2_{G^{-1}(Q')} + \sum_{i=1}^{m-1} \sum_{Q' \in \mathcal{Q}_h} |\hat{\mathcal{V}}|^2_{H^i(G^{-1}(Q'))} + |\hat{\mathcal{V}}|_{H^m}^2 \]
\[ \leq \sum_{Q' \in \mathcal{Q}_h} |\hat{\mathcal{V}}|_{H^i}^2_{G^{-1}(Q')} + C \cdot \sum_{Q' \in \mathcal{Q}_h} |\hat{\mathcal{V}}|_{H^i(G^{-1}(Q'))} + |\hat{\mathcal{V}}|_{H^m}^2 \]
\[ \leq C \left( ||\hat{\mathcal{V}}||_{L^2(Q)}^2 + |\hat{\mathcal{V}}|_{H^m}^2 \right).\]

The case \( m = 0 \) yields \( ||\hat{\mathcal{V}}||_{H^0}^2 = |\hat{\mathcal{V}}|_{H^0}^2 = ||\hat{\mathcal{V}}||_{L^2(Q)}^2.\)

Hence in general for \( m \geq 0 \)

\[ ||\hat{\mathcal{V}}||_{H^m}^2 \leq C \left( ||\hat{\mathcal{V}}||_{L^2(Q)}^2 + |\hat{\mathcal{V}}|_{H^m}^2 \right). \quad (5.4)\]
**Step 4** Introduction of the piecewise polynomials of degree $\leq l - 1$.

Let $\hat{P}$ be the set of piecewise polynomials of degree $\leq l - 1$ on each element of $\hat{Q}$. Further let $\hat{P}_c := \hat{P} \cap \hat{H}_c$.

Thus every $\hat{v} \in \hat{P}_c$ is piecewise polynomial of degree $\leq l - 1$ and can be represented as $\hat{v} = v \circ G$ with $v \in \mathcal{H}_h^m(\hat{Q})$. Since $G$ is piecewise affine, $v$ must be piecewise polynomial of degree $\leq l - 1$ as well. Furthermore, since $v \in \mathcal{H}_h^m(\hat{Q})$, its derivatives are continuous at internal boundaries $\partial Q_1 \cap \partial Q_2$ up to order $\min\{m, l - 1\}$.

Thus by the Theorem of Curry and Schoenberg, Theorem 4.7, it is representable as a B-Spline, $v \in S_h$.

Hence, since $\hat{v} \in \hat{P}_c$ was chosen arbitrarily, $\hat{P}_c \subset \{ \hat{v} | \hat{v} = v \circ G, v \in S_h \}$.

**Step 5** An inequality for hypercubes.

In this step it is proven, that for a given $\hat{v} \in \hat{H}_c^l$ there exists an $\hat{s} \in \hat{P}_c$ such that

$$\|\hat{v} - \hat{s}\|_{L^2(\hat{Q})} + |\hat{v} - \hat{s}|_{H^l} \leq C \|\hat{v}\|_{H^l}$$

with $C$ independent of $\hat{v}$ and $c$.

To this end let $\hat{\Pi}_c : \hat{H}_c^l \rightarrow \hat{P}_c$ be the $L^2(\hat{Q})$ projection onto the set of piecewise polynomials. For given $\hat{v}$ we choose $\hat{s} := \hat{\Pi}_c \hat{v}$ and prove that (5.5) holds.

As $\hat{\Pi}_c \hat{v}$ is piecewise polynomial of degree $\leq l - 1$ its $l$-th derivative is zero. Hence

$$|\hat{\Pi}_c \hat{v}|_{H^l} = \sum_{Q' \in Q_b, Q' \cap Q \neq \emptyset} |\hat{\Pi}_c \hat{v}|_{H^l(G^{-1}(Q'))} = 0 \text{ for all } \hat{v} \in \hat{H}_c^l \text{ and for all } c.$$ 

Assume now that (5.5) is false. Then there exist sequences $\{c_j\}_{j \in \mathbb{N}}$ and $\{\hat{v}_j\}_{j \in \mathbb{N}} \subset \hat{H}_c^l$ with

$$\|\hat{v}_j - \hat{\Pi}_c \hat{v}_j\|_{L^2(\hat{Q})} = 1 \text{ and } |\hat{v}_j|_{H^l} = \frac{1}{j}.$$ 

The $c_j$ belong to a compact set as seen in step 2. Hence they converge component-by-component to a vector $c_\infty$.

Let $\hat{\eta}_j := \hat{\eta}_j - \hat{\Pi}_c \hat{\eta}_j$. Then (5.4) in combination with the assumption yields the uniform boundedness of $\|\hat{\eta}_j\|_{H^l}$

$$\|\hat{\eta}_j\|_{H^l} \leq C \cdot \|\hat{v}_j - \hat{\Pi}_c \hat{v}_j\|_{L^2(\hat{Q})} + C \cdot |\hat{v}_j|_{H^l(\hat{Q})} \leq C \cdot 1 + C \cdot \frac{1}{j}.$$ 

Thus, since $\hat{H}_c^l$ is compactly embedded in $L^2(\hat{Q})$ according to Theorem 2.7, there
exists a subsequence that converges towards some $\hat{\eta}_\infty$ in $L^2(\hat{Q})$. Without loss of generality the subsequence is denoted by $\hat{\eta}_j$.

Applying (5.4) yields

$$\|\hat{\eta}_j - \hat{\eta}_i\|_{\hat{H}^l(\hat{Q})} \leq C \cdot \|\hat{\eta}_j - \hat{\eta}_i\|_{L^2(\hat{Q})} + C \cdot \left(\|\hat{\eta}_j - \hat{\eta}_i\|_{L^2(\hat{Q})} + \frac{1}{j} + \frac{1}{i} \right),$$

where $i, j \to \infty$.

Hence $\{\hat{\eta}_j\}_{j \in \mathbb{N}}$ is a Cauchy sequence in $\hat{H}^l$. Since $\hat{H}^l$ is complete, $\hat{\eta}_j$ converges to $\hat{\eta}_\infty$ in $\hat{H}^l$ as well. Therefore

$$\|\hat{\eta}_\infty\|_{\hat{H}^l} = \lim_{j \to \infty} \|\hat{\eta}_j\|_{\hat{H}^l} = 0,$$

meaning that the $l$-th derivative of $\hat{\eta}_\infty$ is 0 and thus $\hat{\eta}_\infty \in \hat{P}$.

Since $\hat{\eta}_j \to \hat{\eta}_\infty$ in $\hat{H}^l$ the $i$-th derivative for $i \leq l$ converges as well (in an $L^2$-sense). Hence for two adjacent elements $Q_1$ and $Q_2$ with common boundary $e$ Equality (5.3) passes to the limit:

$$\frac{\partial^i}{\partial n_e^i} (\nu \circ G)\big|_{G^{-1}(Q_1)} = \lim_{j \to \infty} \frac{\partial^i}{\partial n_e^i} (\nu \circ G)\big|_{G^{-1}(Q_1)} = \lim_{j \to \infty} \left(c_{i,j} \right)^i \frac{\partial^i}{\partial n_e^i} (\nu \circ G)\big|_{G^{-1}(Q_2)} = (c_\infty \epsilon) \frac{\partial^i}{\partial n_e^i} (\nu \circ G)\big|_{G^{-1}(Q_2)}.$$

Thus $\hat{\eta}_\infty \in \hat{H}^l_{c_\infty}$ and finally $\hat{\eta}_\infty \in P_{c_\infty}$ which implies $\hat{\eta}_\infty = \hat{\Pi}_{c_\infty} \hat{\eta}_\infty$.

Using this equality, the norm of $\hat{\Pi}_{c_\infty} \hat{\eta}_\infty$ is

$$\|\hat{\Pi}_{c_\infty} \hat{\eta}_\infty\|_{L^2(\hat{Q})} = \|\hat{\eta}_\infty\|_{L^2(\hat{Q})} = \lim_{j \to \infty} \|\hat{\eta}_j\|_{L^2(\hat{Q})} = 1.$$

On the other hand

$$\|\hat{\Pi}_{c_\infty} \hat{\eta}_\infty\|_{L^2(\hat{Q})} \leq \|\hat{\Pi}_{c_\infty} \hat{\eta}_\infty - \hat{\Pi}_{c_j} \hat{\eta}_\infty\|_{L^2(\hat{Q})} + \|\hat{\Pi}_{c_j} \hat{\eta}_\infty - \hat{\Pi}_{c_j} \hat{\eta}_j\|_{L^2(\hat{Q})} + \|\hat{\Pi}_{c_j} \hat{\eta}_j\|_{L^2(\hat{Q})} = I + II + III.$$

Due to the fact that $c_j \to c_\infty$ component-by-component for $j \to \infty$ there exists a
basis of $P_c$ that converges to a basis of $P_{c^\infty}$. Hence the first summand converges to zero.

The second summand converges to zero as a projection is always non-expansive:

$$II = \|\hat{I}_{c_j} \hat{\eta}_{c^\infty} - \hat{I}_{c_j} \hat{\eta}\|_{L^2(Q)} \leq \|\hat{\eta}_{c^\infty} - \hat{\eta}\|_{L^2(Q)} \rightarrow 0 \text{ for } j \rightarrow \infty.$$

Finally the third summand is zero by definition. The combination of $I$, $II$, and $III$ leads to

$$\|\hat{I}_{c^\infty} \hat{\eta}_{c^\infty}\|_{L^2(Q)} \rightarrow 0.$$

As this contradicts $\|\hat{I}_{c^\infty} \hat{\eta}_{c^\infty}\|_{L^2(Q)} = 1$ the assumption must be false and (5.5) is proven.

**Step 6** Conclusion of the proof.

Let $v \in H^1_h$, $\hat{\eta} := v \circ G$. Due to (5.5) there exists $\hat{s} \in \hat{P}_c$ such that

$$\|\hat{\eta} - \hat{s}\|_{L^2(Q)} + |\hat{\eta} - \hat{s}|_{H^{l}(\hat{Q}')} \leq C |\hat{\eta}|_{H^{l}(\hat{Q}')}.$$

The inclusion $\hat{P}_c \subset \{\hat{\eta} = v \circ G, v \in S_h\}$ yields $s \in S_h$ with $s \circ G = \hat{s}$. The classical Bramble-Hilbert Lemma 2.15 for $|v - s|_{H^{l}(Q')}^{2}$ leads to

$$|v - s|_{H^{l}(Q')}^{2} = \sum_{Q' \in \mathcal{Q}_h} |v - s|_{H^{l}(Q')}^{2}$$

$$\leq \sum_{Q' \in \mathcal{Q}_h} |v - s|_{H^{l}(Q')}^{2}$$

$$\leq C \cdot \sum_{Q' \in \mathcal{Q}_h} \sum_{Q' \cap Q \neq \emptyset} h^{2(l-k)}_{Q'} |v - s|_{H^{l}(Q')}^{2}$$

$$\leq C \cdot \sum_{Q' \in \mathcal{Q}_h} \sum_{Q' \cap Q \neq \emptyset} h^{2(l-k)}_{Q'} |v - s|_{H^{l}(Q')}^{2}$$

$$= C \cdot \sum_{i=0}^{1} \sum_{Q' \in \mathcal{Q}_h} \sum_{Q' \cap Q \neq \emptyset} h^{2(l-k)}_{Q'} |v - s|_{H^{l}(Q')}^{2}$$

The inequality $h_{Q'} \leq C \cdot h_{Q}$ follows from the local quasi-uniformity of the mesh since $Q$ consists only of a finite number of elements which only depends on the
dimension $d$ and the degree $p^{(\alpha)}$. Hence
\[ |v - s|_{H^k_\alpha(Q)} \leq C \cdot h_l^{-k} \|v - s\|_{H^k_\alpha(Q)}. \]

The function $G$ is piecewise affine, hence its derivative is piecewise constant. Since only a finite number of $Q'$ exist, an upper bound for the determinants of $DG$ and $DG^{-1}$ can easily be found. This fact leads to
\[
\|v - s\|^2_{H^k_\alpha(Q)} = \sum_{i=0}^l \sum_{Q' \subseteq Q} \sum_{Q' \cap Q = \emptyset} |\nabla^\alpha (v - s)|^2 \int_{Q'} dx
\]
\[
= \sum_{i=0}^l \sum_{Q' \subseteq Q} \sum_{Q' \cap Q = \emptyset} \int_{Q'} |\nabla^\alpha (\tilde{\theta} - \tilde{s})|^2 \cdot |\det (DG^{-1})| dy
\]
\[
\leq C \cdot \left\| \det (DG^{-1}) \right\|_{L^\infty(\mathcal{K})} \cdot \sum_{i=0}^l \sum_{Q' \subseteq Q} \sum_{Q' \cap Q = \emptyset} \|\tilde{\theta} - \tilde{s}\|^2_{H^k_\alpha(G^{-1}(Q'))}
\]
\[
= C \cdot \left\| \det (DG^{-1}) \right\|_{L^\infty(\mathcal{K})} \|\tilde{\theta} - \tilde{s}\|^2_{H^k_\alpha}. \]

Finally the application of (5.4) and (5.5) together with the same argument as above with $G^{-1}$ taking the role of $G$ yields
\[
\|\tilde{\theta} - \tilde{s}\|^2_{H^k_\alpha} \leq C \left( \|\tilde{\theta} - \tilde{s}\|_{L^2(Q)} + \|\tilde{\theta} - \tilde{s}\|^2_{H^k_\alpha} \right) \leq C \cdot \|\tilde{\theta}\|^2_{H^k_\alpha} \leq C \cdot \|\det (DG)\|_{L^\infty(\mathcal{Q})} \|v\|^2_{H^k_\alpha(Q)}. \]

Putting the inequalities together leads to
\[
|v - s|_{H^k_\alpha(Q)} \leq C \left\| \det (DG^{-1}) \right\|_{L^\infty(\mathcal{K})} \left\| \det (DG) \right\|_{L^\infty(\mathcal{Q})} h_l^{-k} \|v\|_{H^k_\alpha(Q)}
\]
\[
\leq C \cdot h_l^{-k} \|v\|_{H^k_\alpha(Q)}. \]

\[\square\]
The next step is to verify that (5.1) holds for a projection of \(v\) onto the spline space \(S_h\). Therefore we use the projector defined in [Sch07] by

\[
\Pi_{S_h} : L^2 \left( (0, 1)^d \right) \rightarrow S_h \\
v \mapsto \sum_{i_1=1,\ldots,i_d=1}^{n(1)\ldots n(d)} (\lambda_{i_1\ldots i_d} v) N_{i_1\ldots i_d}.
\]

The \(\lambda_{i_1\ldots i_d}\) denote dual basis functionals, hence

\[
\lambda_{j_1\ldots j_d} N_{i_1\ldots i_d} = \begin{cases} 
1, & \text{if } j_\alpha = i_\alpha, 1 \leq \alpha \leq d, \\
0, & \text{otherwise.}
\end{cases}
\]

Refer to [Sch07] for an explicit definition.

