NTNU
Norwegian University of
Science and Technology

Isogeometric Collocation

Numerical solution of differential equations

Pål Oskar Engen

Master of Science in Physics and Mathematics
Submission date: March 2016
Supervisor: Trond Kvamsdal, MATH
Co-supervisor: Kjetil Andrè Johannessen, MATH

Norwegian University of Science and Technology
Department of Mathematical Sciences

NTNU – Trondheim
Norwegian University of
Science and Technology

Isogeometric Collocation

Pål Oskar Engen

March 20, 2016

Master Thesis

Department of Mathematical Sciences

Norwegian University of Science and Technology

Supervisor 1: Trond Kvamsdal

Supervisor 2: Kjetil Andrè Johannessen

Abstract

This thesis present the theory of Lagrange polynomials and the implementation of a Lagrange interpolation method. We also study spline theory and the implementation of splines intended on geometric representation and interpolation methods. Last but not least we explore and implement a one-dimensional numerical collocation method for solving differential equations numerically, using splines. Along the way we discuss and analyze some of the main issues related to splines, interpolation and collocation.

Sammendrag

Denne masteroppgaven presenterer teorien rundt Lagrange-polynomer og implementasjonen av Lagrange-interpolasjon. Vi studerer også teorien rundt spline-funksjoner og implementasjonen av splines tilknyttet interpolasjon, samt representasjon av geometriske objekter. Sist men ikke minst så utforsker vi og implementerer en endimensjonal kollokasjonsmetode for å løse differensialligninger numerisk, ved bruk av splines. Underveis vil vi diskutere og analysere kjente problemstillinger tilknyttet splines, interpolasjon og kollokasjon.

Preface

This master thesis completes my master's degree in Industrial Mathematics at the Norwegian University of Science and Technology (NTNU). The degree includes a minimum of 300 credits, where 30 of these credits are from this thesis. The work I present was written in the period October 2015 - March 2016. I have been supervised by Professor Trond Kvamsdal and Post. doc. Kjetil André Johannessen from the Department of Mathematical Sciences at NTNU, and I want to express my gratitude for their assistance and advice during this period.

Contents

Nomenclature

BVP	Boundary Value Problem.
CAD	Computer Aided Design.
FEA	Finite Element Analysis.
IGA	Isogeometric Analysis.
$\mathbb{P}$	Space of polynomials.
$\mathbb{R}$	Space of real numbers.
p, q	Polynomial degrees.
n, m	Degrees of freedom.
h	Length of a grid interval.
i, j, k, l	Index variables.
L_k	Lagrange basis function.
$H_\alpha(x)$	One dimensional step function.
$H_\alpha(x, y)$	Two dimensional step function.

$\hat{\Omega}$	Parameter space.
ξ, η	Coordinates in parameter space.
Ω	Physical space.
x, y	Coordinates in physical space.
$\mathcal{F}$	Geometrical map.
$S(\xi, \eta)$	Spline surface map.
$C(\xi)$	Spline curve map.
$x_L(\xi)$	One dimensional linear map.
$\Xi, \mathcal{H}$	Knot vectors.
ξ_i, η_j	Elements of knot vectors.
$M_{j,q}(\eta)$	B-spline basis function.
$N_{i,p}(\xi)$	B-spline basis function.
$\mathbb{S}_{p,\Xi}$	Spline space.
$\mathbb{S}_{q,\mathcal{H}}$	Spline space.
$\hat{\xi}_i, \hat{\eta}_j$	Interpolation points.
$\hat{\tau}_i$	Collocation points.
$u_h(x)$	Numerical solution.
$u(x)$	Exact solution.
$\mathcal{L}, \mathcal{G}$	Linear differential operators.

Chapter 1

Introduction

Spline basis functions have been a mainstay of Computer Aided Design (CAD) for many years due to their flexibility and precision. [21] and [8] has done great work in uniting the field of CAD and the field of Finite Element Analysis (FEA). They introduced the concept of Isogeometric Analysis (IGA) and brought splines into analysis, an area to which their unique properties are also ideally suited.

The collocation method is a numerical method for solving partial differential equations. The numerical solutions in this method are built from basis functions such as Lagrange or Chebyshev polynomials, or standard Finite Element basis functions. [17] has explored the collocation method using Radial Basis Functions (RBF). [13] initiated research on Isogeometric Collocation methods where the solution space was built on spline basis functions.

This thesis aims towards exploring the world of splines. We look at methods of representing geometrical objects using these functions. We also explore interpolation methods and collocation methods, in particular by seeking solutions that are built from spline basis functions.

Chapter 2

Polynomial Interpolation

Polynomial interpolation involves finding a polynomial that agrees exactly with some information that we have about a real-valued function f of a single variable x. This information may be in the form of values $f(x_0),\ldots,f(x_p)$ of the function f evaluated at some finite set of $p+1$ points $x_0,\ldots,x_p$ on the real line.

2.1 Interpolation using the Vandermonde Matrix

Given that p is a non-negative integer, let $\mathbb{P}_p$ denote the set of all polynomials of degree less than or equal to p defined over the set $\mathbb{R}$ of real numbers. A polynomial $r_p(x)$ can be written in its general form as

$$r_p(x) = a_p x^p + a_{p-1} x^{p-1} + \ldots + a_2 x^2 + a_1 x + a_0. \tag{2.1}$$

One can obtain a polynomial $r_p(x)$ passing through the points $(x_i, y_i) \in \mathbb{R}^2$ for $i = 0, 1, \ldots, p$ by solving the equations $r_p(x_i) = y_i$ for $i = 0, 1, \ldots, p$. The linear system of equations to be solved is

$$a_p x_i^p + a_{p-1} x_i^{p-1} + \ldots + a_2 x_i^2 + a_1 x_i + a_0 = y_i \quad \forall \quad i = 0, 1, \ldots, p. \tag{2.2}$$

The coefficients $a_i \in \mathbb{R}$, $i = 0, 1, \ldots, p$ are the unknowns, and the coefficient matrix

$$V = \begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \\ 1 & x_2 & x_2^2 & \ldots & x_2^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_p & x_p^2 & \ldots & x_p^p \end{bmatrix}, \tag{2.3}$$

is known as the *Vandermonde matrix.* The solution is then sought in the *Monomial basis* $\{1, x, x^2, \ldots, x^p\}$.

Theorem 1. *Suppose that x_i, $i = 0, 1, \ldots, p$ are distinct real numbers and y_i, $i = 0, 1, \ldots, p$ are real numbers. Then there exists a unique polynomial $r_p(x) \in \mathbb{P}_p$ such that $r_p(x_i) = y_i$ for all $i = 0, 1, \ldots, p$.*

Proof. The proof consists of showing that the corresponding Vandermonde matrix has a non-zero determinant and hence is nonsingular, implying that r_p exists and is unique. See the *Unisolvence Theorem* [1, p. 31-32] or [2, p. 2-3] for complete proofs. □

This method is the most basic procedure of finding an interpolating polynomial. However, the method is numerically unstable for large p, while it also is very costly to solve a dense $(p+1) \times (p+1)$ linear system.

2.2 Lagrange Interpolation

The Lagrange interpolation method is an alternative way to define r_p without having to solve computationally expensive systems of equations.

Definition 1. ***Lagrange interpolation polynomial.*** *Let $p \geq 0$. Given the real-valued function f, defined and continuous on a closed real interval $[a,b]$, and the distinct interpolation points $x_i \in [a,b]$, $i = 0,\ldots,p$, the polynomial r_p defined by*

$$r_p(x) = \sum_{k=0}^{p} L_k(x) f(x_k) \tag{2.4}$$

$$L_k(x) = \prod_{\substack{i=0 \\ i\neq k}}^{p} \frac{x - x_i}{x_k - x_i} \tag{2.5}$$

is the Lagrange interpolation polynomial of degree p with interpolation points x_i, $i = 0,\ldots,p$, for the function f.

It is proved by [3, p. 181-182] that the Lagrange interpolation polynomial (2.4) is the unique polynomial satisfying theorem (1). Therefore, we can conclude that the Lagrange basis $\{L_0, L_1, L_2, \ldots, L_p\}$ and the Monomial basis both span the same space of polynomials $\mathbb{P}_p$, and the two methods will obtain the same interpolating polynomial. The relationship between the Lagrange basis and the Monomial basis are further investigated by [4].

From (2.5) it follows that all Lagrange basis functions are constructed such that

$$L_k(x_i) = \begin{cases} 1, & \text{if } i = k, \\ 0, & \text{if } i \neq k, \end{cases} \tag{2.6}$$

for all $i, k = 0, 1, \ldots, p$.

Example 1. *The Lagrange basis functions $L_k(x)$, $k = 0,\ldots,4$ are shown in figure 2.1. One can verify the conditions (2.6) on all points marked by black dots.*

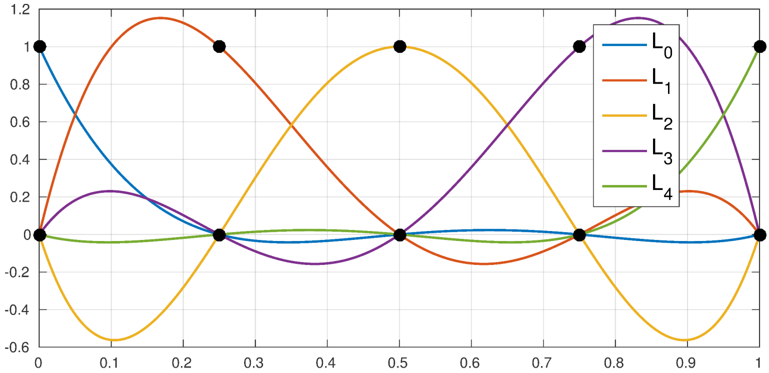


Figure 2.1: The Lagrange basis $\{L_0, L_1, L_2, L_3, L_4\}$ defined in equation (2.5) is plotted for $x \in [0,1]$ using interpolation points $x = \{0, 0.25, 0.5, 0.75, 1\}$.

Example 2. *$f(x) = \sin(2\pi x)$ is interpolated for $x \in [0,1]$. The uniform grid space $h = 0.25$ is such that $x_i = i \cdot h$ for $i = 0, \ldots, 4$. Hence, the interpolation points are given as $(x_i, f(x_i))$ for $i = 0, \ldots, 4$. The interpolating Lagrange polynomial r_4 is therefore a linear combination of the Lagrange basis functions shown in figure 2.1. The exact function $f(x)$ and the interpolating function r_4 can be seen in figure 2.2.*

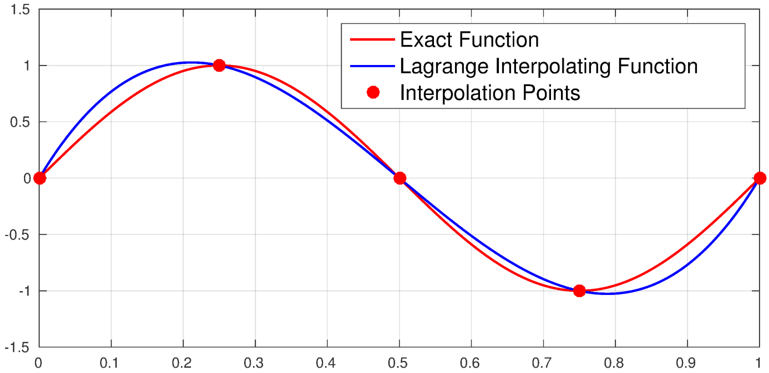


Figure 2.2: Polynomial interpolation of $f(x) = \sin(2\pi x)$ for $x \in [0,1]$. We have used equally spaced interpolation points. The interpolating Lagrange polynomial is of degree $p = 4$.

2.3 Runge's Phenomenon

In general we expect that the Lagrange interpolation polynomial becomes more accurate as the polynomial degree p increases. But this is not the case for all functions.

Example 3. *In figure 2.3 we have interpolated the discontinuous step function $H_a(x)$ defined by*

$$H_a(x) = \begin{cases} 1 & \textit{if } x < a, \\ \frac{1}{2} & \textit{if } x = a, \\ 0 & \textit{if } x > a, \end{cases} \tag{2.7}$$

for $a = 0.5$. The interpolation points are chosen to be uniformly distributed along the x-axis. This has been done for different polynomial degrees $p = 6, 8, 10, 12$ and as we can see in figure 2.3, the interpolating polynomials tend to blow up as p increases.

This is an example of what is referred to as the *Runge's Phenomenon.* It was first discovered by interpolating the function $f(x) = 1/(1+x^2)$ on $x \in [-5, 5]$ using equidistant interpolation points [5]. The latter example is further discussed in [3, 186-187] and the complex analysis of the problem, explaining its diverging properties, is discussed in [6].

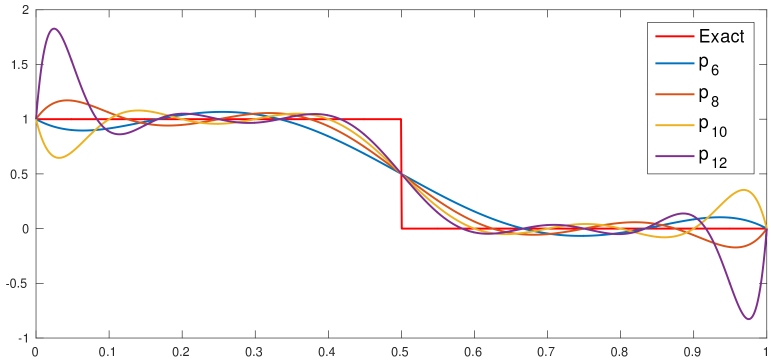


Figure 2.3: Lagrange interpolation for the discontinuous step function. The continuous curves are the associated Lagrange interpolation polynomials r_p for $p = 6, 8, 10, 12$ using equally spaced interpolation points. This phenomena, where the approximated function blow up for large p, is called Runge's Phenomenon.

Example 4. *$H_{0.5}(x)$ is now interpolated for polynomial degrees $p = 6, 8, 10, 12$ on a non-uniform set of interpolation points x_i, namely the points defined by*

$$x_i = \frac{\arctan(2\pi(\tilde{x}_i - 0.5))}{2\arctan(\pi)} + 0.5, \tag{2.8}$$

where $\tilde{x}_i$, $i = 1, \ldots, p$ are the p uniform distributed points on the interval $[0,1]$. As we can see in figure 2.4 the interpolating polynomials r_p now converge as p increases.

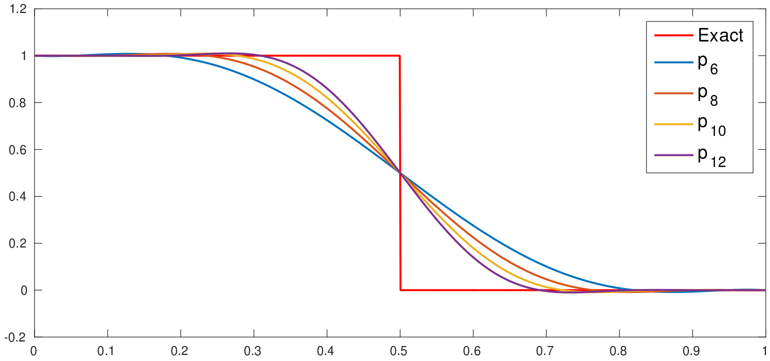


Figure 2.4: Lagrange interpolation for the discontinous step function. The function is interpolated by Lagrange polynomials of degree $p = 6, 8, 10, 12$ using the points defined by (2.8) as interpolation points.

The non-uniform points (2.8) are more densely distributed towards the ends of the domain, the same area as we observe the largest errors in figure (2.3). These examples motivates further investigation of interpolation points, and this will be discussed in later chapters.

Chapter 3

Spline Theory

The Lagrange interpolation polynomial is global in nature, meaning that the approximated function is defined by the same analytical expression on the whole interval. An alternative and more flexible way to interpolate data is to divide the interval into several subintervals and look for piecewise polynomial approximations of a lower degree. Such piecewise polynomials are called *splines* and the endpoints of the subintervals are called *knots*. Splines are built from *B-splines* (basis splines) so a discussion of these functions, and their properties, is a natural starting point. But first we introduce the knot vector.

3.1 Knot Vectors

A *knot vector* in one dimension is a non-decreasing set of coordinates, written $\Xi = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$ where $\xi_i \in \mathbb{R}$ is the i^{th} knot, p is the polynomial order, n is the number of basis functions defined on our domain and i is the knot index $i = 1, 2, \ldots, n+p+1$. The knot vector is used to describe element boundaries. When constructing the B-spline basis we define separate polynomials on each element $[\xi_i, \xi_{i+1}]$. Knot vectors may be uniform if the knots are equally spaced, or non-uniform if the knots are unequally spaced, for example in presence of a repeated knot value. If a knot is repeated m times in the knot vector, we say that the knot has *multiplicity* m. Knots of multiplicity one, two or three are also called *simple, double* and *triple* knots. A knot vector is said to be *open* if its first and last knot values has multiplicity $p+1$. If in addition no knot value occurs with a higher multiplicity then $p+1$, then the knot vector is said to be $p+1$-*regular.*

3.2 B-splines

Definition 2. ***B-spline basis function.*** *Given a knot vector* $\Xi = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$*, where* $\xi_i \in \mathbb{R}$ *is the* i^{th} *knot,* i *is the knot index,* p *is the polynomial order and* n *is the number of basis functions used to construct the basis. The B-spline basis functions are defined recursively starting with piecewise constants (*$p = 0$*):*

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

For $p = 1, 2, 3, \ldots,$ *they are defined by*

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi). \tag{3.2}$$

A characteristic feature of B-splines is their smoothness properties. The multiplicity m_i of knot value ξ_i has important implication on smoothness. In general, basis functions of order p has $p - m_i$ continuous derivatives across knot ξ_i, as stated by [7, p. 26-27]. The first and last value of open knot vectors are repeated $p+1$ times, meaning that B-splines constructed from open knot vectors are discontinuous (C^{-1}-continuous) at the endpoints of the domain.

Example 5. *In figure 3.1 we have demonstrated the continuity property of the B-splines using* 3^{rd} *degree basis functions. The B-splines are discontinuous at the endpoints, since we used an open knot vector. The B-splines have the maximum level of continuity, i.e.* C^2*-continuity, across the simple knot value* $\xi = 1$*. The knot values* $\xi = 2$ *and* $\xi = 3$ *are repeated two and three times, respectively. Therefore, the basis functions across these knots are only* C^1*- and* C^0*-continuous, respectively. A knot vector of length* 14 *implies a number of basis functions* $n = 14 - p - 1 = 10$*, which are ordered from left to right using the index* j*. Hence, in figure 3.1 we look at the set of basis functions* $\left\{N_{j,3}(\xi)\right\}_{j=1}^{10}$*.*

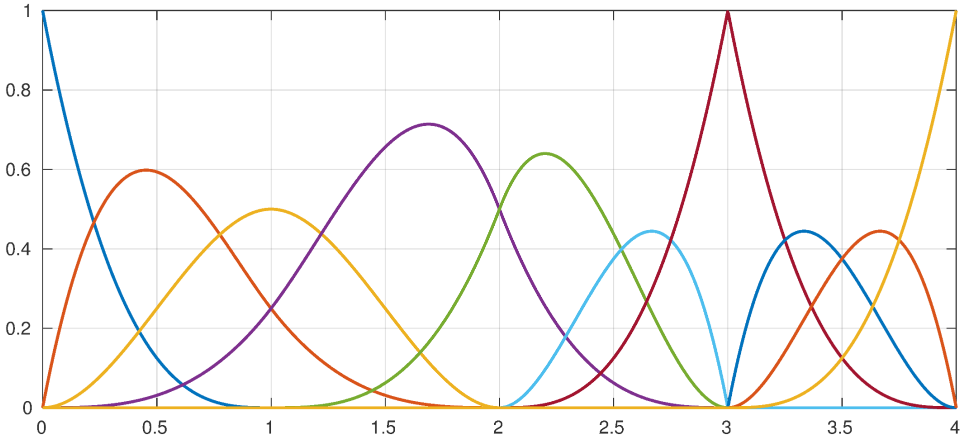


Figure 3.1: Cubic B-spline basis functions calculated from the knot vector $\Xi = [0,0,0,0,1,2,2,3,3,3,4,4,4,4]$. Some knot values are repeated, and the continuity properties of each B-spline basis function are depending on the multiplicity of each knot.

In general, if there is one basis function $N_{j,p}(\xi)$ for some $j \in \{1,\ldots,n\}$ such that $N_{j,p}(\xi_i) = 1$ for some $i = l$, where $l \in \left\{1,\ldots,n+p+1\right\}$ and $N_{j,p}(\xi_i) = 0$ for all other $i \in \left\{1,\ldots,n+p+1\right\}$, then we can say that the basis $\left\{N_{j,p}(\xi)\right\}_{j=1}^{n}$ is *interpolatory* at ξ_i, and it is the function $N_{j,p}(\xi)$ which interpolates at ξ_i. That is, if one basis function has value 1 at ξ_i and all other basis functions are zero at ξ_i, then the basis is interpolatory on ξ_i. This property will prove to be helpful in collocation methods when imposing boundary conditions.

3.3 Derivatives of B-splines

The derivatives of B-spline functions are efficiently represented in terms of B-spline lower order bases. For a given polynomial order p and knot vector Ξ, the k^{th} derivative of the i^{th} basis function is given by

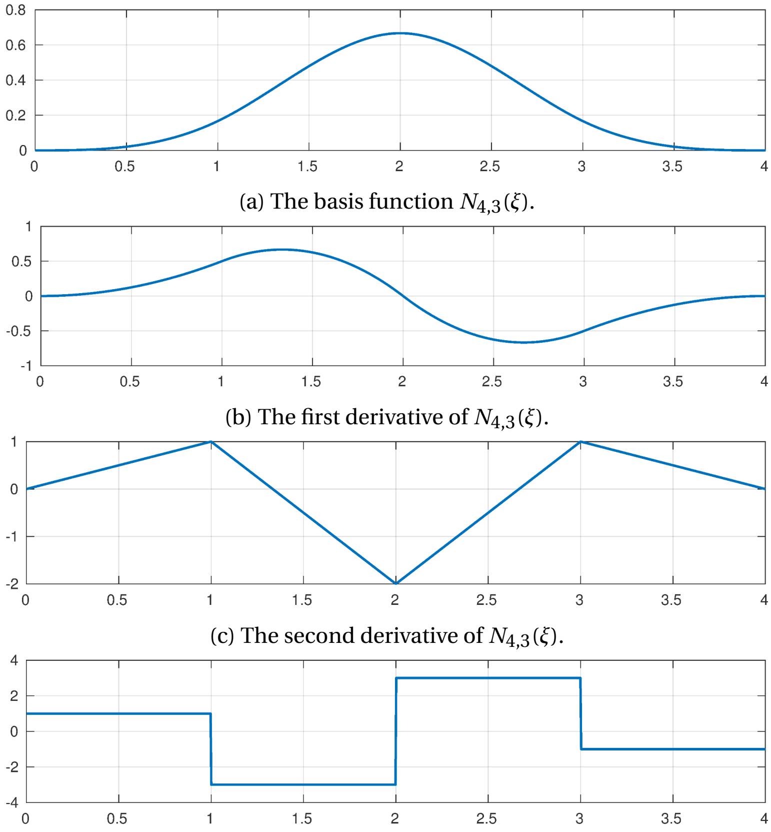
$$\frac{d^k}{d^k\xi}N_{i,p}(\xi) = \frac{p!}{(p-k)!}\sum_{j=0}^{k}\alpha_{k,j}N_{i+j,p-k}(\xi), \tag{3.3}$$

with

$$\begin{aligned}
\alpha_{0,0} &= 1,\\
\alpha_{k,0} &= \frac{\alpha_{k-1,0}}{\xi_{i+p-k+1}-\xi_i},\\
\alpha_{k,j} &= \frac{\alpha_{k-1,j}-\alpha_{k-1,k-1}}{\xi_{i+p+j-k+1}-\xi_{i+j}} \qquad j=1,\dots,k-1,\\
\alpha_{k,k} &= \frac{-\alpha_{k-1,k-1}}{\xi_{i+p+1}-\xi_{i+k}}.
\end{aligned}$$

Some of the denominators of these coefficients can be zero in presence of repeated knots. The coefficients is defined to be zero when this happens.

Example 6. *The continuity property can also be interpreted in terms of the number of continuous derivatives across repeated knots. Figure 3.2, 3.3 and 3.4 shows a* 3^{rd} *order B-spline and its* k^{th} *derivatives for* $k = 1,\dots,3$. *In each figure we can observe that the basis function* $N_{4,3}(\xi)$ *has* $p-m$ *continuous derivatives across the knot* $\xi = 2$, *where the multiplicity* m *is different in each figure.*



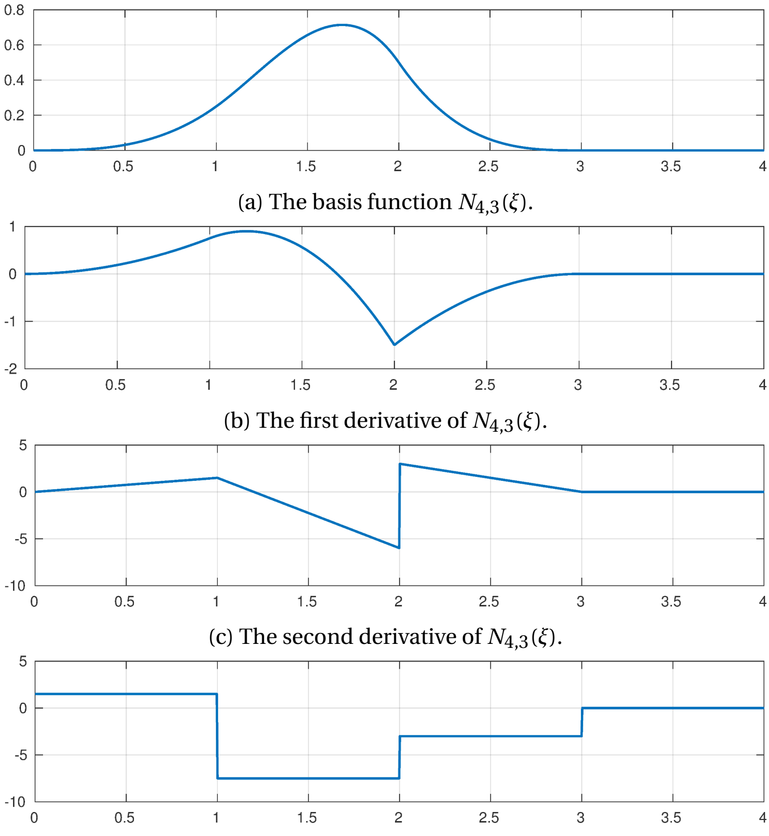
(a) The basis function $N_{4,3}(\xi)$.

(b) The first derivative of $N_{4,3}(\xi)$.

(c) The second derivative of $N_{4,3}(\xi)$.

(d) The third derivative of $N_{4,3}(\xi)$.

Figure 3.2: The B-spline $N_{4,3}(\xi)$ and its derivatives for the open knot vector $\Xi = [0,0,0,0,1,2,3,4,4,4,4]$. Each interior knot has multiplicity one and hence the basis function $N_{4,3}(\xi)$ has $p-1=2$ continuous derivatives.



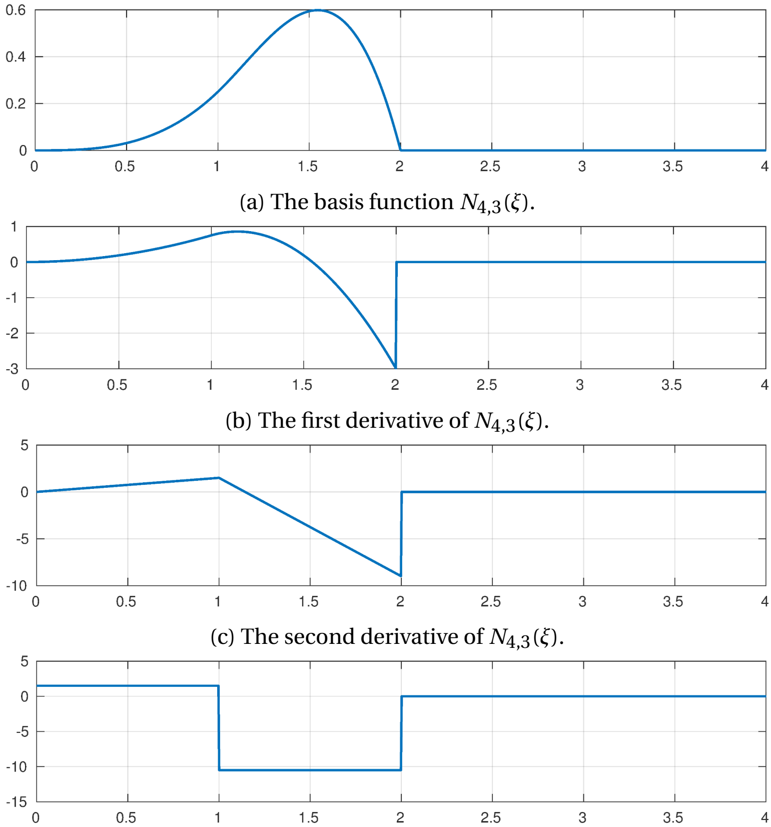
(a) The basis function $N_{4,3}(\xi)$.

(b) The first derivative of $N_{4,3}(\xi)$.

(c) The second derivative of $N_{4,3}(\xi)$.

(d) The third derivative of $N_{4,3}(\xi)$.

Figure 3.3: The B-spline $N_{4,3}(\xi)$ and its derivatives for the open knot vector $\Xi = [0,0,0,0,1,2,2,3,4,4,4,4]$. The knot value $\xi = 2$ is repeated two times, hence $N_{4,3}(\xi)$ have one continuous derivative across the repeated knot.



(a) The basis function $N_{4,3}(\xi)$.

(b) The first derivative of $N_{4,3}(\xi)$.

(c) The second derivative of $N_{4,3}(\xi)$.

(d) The third derivative of $N_{4,3}(\xi)$.

Figure 3.4: The B-spline $N_{4,3}(\xi)$ and its derivatives for the open knot vector $\Xi = [0,0,0,0,1,2,2,2,3,4,4,4,4]$. The knot value $\xi = 2$ is repeated three times, hence $N_{4,3}(\xi)$ have zero continuous derivatives across the repeated knot.

3.4 Properties of B-splines

We have already mentioned some important properties of B-splines, but there are several other important ones. The following lemma sums up some of the most basic properties found in [8] and [7].

Lemma 1. *Let p be a non-negative polynomial degree and let $\Xi = (\xi_j)$ be a knot sequence. The B-splines on Ξ have the following properties:*

1. *Each basis function is pointwise non-negative over the entire domain, i.e. $N_{i,p}(\xi) \geq 0 \; \forall \; \xi$.*

2. *The basis constitutes a partition of unity, i.e. $\sum_{i=1}^{n} N_{i,p}(\xi) = 1 \quad \forall \; \xi$, given that the knot vector is $p+1$-regular.*

3. *A p^{th} order B-spline has $p-m$ continuous derivatives across the knot with multiplicity m.*

4. *The support of a B-spline of order p is always $p+1$ knot spans, meaning that a basis function $N_{i,p}(\xi)$ only has non-zero values in the interval $[\xi_i, \xi_{i+p+1}]$.*

5. *Each B-spline function shares support (overlaps with) with $2p+1$ B-splines, including itself.*

6. *The j^{th} B-spline $N_{j,p}$ depends only on the knots $\xi_j, \xi_{j+1}, \ldots, \xi_{j+p+1}$.*

Chapter 4

Spline Geometry

The next topic is to present how linear combinations of B-splines can be applied to create different spline geometries, such as *spline functions*, *spline curves* and *spline surfaces*.