**Lemma 5.11 (Properties of the Projector \(\Pi_{S_h}\)):**

For the projector defined in (5.6) it holds

\[
\Pi_{S_h} s = s \text{ for all } s \in S_h \quad \text{(spline preserving)}, \\
\left\| \Pi_{S_h} v \right\|_{L^2(Q)} \leq C \|v\|_{L^2(Q)} \text{ for all } v \in L^2 \left( (0, 1)^d \right) \text{ and } Q \in Q_h \quad \text{(stability)}.
\]

**Proof.**

The first property follows from the dual basis character of the \(\lambda_{i_1\ldots i_d}\). For an arbitrary spline \(s = \sum_{i_1=1,\ldots,i_d=1}^{n(1)\ldots n(d)} N_{i_1\ldots i_d} P_{i_1\ldots i_d} \in S_h\) it holds

\[
\Pi_{S_h} s = \Pi_{S_h} \left( \sum_{i_1=1,\ldots,i_d=1}^{n(1)\ldots n(d)} N_{i_1\ldots i_d} P_{i_1\ldots i_d} \right) \\
= \sum_{i_1=1,\ldots,i_d=1}^{n(1)\ldots n(d)} \left( \sum_{j_1=1,\ldots,j_d=1}^{n(1)\ldots n(d)} \lambda_{i_1\ldots i_d} N_{j_1\ldots j_d} P_{j_1\ldots j_d} \right) N_{i_1\ldots i_d} \\
= \sum_{i_1=1,\ldots,i_d=1}^{n(1)\ldots n(d)} \left( \sum_{j_1=1,\ldots,j_d=1}^{n(1)\ldots n(d)} (\lambda_{i_1\ldots i_d} N_{j_1\ldots j_d} P_{j_1\ldots j_d}) \right) N_{i_1\ldots i_d} \\
= \sum_{i_1=1,\ldots,i_d=1}^{n(1)\ldots n(d)} P_{i_1\ldots i_d} N_{i_1\ldots i_d} \\
= s.
\]

The proof of the stability of the projector requires knowledge of the exact definition of the \(\lambda_{i_1\ldots i_d}\) and is therefore omitted in this thesis. It can be found in [Sch07].

□

This projector yields an estimate for the approximation error with B-splines on some \(Q\).
Lemma 5.12 ([BBdVC+06]):
Let $\Pi_{S_h} : L^2 \left( (0,1)^d \right) \rightarrow S_h$ be a projector satisfying the conditions of Lemma 5.11. Further let $k,l \in \mathbb{N}$ satisfy $0 \leq k \leq l \leq p + 1$. Then for all $Q \in Q_h$ it holds

$$\left| v - \Pi_{S_h} v \right|_{H^k(Q)} \leq C h^{l-k} \left| v \right|_{H^l(Q)} \text{ for all } v \in H^l \left( \tilde{Q} \right) \cap L^2 \left( (0,1)^d \right).$$

Proof.
Let $s \in S_h$ be chosen as in Lemma 5.10. Hence the spline preserving property of the projection $\Pi_{S_h}$ and the triangle inequality lead to

$$\left| v - \Pi_{S_h} v \right|_{H^k(Q)} = \left| v - s - \Pi_{S_h} (v - s) \right|_{H^k(Q)} \leq \left| v - s \right|_{H^k(Q)} + \left| \Pi_{S_h} (v - s) \right|_{H^k(Q)} = I + II.$$

By Lemma 5.10 the first summand can be estimated by

$$I = \left| v - s \right|_{H^k(Q)} \leq \sum_{Q' \in Q_h \cap \tilde{Q} \neq \emptyset} \left| v - s \right|_{H^k(Q')} = \left| v - s \right|_{H^k(Q)} \leq C h^{l-k} \left| v \right|_{H^l(Q)}.$$

Since $\Pi_{S_h} (v - s) \in S_h$ it is a polynomial on $Q$. Therefore the inverse inequality Lemma 2.16 for the second summand yields

$$II = \left| \Pi_{S_h} (v - s) \right|_{H^k(Q)} \leq C h^{l-k} \left\| \Pi_{S_h} (v - s) \right\|_{L^2(Q)}.$$

Finally the stability of the projector and Lemma 5.10 once again lead to

$$C h^{l-k} \left\| \Pi_{S_h} (v - s) \right\|_{L^2(Q)} \leq C h^{l-k} \left\| v - s \right\|_{L^2(Q)} \leq C h^{l-k} \left\| v \right\|_{H^0(Q)} \leq C h^{l-k} \cdot h^{l-0} \left| v \right|_{H^l(Q)} = C h^{l-k} \left| v \right|_{H^l(Q)}.$$

The summation of $I$ and $II$ concludes the proof. □

The next step is to find a generalization of Lemma 5.10 to the approximation with NURBS. Therefore a new projector is defined by

$$\Pi_{N_h} : L^2 \left( (0,1)^d \right) \rightarrow N_h$$

$$v \mapsto \frac{\Pi_{S_h} \left( \omega v \right)}{\omega} \text{ for all } v \in L^2 \left( (0,1)^d \right),$$

with weighting function $\omega$ as in Definition 5.2.
Lemma 5.13 ([BBdVC+06]):

Let \(k, l \in \mathbb{N}\) satisfy \(0 \leq k \leq l \leq p + 1\), then

\[
||v - \Pi_{\mathcal{N}_h}v||_{H^k(Q)} \leq C \sum_{Q' \in \mathcal{Q}_h} ||v||_{H^{k-1}(Q')} \text{ for all } v \in \mathcal{H}_h^l \text{ and } Q \in \mathcal{Q}_h.
\]

**Proof.**

\(S_h\) is a refinement developed from \(S_{h_0}\) by knot insertion. Thus a function \(s \in S_{h_0}\) remains parametrically and geometrically unchanged during this process (see Section 4.3.1). Therefore \(s \in S_h\) and since \(s\) was chosen arbitrarily \(S_{h_0} \subseteq S_h\). By definition of the weighting function \(\omega \in S_{h_0} \subseteq S_h\), hence \(\omega\) is piecewise polynomial and \(\nabla^k \omega|_{Q_1} = \nabla^k \omega|_{Q_2}\) for \(k \leq m_{Q_1, Q_2}\). Together with \(v \in \mathcal{H}_h^l(\tilde{Q})\) this leads to

\[
\omega \cdot v \in L^2((0, 1)^l),
\]

\[
\omega \cdot v \in H^l(Q) \text{ for all } Q \in \tilde{Q},
\]

\[
\nabla^k (\omega \cdot v)|_{Q_1} = \sum_{i=0}^{k} \binom{k}{i} \nabla^k-i \omega|_{Q_1} \cdot v|_{Q_1} = \sum_{i=0}^{k} \binom{k}{i} \nabla^k-i \omega|_{Q_2} \cdot v|_{Q_2} = \nabla^k (\omega \cdot v)|_{Q_2}.
\]

Hence \(\omega \cdot v \in \mathcal{H}_h^l(\tilde{Q})\). The definition of \(\Pi_{\mathcal{N}_h}\) leads to

\[
||v - \Pi_{\mathcal{N}_h}v||_{H^k(Q)} = \left| \frac{1}{\omega} \nabla^k (\omega v - \Pi_{\mathcal{N}_h} \omega v)|_{H^k(Q)} \right| = \sum_{i=0}^{k} \binom{k}{i} \left| \nabla^i \frac{1}{\omega} \cdot \nabla^k-i (\omega v - \Pi_{\mathcal{N}_h} \omega v)|_{L^2(Q)} \right| \leq C \sum_{i=0}^{k} \left| \nabla^i \omega|_{W^{l-1}(Q)} \right| ||v - \Pi_{\mathcal{N}_h} \omega v||_{H^{k-i}(Q)}.
\]

(5.7)

Lemma 5.12 applied with the indices \((k - i)\) and \((l - i)\) leads to

\[
||\omega v - \Pi_{\mathcal{N}_h} \omega v||_{H^{k-i}(Q)} \leq C \cdot \nabla^k \omega|_{H^{k-i}(Q)} \sum_{Q' \in \mathcal{Q}_h} ||v||_{H^{k-1}(Q')} \sum_{Q' \in \mathcal{Q}_h} \left| \nabla^i \omega|_{W^{l-1}(Q')} \right| ||v||_{H^{k-i}(Q')}.
\]

\[
(5.8)
\]

53
This can be inserted into (5.8) yielding

\[ |v - \Pi_{N_k} v|_{H^l(Q)} \leq C \cdot \| h_l^{|\cdot - k} \sum_{i=0}^{k} \sum_{j=0}^{l-i} \sum_{\substack{Q' \in Q_h \cap Q \neq \emptyset}} \| \omega \|_{W^{m+\omega}(Q')} |v|_{H^{|v|}(Q')} . \]

Now we change C to the constant C4 which depends on \( \| \omega \|_{W^{m+\omega}(Q')} \). Furthermore the summation over \( i \) and \( j \) is combined. Since \( l - (i + j) \) runs from 0 to \( l \) the inequality is simplified to

\[ |v - \Pi_{N_k} v|_{H^l(Q)} \leq C_4 h_l^{|\cdot - k} \sum_{i=0}^{k} \sum_{j=0}^{l-i} \sum_{\substack{Q' \in Q_h \cap Q \neq \emptyset}} |v|_{H^{|v|}(Q')} \]

\[ \leq C_4 h_l^{|\cdot - k} \sum_{i=0}^{k} \sum_{Q' \in Q_h \cap Q \neq \emptyset} |v|_{H^{l}(Q')} = C_4 h_l^{|\cdot - k} \| v \|_{H^l(Q')} . \]

\( \square \)

Until now the estimates describe bounds for the approximation with NURBS on the parametric domain. On the other hand the PDE is solved on the physical domain. Thus, following [BBdVC+06], an estimate for the transformation from parametric to physical domain is derived.

**Lemma 5.14 ([BBdVC+06]):**

Let \( m \in \mathbb{N}, Q \in Q_h \), and \( K = S(Q) \), then for all functions \( v \in H^m(K) \) it holds that

\[ |v \circ S|_{H^m(Q)} \leq C_s \| \det V S^{-1} \|_{L^\infty(K)}^{1/2} \sum_{j=0}^{m} \| V S \|_{L^\infty(Q)}^{j} |v|_{H^j(K)} , \quad (5.9) \]

\[ |v|_{H^m(K)} \leq C_s \| \det V S \|_{L^\infty(Q)}^{1/2} \sum_{j=0}^{m} |v \circ S|_{H^j(Q)} . \quad (5.10) \]

**Proof.**

Let \( m = 0 \), then an integral transformation from \( Q \) to \( K = S(Q) \) yields

\[ |v \circ S|_{H^0(Q)} = |v \circ S|_{L^2(Q)} = \left( \int_Q (v \circ S)^2 \, dx \right)^{1/2} \]

\[ = \left( \int_{S(Q)} v^2 \left| \det V S^{-1} \right| \, dy \right)^{1/2} \]

\[ \leq \| \det V S^{-1} \|_{L^\infty(K)}^{1/2} \| v \|_{L^2(Q)} = \| \det V S^{-1} \|_{L^\infty(K)}^{1/2} \| v \|_{H^0(K)} . \]

Analogously an integral transformation from \( K \) to \( Q = S^{-1}(K) \) yields (5.10).

Let us now consider the more interesting case \( m \geq 1 \). Following the proof in [BBdVC+06],
the auxiliary function

\[
\tilde{S} : Q \to K
\]

\[
\xi \mapsto \frac{F(\xi)}{\|\nabla S\|_{L^\infty(Q)}}
\]
is defined. Differentiating this expression \(k\)-times with \(1 \leq k \leq m\) yields

\[
\|\nabla^k \tilde{S}\|_{L^\infty(Q)} \leq \|\nabla S\|_{L^\infty(Q)} \|\nabla^k \tilde{S}\|_{L^\infty(Q)}.
\]  (5.11)

In addition it holds

\[
\|\nabla \tilde{S}\|_{W^{m,\infty}(Q)} = \frac{\|\nabla S\|_{W^{m,\infty}(Q)}}{\|\nabla S\|_{L^\infty(Q)}} \leq \|\nabla S\|_{W^{m,\infty}(Q)} \inf_{x \in K} \|\nabla S^{-1}(x)\| \leq C_s.
\]  (5.12)

In the following, Lemma 2.12 is applied. Youngs inequality Lemma 2.4 yields that Lemma 2.12 holds for squared norms with a slightly different constant \(C\). Followed by the H"older inequality Lemma 2.3 and a change of variables this leads to

\[
|v \circ S|^2_{H^m(Q)} = \int_Q |\nabla^m (v \circ S)(\xi)|^2 d\xi
\]

\[
\leq C \int_Q \sum_{j=1}^m |\nabla^j (S(\xi))|^2 \sum_{i \in I(j, m)} |\nabla S(\xi)|^{2i_1} \cdots |\nabla^m S(\xi)|^{2i_m} d\xi
\]

\[
\leq C \sum_{j=1}^m \int_Q |\nabla^j (S(\xi))|^2 d\xi \sum_{i \in I(j, m)} \|\nabla S(\xi)\|_{L^\infty(Q)}^{2i_1} \cdots \|\nabla^m S(\xi)\|_{L^\infty(Q)}^{2i_m}
\]

\[
\leq C \|\det \nabla S^{-1}\|_{L^\infty(K)} \sum_{j=1}^m \int_K |\nabla^j (v)(x)|^2 dx
\]

\[
\times \sum_{i \in I(j, m)} \|\nabla S(\xi)\|_{L^\infty(Q)}^{2i_1} \cdots \|\nabla^m S(\xi)\|_{L^\infty(Q)}^{2i_m}
\]

\[
\leq C \|\det \nabla S^{-1}\|_{L^\infty(K)} \sum_{j=1}^m |\alpha|^2_{H^j(K)}
\]

\[
\times \sum_{i \in I(j, m)} \|\nabla S(\xi)\|_{L^\infty(Q)}^{2i_1} \cdots \|\nabla^m S(\xi)\|_{L^\infty(Q)}^{2i_m}.
\]
Taking the square root and the application of (5.11) lead to

\[
|v \circ S|_{L^2(Q)} \leq C \| \det \nabla S^{-1} \|_{L^2(K)}^{1/2} \sum_{j=1}^m \|v\|_{H^j(K)} \times \sum_{i \in I(j,m)} \|\nabla S\|_{L^2(Q)}^{i_1+i_2+\cdots+i_m} \|\nabla \tilde{S}(\xi)\|_{L^2(Q)}^{i_1} \cdots \|\nabla^m \tilde{S}(\xi)\|_{L^2(Q)}^{i_m}.
\]

Recall that

\[
I(j,m) := \{ i = (i_1, i_2, \ldots, i_m) \in \mathbb{N} | i_1 + i_2 + \cdots + i_m = j, i_1 + 2i_2 + \cdots + mi_m = m \}.
\]

Thus (5.12) finally leads to

\[
|v \circ S|_{L^2(Q)} \leq C_s \| \det \nabla S^{-1} \|_{L^2(K)}^{1/2} \sum_{j=1}^m \|v\|_{H^j(K)} \|\nabla S\|_{L^2(Q)}^j,
\]

thus concluding the proof of (5.9).

The proof of (5.10) follows analogously considering \( \tilde{S}^{-1} \) instead of \( \tilde{S} \). The details can be found in [BBdVC+06].

The final step in this section combines the approximation estimate on the parametric domain from Lemma 5.13 and the estimate for the change of variables from the parametric to the physical domain from Lemma 5.14. To this end we define a projector \( \Pi_{V_h} \) to the space of NURBS on the physical domain as a push-forward of the NURBS projector \( \Pi_{N_h}(v \circ S) \):

\[
\Pi_{V_h} : L^2(\Omega) \to V_h
\]

\[
v \mapsto \left( \Pi_{N_h}(v \circ S) \right) \circ S^{-1}.
\]

This projector combined with Lemma 5.14 lead to a local and a global error estimate for isogeometric analysis.

**Theorem 5.15** (Local Approximation Error Estimate [BBdVC+06]):

*Let \( k, l \in \mathbb{N} \) satisfy \( 0 \leq k \leq l \leq p + 1 \), \( q \in \mathbb{Q}_h \), \( K = S(Q) \), and \( \tilde{Q}, \tilde{K} \) as in Definition 5.3 and Definition 5.4. Then for all \( v \in L^2(\Omega) \cap H^l(\tilde{K}) \) it holds that

\[
\|v - \Pi_{V_h} v\|_{H^l(K)} \leq C_e h_K^{l-k} \sum_{i=0}^l \|\nabla^i S\|_{L^2(Q)} \|v\|_{H^l(\tilde{K})}.
\]

(5.13)

Here \( h_K := \|\nabla S\|_{L^2(Q)} h_Q \) denotes the element size in the physical domain.*

56
**Theorem 5.16** (Global Approximation Error Estimate [BBdVC+06]):

Let \( k, l \in \mathbb{N} \) be defined as in Theorem 5.15. Then, for all \( v \in H^l(\Omega) \), it holds

\[
\sum_{k \in K_0} \left| v - \Pi_{V_k} v \right|^2_{H^2(K)} \leq C_s \sum_{k \in K_0} h_k^{2(l-k)} \sum_{i=0}^l \|\nabla v\|^2_{L^\infty(S^{-1}(K))} \|v\|^2_{H^2(K)}.
\]

**Proof of Theorem 5.15.**

The definition of the projector followed by (5.10) leads to

\[
\left| v - \Pi_{V_k} v \right|_{H^2(K)} = \left| v - \left( \Pi_{N_k} (v \circ S) \right) \circ S^{-1} \right|_{H^2(K)}
\]

\[
\leq C_s \|\det \nabla v\|^2_{L^\infty(Q)} \|\nabla v\|^2_{L^\infty(Q)} \sum_{i=0}^k \left| v - \left( \Pi_{N_k} (v \circ S) \right) \circ S \right|_{H^2(Q)}
\]

\[
= C_s \|\det \nabla v\|^2_{L^\infty(Q)} \|\nabla v\|^2_{L^\infty(Q)} \sum_{i=0}^k \left| v \circ S - \Pi_{N_k} (v \circ S) \right|_{H^2(Q)}.
\]

(5.14)

Since \( v \in H^l(\hat{K}) \) and \( v(K) = v(S(Q)) = v \circ S(Q) \) it holds \( v \circ S \in H^l(\hat{Q}) \) for all \( Q \in \hat{Q} \).

Further \( v \circ S \in L^2(\hat{Q}) \). Finally since by definition

\[
V^k(v)|_{S(Q_1)} = V^k(v)|_{K_1} = V^k(v)|_{K_2} = V^k(v)|_{S(Q_2)} \text{ and }
\]

\[
V^k(S)|_{Q_1} = V^k(S)|_{Q_2} \text{ for } k \leq \min\{m_{Q_1,Q_2},l-1\},
\]

the generalized chain rule, Lemma 2.10, yields

\[
V^k(v \circ S)|_{Q_1} = V^k(v \circ S)|_{Q_2} \text{ for } k \leq \min\{m_{Q_1,Q_2},l-1\}.
\]

Hence \( v \circ S \in H^l(\hat{Q}) \).

Lemma 5.13 applied with the indices \( i \) and \( l+i-k \), that satisfy \( 0 \leq i \leq l+i-k \leq p+1 \), yields for every \( i = 0,\ldots,k \)

\[
\left| v \circ S - \Pi_{N_k} (v \circ S) \right|_{H^2(Q)} \leq C_s h_{Q_1}^{l-k} \|v \circ S\|_{H^{l+i-k}(Q)}
\]

\[
\leq C_s h_{Q_1}^{l-k} \sum_{j=0}^{l+i-k} |v \circ S|_{H^j(\hat{Q})}.
\]

As in the proof of Lemma 5.13 we can combine the two sums over \( i = 0,\ldots,k \) and
As a corollary of these interpolation error estimates an a-priori estimate for the Poisson problem follows from squaring (5.13) and the summation over all $K$

The global error estimate is an immediate consequence of the local error estimate. It follows from squaring (5.13) and the summation over all $K \in \mathcal{K}_h$. \hfill\Box

As a corollary of these interpolation error estimates an a-priori estimate for the Poisson problem can be found.
Theorem 5.17 (A-Priori Error Estimate):
Let \( u \in H^l(K) \cap L^2(\Omega) \) with \( 1 \leq l \leq p + 1 \) be the solution of (3.3) and \( u_h \) the solution of (3.5). Then it holds for the energy norm in \( K \) that

\[
\| u - u_h \|_K \leq C \| \nabla S \|_{L^\infty(\Omega)} \| u \|_{H^l(K)}.
\]

Proof.
The proof of Theorem 5.17 uses the so-called Galerkin orthogonality:

\[
a(u - u_h, v) = a(u, v) - a(u_h, v) = I(v) - I(v) = 0 \quad \text{for all } v \in V_h.
\]

As the discrete approximation \( u_h \in V_h = V_h \) of the solution \( u \) and the projection \( \Pi_{V_h} u \in V_h \), Galerkin orthogonality yields

\[
\| u - u_h \|_K^2 = a(u - u_h, u - u_h) = a(u - u_h, u) = a(u - u_h, u - \Pi_{V_h} u) \leq \| u - u_h \| \| u - \Pi_{V_h} u \|.
\]

Since this general statement likewise holds for the restriction to \( K \) and in the case of the Poisson problem it holds \( \| u - \Pi_{V_h} u \|_{H^1(K)} = \| u - \Pi_{V_h} u \|_{H^1(K)} \), a division by \( \| u - u_h \| \) and insertion of (5.13) finish the proof. \( \Box \)

5.2.3 Inverse Inequality

Theorem 5.18 (Inverse Inequality for NURBS [BBdVC'+06]):
It holds

\[
|v|_{H^l(K)} \leq C_l h^{-1} \| v \|_{H^l(K)} \quad \text{for all } K \in \mathcal{K}_h \text{ and for all } v \in V_h.
\]

Proof.
Let \( v \in V_h \) be arbitrary. Obviously \( v \in H^2(K) \), hence Lemma 5.14 yields

\[
|v|_{H^l(K)} \leq C_l \| \nabla S \|_{L^\infty(Q)}^{1/2} \| \nabla S \|_{L^\infty(Q)}^{1/2} \sum_{j=0}^2 |v \circ S|_{H^j(Q)}
\]

\[
= C_l \| \nabla S \|_{L^\infty(Q)}^{1/2} \| \nabla S \|_{L^\infty(Q)}^{1/2} |v \circ S|_{H^l(Q)}.
\]

(5.17)

As \( v \in V_h \) it holds \( v \circ S \in N_h \) and thus the product with the weighting function \( \omega \) \( v \circ S \)
is polynomial. Therefore the inverse inequality for polynomials Lemma 2.16 leads to

\[ \|v \circ S\|_{H^2(Q)} \leq C \left\| \frac{1}{\omega} \right\|_{W^{2,\infty}(Q)} \|\omega (v \circ S)\|_{H^2(Q)} \]

\[ \leq C h^{-1}_Q \left\| \frac{1}{\omega} \right\|_{W^{2,\infty}(Q)} \|\omega (v \circ S)\|_{H^1(Q)}. \] (5.18)

A further application of Lemma 5.14 yields

\[ \|\omega (v \circ S)\|_{H^1(Q)} \leq C \|\omega\|_{W^{1,\infty}(Q)} \|v \circ S\|_{H^1(Q)} \]

\[ \leq C_s \|\omega\|_{W^{1,\infty}(Q)} \left\| \det VS^{-1} \right\|_{L^{\infty}(K)}^{1/2} \sum_{j=0}^1 \|\nabla S\|_{L^{\infty}(Q)} \|v\|_{H^j(K)}. \] (5.19)

Combining (5.17), (5.18), and (5.19) leads to

\[ |v|_{H^2(K)} \leq C_j h^{-1}_Q \left\| \frac{\det VS}{L^{\infty}(Q)} \right\| \left\| \frac{\nabla S}{L^{\infty}(Q)} \right\| \left\| \frac{1}{\omega} \right\|_{W^{2,\infty}(Q)} \|\omega\|_{W^{1,\infty}(Q)} \]

\[ \times \left\| \det VS^{-1} \right\|_{L^{\infty}(K)}^{1/2} \sum_{j=0}^1 \|\nabla S\|_{L^{\infty}(Q)} \|v\|_{H^j(K)} \]

\[ \leq C_j h^{-1}_Q \sum_{j=0}^1 \|\nabla S\|_{L^{\infty}(Q)} \|v\|_{H^j(K)}. \]

Finally let \( v_K = \int_K v \, dx \) denote the constant integral mean of \( v \) on \( K \). Hence \( v_K \in V_h \) and \( |v_K|_{H^1(K)} = |v_K|_{H^1(K)} = 0 \). Hence the Poincaré inequality Lemma 2.14 together with \( h_K = \|\nabla S\|_{L^{\infty}(Q)} h_Q \) yield

\[ |v|_{H^2(K)} = |v - v_K|_{H^2(K)} \]

\[ \leq C_j h^{-1}_Q \sum_{j=0}^1 \|\nabla S\|_{L^{\infty}(Q)} \|v - v_K|_{H^j(K)} \]

\[ \leq C_j h^{-1}_K \|v - v_K|_{H^1(K)} \]

\[ = C_j h^{-1}_K |v|_{H^1(K)}. \]

\[ \square \]

Remark 5.19: A generalization of Theorem 5.18 for higher order derivatives can be found in [BBdVC+06].
6 Application to Convection-Diffusion Equations

This chapter gives a first example of the application of isogeometric analysis. First the convection-diffusion equation is introduced and its explicit numerical representation with NURBS is derived. Afterwards the SUPG stabilization method is explained. Finally three model problems are described.

6.1 Convection-Diffusion Equation

Definition 6.1 (Convection-Diffusion Equation):
A sufficiently smooth function $u$ is called solution to the convection-diffusion problem on a bounded Lipschitz-domain $\Omega$ with boundary $\partial\Omega = \Gamma_N \cup \Gamma_D$ if it solves

$$
\frac{\partial u}{\partial t} - \nabla \cdot (E \nabla u) + \nabla \cdot (bu) + ru = f \quad \text{in the domain } \Omega,
$$

$$
u = u_D \quad \text{on the Dirichlet boundary } \Gamma_D,
$$

$$
n \cdot E \· \nabla u = g \quad \text{on the Neumann boundary } \Gamma_N.
$$

Here $E(x) : \Omega \to \mathbb{R}^{2 \times 2}$ denotes the diffusion coefficient, $b(x) : \Omega \to \mathbb{R}^2$ the convection field, and $r : \Omega \to \mathbb{R}$ is the reaction coefficient. The RHS $f : \Omega \to \mathbb{R}$ takes account of sources and sinks. The functions $u_D$ and $g$ describe the boundary conditions.

In the following further simplifications are made. The situation is assumed to be stationary, i.e. $\partial u/\partial t = 0$. In addition only scalar equations are considered. That means

$$
E(x) = \begin{pmatrix} \varepsilon & 0 \\ 0 & \varepsilon \end{pmatrix}
$$

with $\varepsilon > 0$. The Neumann boundary condition is set to be $g \equiv 0$. Finally we will only deal with incompressible flows $\nabla \cdot b = 0$ and omit the reaction term, that is $r \equiv 0$. Hence the resulting model problem is given by

$$
-\varepsilon \Delta u + b \· \nabla u = f \quad \text{in } \Omega,
$$

$$
u = u_D \quad \text{on } \Gamma_D,
$$

$$
\varepsilon \frac{\partial u}{\partial n} = 0 \quad \text{on } \Gamma_N. \tag{6.1}
$$

If $u_D \equiv 0$, the discrete solution to this problem $u_h \in V_h$ satisfies

$$
a(u_h, v_h) = l(v_h) \text{ for all } v_h \in V_h, \tag{6.2}
$$

where $V_h$ is a finite dimensional Hilbert space.
The bilinear form \( a \) and linear form \( l \) corresponding to (6.1) are given by

\[
a(u_h, v_h) = \varepsilon \int_{\Omega} \nabla u_h \cdot \nabla v_h \, dx + \int_{\Omega} (b \cdot \nabla u_h) v_h \, dx
\]

and

\[
l(v_h) = \int_{\Omega} f \cdot v_h \, dx + \int_{\Gamma_N} g \cdot n \cdot v_h \, ds = \int_{\Omega} f \cdot v_h \, dx.
\]

Hence in matrix notation the problem becomes

Find \( c \in \mathbb{R}^n \) such that

\[
A \cdot c = b \tag{6.3}
\]

where

\[
A = (a_{ij})_{i,j=0}^{n,n} = a(\Psi_j, \Psi_i) \tag{6.4}
\]

and

\[
b = (b_i)_{i=0}^n = l(\Psi_i),
\]

where the functions \( \{\Psi_i\}_{i=0}^n \) denote a basis of \( V_h \).

Taking Dirichlet boundary conditions into account an extended solution space is necessary. Therefore let

\[
V_h^D := \left\{ v_h \mid v_h|_{\Gamma_D} = u_D, v_h = \sum_{j=1}^{n} w^j v_j \right\}.
\]

Further let \( w_h \in V_h^D \) denote an arbitrary function satisfying the boundary condition. Thus a solution \( \tilde{u}_h \) of (6.1) can be decomposed as \( \tilde{u}_h = u_h + w_h \) with \( u_h \in V_h \). Hence \( a(\tilde{u}_h, v_h) = a(u_h + w_h, v_h) = l(v_h) \) implies \( a(u_h, v_h) = l(v_h) - a(w_h, v_h) =: \tilde{l}(v_h) \). Therefore in order to solve (6.1) with nonzero Dirichlet boundary function \( u_D \), a function \( w_h \in V_h^D \) is chosen and (6.3) is solved with the modified RHS \( \tilde{l} \) to get \( u_h \). The solution is given by \( \tilde{u}_h = u_h + w_h \). As \( w_h \in V_h^D \), it has a representation of the form \( w_h = \sum_{i=0}^{n} w^i \Psi_i \). Hence \( a(w_h, \Psi_i) \) can be calculated as

\[
a(w_h, \Psi_i) = \sum_{j=0}^{n} a(w^j \Psi_j, \Psi_i) = \sum_{j=0}^{n} w^j a_{ij}.
\]

Let us now consider the case of NURBS. Then the physical domain \( \Omega \) is given by a parametrization

\[
S(\xi^{(1)}, \xi^{(2)}) = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,j}(\xi^{(1)}, \xi^{(2)}) p_{i,j}^{w}.
\]
of degree \( p^{(1)} \) (resp. \( p^{(2)} \)) defined on the given knot vectors

\[
\Xi^{(1)} = \left\{ 0, \ldots, 0, \xi_{p^{(1)}+1}^{(1)}, \ldots, \xi_{m_1-p^{(1)}-1}^{(1)}, 1, \ldots, 1 \right\}
\]

and

\[
\Xi^{(2)} = \left\{ 0, \ldots, 0, \xi_{p^{(2)}+1}^{(2)}, \ldots, \xi_{m_2-p^{(2)}-1}^{(2)}, 1, \ldots, 1 \right\},
\]

and with given homogeneous control points \( P^w_{i,j} \). This parametrization should be continuously differentiable to allow the following analysis.

Applying the isogeometric concept we now want the functions \( N_{i,j} \), which describe the geometry, to be the basis for the solution space. Thus the solution space becomes

\[
V^D_h = \mathcal{N}^{n^{(1)},n^{(2)}}(\Xi^{(1)}, \Xi^{(2)}) = \left\{ S\left(\xi^{(1)}, \xi^{(2)}\right) | S\left(\xi^{(1)}, \xi^{(2)}\right) = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,j} \left(\xi^{(1)}, \xi^{(2)}\right) c_{i,j} \right\},
\]

the space of NURBS defined by the knot vectors \( \Xi^{(1)} \) and \( \Xi^{(2)} \) with degree \( p^{(1)} = n^{(1)} - 1 \) in \( \Xi^{(1)} \)-direction and \( p^{(2)} = n^{(2)} - 1 \) in \( \Xi^{(2)} \)-direction. The dimension of \( V^D_h \) is given by

\[
n^{(D)} = \left(n^{(1)} + 1\right) \cdot \left(n^{(2)} + 1\right).
\]

The space of test functions is spanned by the same basis functions. To guarantee that the test functions \( v_h \) equal zero on the boundary of the domain, they have to satisfy

\[
v_h(x, 0) = v_h(x, 1) = v_h(0, y) = v_h(1, y) = 0.
\]

For an arbitrary \( v \in \mathcal{N}^{n^{(1)},n^{(2)}}(\Xi^{(1)}, \Xi^{(2)}) \), \( v = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} N_{i,j} \left(\xi^{(1)}, \xi^{(2)}\right) c_{i,j} \) this requirement results in

\[
v(i, 0) = c_{i,0} = 0,
\]
\[
v(i, 1) = c_{i,n^{(2)}} = 0,
\]
\[
v(0, j) = c_{0,j} = 0,
\]
\[
v(1, j) = c_{n^{(1)},j} = 0.
\]

Thus a function in \( V_h = \mathcal{N}^{n^{(1)},n^{(2)}}(\Xi^{(1)}, \Xi^{(2)}) \cap H^1_0 \) does not depend on \( N_{0,j} \), \( N_{n^{(1)},j} \), \( N_{i,0} \) nor \( N_{i,n^{(2)}} \) and a basis for the test function space \( V_h \) is \( \{N_{i,j}|1 \leq i \leq n^{(1)} - 1, 1 \leq j \leq n^{(2)} - 1\} \).