We can visualize spline geometries in two different coordinate systems, in what we call the *parameter space* on the one hand, and the *physical space* on the other. The parameter space in two dimensions is the rectangular space spanned by the coordinates ξ and η. The physical space is spanned by the usual coordinates x and y, and is what often represents the real world. An object in the physical space is a representation of the true geometry which is approximated by splines. The physical space is denoted Ω. In this thesis we only consider structured grids. However, one can be very creative with the physical space, and the objects living here can be quite complex. The parameter space is denoted by $\hat{\Omega}$. This space has more order to it, at least visually. The grid in $\hat{\Omega}$ is defined by the tensor product of the knot vectors Ξ and $\mathcal{H}$, so the parameter space domain is $\hat{\Omega} = [\xi_1, \xi_{n+p+1}] \times [\eta_1, \eta_{m+q+1}]$. The extent of this space is therefore limited by the non-decreasing values of the knot vectors.

We recognize that the parameter space and the physical space have the same topology, and that the one is merely a continuous deformation of the other, using transformations such as stretching and bending. Objects in the parameter space can be mapped into physical space by a *geometrical map*, and we denote it by $\mathcal{F}:\hat{\Omega}\to\Omega$. Any object in parameter space has to be kind of trivial. A curve is simply a straight line, and a surface is simply a flat rectangle. It is easier to do calculations and analysis in parameter space because of its trivial structure. The geometrical map helps us transform the result into physical space. These tools can help us to do analysis on complex geometries, for example to solve *boundary value problems* (BVP). An illustration of the two spaces can be seen in figure 4.1.

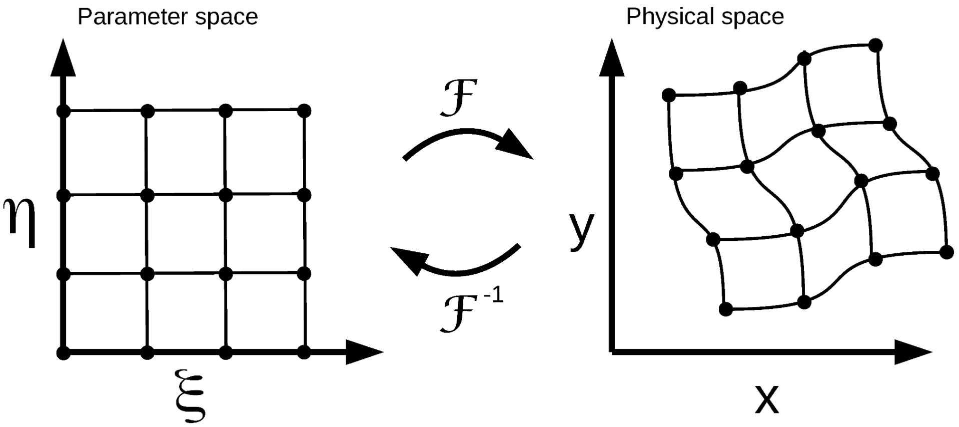


Figure 4.1: Parameter space $\hat{\Omega}$ (left) and physical space Ω (right) and the geometrical map $\mathcal{F}:\hat{\Omega}\to\Omega$ between them.

4.1 Spline Curves

Spline curves are linear combinations of B-splines. First we define the space where these curves live, namely the *spline space.*

Definition 3. ***Spline space.*** *Let* $\Xi = (\xi_j)_{j=1}^{n+p+1}$ *be a non-decreasing sequence of real numbers, i.e. a knot vector for a total of n B-splines. Also let* $d \geq 1$ *be an integer. The linear space of all linear combinations of these B-splines defined by*

$$\begin{aligned} \mathbb{S}_{p,\Xi} &= \operatorname{span}\left\{N_{1,p}(\xi),\dots,N_{n,p}(\xi)\right\} \\ &= \left\{\sum_{j=1}^{n} c_j N_{j,p}(\xi) | c_j \in \mathbb{R}^d \quad \mathit{for} \quad 1 \leq j \leq n\right\}. \end{aligned} \tag{4.1}$$

is called the spline space $\mathbb{S}_{p,\Xi}$.

The building blocks of the spline curve is the elements of the spline space, which is the set of B-splines. One can create different curves by choosing different coefficients in the linear combination. Sometimes we represent splines where the coefficients c_j are scalars on the real line. We then call the spline a *spline function.*

Definition 4. ***Spline function.*** *An element* $f(\xi) = \sum_{j=1}^{n} c_j N_{j,p}(\xi)$ *of the spline space* $\mathbb{S}_{p,\Xi}$, *where* $c_j \in \mathbb{R}$ *for* $j = 1,\dots,n$, *is called a spline function, or just a spline, of degree p with knots* Ξ. *The real numbers* $(c_j)_{j=1}^{n}$ *are the B-spline coefficients of* $f(\xi)$.

We can also represent spline curves in the d-dimensional space $\mathbb{R}^d$ by taking a linear combination of B-spline basis functions using points $c_i \in \mathbb{R}^d$ as coefficients. The coefficients c_i are called *control points.*

Definition 5. ***Spline curve.*** *Given n basis functions* $N_{i,p}(\xi)$ *for* $i = 1,\dots,n$ *and control points* $c_i \in \mathbb{R}^d$ *for* $i = 1,\dots,n$, *an element of the spline space* $\mathbb{S}_{p,\Xi}$

$$C(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) c_i \tag{4.2}$$

is a spline curve, or parametric spline curve.

The smoothness of the spline curve follow from that of the B-spline, as stated in [7, p. 73-74], i.e. the spline curve has $p - m$ continuous derivatives at the knot with multiplicity m, as described in section 3.2. So a quadratic spline curve will only be continuous at a double knot, whereas it will be continuous *and* have a continuous derivative at a single knot. The ability to control the smoothness of a spline curve can have important practical application. For example if it is necessary to represent a sharp corner with a C^0-continuous curve, this can be done by letting the knot appear p times at that corner, see figure 4.2.

The spline curve representation is rather flexible for approximating geometries. In spline literature it is often the control points which are adjusted if the spline curve needs to change. Repeated knots are also used to control smoothness, as already discussed. Another way to adjust a spline curve is to change a knot value in the knot vector. Since we already know that a knot vector has to be a non-decreasing set of numbers

$\xi_i \in [\xi_{i-1}, \xi_{i+1}]$, we can only change ξ_i in its permitted interval. This method is not that common. However, it is a flexibility feature that is worth knowing about. An example is shown in figure 4.2.

Example 7. *The example curves shown in figure 4.2 are built from quadratic basis functions. The control points are given as*

$$\boldsymbol{c} = \begin{bmatrix} 1 & 2 & 4 & 4 & 6 & 8 & 4 & 2 \\ 2 & 1 & 1 & 3 & 3 & 5 & 6 & 4 \end{bmatrix}, \tag{4.3}$$

for $d = 2$. Piecewise linear interpolation of the control points gives the control polygon. Curve (a) is built from the knot vector $\Xi_a = [0,0,0,1,2,3,3.2,3.2,5,5,5]$ and curve (b) is built from the knot vector $\Xi_b = [0,0,0,1,2,3,4.8,4.8,5,5,5]$. Both knot vectors are open, so the curves are interpolatory at the first and last end points. The curves are also interpolatory at the sixth control points due to the repeated knot value. Everywhere else they are $C^{p-1} = C^1$-continuous. The repeated knot value are slightly different for the two curves and therefore we can see that the image of the repeated knot value in Ξ_b lies to the left of the repeated knot value in Ξ_a, so the curves passing through the repeated knot does not overlap each other.

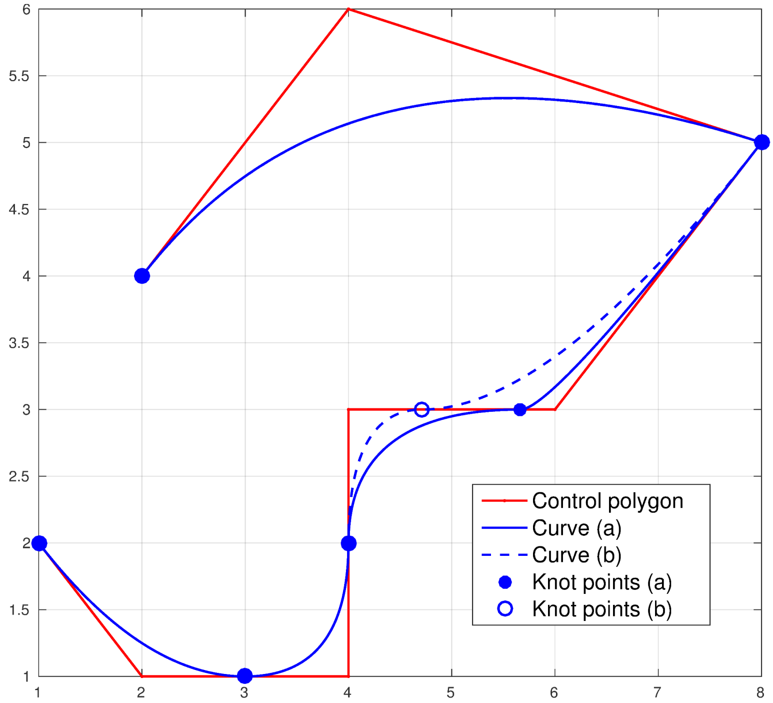


Figure 4.2: B-spline curve for the control points **c** (4.3) using second order B-splines. We have used two knot vectors Ξ_a and Ξ_b. The curves have reduced smoothness due to the repeated knots, and are slightly different since the repeated knots does not have the same value. The knots, which define a mesh by partitioning the curve into elements, are marked by blue circles and dots.

There are alternative methods for representing geometry using spline functions. The *Variation Diminishing Spline Approximation* is one of them.

Definition 6. ***The Variation Diminishing Spline Approximation*** *Let f be a given continuous function on the interval $[a,b]$, let p be a given positive integer, and let $\Xi = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$ be a $p+1$-regular knot vector with boundary knots $\xi_{p+1} = a$ and $\xi_{n+1} = b$. The spline given by*

$$(Vf)(x) = \sum_{j=1}^{n} f(\xi_j^*) N_{j,p}(x), \tag{4.4}$$

where $\xi_j^* = (\xi_{j+1} + \ldots + \xi_{j+p})/p$ are the knot averages, is called the Variation Diminishing Spline (VDS) approximation of degree p to f on the knot vector Ξ.

VDS is a subgroup of the previous considered B-spline function. The difference is that now the control points are defined to be the function f sampled and evaluated at knot averages. It is also a generalization of piecewise linear interpolation and it has a nice shape preserving behavior called the *variation diminishing property*.

4.2 Spline Surfaces

The tensor product structure is essential when it comes to defining two dimensional spline geometries. We will see that the two dimensional spline space is a tensor product of two one dimensional spline spaces. Two dimensional B-splines are tensor products of one dimensional B-splines, and spline surfaces are tensor products of spline curves.

First we need to establish the linear independence of the B-splines in $\mathbb{S}_{p,\Xi}$. The following theorem use that the $p+1$-regular knot vector is a subgroup of the more general $p+1$-*extended knot vector* [7, p. 66].

Theorem 2. *Suppose that Ξ is a $p+1$-extended knot vector. Then the B-splines in $\mathbb{S}_{p,\Xi}$ are linearly independent on the interval $[\xi_{p+1}, \xi_{n+1})$.*

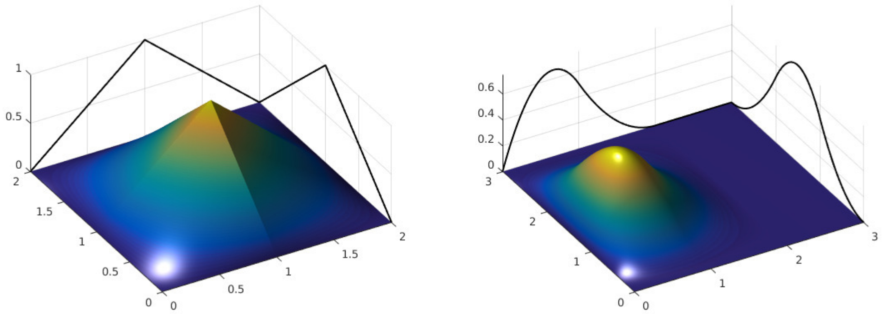
Proof. The proof consists of showing that the linear system $\sum_{j=1}^{n} c_j N_{j,p}(\xi) = 0$ only admits the trivial solution $c_j = 0 \quad \forall j$. For the complete proof, see [7, p. 66]. □

Definition 7. ***Tensor product spline space.*** *Given spline orders p and q, and knot vectors $\Xi = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$ and $\mathcal{H} = [\eta_1, \eta_2, \ldots, \eta_{m+q+1}]$, the tensor product of the two spline spaces $\mathbb{S}_{p,\Xi}$ and $\mathbb{S}_{q,\mathcal{H}}$ is defined to be the space*

$$\mathbb{S}_{p,\Xi} \bigotimes \mathbb{S}_{q,\mathcal{H}} = \text{span}\left\{N_{1,p}(\xi), \ldots, N_{n,p}(\xi)\right\} \bigotimes \text{span}\left\{M_{1,q}(\eta), \ldots, M_{m,q}(\eta)\right\} \tag{4.5}$$

where $N_{i,p}(\xi)$ is the univariate B-spline basis functions of order p corresponding to the knot vector Ξ, and $M_{j,q}(\eta)$ is the univariate B-spline basis function of order q corresponding to the knot vector $\mathcal{H}$.

The space $\mathbb{S}_{p,\Xi} \otimes \mathbb{S}_{q,\mathcal{H}}$ is spanned by the functions $\left\{N_{i,p}(\xi) M_{j,q}(\eta)\right\}_{i,j=1}^{n,m}$. Since these functions are linearly independent the space has dimension nm.



(a) Bilinear.

(b) Biquadratic.

Figure 4.3: Two-dimensional basis functions. Figure 4.3a is a bilinear basis function, meaning that both spline degrees $p = q = 1$. It is constructed by the tensor product $N_{2,1}(\xi) M_{2,1}(\eta)$ built on the open knot vectors $\Xi = \mathcal{H} = [0,0,1,2,2]$ in ξ- and η-direction, respectively.
Figure 4.3b is a biquadratic basis function, meaning that both spline degrees $p = q = 2$. It is constructed by the tensor product $N_{2,2}(\xi) M_{3,2}(\eta)$ built on the open knot vectors $\Xi = \mathcal{H} = [0,0,0,1,2,3,3,3]$ in ξ- and η-direction, respectively.

In lemma 1 we learned that the support of a B-spline $N_{i,p}(\xi)$ is the interval $[\xi_i, \xi_{i+p+1}]$. In two dimensions we get that the support of $N_{i,p}(\xi)M_{j,q}(\eta)$ is the rectangular interval $[\xi_i, \xi_{i+p+1}] \times [\eta_j, \eta_{j+q+1}]$. The support of the bivariate basis function $N_{2,1}(\xi)M_{2,1}(\eta)$ is therefore $[0,2] \times [0,2]$ in figure 4.3a, and the support of $N_{2,2}(\xi)M_{3,2}(\eta)$ is therefore $[0,2] \times [0,3]$ in figure 4.3b.