With these functions we now have to calculate \( A \) and \( b \). Therefore the integrals are transformed to the domain in the parametric space i.e. the unit square and then a Gaussian quadrature rule is applied. Using the NURBS as basis functions, a transformation yields
\[ A = \left( q_{k_1, l_1, k_2, l_2} \right)_{k_1 l_1, k_2 l_2 = 0} \]
\[ = \varepsilon \int_{\Omega} \nabla_{x, y} \Psi_{k_2 l_2} \cdot \nabla_{x, y} \Psi_{k_1 l_1} \, dx \, dy + \int_{\Omega} \left( b \cdot \nabla \Psi_{k_2 l_2} \right) \Psi_{k_1 l_1} \, dx \, dy \]
\[ = \varepsilon \int_{\Omega} \nabla_{x, y} \tilde{N}_{k_2 l_2} (x, y) \cdot \nabla_{x, y} \tilde{N}_{k_1 l_1} (x, y) \, dx \, dy \]
\[ + \int_{\Omega} \left( b \cdot \nabla \tilde{N}_{k_2 l_2} (x, y) \right) \tilde{N}_{k_1 l_1} (x, y) \, dx \, dy \]
\[ = \varepsilon \int_{[0, 1] \times [0, 1]} \left( \nabla_{x, y} \tilde{N}_{k_2 l_2} \left( \xi^{(1)}, \xi^{(2)} \right) \cdot \nabla_{x, y} \tilde{N}_{k_1 l_1} \left( \xi^{(1)}, \xi^{(2)} \right) \right) \left| \det \left( DS^{-1} \left( \xi^{(1)}, \xi^{(2)} \right) \right) \right| \, d\xi^{(1)} \, d\xi^{(2)}. \]

Thereby \( \tilde{N}_{k_1 l_1} (x, y) = N_{k_1 l_1} \left( S^{-1} (x, y) \right) \) describe the basis functions corresponding to a point in the physical domain \( \Omega \). At this point one should pay attention to the gradient. It is meant to be taken with respect to \( x \) and \( y \). As the integral is with respect to \( \xi^{(1)} \) and \( \xi^{(2)} \), the gradient should be transformed as well. The chain rule implies

\[ \nabla_{x, y} \tilde{N}_{k_1 l_1} \left( \xi^{(1)}, \xi^{(2)} \right) = \frac{\partial N_{k_1 l_1} \left( \xi^{(1)}, \xi^{(2)} \right)}{\partial x} \frac{\partial \xi^{(1)}}{\partial x} + \frac{\partial N_{k_1 l_1} \left( \xi^{(1)}, \xi^{(2)} \right)}{\partial y} \frac{\partial \xi^{(2)}}{\partial y} \]
\[ = DS^{-T} \left( \nabla_{\xi^{(1)}, \xi^{(2)}} N_{k_1 l_1} \left( \xi^{(1)}, \xi^{(2)} \right) \right) \]
\[ = \left( DS \left( \xi^{(1)}, \xi^{(2)} \right) \right)^{-T} \cdot \nabla_{\xi^{(1)}, \xi^{(2)}} N_{k_1 l_1} \left( \xi^{(1)}, \xi^{(2)} \right). \]
\( DS \) (resp. \( DS^{-1} \)) is thereby representing the Jacobian matrix of \( S \) (resp. \( S^{-1} \)). Hence

\[
A = \int_{[0,1] \times [0,1]} \left( \left( DS\left( \xi^{(1)}, \xi^{(2)} \right) \right)^{-T} \cdot V_{\xi^{(1)}, \xi^{(2)}} N_{k_2, l_2} \left( \xi^{(1)}, \xi^{(2)} \right) \right)
\times \left( \left( DS\left( \xi^{(1)}, \xi^{(2)} \right) \right)^{-T} \cdot V_{\xi^{(1)}, \xi^{(2)}} N_{k_1, l_1} \left( \xi^{(1)}, \xi^{(2)} \right) \right)
+ \left( b \cdot \left( DS\left( \xi^{(1)}, \xi^{(2)} \right) \right)^{-T} \cdot V_{\xi^{(1)}, \xi^{(2)}} N_{k_2, l_2} \left( \xi^{(1)}, \xi^{(2)} \right) \right) N_{k_1, l_1} \left( \xi^{(1)}, \xi^{(2)} \right)
\times \left| \det \left( DS^{-1}\left( \xi^{(1)}, \xi^{(2)} \right) \right) \right| d\xi^{(1)} d\xi^{(2)}.
\]

Next the unit square is decomposed into the knot spans:

\[
da_{k_1, l_1} = \sum_{i=0}^{m_1-1} \sum_{j=0}^{m_2-1} \int_{\left[ \xi^{(1)}_i, \xi^{(1)}_{i+1} \right] \times \left[ \xi^{(2)}_j, \xi^{(2)}_{j+1} \right]} \left( \left( DS\left( \xi^{(1)}, \xi^{(2)} \right) \right)^{-T} \cdot V_{\xi^{(1)}, \xi^{(2)}} N_{k_2, l_2} \left( \xi^{(1)}, \xi^{(2)} \right) \right)
\times \left( \left( DS\left( \xi^{(1)}, \xi^{(2)} \right) \right)^{-T} \cdot V_{\xi^{(1)}, \xi^{(2)}} N_{k_1, l_1} \left( \xi^{(1)}, \xi^{(2)} \right) \right)
+ \left( b \cdot \left( DS\left( \xi^{(1)}, \xi^{(2)} \right) \right)^{-T} \cdot V_{\xi^{(1)}, \xi^{(2)}} N_{k_2, l_2} \left( \xi^{(1)}, \xi^{(2)} \right) \right) N_{k_1, l_1} \left( \xi^{(1)}, \xi^{(2)} \right)
\times \left| \det \left( DS^{-1}\left( \xi^{(1)}, \xi^{(2)} \right) \right) \right| d\xi^{(1)} d\xi^{(2)}.
\]

Let us now assume we have defined the nodes \( \gamma_1, \ldots, \gamma_g \) on \([-1,1]\) and weights \( w_1, \ldots, w_g \) for the Gaussian quadrature rule. These points are then transformed to \( \left[ \xi^{(1)}_i, \xi^{(1)}_{i+1} \right] \) and \( \left[ \xi^{(2)}_j, \xi^{(2)}_{j+1} \right] \), respectively, via

\[
\bar{\gamma}_{i, \alpha} = \frac{\left( \xi^{(1)}_{i+1} - \xi^{(1)}_i \right) \cdot \gamma_\alpha + \left( \xi^{(1)}_i + \xi^{(1)}_{i+1} \right)}{2} \quad \text{and} \quad \bar{\gamma}_{j, \beta} = \frac{\left( \xi^{(2)}_{j+1} - \xi^{(2)}_j \right) \cdot \gamma_\beta + \left( \xi^{(2)}_j + \xi^{(2)}_{j+1} \right)}{2}.
\]
Applying the Gaussian quadrature rule and re-transforming the integral we finally get

\[
\begin{align*}
ad_{k_1l_1,k_2l_2} &= \sum_{i=0}^{m_1-1} \sum_{j=0}^{m_2-1} \left[ \frac{\xi^{(1)}_{i+1} - \xi^{(1)}_i}{2} \cdot \frac{\xi^{(2)}_{j+1} - \xi^{(2)}_j}{2} \right] \\
&\quad \times \sum_{a=1}^g \sum_{b=1}^g w_a w_b \cdot \left( \left( \left( DS \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right)^T \cdot \nabla_{\xi^{(1)}, \xi^{(2)}} N_{k_2l_2} \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) \\
&\quad \times \left( \left( DS \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right)^T \cdot \nabla_{\xi^{(1)}, \xi^{(2)}} N_{k_1l_1} \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) \\
&\quad + \left( b \cdot \left( DS \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right)^T \cdot \nabla_{\xi^{(1)}, \xi^{(2)}} N_{k_2l_2} \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) N_{k_1l_1} \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) \\
&\quad \cdot \left| \det \left( DS^{-1} \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) \right| \\
&\quad \text{for all } 0 \leq k_1 \leq n^{(1)}, 0 \leq l_1 \leq n^{(2)}, 0 \leq k_2 \leq n^{(1)}, 0 \leq l_2 \leq n^{(2)}. \quad (6.5)
\end{align*}
\]

The entry \(d_{k_1l_1,k_2l_2}\) is then written into the matrix \(A\) at row \(k_1 \cdot n^{(2)} + l_1 + 1\) and column \(k_2 \cdot n^{(2)} + l_2 + 1\).

Similarly the entry \(b_{k_1l_1}\) in \(b\) at the position \(k_1 \cdot n^{(2)} + l_1 + 1\) is given by

\[
\begin{align*}
b_{k_1l_1} &= \int_{\Omega} f \cdot \Psi_{k_1l_1} \, dx \, dy = \int_{\Omega} f(x, y) \cdot N_{k_1l_1} (x, y) \, dx \, dy \\
&= \int_{[0,1] \times [0,1]} f \left( S \left( \xi^{(1)}, \xi^{(2)} \right) \right) N_{k_1l_1} \left( \xi^{(1)}, \xi^{(2)} \right) \left| \det \left( DS^{-1} \left( \xi^{(1)}, \xi^{(2)} \right) \right) \right| \, d\xi^{(1)} \, d\xi^{(2)} \\
&= \sum_{i=0}^{m_1-1} \sum_{j=0}^{m_2-1} \int_{\left[ \xi^{(1)}_i, \xi^{(1)}_{i+1} \right]} \int_{\left[ \xi^{(2)}_j, \xi^{(2)}_{j+1} \right]} f \left( S \left( \xi^{(1)}, \xi^{(2)} \right) \right) N_{k_1l_1} \left( \xi^{(1)}, \xi^{(2)} \right) \\
&\quad \times \left| \det \left( DS^{-1} \left( \xi^{(1)}, \xi^{(2)} \right) \right) \right| \, d\xi^{(1)} \, d\xi^{(2)} \\
&= \sum_{i=0}^{m_1-1} \sum_{j=0}^{m_2-1} \frac{\xi^{(1)}_{i+1} - \xi^{(1)}_i}{2} \cdot \frac{\xi^{(2)}_{j+1} - \xi^{(2)}_j}{2} \\
&\quad \times \sum_{a=1}^g \sum_{b=1}^g w_a w_b \cdot \left( f \left( S \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) \right) N_{k_1l_1} \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) \\
&\quad \cdot \left| \det \left( DS^{-1} \left( \overline{r}_{i,a}, \overline{r}_{j,b} \right) \right) \right| \\
&\quad \text{for all } 0 \leq k_2 \leq n^{(1)}, 0 \leq l_2 \leq n^{(2)}. \quad (6.5)
\end{align*}
\]
The modified RHS for nonhomogeneous Dirichlet boundary with the auxiliary function
\(w_h = \sum_{i=0}^{n^{(1)}} \sum_{j=0}^{n^{(2)}} w_{h}^{i,j} N_{i,j}\) is calculated as

\[\tilde{b}_{k_{1}l_{1}} = b_{k_{1}l_{1}} - \sum_{k_{2}=0}^{n^{(1)}} \sum_{l_{2}=0}^{n^{(2)}} w_{h}^{k_{2}l_{2}} a_{k_{1}l_{1},k_{2}l_{2}}.\]

To find \(u_h \in V_h\) the equation \((a_{i,j})_{i,j=1}^{n^{(1)}-1,n^{(2)}-1} \cdot (x)_{i=1}^{n^{(1)}-1} = (\tilde{b}_{i})_{i=1}^{n^{(1)}-1}\) is solved.

### 6.2 SUPG Stabilization

To reduce unphysical oscillations in the numerical solution of the convection-diffusion equation a stabilization is necessary. In this thesis the popular Streamline-Upwind/Petrov-Galerkin Method (SUPG) cf. [BH82] is used. It modifies the test function \(v_h\) to \(\tilde{v}_h = v_h + \delta b \cdot \nabla v_h\) by adding its weighted derivative. Hence the weak problem (6.2) becomes

\[
\begin{align*}
\varepsilon & \int_{\Omega} \nabla u_h \cdot \nabla v_h \, dx + \int_{\Omega} (b \cdot \nabla u_h) v_h \, dx \\
- \varepsilon & \int_{\Omega} \Delta u_h \cdot \delta (b \cdot \nabla v_h) \, dx + \int_{\Omega} (b \cdot \nabla u_h) \delta (b \cdot \nabla v_h) \, dx \\
= & \int_{\Omega} f \cdot v_h \, dx + \int_{\Omega} f \cdot \delta (b \cdot \nabla v_h) \, dx.
\end{align*}
\]

(6.6)

The term containing \(\Delta u_h\) has to be interpreted element-wise

\[
\varepsilon \int_{\Omega} \Delta u_h \cdot \delta (b \cdot \nabla v_h) \, dx = \varepsilon \sum_{i=0}^{n_{1}-1} \sum_{j=0}^{n_{2}-1} \int_{S} [\xi_{i}^{(1)} \xi_{j}^{(1)}] [\xi_{i}^{(2)} \xi_{j+1}^{(2)}] \Delta u_h \cdot \delta (b \cdot \nabla v_h) \, dx.
\]