Definition 8. ***Spline surface.*** *Given a control net* (c_{ij})*,* $i = 1,2,\ldots,n$*,* $j = 1,2,\ldots,m$*, polynomial orders* p *and* q*, and knot vectors* $\Xi = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$ *and*
$\mathcal{H} = [\eta_1, \eta_2, \ldots, \eta_{m+q+1}]$*, a bivariate spline, or spline surface, is defined by the geometrical map* $S : \hat{\Omega} \to \Omega$

$$S(\xi,\eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi)M_{j,q}(\eta)c_{ij}. \tag{4.6}$$

The surface $S(\xi,\eta)$ *is an element in the tensor product spline space* $\mathbb{S}_{p,\Xi} \otimes \mathbb{S}_{q,\mathcal{H}}$*.*

4.3 More on Geometry

The one dimensional parameter space is partitioned into *elements* by the knots. The two dimensional analogue is the *knot lines.* Knot lines in physical space are the image of knot lines in parameter space.

Every knot line in the parameter space ξ-direction is perpendicular on all other knot lines in the parameter space η-direction. A realization of the spline geometry is made by mapping each element from the parameter space into the physical space by making use of the equation (4.6).

The control net $(c_{ij})_{i,j=1}^{n,m}$ is what governs the surface in physical space. Each point $c_{ij} \in \mathbb{R}^3$ in the control net has x-,y- and z-components. We recognize $(x_{ij}, y_{ij}, z_{ij})^T$ as the components of control point c_{ij}. Lets collect all x-components in the matrix $\mathbf{X} = (x_{ij})_{i,j=1}^{n,m}$, where $x_{ij} \in \mathbb{R}$ is a scalar. Now do the same for the y- and z-components to get the matrices $\mathbf{Y}$ and $\mathbf{Z}$. Each component has to undergo the transformation of equation (4.6), i.e.

$$S(\xi,\eta) = \begin{pmatrix} x(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{pmatrix} = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) \begin{pmatrix} x_{ij} \\ y_{ij} \\ z_{ij} \end{pmatrix}, \tag{4.7}$$

which can be written conveniently in a matrix-vector form as

$$\begin{aligned} x(\xi,\eta) &= \mathbf{N}^{\mathbf{T}}\,\mathbf{X}\,\mathbf{M} \\ y(\xi,\eta) &= \mathbf{N}^{\mathbf{T}}\,\mathbf{Y}\,\mathbf{M} \\ z(\xi,\eta) &= \mathbf{N}^{\mathbf{T}}\,\mathbf{Z}\,\mathbf{M} \end{aligned} \tag{4.8}$$

where

$$\mathbf{N} = \begin{bmatrix} N_{1,p}(\xi) \\ \vdots \\ N_{n,p}(\xi) \end{bmatrix} \quad \text{and} \quad \mathbf{M} = \begin{bmatrix} M_{1,q}(\eta) \\ \vdots \\ M_{m,q}(\eta) \end{bmatrix} \tag{4.9}$$

are the vectors holding all B-splines in the basis. When the z-components $z(\xi,\eta)$ of the surface is plotted in the coordinates of $x(\xi,\eta)$ and $y(\xi,\eta)$ in physical space we can see the realization of the spline surface defined by the control net.

Example 8. *A biquadratic spline surface, seen in figure 4.4, is built on the open knot vectors $\Xi = [0,0,0,0.5,1,1,1]$ and $\mathcal{H} = [0,0,0,1,1,1]$. A control net is defined in table 4.1. The surface, along with its black knot lines, is mapped from parameter space into physical space using the equations (4.8). The bivariate basis function $\{N_{2,2}(\xi)M_{2,2}(\eta)\}$ is plotted in parameter space and physical space, as shown in figure 4.5.*

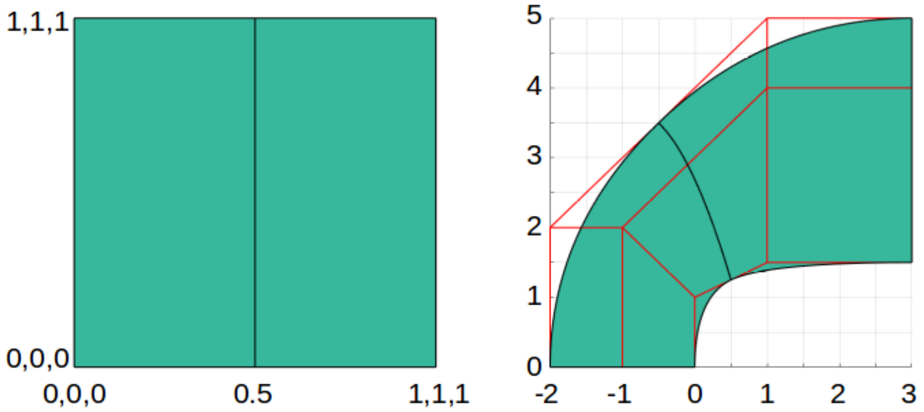


Figure 4.4: Parameter space (left) and Physical space (right). The knots and the knot lines, when mapped into physical space, is what partition the surface into elements.

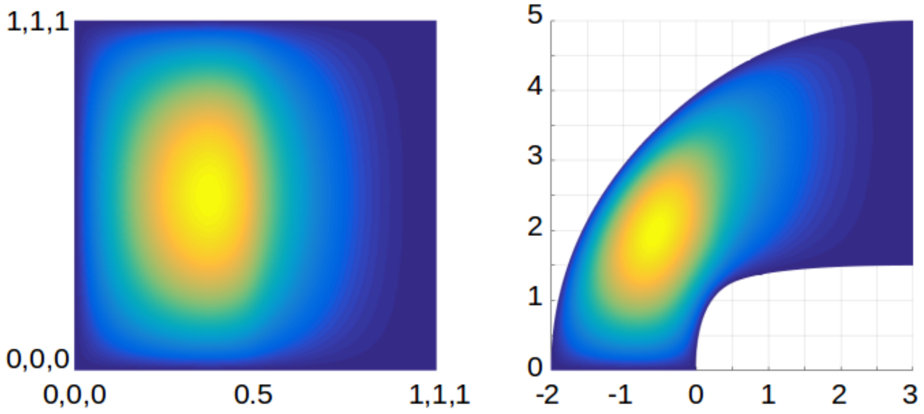
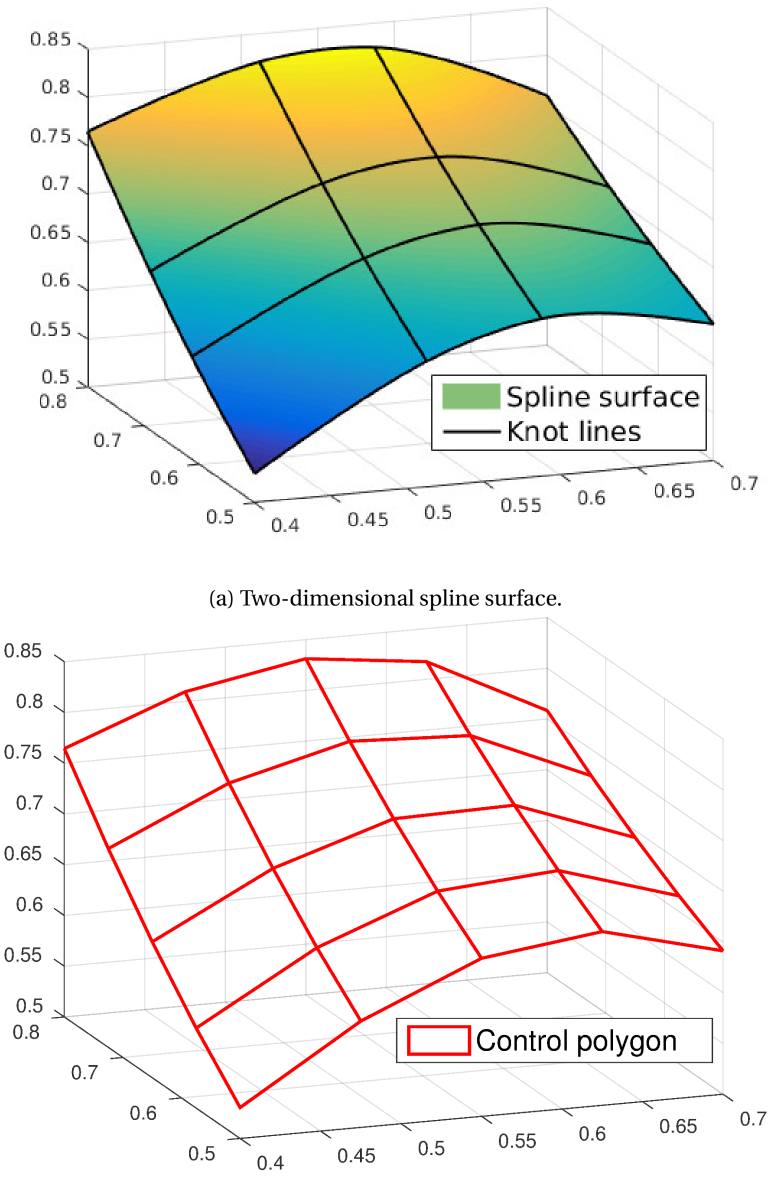


Figure 4.5: Parameter space (left) and Physical space (right). The bivariate basis function $\{N_{2,2}(\xi)M_{2,2}(\eta)\}$ is mapped from the parameter space (left) into the physical space (right).

Example 9. *Spline surfaces can also have control points where the z-components are non-zero. In figure 4.6, the function* $f(x,y) = (1-x)y^2 + x\sin(\pi x)$ *is sampled on the points* $x \in \{0.4, 0.475, 0.55, 0.625, 0.7\}$ *and* $y \in \{0.5, 0.575, 0.65, 0.725, 0.8\}$ *to create a control net (figure 4.6b). The surface (figure 4.6a) is build on the knot vectors*

$\Xi = \mathcal{H} = [0,0,0,1,2,3,3,3]$ *using quadratic polynomials* $p = q = 2$.



(a) Two-dimensional spline surface.

(b) The corresponding control polygon.

Figure 4.6: The control polygon in figure 4.6b is a net of sampled points from a real-valued function $f(x, y)$, while the surface in 4.6a is the biquadratic tensor product spline surface approximating the control polygon.

Table 4.1: Control points c_{ij} for the surface in figure 4.4 and 4.5. Only (x, y)-coordinates are shown. All z-components of the control points are defined to be zero, and are therefore excluded from the table.

i	j	c_{ij}
1	1	(0,0)
1	2	(-1,0)
1	3	(-2,0)
2	1	(0,1)
2	2	(-1,2)
2	3	(-2,2)
3	1	(1,1.5)
3	2	(1,4)
3	3	(1,5)
4	1	(3,1.5)
4	2	(3,4)
4	3	(3,5)

When constructing surfaces, it is interesting to notice that, the knot vectors used are invariant under translation and scaling. For example, if we build the surface in figure 4.6 on the translated knot vectors $\Xi = \mathcal{H} = [1, 1, 1, 2, 3, 4, 4, 4]$, or the scaled knot vectors $\Xi = \mathcal{H} = [0, 0, 0, 2, 4, 6, 6, 6]$, it will result in an equivalent surface when plotted in physical space. This property is called *translation invariance* and is discussed in [7, p.42-43].

4.3.1 A Linear Map

A linear map $x_L : \hat{\Omega} \to \Omega$ is given by the linear function

$$x_L(\xi) = A\xi + B. \tag{4.10}$$

Lets say the parameter space is $\hat{\Omega} = \left[\hat{a}, \hat{b}\right]$ and the physical space is $\Omega = [a, b]$. The two conditions $x(\hat{a}) = a$ and $x(\hat{b}) = b$ helps us determine A and B so that

$$x_L(\xi) = \frac{b-a}{\hat{b}-\hat{a}}\,\xi + \frac{a\hat{b}-\hat{a}b}{\hat{b}-\hat{a}}, \tag{4.11}$$

and hence the linear map is governed by the control points $\{x_i\}_{i=1}^{n}$ which are the solution of the linear system

$$x_L(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\; x_i. \tag{4.12}$$

Most figures in this chapter, and the next chapter, have used control points with equidistant spacing in physical space. This *will not* result in a linear map, and is the result that the reader will observe that the spacing between the coordinates of knot lines and knot points is not equidistant, as for example in figure 4.6. A linear map, and the nonlinear map used in chapter (4) and (5), is plotted in 4.7.

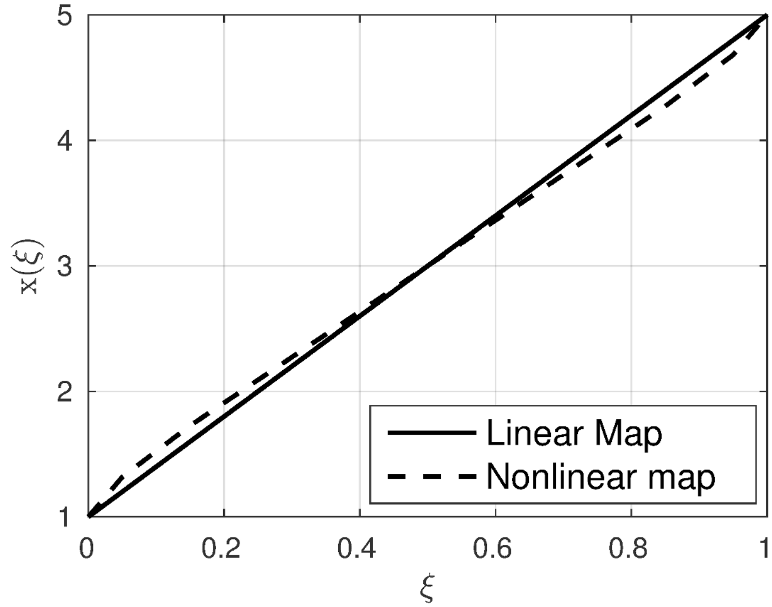


Figure 4.7: An illustration of a linear map $x_L(\xi)\ :\ [0,1] \to [1,5]$ and a nonlinear map $x_N(\xi)\ :\ [0,1] \to [1,5]$ between the spaces.

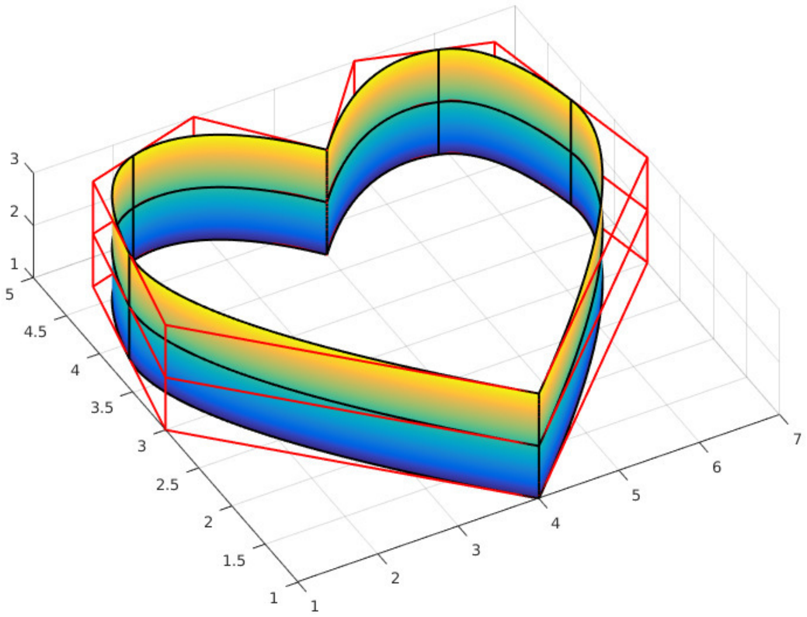


Figure 4.8: This heart is made of quadratic splines in ξ direction and linear splines in η direction. The surface is defined by $9 \cdot 3$ control points in $\mathbb{R}^3$. We have used open knot vectors defining a rectangular domain where the surface lives in parameter space. The control points and the basis functions defines a geometrical map into physical space transforming the rectangular surface into a heart using only continuous bending and stretching. Ξ has a repeated knot value to make a C^0 continuity in ξ direction. The image of this is the middle top part of the heart. On all other interior knot lines we have C^1 continuity. The middle bottom part is clamped together simply by defining the first and last control points to meet at the same points.

Chapter 5

Spline Interpolation

So far we have mainly considered spline approximations defined by a given set of control points. However, they do not pass through the control points. Now we want to explore methods where the spline approximation is exact at a given set of *interpolation points*. We first develop a one dimensional method and then move on to cover two dimensions.

5.1 1D General Spline Interpolation

Splines are piecewise polynomials, as we already know. The degree of a Lagrange interpolating polynomial will increase proportional to the number of control points being interpolated. For splines it is different. Here we need to increase the number of piecewise polynomials proportional to the number of control points. Higher degree polynomials tend to oscillate a lot, and this is the case for the Lagrange interpolating polynomials, since they are global in nature. As a result of this we get large errors in our approximation. The advantage of splines is that it is often enough to use quadratic or cubic polynomials locally, on each interval. Hence we can avoid the large oscillations and the large errors in our approximation. Even though cubic splines are undoubtedly the most common, there is an advantage of having methods available for all spline degrees, so that is what we want to develop.

We want a method that creates a spline passing through all control points, or interpolation points. Given interpolation points $(\hat{\xi}_i, u_i),\ i = 1, \ldots, n$ we then consider the problem of finding a spline function g such that

$$g(\hat{\xi}_i) = u_i, \quad i = 1, \ldots, n. \tag{5.1}$$

Usually there is some sort of unknown field $u(x)$ on the physical domain, such as a distribution of temperature, or a magnetic field. The response variable $u_i = u(\hat{x}_i)$ represent one distinct measurement of this field, where $\hat{x}_i$ in physical space is the image of the interpolation point $\hat{\xi}_i$ in parameter space. Since there is some analogous function $\hat{u}(\xi)$ in $\hat{\Omega}$ such that $u(x(\xi)) = \hat{u}(\xi)$, we can directly transfer the function values on interpolation points $u_i = u(\hat{x}_i) = \hat{u}(\hat{\xi}_i)$ for all $i = 1, \ldots, n$. The resulting spline function from (5.1) is an approximation of the unknown field, i.e. $g(\xi) \approx u(x(\xi))$.

There can be various goals we would want to achieve, depending on the specific interpolation problem, and we would like to use the most suitable spline tools for the situation. If quadratic splines are chosen we will be able to get a smooth representation of the first derivative of the spline. If cubic splines are chosen we can get a smooth second derivative. If we want the third degree derivative to be continuous then the degree must be higher than three, and so on. For the sake of robustness, we want to develop a method which can handle all these situations. We therefore consider the following interpolation problem

Definition 9. ***1D spline interpolation problem.*** *Let there be given data $(\hat{\xi}_i, u_i)_{i=1}^n$ and a spline space $\mathbb{S}_{p,\Xi}$ whose knot vector $\Xi = (\xi_i)_{i=1}^{n+p+1}$ is open and non-decreasing such that $\xi_{i+p+1} > \xi_i$, for $i = 1, \ldots, n$. Find a spline g in $\mathbb{S}_{p,\Xi}$ such that*

$$g(\hat{\xi}_i) = \sum_{j=1}^{n} c_j N_{j,p}(\hat{\xi}_i) = u_i \quad for \quad i = 1, \ldots, n. \tag{5.2}$$

The equations in (5.2) form a system of n equations in n unknowns. In matrix form these equations can be written

$$\mathbf{N}^{\mathbf{T}}\,\mathbf{c} = \begin{pmatrix} N_{1,p}(\hat{\xi}_1) & \dots & N_{n,p}(\hat{\xi}_1) \\ \vdots & \ddots & \vdots \\ N_{1,p}(\hat{\xi}_n) & \dots & N_{n,p}(\hat{\xi}_n) \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} = \mathbf{u}. \tag{5.3}$$

The matrix **N** is often referred to as the *B-spline collocation matrix*. Since $N_{i,p}(\xi)$ is non-zero only in its region of support $[\xi_i, \xi_{i+p+1}]$, the matrix **N** will in general be sparse. The collocation matrix possesses the property of being *totally positive* which is defined in the following

Definition 10. ***Totally positive matrices.*** *A matrix* $\mathbf{A}$ *in* $\mathbb{R}^{n\times m}$ *is said to be totally positive if all its square submatrices have non-negative determinant. More formally, let* $\mathbf{i} = (i_1, i_2, \dots, i_l)$ *and* $\mathbf{j} = (j_1, j_2, \dots, j_l)$ *be two integer sequences such that*

$$1 \le i_1 < i_2 \le \dots < i_l \le m, \tag{5.4}$$

$$1 \le i_1 < i_2 \le \dots < i_l \le n, \tag{5.5}$$

and let $\mathbf{A}(\mathbf{i},\mathbf{j})$ *denote the submatrix of* $\mathbf{A}$ *with entries* $(a_{i_p,j_q})_{p,q=1}^{l}$. *Then* $\mathbf{A}$ *is totally positive if* $\det(\mathbf{A}(\mathbf{i},\mathbf{j})) \ge 0$ *for all sequences* **i** *and* **j** *on the form* (5.4) *and* (5.5), *for all* l *such that* $1 \le l \le \min\{m, n\}$.

For the system of equations (5.3) to have a unique solution, it is necessary and sufficient that **N** is non-singular. The next theorem follows from the fact that **N** is totally positive, and it tells us exactly when the matrix is non-singular.

Theorem 3. *Let $\mathbb{S}_{p,\Xi}$ be a given spline space, and let $\hat{\xi}_1 < \hat{\xi}_2 < \ldots < \hat{\xi}_n$ be n distinct numbers. The collocation matrix* $\mathbf{N}$ *with entries $(N_{j,p}(\hat{\xi}_i))_{i,j=1}^n$ is non-singular if and only if its diagonal elements is positive, i.e.*

$$N_{i,p}(\hat{\xi}_i) > 0 \quad \textit{for} \quad i = 1, \ldots, n. \tag{5.6}$$

Proof. See the proof in [7, p.131-132,208] □

The condition that the diagonal elements of $\mathbf{N}$ should be non-zero can be written

$$\xi_i < \hat{\xi}_i < \xi_{i+p+1} \quad \text{for} \quad i = 1, \ldots, n \tag{5.7}$$

provided we allow that $\hat{\xi}_i = \xi_i$ if $\xi_i = \cdots = \xi_{i+p}$. For an open knot vector this means that the outermost interpolation points is allowed to lay strictly on the domain boundary. Condition (5.7) goes under the name *Schoenberg-Whitney nesting conditions*. It describes that the interpolation points $\hat{\xi}_i$ for $i = 1, \ldots, n$ are required to be placed in the region where the corresponding basis function $N_{i,p}(\xi)$ is non-zero, for a solution to exist. That is the region of $p+1$ knot spans, as we remember from item 4 in lemma 1.

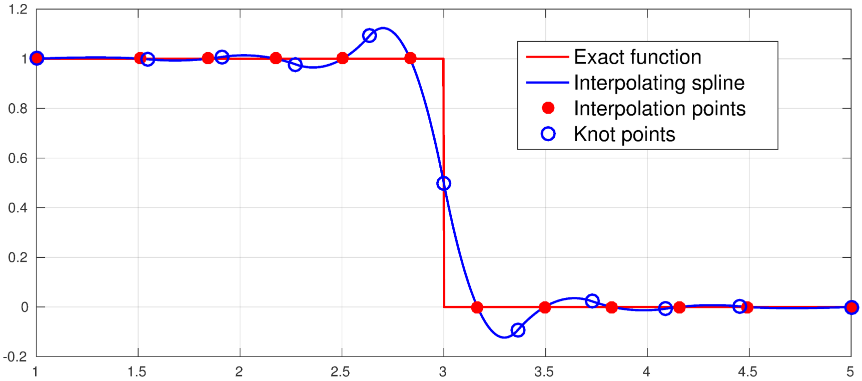


Figure 5.1: The step function $H_{0.5}(x)$ is sampled on the physical image of $n = 12$ uniformly distributed interpolation points in the parameter space $\hat{\Omega} = [0, 1]$. These points are interpolated by a quadratic spline which is built on the uniform and open knot vector Ξ. The solution is plotted in the physical space $\Omega = [1, 5]$.

5.2 2D General Spline Interpolation

We now consider the two dimensional analogy of the one dimensional spline interpolation. Given a net of interpolation points $(\hat{\xi}_i, \hat{\eta}_j)_{i,j=1}^{n,m}$ and the values of an unknown function $f = f(x,y)$ at the interpolation points in physical space, we want to find a tensor product spline surface such that

$$g(\hat{\xi}_i, \hat{\eta}_j) = f_{ij} \quad \text{for} \quad i = 1, \ldots, n \quad \text{for} \quad j = 1, \ldots, m, \tag{5.8}$$

where we have used the notation $f_{ij} = f(\hat{x}_i, \hat{y}_j)$ and that $(\hat{x}_i, \hat{y}_j)$ is the interpolation point $(\hat{\xi}_i, \hat{\eta}_j)$ mapped to physical space. We define the two dimensional interpolation problem in the following.

Definition 11. ***2D spline interpolation problem.*** *Let there be given data $(\hat{\xi}_i, \hat{\eta}_j, f_{ij})_{i,j=1}^{n,m}$ and a spline space $\mathbb{S}_{p,\Xi} \otimes \mathbb{S}_{q,\mathcal{H}}$ whose knot vectors $\Xi = (\xi_i)_{i=1}^{n+p+1}$ and $\mathcal{H} = (\eta_j)_{j=1}^{m+q+1}$ are open and non-decreasing. Find a spline $g \in \mathbb{S}_{p,\Xi} \otimes \mathbb{S}_{q,\mathcal{H}}$ such that*

$$g(\hat{\xi}_i, \hat{\eta}_j) = \sum_{k=1}^{n} \sum_{l=1}^{m} c_{kl} N_{k,p}(\hat{\xi}_i) M_{l,q}(\hat{\eta}_j) = f_{ij}, \tag{5.9}$$

for $i = 1, \ldots, n$ and $j = 1, \ldots, m$.

We define the collocation matrices $\mathbf{N} \in \mathbb{R}^{n \times n}$ and $\mathbf{M} \in \mathbb{R}^{m \times m}$ by

$$\mathbf{N} = \begin{pmatrix} N_{1,p}(\hat{\xi}_1) & \ldots & N_{1,p}(\hat{\xi}_n) \\ \vdots & \ddots & \vdots \\ N_{n,p}(\hat{\xi}_1) & \ldots & N_{n,p}(\hat{\xi}_n) \end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} M_{1,q}(\hat{\eta}_1) & \ldots & M_{1,q}(\hat{\eta}_m) \\ \vdots & \ddots & \vdots \\ M_{m,q}(\hat{\eta}_1) & \ldots & M_{m,q}(\hat{\eta}_m) \end{pmatrix}. \tag{5.10}$$

If $\mathbf{N}$ and $\mathbf{M}$ are non-singular then there is a unique tensor product spline $g \in \mathbb{S}_{p,\Xi} \otimes \mathbb{S}_{q,\mathcal{H}}$ such that the interpolation conditions (5.8) holds. It is convenient to denote the coefficient matrix $\mathbf{C} = (c_{kl})_{k,l=1}^{n,m}$ and to express (5.9) in the matrix notation

$$\mathbf{N}^{\mathbf{T}} \, \mathbf{C} \, \mathbf{M} = \mathbf{F}. \tag{5.11}$$

where $\mathbf{F} = (f_{ij})_{i,j=1}^{n,m}$. The matrix **C** can now be computed in the following steps

$$\text{Define } \mathbf{D} \equiv \mathbf{CM}. \tag{5.12}$$

$$\text{Solve } \mathbf{N}^{\mathbf{T}}\,\mathbf{D} = \mathbf{F} \text{ for the unknown } \mathbf{D}. \tag{5.13}$$

$$\text{Solve } \mathbf{D} = \mathbf{C}\,\mathbf{M} \Leftrightarrow \mathbf{M}^{\mathbf{T}}\,\mathbf{G} = \mathbf{D}^{\mathbf{T}} \text{ for the unknown } \mathbf{G} = \mathbf{C}^{\mathbf{T}}. \tag{5.14}$$

$$\text{Transpose } \mathbf{C} = \mathbf{G}^{\mathbf{T}}. \tag{5.15}$$

This splits the computation into two univariate interpolation problems. It involves solving two banded linear systems with several right hand sides, and two transpose operations.

Figure 5.2 shows an example where the two dimensional step function

$$H_a(x,y) = \begin{cases} 1 & \text{if } (x < a \text{ and } y < a) \text{ or } (x > a \text{ and } y > a), \\ \frac{1}{2} & \text{if } x = a \text{ or } y = a, \\ 0 & \text{if } (x > a \text{ and } y < a) \text{ or } (x < a \text{ and } y > a). \end{cases} \tag{5.16}$$

for $a = 0.5$ is interpolated by a tensor spline surface.

5.3 Interpolation Points

It has been clear from the discussion in section 2.3 that the choice of interpolation points can be crucial, at least for Lagrange interpolation. We now want to show that this is the case for spline interpolation as well.

Example 10. *The step function $H_{0.5}(x)$ is sampled on the physical image of 12 uniformly distributed interpolation points in the parameter space $\hat{\Omega} = [0,1]$. These points are interpolated by a quadratic spline. Three different cases, using three different knot vectors, are being considered. The solutions is plotted in the physical space $\Omega = [1,5]$. In each sub-figure we have plotted the spline basis and interpolation points on the left hand side, and the numerical solution of the interpolating spline on the right hand side. The knot*

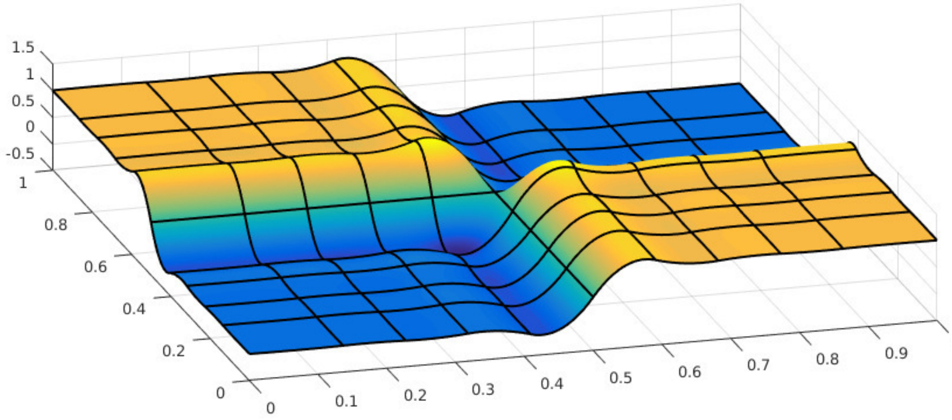


Figure 5.2: The step function $H_{0.5}(x, y)$ defined in (5.16) is sampled on the physical image of $n = m = 12$ uniformly distributed interpolation points in each direction in the parameter space $\hat{\Omega} = [0,1] \times [0,1]$, creating a total of 12^2 interpolation points. These are interpolated by a biquadratic spline which is built on the uniform and open knot vectors Ξ and $\mathcal{H}$. The solution is plotted in the physical space $\Omega = [0,1] \times [0,1]$.