The parameter \(\delta\) is called the stabilization parameter. For details on the choice of an optimal parameter see [JK07]. The additional terms in (6.6) are treated as in Section 6.1. Similarly to the gradient \(\nabla\) the Laplacian \(\Delta N_{ij} = \Delta_{xy} N_{ij} = \partial^{2}N_{ij}/\partial x^{2} + \partial^{2}N_{ij}/\partial y^{2}\) is...
transformed using
\[
\frac{\partial^2 N_{ij}}{\partial x^2} = \frac{\partial}{\partial x} \left( \frac{\partial N_{ij}}{\partial \xi(1)} \frac{\partial \xi(1)}{\partial x} + \frac{\partial N_{ij}}{\partial \xi(2)} \frac{\partial \xi(2)}{\partial x} \right)
\]
\[
= \frac{\partial^2 N_{ij}}{\partial \xi(1)^2} \left( \frac{\partial \xi(1)}{\partial x} \right)^2 + \frac{\partial^2 N_{ij}}{\partial \xi(2)^2} \left( \frac{\partial \xi(2)}{\partial x} \right)^2 + \frac{\partial^2 N_{ij}}{\partial \xi(1) \partial \xi(2)} \left( \frac{\partial \xi(1)}{\partial x} \right) \left( \frac{\partial \xi(2)}{\partial x} \right)
\]
and the analogous formula for \( \frac{\partial^2 N_{ij}}{\partial y^2} \). Thus passing to the matrix notation and applying the transformations finally leads to
\[
[a_{k_1,l_1,k_2,l_2}] = \sum_{i=0}^{n_{1}-1} \sum_{j=0}^{n_{2}-1} \left[ \xi_i^{(1)} - \xi_{j+1}^{(1)} \right] \left[ \xi_j^{(2)} - \xi_{j+1}^{(2)} \right] \times \sum_{a=1}^{8} \sum_{\beta=1}^{8} w_a w_{\beta}
\]
\[
\times \left( DS \left( \gamma_{ij}^{a\beta} \right)^{-T} \cdot \nabla \xi N_{k_2,l_2} \left( \gamma_{ij}^{a\beta} \right) \right) \times \left( DS \left( \gamma_{ij}^{a\beta} \right)^{-T} \cdot \nabla \xi N_{k_1,l_1} \left( \gamma_{ij}^{a\beta} \right) \right) + \left( b \cdot DS \left( \gamma_{ij}^{a\beta} \right)^{-T} \cdot \nabla \xi N_{k_2,l_2} \left( \gamma_{ij}^{a\beta} \right) \right) \times \left( DS \left( \gamma_{ij}^{a\beta} \right)^{-T} \cdot \nabla \xi N_{k_1,l_1} \left( \gamma_{ij}^{a\beta} \right) \right)
\]
\[
- \varepsilon \Delta \xi N_{k_2,l_2} \left( \gamma_{ij}^{a\beta} \right) \cdot \delta \left( b \cdot DS \left( \gamma_{ij}^{a\beta} \right)^{-T} \cdot \nabla \xi N_{k_1,l_1} \left( \gamma_{ij}^{a\beta} \right) \right)
\]
\[
+ \left( \gamma_{ij}^{a\beta} \right) \cdot \delta \left( \gamma_{ij}^{a\beta} \right)^{-T} \cdot \nabla \xi N_{k_1,l_1} \left( \gamma_{ij}^{a\beta} \right)
\]
\[
\times \delta \left( \gamma_{ij}^{a\beta} \right)^{-T} \cdot \nabla \xi N_{k_1,l_1} \left( \gamma_{ij}^{a\beta} \right)
\]
\[
\cdot \left| \det \left( DS^{-1} \left( \gamma_{ij}^{a\beta} \right) \right) \right|
\]
for all \( 0 \leq k_1 \leq n^{(1)}, 0 \leq l_1 \leq n^{(2)}, 0 \leq k_2 \leq n^{(1)}, 0 \leq l_2 \leq n^{(2)}, \)

where \( \gamma_{ij}^{a\beta} = \left( \gamma_{i\alpha}^{a\beta}, \gamma_{j\beta}^{a\beta} \right) \) and \( \Delta \xi \) denotes the transformed Laplacian as described above.
The right hand side is
\[
b_{k,l} = \sum_{i=0}^{m_1-1} \sum_{j=0}^{m_2-1} \frac{\xi_i^{(1)} - \xi_{i+1}^{(1)}}{2} \cdot \frac{\xi_j^{(2)} - \xi_{j+1}^{(2)}}{2} \times \sum_{a=1}^{\gamma} \sum_{\beta=1}^{\delta} \left( w_\alpha w_\beta \right) \times \left( f \left( S \left( y_{i}^{(1)} a \beta \right) \right) \cdot N_{k,l,1} \left( y_{i}^{(1)} a \beta \right) \right) + f \left( S \left( y_{i}^{(1)} a \beta \right) \right) \cdot \delta \left( b \cdot DS \left( y_{i}^{(1)} a \beta \right) \right)^{-T} \nabla_{\xi_{i}^{(1)},\xi_{j}^{(2)}} N_{k,l,1} \left( \gamma_{i}^{(1)}, \gamma_{j}^{(2)} \right)
\]
\[
\times \det \left( DS^{-1} \left( y_{i}^{(1)} a \beta \right) \right)
\]
\[
for all 0 \leq k_2 \leq n^{(1)}, 0 \leq l_2 \leq n^{(2)}.
\]

6.3 Example I: Skewed Advection on the Unit Square

The first example is taken from [HCB05, Section 5.1].
The domain \( \Omega \) is the unit square \( (0, 1) \times (0, 1) \). Its coarsest parametrization (cf. Figure 2) is defined on the knot vectors
\[
\Xi^{(1)} = \{0, 0, 0, 1, 1, 1\} \quad \text{and} \quad \Xi^{(2)} = \{0, 0, 0, 1, 1, 1\}
\]
with the homogeneous control points

\[
p_{0,0}^0 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad p_{0,1}^0 = \begin{pmatrix} 0.5 \\ 0 \\ 1 \end{pmatrix}, \quad p_{0,2}^0 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}
\]
\[
p_{1,0}^0 = \begin{pmatrix} 0.5 \\ 0 \\ 1 \end{pmatrix}, \quad p_{1,1}^0 = \begin{pmatrix} 0.5 \\ 0.5 \\ 1 \end{pmatrix}, \quad p_{1,2}^0 = \begin{pmatrix} 0.5 \\ 1 \\ 1 \end{pmatrix}
\]
\[
p_{2,0}^0 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \quad p_{2,1}^0 = \begin{pmatrix} 1 \\ 0.5 \\ 1 \end{pmatrix}, \quad p_{2,2}^0 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}
\]

The boundary condition is a Dirichlet condition on the entire boundary \( \Gamma_D = \partial \Omega \):
\[
u_D = \begin{cases} 1, & \text{on } \Gamma_1 := (\{0, 1\} \times \{0\}) \cup (\{0\} \times [0, 0.2]), \\ 0, & \text{on } \Gamma_0 = \partial \Omega \setminus \Gamma_1. \end{cases}
\]
To implement this boundary condition the third dimension of the control points of the auxiliary function \( w_h \) is set to one times the weight \( \omega \) if and only if the first and second
coordinate \((x, y)\) are on \(\Gamma_1\), and to zero otherwise. The parameters are chosen to be \(\varepsilon = 10^{-6}\), \(b = (\cos \theta, \sin \theta)\), and \(f \equiv 0\). The two angles \(\theta = 45^\circ\) and \(\theta = \arctan(2)\) are tested. The stabilization parameter is \(\delta = \tilde{h}/2\), where \(\tilde{h} = h_Q / \max(\sin \theta, \cos \theta)\) and \(h_Q\) denotes the mesh size.

A numerical solution to this problem is depicted in Figure 4.
Figure 3: Boundary approximation $w$ for Example I. Left: $p = 3$. Right: $p = 12$.

Figure 4: Solution of Example I. Left: $\theta = 45^\circ$. Right: $\theta = \arctan(2)$. 
6.4 Example II: Sine Hill on a Circle

The second example is taken from [HCB05, Section 5.2] with some minor changes. To show one advantage of NURBS in the parametrization, the unit circle is chosen as domain $\Omega = \{(x, y) \in \mathbb{R}^2 | x^2 + y^2 < 1\}$. Its coarsest parametrization (cf. Figure 5) is realized by the knot vectors

$$\Xi^{(1)} = \{0, 0, 0, 0.25, 0.25, 0.5, 0.5, 0.75, 0.75, 1, 1, 1\} \quad \text{and} \quad \Xi^{(2)} = \{0, 0, 0, 1, 1, 1\},$$

and the control points

$$P_{0,0}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{1,0}^\omega = \begin{pmatrix} 0 \\ 0 \\ \sqrt{2} \end{pmatrix}, \quad P_{2,0}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{3,0}^\omega = \begin{pmatrix} 0 \\ 0 \\ \sqrt{2} \end{pmatrix}, \quad P_{4,0}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

$$P_{5,0}^\omega = \begin{pmatrix} \frac{1}{2} \\ 0 \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad P_{6,0}^\omega = \begin{pmatrix} \frac{1}{2} \\ 0 \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad P_{7,0}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{8,0}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

$$P_{0,1}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{1,1}^\omega = \begin{pmatrix} \frac{1}{2} \sqrt{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{2,1}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{3,1}^\omega = \begin{pmatrix} -\frac{1}{2} \sqrt{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{4,1}^\omega = \begin{pmatrix} -0.5 \\ 0 \sqrt{2} \end{pmatrix},$$

$$P_{5,1}^\omega = \begin{pmatrix} -\frac{1}{2} \sqrt{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{6,1}^\omega = \begin{pmatrix} -\frac{1}{2} \sqrt{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{7,1}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{8,1}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

$$P_{0,2}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{1,2}^\omega = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{2,2}^\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad P_{3,2}^\omega = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{4,2}^\omega = \begin{pmatrix} -1 \\ 0 \sqrt{2} \end{pmatrix},$$

$$P_{5,2}^\omega = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{6,2}^\omega = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{7,2}^\omega = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ 0 \sqrt{2} \end{pmatrix}, \quad P_{8,2}^\omega = \begin{pmatrix} 0 \\ 0 \sqrt{2} \end{pmatrix}. $$

Once again only Dirichlet boundary conditions are implemented. The boundary $(\xi^{(1)}, \xi^{(2)}) \in [0, 1] \times ([0] \cup \{1\})$ is mapped onto the pairs $(x, y) \in \mathbb{R}^2$ satisfying $x^2 + y^2 = 0$ and $x^2 + y^2 = 1$, respectively. Here the Dirichlet boundary condition is homogeneous, $u_D = 0$.

In [HCB05] an additional condition for the solution $u$ is imposed. On the slit $[0, 1] \times \{0\}$ it
is defined to be $u = \sin \pi x$. As the boundaries $[0] \times [0, 1]$ and $[1] \times [0, 1]$ of the parametric domain are both mapped to $\{(x, y) | (x, y) \in [0, 1] \times \{0\}\}$, this condition is implemented as an artificial boundary condition

$$u_D(x, y) = \sin (\pi x), \quad \text{for } (x, y) \in S\left( ([0] \cup [1]) \times [0, 1] \right).$$

The approximation of the boundary conditions for $p = 2$ and $p = 6$ is depicted in Figure 6.

The parameters are $\varepsilon = 10^{-6}$ and $f \equiv 0$ as before and $b(x, y) = (-y, x)^T$. The stabilization parameter is $\delta = \tilde{h}/(2 \cdot |a|)$ where $\tilde{h}$ denotes the diagonal of an element.
6.5 Example III: Hemker Problem

The last example is the so-called Hemker problem. In [ACF+11] it is studied in detail and different stabilization methods as well as different FEMs are compared.

The Hemker problem is defined in \( \Omega = \{ (x, y) \in \mathbb{R}^2 | x \in (-3, 9), y \in (-3, 3), x^2 + y^2 > 1 \} \). The coarsest parametrization (cf. Figure 7) is defined on the knot vectors

\[
\Xi^{(1)} = \{0, 0, 0, 1, 1, 1, 2, 2, 1, 1, 3, 3, 3, 3, 7, 7, 8, 8, 1, 1, 1\},
\]

\[
\Xi^{(2)} = \{0, 0, 1, 1\},
\]

by the control points

\[
\begin{align*}
& p_{0,0}^{\omega} = \begin{pmatrix} -3 \\ 0 \\ 1 \end{pmatrix}, p_{1,0}^{\omega} = \begin{pmatrix} -3 \\ 0 \\ 1 \end{pmatrix}, p_{2,0}^{\omega} = \begin{pmatrix} -3 \\ 0 \\ 1 \end{pmatrix}, p_{3,0}^{\omega} = \begin{pmatrix} -3 \\ 0 \\ 1 \end{pmatrix}, p_{4,0}^{\omega} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, p_{5,0}^{\omega} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \\
& p_{6,0}^{\omega} = \begin{pmatrix} 9 \\ 0 \\ 1 \end{pmatrix}, p_{7,0}^{\omega} = \begin{pmatrix} 9 \\ 0 \\ 1 \end{pmatrix}, p_{8,0}^{\omega} = \begin{pmatrix} 9 \\ 0 \\ 1 \end{pmatrix}, p_{9,0}^{\omega} = \begin{pmatrix} 9 \\ 0 \\ 1 \end{pmatrix}, p_{10,0}^{\omega} = \begin{pmatrix} 3 \\ 0 \\ 1 \end{pmatrix}, p_{11,0}^{\omega} = \begin{pmatrix} 3 \\ 0 \\ 1 \end{pmatrix}, \\
& p_{12,0}^{\omega} = \begin{pmatrix} 0 \\ 3 \\ 1 \end{pmatrix}, p_{13,0}^{\omega} = \begin{pmatrix} -1.5 \\ 3 \\ 1 \end{pmatrix}, p_{14,0}^{\omega} = \begin{pmatrix} -3 \\ 3 \\ 1 \end{pmatrix}, p_{15,0}^{\omega} = \begin{pmatrix} -3 \\ 3 \\ 1 \end{pmatrix}, p_{16,0}^{\omega} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \\
& p_{17,0}^{\omega} = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}, p_{18,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0 \\ 0 \end{pmatrix}, p_{19,0}^{\omega} = \begin{pmatrix} -1.5 \\ 0 \\ 0 \end{pmatrix}, p_{20,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0 \\ 0 \end{pmatrix}, p_{21,0}^{\omega} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \\
& p_{22,0}^{\omega} = \begin{pmatrix} 0.4243 \\ 0.8594 \\ 0.8536 \end{pmatrix}, p_{23,0}^{\omega} = \begin{pmatrix} 0.6994 \\ 0.8594 \\ 0.8536 \end{pmatrix}, p_{24,0}^{\omega} = \begin{pmatrix} 0.6994 \\ 0.8594 \\ 0.8536 \end{pmatrix}, p_{25,0}^{\omega} = \begin{pmatrix} 0.2828 \\ 0.8536 \\ 0.8536 \end{pmatrix}, p_{26,0}^{\omega} = \begin{pmatrix} 0.8828 \\ 0.2828 \\ 0.2828 \end{pmatrix}, \\
& p_{27,0}^{\omega} = \begin{pmatrix} -0.8243 \\ 0.8594 \\ 0.8536 \end{pmatrix}, p_{28,0}^{\omega} = \begin{pmatrix} -0.2828 \\ 0.8594 \\ 0.8536 \end{pmatrix}, p_{29,0}^{\omega} = \begin{pmatrix} -0.3536 \\ 0.8594 \\ 0.8536 \end{pmatrix}, p_{30,0}^{\omega} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, p_{31,0}^{\omega} = \begin{pmatrix} -0.6036 \\ 0.8536 \\ 0.8536 \end{pmatrix}, p_{32,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0.8536 \\ 0.8536 \end{pmatrix}, \\
& p_{33,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0.8536 \\ 0.8536 \end{pmatrix}, p_{34,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0.8536 \\ 0.8536 \end{pmatrix}, p_{35,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0.8536 \\ 0.8536 \end{pmatrix}, p_{36,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0.8536 \\ 0.8536 \end{pmatrix}, p_{37,0}^{\omega} = \begin{pmatrix} -0.536 \\ 0.8536 \\ 0.8536 \end{pmatrix}. 
\end{align*}
\]

For this problem a Dirichlet and a Neumann boundary condition are implemented on
\[ \Gamma_D = \{-3\} \times [-3,3] \cup \{(x,y)|x^2 + y^2 = 1\} \text{ and } \Gamma_N = \partial \Omega \setminus \Gamma_D. \] The Neumann boundary condition is homogeneous, \( \varepsilon \nabla u \cdot n = g \equiv 0 \) on \( \Gamma_N \). The Dirichlet boundary is defined by

\[
u_D(x,y) = \begin{cases} 0, & \text{if } (x, y) \in \{-3\} \times [-3,3], \\ 1, & \text{if } x^2 + y^2 = 1. \end{cases}
\]

In terms of \( \xi^{(1)} \) and \( \xi^{(2)} \) according to the parametrization this means that

\[
u_D(S(\xi^{(1)}, \xi^{(2)})) = \begin{cases} 0, & \text{if } \xi^{(2)} = 0 \text{ and } \xi^{(1)} \in [0, 0.125] \cup [0.875, 1], \\ 1, & \text{if } \xi^{(2)} = 1 \text{ and } \xi^{(1)} \in [0, 1], \end{cases}
\]

and for \( \xi^{(2)} = 0 \) and \( \xi^{(1)} \in (0.125, 0.875) \) the Neumann condition is applied.