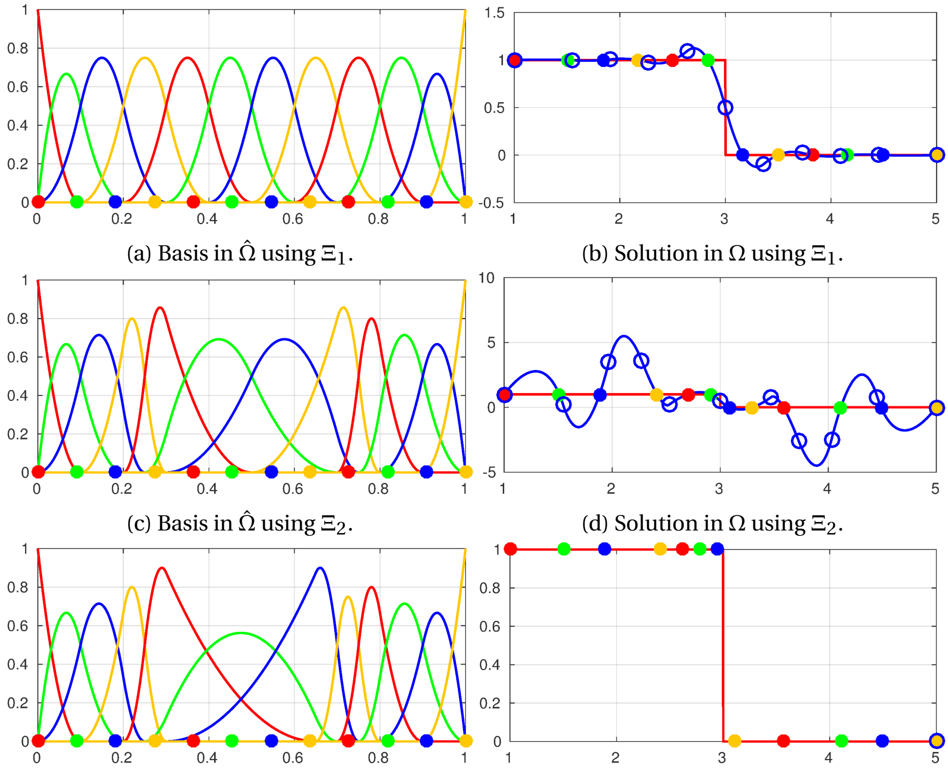
vectors used are defined by

$$\Xi_1 = [0,0,0,.1,.2,.3,.4,.5,.6,.7,.8,.9,1,1,1], \tag{5.17}$$

$$\Xi_2 = [0,0,0,.1,.2,.25,.3,.5,.7,.75,.8,.9,1,1,1], \tag{5.18}$$

$$\Xi_3 = [0,0,0,.1,.2,.25,.3,.65,.7,.75,.8,.9,1,1,1]. \tag{5.19}$$

The solution in Figure 5.3b is the same as in Figure 5.1. Figure 5.3d and 5.3d is the exact same problem except for some changes made in the knot vectors used. The basis functions, and hence the interpolating spline, change according to the knot vectors. When Ξ_3 is used condition (5.7) in theorem 3 is violated, so there is no solution available in this case. The basis function plots are color-coded to illustrate that each interpolation points has its corresponding basis function. The middle yellow point is not under the non-zero part of its basis function, causing the collocation matrix to have one zero diagonal entry, and hence become singular.



(a) Basis in $\hat{\Omega}$ using Ξ_1.

(b) Solution in Ω using Ξ_1.

(c) Basis in $\hat{\Omega}$ using Ξ_2.

(d) Solution in Ω using Ξ_2.

(e) Basis in $\hat{\Omega}$ using Ξ_3.

(f) No existing solution using Ξ_3.

Figure 5.3: The step function is sampled on $n = 12$ uniformly distributed points, and interpolated by quadratic splines. Different knot vectors are used in each case. The knot vector used in the bottom figure violates the Schoenberg-Whitney nesting conditions, as explained in example 10.

The image of the so-called Greville abscissae is a different selection of interpolation points. We want to consider these points as an alternative to uniform points, and see if they can give better results for interpolation problems.

Definition 12. ***Greville Abscissae*** *For a given knot vector* $\Xi = [\xi_1, \dots, \xi_{n+p+1}]$ *the associated* n *points are calculated from*

$$\hat{\xi}_i = \frac{\xi_{i+1} + \xi_{i+2} + \ldots + \xi_{i+p}}{p}, \tag{5.20}$$

where p *is the polynomial degree.*

To illustrate the difference between Greville points and uniform points, we have plotted them in figure 5.4.

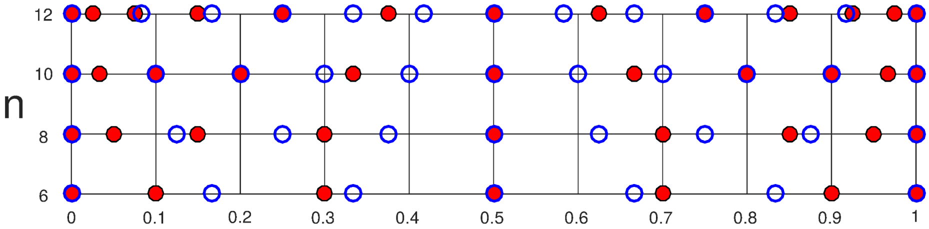
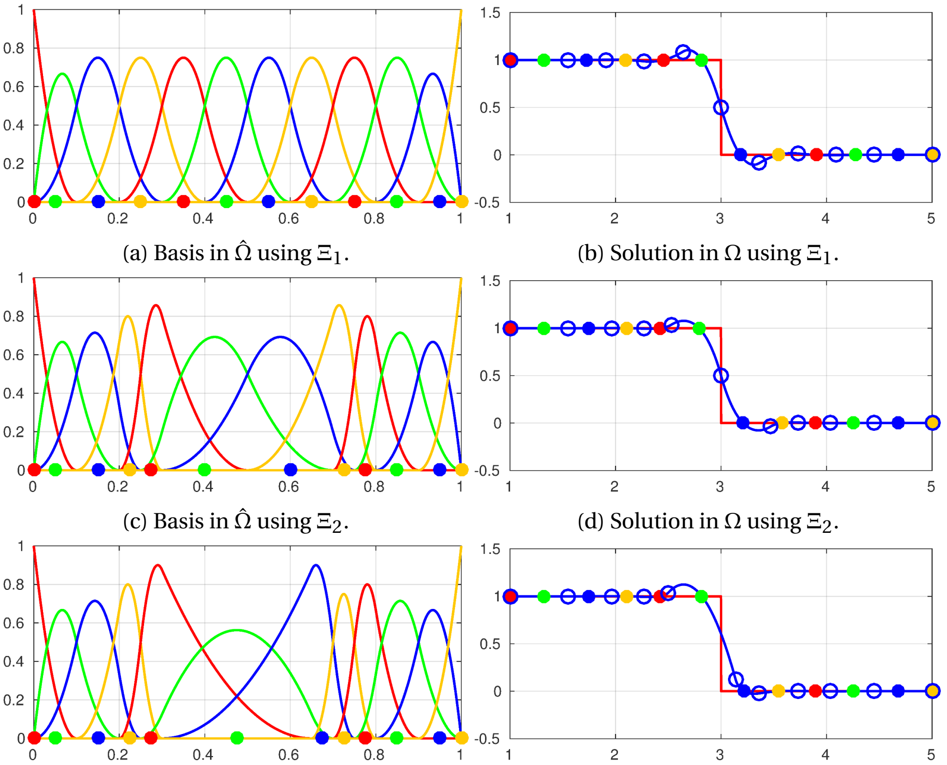


Figure 5.4: Interpolation points in the domain $\bar{\Omega} = [0,1]$ for $n = 6, 8, 10, 12$. Uniformly distributed points are marked by blue circles and Greville abscissae are marked by red dots.

When applying (5.20) on an open knot vector, the two outermost interpolation points is placed at the domain boundary. For p even, the interior interpolation points are located exactly in the center of an element in parameter space, i.e. the average of two neighboring knots. This is the reason the Greville abscissae also goes under the name *knot averages*. For p odd, the interior points are located exactly at the knots in parameter space. This can however not be observed in either case in figure 5.3 and 5.5. This is because we have plotted in physical space and used a non-linear map between spaces.



(a) Basis in $\hat{\Omega}$ using Ξ_1.

(b) Solution in Ω using Ξ_1.

(c) Basis in $\hat{\Omega}$ using Ξ_2.

(d) Solution in Ω using Ξ_2.

(e) Basis in $\hat{\Omega}$ using Ξ_3.

(f) Solution in Ω using Ξ_3.

Figure 5.5: The step function is sampled on the Greville abscissae for $n = 12$ points, and interpolated by quadratic splines. Different knot vectors are used in each case.

Example 11. *Example 10 are recreated using Greville points, while everything else is unchanged. We can observe descent solutions in all three cases, see Figure 5.5. The color coded plots for basis functions and interpolation points can be used to observe that the conditions* (5.7) *are fulfilled in each case.*

Chapter 6

Isogeometric Analysis

In this chapter we bring the splines into the setting of analysis, an arena to which their unique properties are also ideally suited. The concept of *Isogeometric Analysis*, or IGA, is that the basis used to exactly model the geometry will also serve as the basis for the solution space of the numerical method. However, the B-spline basis also possesses many properties that are quite desirable when approximating solution fields independently of any geometrical consideration. The concept of IGA is today advancing in the field of *Finite Element Analysis*. Within the framework of IGA, *collocation methods* have been recently proposed as an interesting strong form alternative to standard Galerkin approaches, characterized by a significantly reduced computational cost, but still guaranteeing higher order convergence rates. We will develop collocation methods in this chapter. We will continue to build on the ideas from the spline theory, and we will use B-splines to construct the solution spaces.

6.1 Collocation methods

Collocation is used to approximate a solution to a boundary value problem. The approximate solution and the exact solution for the differential equation is equal at a set of discrete points called the *collocation points*, and is an approximation in between. The collocation points are located in the domain Ω and some conditions are given at the boundary $\partial\Omega$ of the domain.

Definition 13. ***Boundary Value Problem (BVP).***
The goal is to find a solution $u : \Omega \to \mathbb{R}$ to the strong form of the BVP

$$\begin{cases} \mathcal{L}u & = f \quad in \quad \Omega \\ \mathcal{G}u & = g \quad on \quad \partial\Omega \end{cases} \tag{6.1}$$

where the functions $f : \Omega \to \mathbb{R}$ and $g : \partial\Omega \to \mathbb{R}$ are given, $\mathcal{L}$ is a linear differential operator, and $\mathcal{G}$ is a vector operator.

The boundary conditions given in definition 13 can be either *Dirichlet* where the function value of u is given at the boundary, or *Neumann* conditions where the gradient of the function is given at the boundary. We will encounter both in later examples.

To solve this problem by the collocation method, two sets of sample collocation points need to be considered, namely the sets $\{x_i^{int}\}$, $i = 1,\ldots,n^{int}$ in the interior of Ω and $\{x_i^{bnd}\}$, $i = 1,\ldots,n^{bnd}$ on $\partial\Omega$, where n^{int} ans n^{bnd} are the number of collocation points chosen in the interior and on the boundary, respectively. The collocation solution u_h is required to satisfy

$$\begin{cases} \mathcal{L}\left[u_h(x_i^{int})\right] & = f(x_i^{int}) \quad \text{in } \Omega, \quad \text{for } i = 1,\ldots,n^{int}, \\ \mathcal{G}\left[u_h(x_i^{bnd})\right] & = g(x_i^{bnd}) \quad \text{on } \partial\Omega, \quad \text{for } i = 1,\ldots,n^{bnd}. \end{cases} \tag{6.2}$$

Example 12. ***From interpolation to collocation.***

It is interesting to notice that interpolation is actually a special case of collocation. The collocation problem (6.2) for $\mathcal{L} = \mathcal{G} = I$ is simply an interpolation problem. The operator I is the differential operator which takes a real function and maps it to itself. Interpolation can therefore be thought of as a special case of collocation. Or in the opposite way, collocation can be thought of as the interpolation of $\mathcal{L}$.

6.2 1D Collocation method

To solve a BVP such as (6.1) in one dimension by a collocation method, a set of collocation points $\{\hat{\tau}_i\}_{i=1}^{n}$ needs to be introduced. The points are defined to be non-decreasing in the domain $\hat{\Omega} = [\hat{a}, \hat{b}]$ such that

$$\hat{a} = \hat{\tau}_1 < \hat{\tau}_2 < \ldots < \hat{\tau}_{n-1} < \hat{\tau}_n = \hat{b}, \tag{6.3}$$

where $\hat{a}$ and $\hat{b}$ are real scalars. The set of $n^{int} = n-2$ interior collocation points is $\{\hat{\tau}_i\}_{i=2}^{n-1}$ and the $n^{bnd} = 2$ boundary collocation points are $\hat{\tau}_1 = \hat{a}$ and $\hat{\tau}_n = \hat{b}$. The collocation solution $u_h(\xi)$, required to satisfy (6.2), is built as linear combinations of B-splines, i.e.

$$u_h(\xi) = \sum_{i=1}^{n} d_i N_{i,p}(\xi). \tag{6.4}$$

The one dimensional physical space is given by $\Omega = [a, b] \subset \mathbb{R}$. Any function in Ω can be directly mapped to $\hat{\Omega}$, meaning that for example $u_h(\xi) = u_h(x(\xi))$. This stands in contrast to coordinates that must undergo the geometrical transformation $\mathcal{F} : \hat{\Omega} \to \Omega$ when mapped between spaces.

We may now restate (6.2) as a general problem formulation for the 1D case.

Definition 14. ***1D Collocation Problem.***
A set of collocation points is given in the parameter space $\hat{\Omega} = [\hat{a}, \hat{b}] \subset \mathbb{R}$ *such that* $\hat{a} = \hat{\tau}_1 < \hat{\tau}_2 < \ldots < \hat{\tau}_{n-1} < \hat{\tau}_n = \hat{b}$. *The collocation solution* $u_h(\xi) \in \mathbb{S}_{p,\Xi}$ *is required to satisfy*

$$\begin{cases} \mathcal{L}\left[u_h(\hat{\tau}_j)\right] = f(\hat{\tau}_j) \quad \textit{in} \quad \Omega, \quad \textit{for} \quad j = 2, \ldots, n-1, \\ \mathcal{G}\left[u_h(\hat{\tau}_j)\right] = g(\hat{\tau}_j) \quad \textit{on} \quad \partial\Omega, \quad \textit{for} \quad i = 1 \quad \textit{and} \quad i = n. \end{cases} \tag{6.5}$$

The differential operator $\mathcal{L}$ is not yet defined. However, we can easily apply it to (6.4) in the following way

$$\mathcal{L}\left[u_h(\xi)\right] = \sum_{i=1}^{n} d_i \mathcal{L}\left[N_{i,p}(\xi)\right], \tag{6.6}$$

since the coefficients $d_i \; \forall \; i$ are scalars. The interpolatory nature of our basis makes it convenient to build the boundary directly into the solution space, and directly into $u_h(\xi)$ by the means of a lifting function. This process is referred to as a *strong imposition* of the boundary conditions. We assume to have a given lifting function $g_h \in \mathbb{S}_{p,\Xi}$ such that $g_h|_{\partial\Omega} = g$ and a function $v_h \in \mathbb{S}_{p,\Xi}$ such that $v_h|_{\partial\Omega} = 0$. To fulfill these conditions, we will in practice always choose $g_h(\xi)$ such that

$$g_h(\xi) = \sum_{i=1}^{n} g_i N_{i,p}(\xi) = g_1 N_{1,p}(\xi) + g_n N_{n,p}(\xi), \tag{6.7}$$

where the coefficients $g_2 = \ldots = g_{n-1} = 0$ have no effect on the boundary, and $v_h(\xi)$ such that

$$v_h(\xi) = \sum_{i=2}^{n-1} d_i N_{i,p}(\xi), \tag{6.8}$$

where $d_1 = d_n = 0$. Finally, recalling (6.4) we can describe the solution $u_h(\xi)$ as the sum

$$u_h(\xi) = v_h(\xi) + g_h(\xi) = \sum_{i=2}^{n-1} d_i N_{i,p}(\xi) + g_1 N_{1,p}(\xi) + g_n N_{n,p}(\xi), \tag{6.9}$$

and we have in fact obtained to build the boundary into the solution. Now going back to the differential operator; for a given p, we can take advantage of linearity to obtain the following relation

$$\mathcal{L}\left[u_h(\hat{\tau}_j)\right] = \sum_{i=2}^{n-1} d_i \mathcal{L}\left[N_{i,p}\left(\hat{\tau}_j\right)\right] + g_1 \mathcal{L}\left[N_{1,p}\left(\hat{\tau}_j\right)\right] + g_n \mathcal{L}\left[N_{n,p}\left(\hat{\tau}_j\right)\right]. \tag{6.10}$$

which we recognize as the left hand side of the equation (6.5). By introducing the right hand side of (6.5), this yields the following general 1D collocation scheme

$$\sum_{i=2}^{n-1} d_i \mathcal{L}_{i,j} + g_1 \mathcal{L}_{1,j} + g_n \mathcal{L}_{n,j} = f_j \quad \text{for} \quad j = 2, \ldots, n-1, \tag{6.11}$$

where we have denoted $\mathcal{L}_{ij} = \mathcal{L}\left[N_{i,p}\left(\hat{\tau}_j\right)\right]$ and $f_j = f\left(\hat{\tau}_j\right)$.

This becomes a $(n-2) \times (n-2)$ linear system written in matrix form as

$$\mathbf{L}^{\mathbf{T}}\, \mathbf{d} \;=\; \mathbf{f} \;-\; g_1\, \mathbf{b_1} \;-\; g_n\, \mathbf{b_n}, \tag{6.12}$$

where the stiffness matrix $\mathbf{L} \in \mathbb{R}^{(n-2)\times(n-2)}$ is given as

$$\mathbf{L} = \begin{pmatrix} \mathcal{L}_{2,2} & \cdots & \mathcal{L}_{2,n-1} \\ \vdots & \ddots & \vdots \\ \mathcal{L}_{n-1,2} & \cdots & \mathcal{L}_{n-1,n-1} \end{pmatrix}, \tag{6.13}$$

the unknown displacement vector $\mathbf{d} \in \mathbb{R}^{n-2}$ and load vector $\mathbf{f} \in \mathbb{R}^{n-2}$ is given as

$$\mathbf{d} = \begin{pmatrix} d_2 \\ \vdots \\ d_{n-1} \end{pmatrix} \quad \text{and} \quad \mathbf{f} = \begin{pmatrix} f_2 \\ \vdots \\ f_{n-1} \end{pmatrix}, \tag{6.14}$$

and the left and right boundary vectors $\mathbf{b_1} \in \mathbb{R}^{n-2}$ and $\mathbf{b_n} \in \mathbb{R}^{n-2}$ are given as

$$\mathbf{b_1} = \begin{pmatrix} \mathcal{L}_{1,2} \\ \vdots \\ \mathcal{L}_{1,n-1} \end{pmatrix} \quad \text{and} \quad \mathbf{b_n} = \begin{pmatrix} \mathcal{L}_{n,2} \\ \vdots \\ \mathcal{L}_{n,n-1} \end{pmatrix}. \tag{6.15}$$

What remains is to determine the coefficients g_1 and g_n and solve the linear system using some sort of linear system solver. The coefficients g_1 and g_n in the lifting function is in fact dependent on whether we deal with a dirichlet or a neumann boundary, so we discuss the two cases in the next two sections.

6.2.1 Dirichlet boundary

The 1D problem with dirichlet boundary on both sides is described by the special case $\mathcal{G} = I$,

$$\begin{cases} \mathcal{L}\left[u_h(\hat{\tau}_i)\right] = f(\hat{\tau}_i) \quad \text{in} \quad \Omega, \quad \text{for} \quad i = 2, \ldots, n-1, \\ u_h(a) = u_a \quad \text{and} \quad u_h(b) = u_b, \end{cases} \tag{6.16}$$

where $u_a, u_b \in \mathbb{R}$. Due to the partition of unity property, and that the B-spline basis is interpolatory on the boundary, it is straight forward to impose the dirichlet boundary by doing the following evaluation

$$u_h(a) = g_1 N_{1,p}(\hat{a}) = g_1 = u_a, \tag{6.17}$$

$$u_h(b) = g_n N_{n,p}(\hat{b}) = g_n = u_b. \tag{6.18}$$

So the coefficients g_1 and g_n are actually equal to the boundary values them self. All other B-splines are zero at the boundaries. Since g_1 and g_n are independent of any unknown values they move over to the right hand side of the linear system we are about to obtain. By inserting (6.17) and (6.18) into the general scheme (6.11) and collecting terms, we get a scheme designed for solving (6.16)

$$\sum_{i=2}^{n-1} d_i \, \mathcal{L}_{i,j} = f_j - u_a \mathcal{L}_{1,j} - u_b \mathcal{L}_{n,j} \quad \text{for} \quad j = 2, \ldots, n-1. \tag{6.19}$$

The linear system written in matrix form is given as

$$\mathbf{L^T} \, \mathbf{d} \; = \mathbf{f} \, - \, u_a \, \mathbf{b_1} \, - \, u_b \, \mathbf{b_n}, \tag{6.20}$$

where $\mathbf{L}$, $\mathbf{d}$, $\mathbf{f}$, $\mathbf{b_1}$ and $\mathbf{b_n}$ are unchanged and given in (6.13), (6.14) and (6.15). Solving for $\mathbf{d}$ and inserting it into (6.9) together with $g_1 = u_a$ and $g_n = u_b$ yields the spline solving problem (6.16).

Example 13. ***Homogeneous Dirichlet boundary.***
It is convenient to notice that the system

$$\mathbf{L}^{\mathbf{T}}\,\mathbf{d} = \mathbf{f} \tag{6.21}$$

will solve a test problem for a given linear differential operator $\mathcal{L}$ and load vector f, where the solution satisfies a homogeneous Dirichlet boundary, i.e. the solution is zero on the boundary.

6.2.2 Neumann boundary

The 1D problem for Neumann boundary on both sides is the special case where the first derivative $\mathcal{G} = d/dx$ of the solution should be equal to some scalar coefficient, when evaluated on the boundary. The problem is given as

$$\begin{cases} \mathcal{L}\left[u_h(\hat{\tau}_i)\right] = f(\hat{\tau}_i) \quad \text{in} \quad \Omega, \quad \text{for} \quad i = 2, \dots, n-1, \\ u_h'(a) = c_a \quad \text{and} \quad u_h'(b) = c_b, \end{cases} \tag{6.22}$$

where c_a and c_b are known scalars. We want to be able to evaluate $u_h'(\xi)$ on the boundary, so we apply relation (6.10) using $\mathcal{L} = d/dx$ to get

$$\frac{d}{dx}u_h(\xi) = u_h'(\xi) = \sum_{i=2}^{n-1} d_i N_{i,p}'(\xi) + g_1 N_{1,p}'(\xi) + g_n N_{n,p}'(\xi). \tag{6.23}$$

When imposing the Neumann boundary conditions directly into the solution space it is necessary to use the result in the following theorem.

Theorem 4. *Given a parameter space $\hat{\Omega} = [\hat{a}, \hat{b}] \subset \mathbb{R}$. On the set of B-spline basis functions $\{N_{i,p}(\xi)\}_{i=1}^{n}$ only $N_{1,p}(\xi)$ and $N_{2,p}(\xi)$ are non-zero when evaluated at the boundary $\xi = \hat{a}$, and only $N_{n-1,p}(\xi)$ and $N_{n,p}(\xi)$ are non-zero when evaluated at the boundary $\xi = \hat{b}$.*

Proof. The proof can be done directly by applying Cox-de Boor recursion formula (3.2). □

We can now write out the evaluation in (6.23)

$$u_h'(a) = g_1 N_{1,p}'(\hat{a}) + d_2 N_{2,p}'(\hat{a}) = c_a, \tag{6.24}$$

$$u_h'(b) = d_{n-1} N_{n-1,p}'(\hat{b}) + g_n N_{n,p}'(\hat{b}) = c_b, \tag{6.25}$$

where only the two outermost B-splines are non-zero. We remember that $u_h(\hat{a}) = u_h(a)$ since $x(\hat{a}) = a$ when using a linear map such as (4.11). Solving for the boundary coefficients g_1 and g_n, we see that

$$g_1 = \frac{c_a - d_2 N_{2,p}'(\hat{a})}{N_{1,p}'(\hat{a})} \quad \text{and} \quad g_n = \frac{c_b - d_{n-1} N_{n-1,p}'(\hat{b})}{N_{n,p}'(\hat{b})}. \tag{6.26}$$

The coefficients are now dependent on the two unknown values d_2 and d_{n-1}. We choose to denote the constant coefficients

$$C_a = \frac{N_{2,p}'(\hat{a})}{N_{1,p}'(\hat{a})} \quad \text{and} \quad C_b = \frac{N_{n-1,p}'(\hat{b})}{N_{n,p}'(\hat{b})}, \tag{6.27}$$

so that (6.26) becomes

$$g_1 = \frac{c_a}{N_{1,p}'(\hat{a})} - d_2 C_a \quad \text{and} \quad g_n = \frac{c_b}{N_{n,p}'(\hat{b})} - d_{n-1} C_b. \tag{6.28}$$

The independent terms of (6.28) will be moved to the right side of the linear system we are about to obtain, and the dependent terms will be moved to the left hand side.

Therefore, by inserting (6.28) into the general scheme (6.11) and collecting terms, we get a scheme designed for solving (6.22)

$$\begin{aligned} &d_2\left[\mathcal{L}_{2,j} - C_a\mathcal{L}_{1,j}\right] + \sum_{i=3}^{n-2} d_i\mathcal{L}_{i,j} + d_{n-1}\left[\mathcal{L}_{n-1,j} - C_b\mathcal{L}_{n,j}\right] \\ &= f_j - \frac{c_a}{N'_{1,p}(\hat{a})}\mathcal{L}_{1,j} - \frac{c_b}{N'_{n,p}(\hat{b})}\mathcal{L}_{n,j} \quad \text{for} \quad j = 2,\ldots,n-1. \end{aligned} \tag{6.29}$$

Written on matrix form, this linear system becomes

$$\left(\mathbf{L}^{\mathbf{T}} - C_a\,\mathbf{B_1} - C_b\,\mathbf{B_n}\right)\mathbf{d} = \mathbf{f} - \frac{c_a}{N'_{1,p}(\hat{a})}\mathbf{b_1} - \frac{c_b}{N'_{n,p}(\hat{b})}\mathbf{b_n}, \tag{6.30}$$

where **L**, **d**, **f**, $\mathbf{b_1}$ and $\mathbf{b_n}$ are unchanged and given in (6.13), (6.14) and (6.15). The left hand side matrices $\mathbf{B_1} \in \mathbb{R}^{(n-2)\times(n-2)}$ and $\mathbf{B_n} \in \mathbb{R}^{(n-2)\times(n-2)}$ are given as

$$\mathbf{B_1} = [\,\mathbf{b_1} \mid \mathbf{0} \mid \cdots \mid \mathbf{0} \mid \mathbf{0}\,], \tag{6.31}$$

$$\mathbf{B_n} = [\,\mathbf{0} \mid \mathbf{0} \mid \cdots \mid \mathbf{0} \mid \mathbf{b_n}\,], \tag{6.32}$$

where **0** is the column vector in $\mathbb{R}^{n-2}$ containing only zeros. Lastly, we need to solve for **d** and construct the resulting spline function to obtain a solution to the 1D neumann problem.

6.2.3 Mixed boundary

In general, the 1D collocation method we have developed can be used to solve all *linear ordinary differential equations*. In addition, the method has been made adaptive in such a way that is is easy to define Dirichlet and Neumann boundaries, both left and right. Of course, there can be some pitfalls. Some we have discovered and most probably some we have not discovered. There will also be restrictions on what will work and what will not work. Even though, we will do some numerical tests in chapter 8 on test problems and try to determine some rules to follow alongside each problem, and hopefully also in general. We will also see what we can obtain in terms of correctness and convergence rates on our test examples. But for now, we want to illustrate how to solve a test problem with mixed boundary.

Example 14. ***Test problem with mixed boundary.***
We solve the 1D Poisson problem $\mathcal{L} = d^2/dx^2$ with Dirichlet condition on the left boundary and Neumann condition on the right boundary

$$\begin{cases} u'' = f \quad in \quad \Omega = [0,1], \\ u(0) = u_a \quad and \quad u'(1) = c_b, \end{cases} \tag{6.33}$$

with some source function f yielding the exact solution u. The approximate solution u_h is given by (6.9) in terms of the B-spline coefficients **d** *which are obtained by solving the system*

$$(\mathbf{L^T} - C_b\mathbf{B_n})\mathbf{d} = \mathbf{f} - u_a\mathbf{b_1} - \frac{c_b}{N'_{n,p}(\hat{b})}\mathbf{b_n}. \tag{6.34}$$

where the entries of **L** *are $\mathcal{L}_{i,j} = N''_{i,p}(\hat{\tau}_j)$.*

6.3 A Change of Basis

It is clear that the differential operator $\mathcal{L}$ is given as a differentiation with respect to the coordinate x in physical space. Therefore, it is necessary to perform a change of basis when applying this operator to a given spline function. We consider the two cases where $\mathcal{L} = d/dx$ and $\mathcal{L} = d^2/dx^2$. The first derivative of some arbitrary function $k(\xi)$ is

$$k_x \;=\; \xi_x k_\xi = (x_\xi)^{-1}k_\xi. \tag{6.35}$$

The second derivative is given as

$$\begin{aligned} k_{xx} &= (k_x)_x \;=\; \xi_{xx}k_\xi \;+\; \xi_x k_{\xi x} \;=\; \xi_{xx}k_\xi + \xi_x(k_\xi)_x \\ &= \xi_{xx}k_\xi \;+\; \xi_x(k_{\xi\xi})\xi_x \;=\; \xi_{xx}k_\xi \;+\; (\xi_x)^2(k_{\xi\xi}) \\ &= -\frac{x_{\xi\xi}}{(x_\xi)^3}k_\xi \;+\; \frac{1}{(x_\xi)^2}(k_{\xi\xi}) = \; -x_{\xi\xi}(x_\xi)^{-3}k_\xi \;+\; (x_\xi)^{-2}(k_{\xi\xi}). \end{aligned} \tag{6.36}$$

The usual subscript notation for derivatives is assumed to be known by the reader. In these calculations we have used the chain rule and the product rule for derivatives, together with

$$\xi_x = (x_\xi)^{-1}, \tag{6.37}$$

$$\xi_{xx} = (\xi_x)_x = \left(\frac{1}{x_\xi}\right)_x = -\frac{1}{(x_\xi)^2}(x_\xi)_x = -\frac{1}{(x_\xi)^2}x_{\xi\xi}\xi_x = -\frac{x_{\xi\xi}}{(x_\xi)^3}. \tag{6.38}$$

By applying equation (6.35) we can for example calculate equation (6.23), and by applying (6.36) we can calculate the entries of the stiffness matrix **L** in example 14. Both of these have been used in the implementation of the 1D collocation solver. The spline functions we use in the collocation method is always a linear combination of B-splines. The clue here is to use the change of basis procedure to express the derivatives of functions in terms of ξ, since we can use the implementation of the B-spline derivative formula from (3.3) to determine them.