Figure 8: Boundary approximation \( \nu \) for Example III. Left: 250 degrees of freedom. Right: 134418 degrees of freedom.

The parameters for the Hemker problem are chosen as in [ACF++11], i.e. \( f \equiv 0 \) as before,
\( \varepsilon = 10^{-4} \) and \( \mathbf{b} = (1,0)^T \). The stabilization parameter is given on each element \( Q \) by
\[
\delta_Q(x) = \frac{\tilde{h}}{2p|\mathbf{b}(x)|} \left( P_{eQ}(x) \right),
\]
\[
P_{eQ}(x) = \frac{|\mathbf{b}(x)|\tilde{h}}{2\varepsilon} \zeta(\alpha) = \coth \alpha - \frac{1}{\alpha},
\]
where \( \tilde{h} = h_Q \) is the length of an element in \( x \)-direction and \( p \) the degree of the NURBS. \( P_{eQ} \) is called the local Péclet number.

For the discussion of the numerical results, two slightly different domains are studied as well. For \( \Omega_1 \) the right border of the rectangle is set to \( x = 4 \) instead of \( x = 9 \). For \( \Omega_2 \) only the upper half \( y \geq 0 \) is considered as this allows a different parametrization. The coarsest meshes of these domains are depicted in Figures 9 and 10.

Figure 9: Control points and resulting mesh on the coarsest level for Example III, \( \Omega_1 \).

Figure 10: Control points and resulting mesh on the coarsest level for Example III, \( \Omega_2 \).
Finally another approach is implemented. The domain is divided into two parts along the axis $x = 3$. The first part $\Omega_3$ is parametrized similarly to $\Omega_1$. The second part $\Omega_4$ is the rectangle $[3, 9] \times [-3, 3]$. The coarsest mesh is depicted in Figure 11. The problem is then solved on $\Omega_3$ with the same parameters and boundary conditions as on $\Omega$. Afterwards the solution at $\{3\} \times [-3, 3]$ is taken as a Dirichlet boundary condition $u_D^*$. On $\Omega_4$ the Hemker problem is solved with the parameters as above and the following boundary conditions. On $\Gamma_D = \{3\} \times [-3, 3]$ it holds $u = u_D^*$ and on $\Gamma_N = \partial \Omega \setminus \Gamma_D$ the homogeneous Neumann boundary condition $\varepsilon \nabla u \cdot n \equiv 0$ is applied.

Figure 11: Control points and resulting mesh on the coarsest level for Example III, $\Omega_3 \cup \Omega_4$. 

---

1This approach was proposed by Yuri Bazilevs, personal communication.
7 Experiments

In this section the results of the numerical studies are presented. The first two examples are compared with the results in [HCB05], the last example is compared with [ACF+11].

7.1 Example I: Skewed Advection on the Unit Square

As in [HCB05] k-refinement was used. The coarsest mesh was first degree elevated and then refined by knot insertion to a $20 \times 20$ mesh. The knots are uniformly distributed. The results are shown in Figure 12 and 13 (resp. Figure 14 and 15) for $\theta = 45^\circ$ (resp. $\theta = \arctan(2)$). For $\theta = \arctan(2)$ the oscillations seem higher. In the ‘eye ball norm’ the results coincide with the results in [HCB05]. This problem was also discussed in [Fre10]. Their results are similar in the ‘eye ball norm’ as well. Therefore we conclude that the implementation in this thesis is correct.

To compare the results, the overshoot was calculated at $100 \times 100$ uniformly distributed points. The resulting maximal percentage overshoot is plotted vs. the polynomial degree of the NURBS in Figure 16. The results confirm, that the oscillations in the case $\theta = \arctan(2)$ are higher than in the case $\theta = 45^\circ$. To make comparison with [HCB05] easier, it is plotted separately for even and odd degrees as well. The graph shows that the convergence rate of the overshoot is similar to the one obtained in [HCB05]. In [Fre10] no overshoot is depicted, hence comparison is not possible.
Figure 12: Solution of Example I with $\theta = 45^\circ$. Left: even degrees from $p = 2$ to $p = 6$. Right: odd degrees from $p = 3$ to $p = 7$. Plot with $100 \times 100$ points.
Figure 13: Solution of Example I with $\theta = 45^\circ$. Left: even degrees from $p = 8$ to $p = 12$. Right: odd degrees from $p = 9$ to $p = 11$. Plot with $100 \times 100$ points.
Figure 14: Solution of Example I with $\theta = \arctan(2)$. Left: even degrees from $p = 2$ to $p = 6$. Right: odd degrees from $p = 3$ to $p = 7$. Plot with $100 \times 100$ points.
Figure 15: Solution of Example 1 with $\theta = \arctan(2)$. Left: even degrees from $p = 8$ to $p = 12$. Right: odd degrees from $p = 9$ to $p = 11$. Plot with $100 \times 100$ points.
Figure 16: Maximum percentage overshoot vs. degree for Example 1. Left: $\theta = 45^\circ$. Right: $\theta = \arctan(2)$. Top: all degrees. Middle: odd degrees. Bottom: even degrees.
7.2 Example II: Sine Hill on a Circle

In this example k-refinement was used once again. The solution is calculated on a 30\times30 mesh with uniformly distributed knots. The results for degree $p = 2$ to $p = 12$ are shown in Figure 17. They appear similar to the ones obtained in [HCB05]. This confirms again the correctness of implementation.

The implementation for this example differs from the one chosen in [HCB05]. Here, only the unit circle is taken as domain in contrast to the domain $[-1, 1] \times [-1, 1]$ chosen in [HCB05]. This choice should have no influence on the solution as in the difference of both domains the solution is identically zero. The unit circle demonstrates the power and the advantage of NURBS. A calculation on the square could easily be implemented with a finite element method. On the other hand in the calculation of such a problem on a circular domain, polygonal meshes would cause approximation errors on the boundary. These errors are prevented by taking NURBS as basis. Another advantage is that the slit condition simply becomes a boundary condition under the parametrization that was chosen. Hence it becomes easier to apply this condition.
Figure 17: Solution of Example II for increasing degrees $p$ and $q$. Plot with $100 \times 100$ points.
7.3 Example III: Hemker Problem

In this example h-refinement is used. The polynomial degree is always \( p = 2 \). It shows both advantages and disadvantages of IGA with NURBS. The major advantage is once again that the circular hole is exactly represented by even the coarsest mesh. That does not only prevent approximation errors at the boundary, it also makes communication with the exact design in each refinement step redundant. On the other hand the parametrization of the domain is difficult. A continuous map from the unit square onto \( \Omega \) is necessary. To deal with this problem, the unit square is ‘bent’ around the hole resulting in a parametrization that resembles polar coordinates, see Figure 7. This produces decent results, which are shown in Figure 18. Nevertheless they are not satisfying. First of all the uniform distribution of knots results in coarser mesh cells for \( x \geq 1 \) than for \( x \leq -1 \). In addition the mesh lines for \( x \geq 3 \) are not horizontal. Hence the smearing effect becomes large.

Another more practical problem occurred while comparing the results with [ACF+11]. For comparison the thickness of the layer \( 0.1 < u(x, y) < 0.9 \) at \( x = 4 \) should be calculated as well. The problem is, that the inverse of the parametrization is not known. Since no single knot line is mapped onto \( x = 4 \), the points \( (\xi^{(1)}, \xi^{(2)}) \) that are mapped onto \( \{(x, y)\mid x = 4\} \) can only be found by trial and error. This method is very expensive in time and memory.

To avoid this problem the domain \( \Omega_1 \) is considered. As the line \( x = 4 \) then becomes part of the boundary here \( u(4, y) \) can easily be evaluated and the calculation of the layer becomes less expensive. The results on \( \Omega_1 \) are depicted in Figure 19. The domain \( \Omega_2 \) was considered to avoid the strong smearing effect along the non-horizontal mesh lines. As \( \Omega_2 \) has no hole a continuous parametrization from the unit square is easy. The results on this domain are shown in Figure 20. The approach to divide the domain into \( \Omega_3 \) and \( \Omega_4 \) also yields good results depicted in Figure 21.
Figure 18: Solution of Example III on $\Omega$ for increasing degrees of freedom. Plot with $300 \times 300$ points.
Figure 19: Solution of Example III on $\Omega_1$ for increasing degrees of freedom. Plot with $300 \times 300$ points.
Figure 20: Solution of Example III on $\Omega_2$ for increasing degrees of freedom. Plot with $300 \times 300$ points. Turned 180° for better visibility of the layer.
Figure 21: Solution of Example III on $\Omega_3 \cup \Omega_4$ for increasing degrees of freedom. Plot with $300 \times 300$ points.
The width of the layer \( \{ y \mid 0.1 < u(4, y) < 0.9 \} \) is plotted in Figure 22. The comparison with [ACF’11] shows that the results in this thesis are similar to the ones obtained by SUPG-stabilization on quadrilateral grids. The layer width resulting from FEM on triangular grids is decreasing slightly faster. The comparison between the four different parametrization shows that the layer width obtained on \( \Omega \) and \( \Omega_2 \) is smaller than the layer width on \( \Omega_1 \) and \( \Omega_3 \cup \Omega_4 \). The mesh lines in the neighborhood of the circular hole are a possible explanation for this. Their angle to the horizontal convection velocity is greater. Hence the boundary condition \( u_D = 1 \) is smeared in the vertical direction. Nevertheless IGA converges for all the parametrizations to a layer width slightly smaller than 0.1 as in [ACF’11].

![Diagram](image)

Figure 22: Layer width of \( \{ y \mid 0.1 < u(4, y) < 0.9 \} \) of Example III.

In the sixth refinement step the parametrization from Figure 7 and Figure 10 show an anomaly. Suddenly the calculated layer becomes much thicker, approximately 0.7. As Figure 23 shows, the layer width is still very small. The problem is, that the upper peak of the layer becomes smaller than 0.9. Hence the first point where the solution passes \( u(x, y) = 0.9 \) is at \( y \approx -0.32 \) and this \( y \)-value is used to calculate the layer width. The reason for this anomaly can be seen in Figure 18. The boundary condition \( u_D = 1 \) is only exactly transported at \( y = 0 \). Away from this line the solution becomes smaller than 1 immediately. To avoid this problem the layer width depicted in Figure 22 was calculated for the layer \( \{ y \mid 0.1 < u(4, y) < 0.89 \} \) in the sixth refinement step for \( \Omega \) and \( \Omega_2 \).
Finally the over- and undershoots are depicted in Figure 24. The results are very similar for the four parametrizations. The overshoots become 0 at around 10000 degrees of freedom. The undershoots tend versus zero, from 10000 degrees of freedom on, as well. For $\Omega$ and $\Omega_1$ they lie on top of each other. The domain $\Omega_2$ shows slightly better results than the other cases.  

In comparison with the SUPG-stabilization in [ACF+11], the undershoots obtained in this thesis are significantly better on all parametrizations than P2-FEM. The Q2-FEM in [ACF+11] with SUPG can compete with the results in this thesis, but the results obtained with IGA on $\Omega_2$ are still better. The overshoots show better results with IGA than with FEM as well. Here even Q2-FEM can not compete with the IGA solutions. Hence concerning over- and especially undershoots, when the SUPG-stabilization is used IGA is superior to standard FEM for the Hemker problem. 

Figure 24: Over- and undershoot of Example III.
8 Implementation

This section gives insight into the code. It starts with an overview of the most important employed functions. This general survey will be followed by some more detailed descriptions of the main routines.

8.1 General Structure of the Code

The basis for most of the computations in these programs are NURBS. There are very detailed descriptions in C-pseudo code of all the used routines concerning NURBS in the book by Piegl and Tiller [PT97]. Most of them were published under a BCD license by D. M. Spink in the nurbs-toolbox. As they were used unchanged, they will not be commented any further.

The main part of the code written in this thesis consists of four routines. First there is test_of_nurbs_convection_hrefine.m. Second there are the slightly different functions test_of_nurbs_convection_krefine.m and test_of_nurbs_convection_2parts.m. All three of them contain the main program. That is they load the parameters of a chosen problem, followed by a loop over the calls of solving, refining and output routines.

Finally there is the solver, solve_FEM_on_NURBS_convection.m. It calculates the stiffness matrix $A$, the right-hand side $rhs$ and solves the equation $A \cdot x = rhs$.

Furthermore there are many minor routines, for example for the calculation of boundary values and the convection field.

8.2 test_of_nurbs_convection_hrefine.m

This is one of the main routines. Its input is a string domain that defines the example to be calculated. The possibilities ‘circle’, ‘rectangle-45’, ‘rectangle-atan2’, ‘hemker’, ‘hemker-x4’, ‘hemker-x3’ and ‘hemker-y0’ are implemented.

The first step is the definition of some input parameters, $gp$ denoting the number of Gaussian nodes, and $nosteps$ the number of steps for the h-refinement. Next there are some parameters that control the output:

```matlab
% Output
plot_solution = 1;
plot_mesh = 1;
plot_coarsest_mesh = 1;
plot_boundary = 1;
plot_overshoot = 1;
plot_layer = 1;
```

The parameters in lines 2 to 5 are set to be equal to 1 if the numerical solution, the mesh in each refinement step, the coarsest mesh and the boundary should be plotted. Setting their value to $= 0$ prevents the output. The next variables state whether the overshoot and the layer width should be calculated and plotted or not.
After some initializations the problem data is loaded.

```matlab
% Load the problem data
[surface, get_boundary, b_func, h_func, w_func, epsilon] = ...
load_geometry_convection(domain);
```

It contains the domain $\Omega$ in its NURBS parametrization in the variable `surface`. Further it contains function handles to calculate, in that order, the boundary points, the convection field function $b$, the function that provides the used $h$ for the calculation of the stabilization parameter $\delta$ and the Dirichlet boundary function. Finally the diffusion parameter $\epsilon$ is provided.

Afterwards the coarsest mesh is plotted if desired and h-refined once.

The next important step is ensuring the exactness of the Gaussian quadrature rule with the chosen parameters.

```matlab
degree(1:2) = surface.order(1:2) - 1;
while any(degree > 2 * gp - 1)
    gp = gp + 1;
    warning(['number of gauss points increased to ' int2str(gp) ... ' to guarantee exactness of the Gaussian quadrature rule']);
end
```

This is followed by the main loop.

```matlab
% Loop over the desired number of steps
while step < nosteps
    time_till_now = cputime - start_time
    step = step + 1
    % knot insertion = h-refinement
    % inserting the midpoint of each knot span
    size_u = size(surfacex.knots{1}, 2);
    size_v = size(surface.knots{2}, 2);
    x_insert = (surface.knots{1}(2:size_u) - surface.knots{1}(1: ... size_u - 1))/2 ... + surfacex.knots{1}(1:size_u - 1);
    y_insert = (surface.knots{2}(2:size_v) - surface.knots{2}(1: ... size_v - 1))/2 ... + surface.knots{2}(1:size_v - 1);
    y_insert = setdiff(y_insert, surfacex.knots{2});
    x_insert = setdiff(x_insert, surfacex.knots{1});
    surface = nrbkntins(surface, {x_insert, y_insert});
```

% get boundary nodes and values
[dbound, nbound] = get_boundary(surface);
w = w_func(surface, surface.number(1), surface.number(2));

% get degrees of freedom for current step
dof(step) = surface.number(1) * surface.number(2) - length(dbound);

% solving equation
surface_solved = ...
solve_FEM_on_NURBS_convection(surface, dbound, nbound, ...
    b_func, h_func, w, epsilon, gp);

% OUTPUT
% plot boundary
if plot_boundary
    ...
end

% plot control points, control polygon and mesh
if plot_mesh
    ...
end

% plot solution
if plot_solution
    ...
end

% calculate under- and overshoot
if plot_overshoot
    x = nrbeval(surface_solved, {[0:0.01:1, 0:0.01:1]});
    overshoot(step) = max(max(squeeze(x(3,:,:)) - 1))
    undershoot(step) = min(min(squeeze(x(3,:,:)) - 1))
    clear x;
end

% calculate layer
if plot_layer
    switch domain
        case 'hemker'
            x = nrbeval(surface_solved...
index1 = find(abs(squeeze(x(1,:,:)) - 4) < 10^(-3));

x2 = squeeze(x(2,index1));
x3 = squeeze(x(3,index1));

index2 = find(x3 > 0.1);
index3 = find(x3 > 0.9);
layerwidth = abs(min(min(x2(index3)) - min(x2(index2...))));

case 'hemker-x4'
    x = nrbeval(surface_solved,[0.4:0.00001:0.5,0]);
    x2 = squeeze(x(2,:,:));
    x3 = squeeze(x(3,:,:));
    index1 = find(x3 > 0.1);
    index2 = find(x3 > 0.9);
    layerwidth = abs(min(x2(1,index2)) - min(x2(1,index1)));
end

case 'hemker-y0'
    x = nrbeval(surface_solved,[10/12,0:0.0001:1]);
    x2 = squeeze(x(2,:,:));
    x3 = squeeze(x(3,:,:));
    index1 = find(0.1 > x3);
    index2 = find(x3 > 0.9);
    layerwidth = abs(max(x2(index2,1)) - min(x2(index1,1)));
end

otherwise
    layerwidth = -100;
end

if ~isempty(layerwidth)
    layer(step) = layerwidth
else
    layer(step) = 0
end

clear x;
end

In lines 5 to 15 the routine starts with an insertion of the midpoint of each knot span into the knot vectors, if it is not already contained in them. Afterwards it uses the functions get_boundary and w_func to calculate the boundary points dbound and nbound and the values w of ud. Then the degrees of freedom dof are calculated. In line 24 the program calls the function solve_FEM_on_NURBS_convection.m to compute a solution on the current NURBS surface surface with Dirichlet boundary dbound and Neumann boundary nbound, the parameter functions b_func and h_func and the parameters w, epsilon and gp.
This is followed by the plot of boundary, mesh and solution. For the mesh the later described functions `nrbplotcontrol.m` and `nrbplotmesh.m` are used. The other two plots are realized with the function `nrbplot.m` of the nurbs-toolbox. Afterwards in lines 44 to 49 overshoot and undershoot are calculated. To this end `surface_solved` is evaluated at $100 \times 100$ points.

Finally the layer is calculated depending on the chosen domain. For the Hemker problem with $\Omega$ this is a trial and error routine. For $\Omega_1$ `surface_solved` is only evaluated at the values of $\xi^{(1)}, \xi^{(2)}$ that correspond to the lower half of the right border of the rectangle limiting $\Omega_1$. For $\Omega_2$ the line $x = 4$ is produced by $\xi^{(1)} = 10/12$. Hence the evaluation is restricted to this line. Finally if the problem is not the Hemker problem the layer width is set to $-100$.

After the loop some final output follows. That is the overshoot and the layer are plotted if `plot_overshoot` and `plot_layer`, respectively, are true.

### 8.3 test_of_nurbs_convection_krefine.m and test_of_nurbs_convection_2parts.m

These functions proceed similarly to `test_of_nurbs_convection_hrefine.m`. In the case of k-refinement the main loop runs through the degree of `surface` till `maxdegree` is reached. At the beginning of each iteration the surface is h-refined. After calculating the boundary values, the convection-diffusion equation on this refined surface is solved. Then the starting surface is p-refined once. Thus the k-refinement strategy is implemented.

After the p-refinement it is important to check the exactness of the Gaussian quadrature rule again (see lines 27 to 30 of the following code-segment).

```matlab
% h-refine at current degree
if length(strfind(domain,'rectangle'))>0
    insert=linspace(0,1,21);
else
    insert=linspace(0,1,31);
end
x_insert=setdiff(insert,surface.knots{1});
y_insert=setdiff(insert,surface.knots{2});
surface_hrefined=nrbkntins(surface,{x_insert,y_insert});

% get boundary nodes and values
[dbound,nbound]=get_boundary(surface_hrefined);
w=w_func(surface_hrefined,surface_hrefined.number(1),...
        surface_hrefined.number(2));
dof(degree(1))=surface.number(1)*surface.number(2)-length(...
        dbound);

% solve equation
surface_solved = ...
```
solve_FEM_on_NURBS_convection(surface_hrefined, dbound,...
nbound, b_func, h_func, w, epsilon, gp);

% OUTPUT

% p-refine for next step
surface=nrbdegelev(surface,[1 1]);
degree=degree+1;
while any(degree>2*gp-1)
    gp=gp+1;
    warning('number of gauss points increased to guarantee ... exactness of the Gaussian quadrature rule');
end

The function test_of_nurbs_convection_2parts.m loads two domains.

[surface, get_boundary, b_func, h_func, w_func, epsilon]=...
load_geometry_convection(domain);
[surface2, get_boundary2, b_func2, h_func2, w_func2, epsilon2]=...
load_geometry_convection('secondpart');

Then the solver is called for the first domain. Afterwards the solution is used as boundary condition for the second domain (lines 25 to 27). Thereby the solution on the second domain is calculated.

surface_solved =...
solve_FEM_on_NURBS_convection(surface, dbound, nbound,...
    b_func, h_func, w, epsilon, gp);

% refinement of second part
minu=find(surface_solved.knots{1}>=0.4,1);
maxu=find(surface_solved.knots{1}>=0.6,1);

size_u=size(surface2.knots{1},2);
size_v=size(surface2.knots{2},2);
x_insert=(surface2.knots{1}(2:size_u)-surface2.knots{1}(1:...
    size_u-1))/2...
    +surface2.knots{1}(1:size_u-1);
y_insert=(surface_solved.knots{1}(minu:maxu)-surface_solved...
    .knots{1}(minu))/...
    (surface_solved.knots{1}(maxu)-surface_solved.knots{1}(...
minu = x_insert( ismembc(x_insert, surface2.knots(1)));
y_insert = y_insert( ismembc(y_insert, surface2.knots(2)));
surface2 = nrbkntins(surface2, {x_insert, y_insert});
[dbound2, nbound2] = get_boundary2(surface2);

% get boundary condition for second part from solution of ...
first part
minu = find(squeeze(surface_solved.coefs(1,:,:)) >= ...
3*squeeze(surface_solved.coefs(4,:,:)), 1, 'first');
maxu = find(squeeze(surface_solved.coefs(1,:,:)) >= ...
3*squeeze(surface_solved.coefs(4,:,:)), 1, 'last');
w2 = surface2;
w2.coefs(3,1,:) = squeeze(surface_solved.coefs(3, minu:maxu, 1));
w2.coefs(4,1,:) = squeeze(surface_solved.coefs(4, minu:maxu, 1));
dof(step) = dof(step) + surface2.number(1) * surface2.number(2) - ...
length(dbound2);

% solve the equation on the second part
surface_solved2 = ...
solve_FEM_on_NURBS_convection(surface2, dbound2, nbound2, ...
b_func2, h_func2, w2, epsilon2, gp);

For the output both parts are depicted in one figure.

nrhplot(surface_solved, [300 300]);
hold on;
rhplot(surface_solved2, [300 300]);

The other parts of the code are similar to test_of_nurbs_convection_hrefine.m.

8.4 solve_FEM_on_NURBS.m

This function contains the solver of the convection-diffusion equation with IGA. It starts with the initialization of all necessary variables:

def function new_surface = solve_FEM_on_NURBS_convection(surface, ... dbound, nbound, b_func, h_func, w, epsilon, gp)

% stabilization
b = zeros(gp, gp, 2);
% getting gauss points and weights
[gauss_nodes, gauss_weights] = gaussparameters(gp);
Then the main loop over all elements of the mesh follows. Within the loop the indices of basis functions whose supports intersect the actual element are determined (lines 5 to 8) followed by the transformation of the Gaussian nodes and weights to the element (lines 10 to 14). \( \tilde{h}, b, \) and \( \delta \) are calculated in lines 22, 39, and 40, respectively. Additionally the parametrization is derivated to get the Jacobian \( \text{jac} \), its determinant \( \text{determinant} \), and its inverse \( \text{invjac} \) at the transformed Gaussian nodes. Attention has to be paid if \( S(x,y) = S(x) \) or \( S(x,y) = S(y) \) as this causes \text{inf} or \text{nan} which are set to zero. These steps are done in lines 24 to 45. The calculation of the derivative of a NURBS in the lines
26 to 32 follows the algorithm in [PT97, pp. 136].

```matlab
% loop through all elements of the mesh
for i = 1:size_u
    for j = 1:size_v
        % getting the nonzero basis functions on [u_{i}, u_{i+1}]*[v_{j}, ...
        % v_{j+1}]
        imin = max(1, i - surface.order(1) + 1);
        imax = min(size_u, i);
        jmin = max(1, j - surface.order(2) + 1);
        jmax = min(size_v, j);

        % transform gauss nodes onto the element
        x(1:gp) = 0.5*(surface.knots{1}(i+1)-surface.knots{1}(i)...)
        *gauss_nodes... + (surface.knots{1}(i+1)+surface.knots{1}(i))
        *ones(1, gp...));
        y(1:gp) = 0.5*((surface.knots{2}(j+1)-surface.knots{2}(j)...)
        *gauss_nodes... + (surface.knots{2}(j+1)+surface.knots{2}(j))
        *ones(1, gp...));

        % get the 1st and 2nd derivative of the curve at the ...
        % gauss nodes
        % calculate the determinant of the transformation at the ...
        % 3*3 gauss nodes
        % stabilization
        gridpoints = nrbeval(surface, {[surface.knots{1}(i) surface....
        knots{1}(i+1)]}, ...
        [surface.knots{2}(j+1) surface.knots{2}(j)]);
        h = h_func(gridpoints);
        if h > eps
            for ii = 1:gp
                for jj = 1:gp
                    Aders = nrbsurfacederivs(size_u-1, surface.order(1)-1,...
                    surface.knots{1},...
                    size_v-1, surface.order(2)-1, surface.knots{2},...
                    surface.coefs(1:3,:), x(ii), y(jj), 2);
                    wders = nrbsurfacederivs(size_u-1, surface.order(1)-1,...
                    surface.knots{1},...
                    size_v-1, surface.order(2)-1, surface.knots{2},...
                    surface.coefs(4,:,:), x(ii), y(jj), 2);
```
\[
\begin{align*}
DN(:, :, :, ii, jj) &= \text{nrbratsurfacederivs} (\text{Aders}, \text{squeeze}(\ldots, \text{wders}), 2); \\
\text{jac}(:, :) &= \text{horzcat} (\text{DN}(1:2, 2, 1, ii, jj), \text{DN}(1:2, 1, 2, ii, jj))'; \\
\text{jac} (\text{abs}(\text{jac}) < \text{eps}) &= 0; \\
\text{warning off}; \\
\text{invjac}(:, :, ii, jj) &= \text{jac}^{-1}; \\
\text{warning on}; \\
\text{determinant} (ii, jj) &= \text{abs} (\text{det} (\text{jac})); \\
\text{b} (ii, jj, :) &= \text{b}_\text{func} (\text{surface}, x(ii), y(jj)); \\
&\quad \% [-\text{DN}(2, 1, 1, ii, jj)]; \\
\text{delta} (ii, jj) &= h/(2*\text{norm} (\text{squeeze} (\text{b} (ii, jj, :) )))); \\
\end{align*}
\]

The next 11 lines calculate the derivatives of the basis functions at the Gaussian nodes \( x \) and \( y \).

\[
\begin{align*}
\text{derivsu} &= \text{zeros} (\text{gp}, 3, \text{imax}, \text{jmax}); \\
\text{derivsv} &= \text{zeros} (\text{gp}, 3, \text{imax}, \text{jmax}); \\
&\text{for k=imin:imax} \\
&\quad \text{for l=jmin:jmax} \\
&\quad \quad \% \text{get the 0'th and 1'st derivatives of the} \\
&\quad \quad \quad \quad \text{basisfunction of the surface} \\
&\quad \quad \text{derivsu} (:, :, k, l) = \text{nrbdersonebasisfun} (\ldots) \\
&\quad \quad \quad \quad (\text{surface}.\text{order}(1)-1, \text{surface}.\text{knots}{1}, k, x, 2); \\
&\quad \quad \text{derivsv} (:, :, k, l) = \text{nrbdersonebasisfun} (\ldots) \\
&\quad \quad \quad \quad (\text{surface}.\text{order}(2)-1, \text{surface}.\text{knots}{2}, 1, y, 2); \\
&\quad \end{align*}
\]

Afterwards in a loop over \( kk \) and \( ll \) through the basis functions \( N_{kkll} \) for the test function and over \( k \) and \( l \) through the basis functions \( N_{kll2} \) for the solution the stiffness matrix \( A \) is assembled. First the necessary derivatives are calculated and transformed (lines 5 to 34). Then the single terms according to (6.5) are calculated in lines 35 to 42. Finally they are summed up and written into the right place of \textbf{stima}.

\[
\begin{align*}
&\% \text{loop trough all basisfunctions} (kk, ll) \text{ with } \ldots \text{ intersecting support}
\end{align*}
\]
% for the test function
% for kk = max(imin,2) : min(imax, size_u -1)
% for ll = max(jmin,2) : min(jmax, size_v -1)

for kk = imin :imax
    for ll = jmin : jmax
        index_test = (kk-1)*size_v + ll;

        derivsu_kkl = derivsu(:, :, kk, ll);
        derivsv_kkl = derivsv(:, :, kk, ll);
        N_kkl (: , :) = derivsu_kkl (: , 1) * derivsv_kkl (: , 1)';
        DN_kkl (1 , :, :) = derivsu_kkl (: , 2) * derivsv_kkl (: , 1)';
        DN_kkl (2 , :, :) = derivsu_kkl (: , 1) * derivsv_kkl (: , 2)';

        % loop through the basis functions for the solution
        for k = imin :imax
            for l = jmin : jmax
                index_basis = (k-1)*size_v + l;

                derivsu_kl = derivsu(:, :, k, l);
                derivsv_kl = derivsv(:, :, k, l);
                N_kl (: , :) = derivsu_kl (: , 1) * derivsv_kl (: , 1)';
                DN_kl (1 , :, :) = derivsu_kl (: , 2) * derivsv_kl (: , 1)';
                DN_kl (2 , :, :) = derivsu_kl (: , 1) * derivsv_kl (: , 2)';

                for ii = 1:gp
                    for jj = 1:gp
                        laplace_kl(ii, jj) = laplace(DN(:, :, ii, jj),
                        derivsu, derivsv);

                        invjac_iijj(:, :) = invjac(:, :, ii, jj);
                        DN_klinv(ii, jj, 1:2) = invjac_iijj * DN_kl(1:2, ii, jj);
                        DN_kklinv(ii, jj, 1:2) = invjac_iijj * DN_kkl(1:2, ii, jj);
                    end
                end
            end
        end

        grad_grad(:, :) = epsilon * (DN_klinv(:, : , 1) .* ...
        DN_kklinv(:, : , 1) + ...
        DN_klinv(:, : , 2) .* DN_kklinv(:, : , 2)) ;
        bgrad_function(:, :) = (b(:, : , 1) .* DN_klinv(:, : , 1) + ...
        b(:, : , 2) .* DN_klinv(:, : , 2) ) ;
\[ \text{laplace} \cdot \text{bgrad} (\cdot, \cdot) = -\epsilon \cdot (\text{laplace} \cdot \text{kl} (\cdot, \cdot) + \text{DN} \cdot \text{kkll} \cdot (\cdot, 1)) ; \]

\[ \text{bgrad} \cdot \text{bgrad} (\cdot, \cdot) = ((\text{b} (\cdot, \cdot, 1) \cdot \text{DN} \cdot \text{kl} \cdot (\cdot, \cdot, 1) + \text{b} (\cdot, \cdot, 2) \cdot \text{DN} \cdot \text{kl} \cdot (\cdot, \cdot, 2))) ; \]

\% calculate the transformed integral over the grad product

\[
\text{integral} = (\text{gauss} \cdot \text{weights} \cdot...
\quad ((\text{grad} \cdot \text{grad} + \text{bgrad} \cdot \text{function} + \text{delta} \cdot \ldots
\quad \text{laplace} \cdot \text{bgrad} + \text{bgrad} \cdot \text{bgrad})) \ldots
\quad \text{*determinant} \cdot \text{gauss} \cdot \text{weights}') \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\quad \ldots
\4
\]

The last step is to get the boundary values at \text{dbound} (line 9), calculate the modified RHS \text{\tilde{b}} (line 10), solve the equation \text{A} \cdot \text{x} = \text{rhs} (line 13), and finally rearrange the calculated values into the 3rd dimension of \text{new_surface}, the output of this function.

% initializing freenodes with all possible indices
freenodes=1:size_u*size_v;
% erasing the indices of the boundary
freenodes=setdiff(freenodes,dbound);
% initializing solution vector
values=zeros(size_u*size_v,1);

end

% modified rhs b=-a(w,v)

w_reshape=reshape(squeeze(w.coefs(3,:,:))',1,size_u*size_v);
rhs(freenodes)= rhs(freenodes)-stima(freenodes,dbound)*...
    w_reshape(dbound)';

% calculate the new values by solving the equation at the ...
% free nodes
values(freenodes)=stima(freenodes,freenodes) \ rhs(freenodes)...

new_surface=surface;

% sorting values in the right order
for i=1:size_u
    new_surface.coefs(3,i,1:size_v)=values((i-1)*size_v+(1:...size_v))+...
    (squeeze(w.coefs(3,i,1:size_v)./w.coefs(4,i,1:size_v)));
    new_surface.coefs(3,i,1:size_v)=new_surface.coefs(3,i,1:...
        size_v).*new_surface.coefs(4,i,1:size_v);
end

8.5 laplace.m and gaussparameters.m

The function laplace.m calculates the transformation of $\Delta_{xy}$ to $\Delta_{\xi(1)\xi(2)}$ as described in Section 6.2. Thereby division by zero must be avoided. That is if a function $f$ is independent of $x$, the corresponding terms in the transformation would entail a division by zero. These terms should simply not occur in the calculation from the beginning as $f(x,y) = f(y)$.

gaussparameters.m is a simple switch case procedure that contains the values of nodes and weights for the Gaussian quadrature rule up to order 7. The input is number containing the desired order. The output are the Gaussian nodes $x$ and the Gaussian weights $w$. For example for number = 3 this looks like

switch number
    case 3
        x=[-sqrt(3/5) 0 sqrt(3/5)];
        w=1/9* [5 8 5];

8.6 load_geometry_convection.m

This function takes the string domain as an input. In a switch case procedure it constructs the surface for the corresponding domain. For example for the Hemker problem on $\Omega$ the rectangle and the circle are constructed as one-dimensional NURBS and then
connected with \texttt{nrbruled.m}:

\begin{verbatim}
1 case 'hemker'
  points=zeros(4,8);
  points(:,1)=[-3 0 0 1];
  points(:,2)=[-3 -3 0 1];
  points(:,3)=[0 -3 0 1];
  points(:,4)=[9 -3 0 1];
  points(:,5)=[9 3 0 1];
  points(:,6)=[0 3 0 1];
  points(:,7)=[-3 3 0 1];
  points(:,8)=[-3 0 0 1];
12 rect=nrbmak(points,[0 0 0.125 0.25 0.4 0.6 0.75 0.875 1 ... 1]);
13 circle=nrbcirc(1,0,0,2*pi);
14 circle=nrbtform(circle,vecrotz(pi));
15 surface=nrbruled(rect,circle);
16 surface=nrbdgelev(surface,[0 1]);
17 epsilon=10^(-4);
\end{verbatim}

In the last line the diffusion coefficient $\varepsilon$ is defined.
After the switch case procedure the function handles corresponding to \texttt{domain} are set.

\begin{verbatim}
1 get_boundary=@(x)boundary_points(domain,x);
2 b=@(x,y,z)convection_field(domain,x,y,z);
3 h=@(x)h_function(domain,x);
4 w=@(x,y,z)boundfunc_conv(domain,x,y,z);
\end{verbatim}

These functions are discussed in the following section.

\textbf{8.7 boundary_points.m, boundfunc_conv.m, convection_field.m, h_function.m}

These four functions contain the parameters and data of the example problems. They mainly consist of a switch case procedure to determine the current example via the parameter \texttt{domain}.
\texttt{boundary_points.m} determines the points belonging to the different boundaries. For example for the rectangle the boundary is:

\begin{verbatim}
% getting length of knot vector
1 size_u=surface.number(1);
2 size_v=surface.number(2);
\end{verbatim}
switch domain
  case { 'rectangle−45', 'rectangle−atan2' }
    % saving the indices of boundary points, if \( P_{ij} \) is a boundary point the
    % value \((i−1)\cdot\text{size}_v+j\) is saved
    % boundary points have \( i=0 \) or \( i=\text{size}_u \) or \( j=0 \) or \( j=\text{size}_v \)
    dbound=\text{zeros}(2*(\text{size}_u+\text{size}_v−2),1);
    dbound(1:MATLAB_RUNNING_OCTAVE\text{size}_v )=[1:MATLAB_RUNNING_OCTAVE\text{size}_v ];
    dbound(\text{size}_v+1:2*\text{size}_v )=(\text{size}_u−1)\cdot\text{size}_v +[1:MATLAB_RUNNING_OCTAVE\text{size}_v ];
    dbound(2*\text{size}_v+1:2*\text{size}_v+\text{size}_u−2)=\text{size}_v *[1:MATLAB_RUNNING_OCTAVE\text{size}_u−2]+1;
    dbound(2*\text{size}_v+\text{size}_u−1:2*(\text{size}_v+\text{size}_u )−4)=[2:MATLAB_RUNNING_OCTAVE\text{size}_u−1]*\text{size}_v ;
    nbound=[];
  end

\text{size}_u and \text{size}_v are the numbers of control points in the \( \xi^{(1)} \) and \( \xi^{(2)} \) direction. Then for the index \( ij \) of a boundary point \( P_{ij} \) the value \((i−1)\cdot\text{size}_v+j\) is saved. Since the whole boundary is a Dirichlet boundary, \text{dbound} are the points where either \( i=0 \), \( j=0 \), \( i=\text{size}_u \) or \( j=\text{size}_v \). The boundary points are calculated similarly in the other cases. \text{boundfunc.conv.m} calculates the boundary function. It takes as additional input \text{surface} and the length of the knot vectors in \( \xi^{(1)} \) and \( \xi^{(2)} \) direction, \text{size}_u and \text{size}_v, respectively. Then it starts by constructing a surface \text{ubound} with the same \( x \) and \( y \) values as \text{surface} but zero as \( z \) component. This third coordinate is then changed according to \text{ubD}. For the sine hill example II this means, that the values at the first and last knot in \( \xi^{(1)} \) direction are set to \( \sin(\pi x) \), see lines 11 and 9, respectively.

\begin{verbatim}
ubound=\text{surface};
ubound.coevs(3,:,:)=\text{zeros}(\text{size}_u,\text{size}_v);

switch domain
  case 'circle'
    %sine hill
    for i=1:\text{size}_v
      ubound.coevs(3,ubound.number(1),i)=ubound.coevs(4,...
        ubound.number(1),i)\ast...
      \text{sin}(\pi*ubound.coevs(1,ubound.number(1),i)/ubound....
        coefs(4,ubound.number(1),i));
      ubound.coevs(3,1,i)=ubound.coevs(4,1,i)\ast...
      \text{sin}(\pi*ubound.coevs(1,1,i)/ubound.coevs(4,1,1))
    end
\end{verbatim}
The other cases are implemented similarly. Finally, h_func.m calculates the size $h$ used for the stabilization parameter $\delta$. To be optimal this should be the mesh size in the direction of the convection $b$. Furthermore it takes the four corner points of an element $Q$ as an input. The output is $\tilde{h}$ as described in Section 6.

```matlab
function h=h_function(domain,points)
    switch domain
    case 'rectangle−45'
        theta=pi/4;
        h=(points(1,2,1)−points(1,1,1))/max(cos(theta),sin(theta));
    case 'rectangle−atan2'
        theta=atan(2);
        h=(points(1,2,1)−points(1,1,1))/max(cos(theta),sin(theta));
    case 'circle'
        h=sqrt(((min(min(points(2,:,:)))-max(max(points(2,:,:))))^2+
                    (min(min(points(1,:,:)))-max(max(points(1,:,:))))^2);
    case {'hemker','hemker−x<4','hemker−y>0'}
        h=abs(min(min(points(1,:,:))−max(max(points(1,:,:)))));
    otherwise
        error(['geometry ' domain ' is not defined!']);
    end
end
```

convection_field.m calculates the convection field vector $b$ at a given point $(\xi^{(1)},\xi^{(2)})$ of the parametric domain. Thus the surface and the coordinates $x$ and $y$ are the input. The most difficult case is Example II as it is the only one where $b$ is not constant. It calculates the value $(x,y,z)^T = S\left(\xi^{(1)},\xi^{(2)}\right)$ in lines 6 to 11 as described in Section 8.4. Then it returns the value $b = (-y,x)^T$.

```matlab
function b=convection_field(domain,surface,x,y)
    switch domain
    case 'circle'
        size_u=surface.number(1);
        size_v=surface.number(2);
        Aders=nrbsurcederivs(size_u−1,surface.order(1)−1,...
                            surface.knots{1},...
                            size_v−1,surface.order(2)−1,surface.knots{2},...
                            surface.coefs(1:3,:,:),x,y,0);
```
wders = nrbbsurfsderivs ( size_u -1, surface . order ( 1 ) -1, ... 
    surface . knots [ 1 ], ... 
    size_v -1, surface . order ( 2 ) -1, surface . knots [ 2 ], ... 
    surface . coefs ( 4 : : : ) , x , y , 0 ) ;
DN ( : : : ) = nrbbratsurfsderivs ( Aders , squeeze ( wders ) , 0 ) ;
b = [ -DN ( 2 , 1 , 1 ) DN ( 1 , 1 , 1 ) ] ;

8.8 nrbplotcontrol.m and nrbplotmesh.m

These functions are tools to visualize the control polygon and the mesh. nrbplotcontrol 
takes a NURBS surface and its boundary as an input. It starts by extracting the real 
coordinates and the weight from the homogeneous coordinates.

function nrbplotcontrol ( srf , boundary )

    w ( : : : ) = srf . coefs ( 4 : : : ) ;

Afterwards it goes through all points \( P_{ij} \) of the control net, plots the value as a red 
diamond and the control polygon as linear interpolation between the points with the same \( i \)-value. It continues by plotting the control net in the other direction, that is 
connecting all points with the same \( j \)-value.

for i = 1 : size ( p , 3 )

    % plotting the control points
    plot3 ( p ( 1 : : : , i ) , p ( 2 : : : , i ) , p ( 3 : : : , i ) , ’ dr ’ , ’LineWidth’ , 2 , ’ ... 
        MarkerSize’ , 10 ) ;
    % plotting the control polygon
end

for i = 1 : size ( p , 2 )

    % plotting the control polygon
    plot3 ( squeeze ( p ( 1 , i : : : ) ) , squeeze ( p ( 2 , i : : : ) ) , squeeze ( p ( 3 , i : : : ) ) , ’ − r ’ , ’LineWidth’ , 1 ) ;
end

If the boundary points are given, the function plots them in green instead of red.
if nargin>1
    size_v=srf.number(2);
    for k=1:size(boundary,1)
        % converting index to (i,j)
        j=mod(boundary(k),size_v);
        if j==0
            j=size_v;
        end
        i=(boundary(k)-j)/size_v+1;
        plot(p(1,i,j),p(2,i,j),’dg’, ’LineWidth’, 2,’MarkerSize’... ,10);
    end
end

To plot the mesh the function **nrbplotmesh.m** evaluates the isoparametric lines of the given NURBS at the knots. These isoparametric lines are then plotted in black.

```matlab
function nrbplotmesh(srf)
    uknots=unique(srf.knots{1});
    vknots=unique(srf.knots{2});
    for i=1:length(uknots)
        q=nrbeval(srf,{uknots(i),linspace(0.0,1.0,50)});
        plot3(q(1,:),q(2,:),q(3,:),’–k’, ’MarkerSize’,10);
    end
    for i=1:length(vknots)
        q=nrbeval(srf,{linspace(0.0,1.0,50),vknots(i)});
        plot3(q(1,:),q(2,:),q(3,:),’–k’);
    end
end
```

8.9 **nrbonebasisfun.m, nrbdersonebasisfun.m, nrbsdersbasisfuns.m, nrbsurfacederivs.m and nrbratsurfacederivs.m**

These functions are taken from ‘The NURBS book’ [PT97]. As they are not provided by the nurbs-toolbox I wrote them myself. They follow the C-pseudo code from the book. One problem was, that arrays start with the index 1 in matlab while they start at 0 in C. Therefore some indices had to be shifted.
9 Summary and Outlook

The aim of this thesis was to apply isogeometric analysis for the Hemker problem.

Therefore introductions into non-uniform rational B-Splines and finite element methods were given.

With the help of [BBdVC*06] a local and a global a-priori error estimate for IGA were developed. In addition an inverse inequality was proven. Hence it is reasonable to choose NURBS as basis functions for solving PDEs numerically.

A solver on the basis of IGA for convection-diffusion equations was implemented and tested.

The goals of this thesis were

- to present the main ideas and the numerical analysis of IGA
- to implement IGA for two-dimensional problems
- to apply IGA in particular to scalar convection-diffusion problems.

The numerical studies show that the application of IGA in combination with SUPG to scalar convection-diffusion problems leads to good approximations of the solution of these problems. The implementation in this thesis leads to results similar to [HCB05]. In addition it emphasizes that circular boundaries are represented exactly in the coarsest mesh and that no further communication with the exact design is necessary during the refinement process.

The most interesting example was the Hemker Problem. Comparison with the solutions obtained by FEM in [ACF+11], the IGA results are equal with respect to layer width and even better with respect to undershoots. Hence IGA is a competitive alternative to FEM. Similarly to FEM it would be interesting to study the behavior of IGA for convection-diffusion equations with other stabilization methods than SUPG.

On the other hand this thesis shows some difficulties with IGA on the basis of NURBS. The parametrization has to be continuous. Therefore the representation of domains that are not simply connected becomes difficult. An approach to deal with this was made by Bazilevs et al. in [BCC+10]. They use generalized splines called T-Splines. These T-splines allow strictly local mesh refinement. Therefore they provide a possibility to implement adaptive algorithms. The development of an analogon to AFEM is one possibility to continue the work done in this thesis.
References


Affidavit

Hereby I certify that I have written this Diploma Thesis independently and that I have used no other aid than the specified recourses and tools. Furthermore this thesis was not handed in or published in this or a similar form before.

Berlin May 8, 2013

Liesel Schumacher
Statements

Non-uniform rational B-Splines are introduced with their major properties.

They provide a possibilities to represent circular shapes exactly.

Three refinement methods that preserve the shape of the coarsest mesh are introduced. One of them, k-refinement, has no equivalent in finite element methods.

Isogeometric analysis on the basis of non-uniform rational B-splines satisfies an a-priori error estimate and it is therefore a reasonable tool for the numerical solution of partial differential equations.

The application to scalar convection-diffusion equations leads to promising results.

The comparison for the Hemker problem shows competitive results with isogeometric analysis then with finite element methods, using streamline-upwind-Petrov-Galerkin stabilization.