6.4 Collocation points

The choice of collocation points is in general difficult. We know that the Schoenberg-Whitney nesting conditions (5.7) are necessary, at least for interpolation. But we do not know if they are sufficient for collocation. We know however that some choices of collocation points will violate these conditions for large n, while other points will not. If they do not, do we achieve convergence? Either way, in this section we will present some different set of points used in literature on the field.

6.4.1 Uniform points

Uniformly distributed points are the most basic type of collocation points. They are equidistant on the domain $\hat{\Omega} = [\hat{a}, \hat{b}]$ and defined by

$$\hat{\tau}_i = (i-1)h + \hat{a} \quad \text{for} \quad i = 1, \ldots, n, \tag{6.39}$$

where $h = (\hat{b} - \hat{a})/(n-1)$.

6.4.2 Greville Abscissae

The image of the Greville abscissae was initially introduced by [9] and have been widely adopted as the default choice for collocation points in the IGA collocation literature. We have already introduced them in chapter 5, but we restate them here. For a given knot vector $\Xi = [\xi_1, \ldots, \xi_{n+p+1}]$ the associated n Greville points in parameter space are calculated from

$$\hat{\tau}_i = \frac{\xi_{i+1} + \xi_{i+2} + \ldots + \xi_{i+p}}{p} \quad \text{for} \quad i = 1, \ldots, n, \tag{6.40}$$

where p is the polynomial degree.

6.4.3 Knot Maxima

Another promising possibility is to locate the collocation points on the parameter values at which a B-spline function achieves its maximum. The formal definition is

$$\hat{\tau}_i = \underset{\xi_i < \xi < \xi_{i+p+1}}{\arg\max} \; N_{i,p}(\xi) = \left\{ \xi \in [\xi_i, \xi_{i+p+1}] \mid \frac{d}{d\xi} N_{i,p}(\xi) = 0 \right\}, \tag{6.41}$$

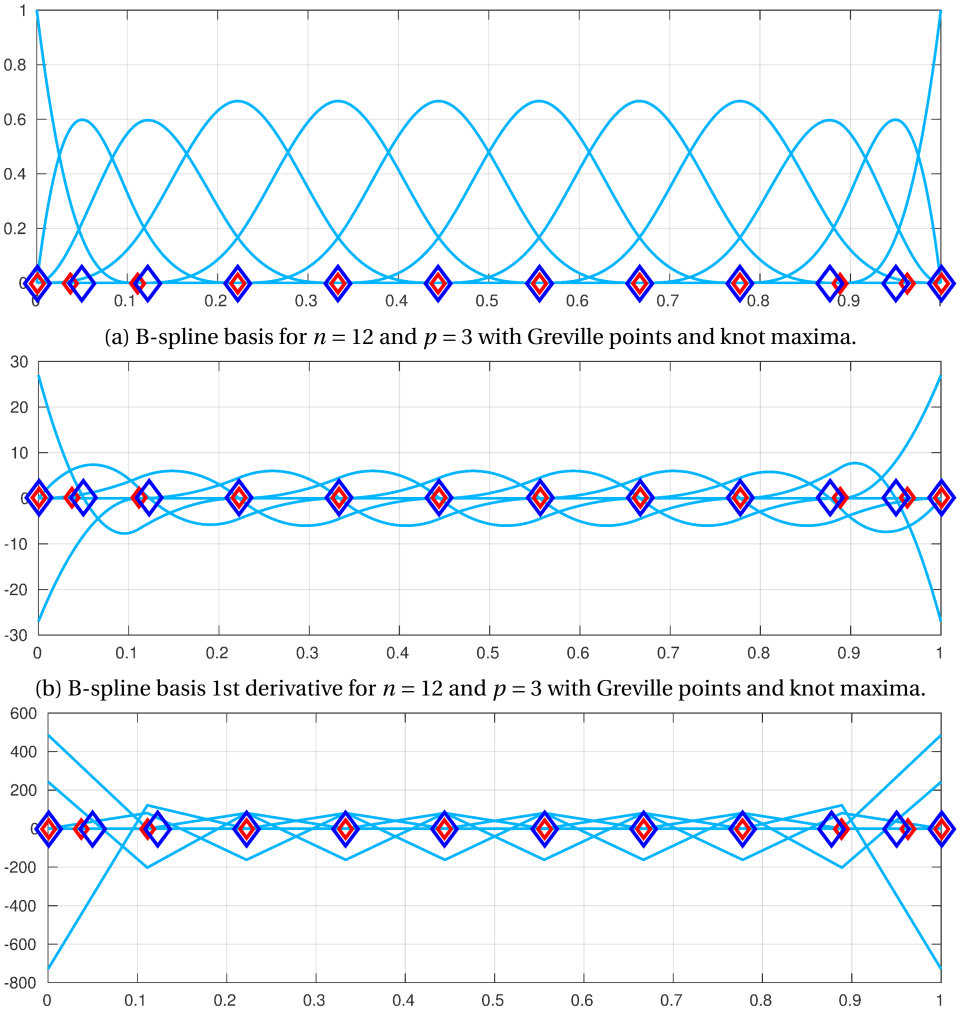
for $i = 1, \ldots, n$. These points were initially introduced by [10].

6.4.4 Gauss-Legendre

These points are associated with the Gauss-Legendre quadrature rule used in numerical integration. They are in fact defined to be *the roots* of the orthogonal Legendre polynomials, and can be found by iterative methods such as the Newton-Raphson method. A fast and accurate method for computing the roots is proposed by [11].

6.4.5 Superconvergent points

Another interesting possibility was, as recently as in late 2014, proposed by [12]. These points are derived from superconvergent theory and exploits some efficient convergence properties.



(a) B-spline basis for $n = 12$ and $p = 3$ with Greville points and knot maxima.

(b) B-spline basis 1st derivative for $n = 12$ and $p = 3$ with Greville points and knot maxima.

(c) B-spline basis 2nd derivative for $n = 12$ and $p = 3$ with Greville points and knot maxima.

Figure 6.1: An illustration of Greville (small red diamonds) and knot maxima (large blue diamonds) collocation points. The first and last collocation points can be ignored in computation of the collocation methods since they correspond to the lifting function. Both choices of collocation points lies under the support of basis functions.

6.5 Error Analysis

In the following, we will use the usual notation for function norms and seminorms, namely for the L^∞-norm, and $W^{\infty,1}$-, $W^{\infty,2}$-seminorms, L^2-norm and H^1-, H^2-seminorms. Given a problem with an exact solution u and a numerical solution u_h, the norms are defined as

$$|| u - u_h ||_{L^\infty} = \max | u - u_h |, \tag{6.42}$$

$$| u - u_h |_{W^{1,\infty}} = \max | u' - u'_h |_{L^\infty}, \tag{6.43}$$

$$| u - u_h |_{W^{2,\infty}} = \max | u'' - u''_h |_{L^\infty}, \tag{6.44}$$

$$|| u - u_h ||^2_{L^2} = \int_\Omega | u - u_h |^2, \tag{6.45}$$

$$| u - u_h |^2_{H^1} = \int_\Omega | u' - u'_h |^2, \tag{6.46}$$

$$| u - u_h |^2_{H^2} = \int_\Omega | u'' - u''_h |^2, \tag{6.47}$$

where u' and u_h are the usual derivatives with respect to x. The number defined by

$$\frac{|| u - u_h ||_{L^\infty}}{|| u ||_{L^\infty}}. \tag{6.48}$$

is the *relative error* of the L^∞ norm. A similar definition holds for the other norm and seminorms also.

We now restrict our attention to one dimensional collocation methods, and to the case of a second order differential equation. It has proved by [13] that if collocation points are chosen suitably the collocation method converges with optimal theoretical rate, which is of order p for L^∞ and $W^{1,\infty}$, and of order $p-1$ for $W^{1,\infty}$. That the collocation points are chosen suitably means that they are chosen according to theorem 3.2 in [13]. The Greville points meets these criteria, and hence shows optimal convergence as shown by [13]. The superconvergent points also exploit optimal convergence rate, at least for 1st derivative norms, as shown by [12].

The Greville abscissae is considered to be the best choice for many problems, and for interpolation problems they are proved to be stable up to degree $p = 3$. Even though, there are numerical examples of instability for degrees higher than 19. These examples are provided by [14] and are calculated on particular non-uniform meshes where the lengths of each consecutive knot span forms a geometric sequence. The only choice of collocation points which is proved to be stable for any mesh and degree is the one proposed by Demko [15]. These points are referred to as *Demko abscissae* and [13] provide numerical results where they converge with optimal rates for given problems.

Chapter 7

Implementation

In this chapter we present some smart choices to make when programming numerical methods in general, and some choices to consider when programming spline and collocation methods. There are a few observations one can make in order to do computation cheaper in terms of time and floating point operations, or just to make the assembly processes simpler. We will also give a general, but shallow, recipe on how to implement a B-spline code and collocation method. Since all functions and are represented in a discrete sense in programming, using vectors and matrices, we will discuss the finite realization of some matrices and geometrical mappings.

7.1 Splines

In good programming practice it can be convenient to verify the conditions in lemma 1 are fulfilled when implementing programs utilizing B-splines. The conditions from theorem 3 that ensures unique solutions in interpolation can also be convenient to verify. However, software such as Matlab will give warnings if the interpolation matrices are close to being singular.

We remember that at most $p+1$ entries on each row of the interpolation matrix $\mathbf{N_I}$ are non-zero. Those entries are consecutive. This gives the matrix a band structure which can be exploited in linear solvers such as the Gaussian elimination method, amongst others. For interpolation problems the backslash operator in Matlab is sufficient.

7.1.1 Evaluation of Basis Functions

B-spline basis functions is in general evaluated in parameter space. Independent of the knot vector used, each basis function in the set $\{N_{i,p}(\xi)\}_{i=1}^{n}$ is non-zero on $p+1$ knot spans, according to property 4 in lemma 1. So, it is redundant to evaluate the function $N_{i,p}(\xi_j)$ if $\xi_j \notin [\xi_i, \xi_{i+p+1}]$. Instead, one can make use of the spline theory and predetermine it to be zero.

The recursive formula for the B-splines, described in equation (3.1) and (3.2), is to be evaluated quite a few times during any collocation solution algorithm. It is therefore essential that it is possible to evaluate them in an efficient way. It is possible to evaluate (3.1) and (3.2) as it stands, that is by creating a recursive algorithm by taking i, p, Ξ, ξ_j as input values, and returning $N_{i,p}(\xi_j)$ simply by calling itself recursively. However, this will not be an efficient implementation. When running through the recursive algorithm, there will be functions with the exact same parameters being evaluated several times. By using dynamic programming one can make an algorithm that runs in polynomial time, instead of exponential time, as described in [27, p.57-60].

7.1.2 The B-spline Basis Matrix

It is convenient to collect the set of basis functions $\{N_{i,p}(\xi)\}_{i=1}^{n}$ in a vector

$$\left[N_{1,p}(\xi), N_{1,p}(\xi), \ldots, N_{n,p}(\xi)\right]^T, \tag{7.1}$$

where each element is a continuous function. In programming however, we have to store function values evaluated on a distinct set of points as a vector. We have already discussed how the spline curve 4.2 and the spline surface 4.6 can be expressed as a linear mapping using matrix notation, and we will now continue that discussion.

In the following we present the three matrix types; *interpolation matrix*, *knot matrix* and *visualization matrix*. The distinction between the three is the set of points in parameter space where the basis functions are evaluated at. We use these matrices in the geometrical map to transform curves, surfaces, interpolation points, knot points and knot lines into physical space. Table 7.1 contains an overview of these transformations. We only consider two dimensions here, but the principles are easy transferable to one dimension and higher dimensions.

Table 7.1: Different geometrical maps for tensor spline surfaces are presented in this table. The (x, y, z)-coordinates of the control points in physical space are collected in the matrices **X**, **Y** and **Z**. The control points and the basis matrices $\mathbf{N_I}$, $\mathbf{N_K}$ and $\mathbf{N_V}$ is what governs the geometrical map.

	X	**Y**	**Z**
Interpolation points	$\mathbf{N_I^T\,X\,M_I}$	$\mathbf{N_I^T\,Y\,M_I}$	$\mathbf{N_I^T\,Z\,M_I}$
Knot points	$\mathbf{N_K^T\,X\,M_K}$	$\mathbf{N_K^T\,Y\,M_K}$	$\mathbf{N_K^T\,Z\,M_K}$
Knot lines x-direction	$\mathbf{N_V^T\,X\,M_K}$	$\mathbf{N_V^T\,Y\,M_K}$	$\mathbf{N_V^T\,Z\,M_K}$
Knot lines y-direction	$\mathbf{N_K^T\,X\,M_V}$	$\mathbf{N_K^T\,Y\,M_V}$	$\mathbf{N_K^T\,Z\,M_V}$
The whole surface	$\mathbf{N_V^T\,X\,M_V}$	$\mathbf{N_V^T\,Y\,M_V}$	$\mathbf{N_V^T\,Z\,M_V}$

Interpolation matrix

The interpolation matrices $\mathbf{N_I} \in \mathbb{R}^{n\times n}$ and $\mathbf{M_I} \in \mathbb{R}^{m\times m}$ consist of basis functions evaluated on the interpolation points in parameter space $\{\hat{\xi}_i\}_{i=1}^{n}$ and $\{\hat{\eta}_i\}_{i=1}^{m}$.

$$\mathbf{N_I} = \begin{pmatrix} N_{1,p}(\hat{\xi}_1) & \dots & N_{1,p}(\hat{\xi}_n) \\ \vdots & \ddots & \vdots \\ N_{n,p}(\hat{\xi}_1) & \dots & N_{n,p}(\hat{\xi}_n) \end{pmatrix}, \quad \mathbf{M_I} = \begin{pmatrix} M_{1,q}(\hat{\eta}_1) & \dots & M_{1,q}(\hat{\eta}_m) \\ \vdots & \ddots & \vdots \\ M_{m,q}(\hat{\eta}_1) & \dots & M_{m,q}(\hat{\eta}_m) \end{pmatrix} \tag{7.2}$$

Knot matrix

The knot matrices $\mathbf{N_K} \in \mathbb{R}^{n\times(n+p+1)}$ and $\mathbf{M_K} \in \mathbb{R}^{m\times(m+q+1)}$ consist of basis functions evaluated on the knot points in parameter space $\{\xi_i\}_{i=1}^{n+p+1}$ and $\{\eta_i\}_{i=1}^{m+q+1}$.

$$\mathbf{N_K} = \begin{pmatrix} N_{1,p}(\xi_1) & \dots & N_{1,p}(\xi_{n+p+1}) \\ \vdots & \ddots & \vdots \\ N_{n,p}(\xi_1) & \dots & N_{n,p}(\xi_{n+p+1}) \end{pmatrix}, \quad \mathbf{M_K} = \begin{pmatrix} M_{1,q}(\eta_1) & \dots & M_{1,q}(\eta_{m+q+1}) \\ \vdots & \ddots & \vdots \\ M_{m,q}(\eta_1) & \dots & M_{m,q}(\eta_{m+q+1}) \end{pmatrix}. \tag{7.3}$$

Visualization matrix

For plotting and visualization of surfaces, the basis matrices needs to be evaluated at a large number n_V of points uniformly distributed in the interval $[\xi_i, \xi_{i+p+1}]$, such that $\xi \in \{u_1, \dots, u_{n_V}\}$ and $\eta \in \{v_1, \dots, v_{n_V}\}$. We denote the visualization matrices as
$\mathbf{N_V} \in \mathbb{R}^{n\times n_V}$ and $\mathbf{M_V} \in \mathbb{R}^{m\times n_V}$ and define them as

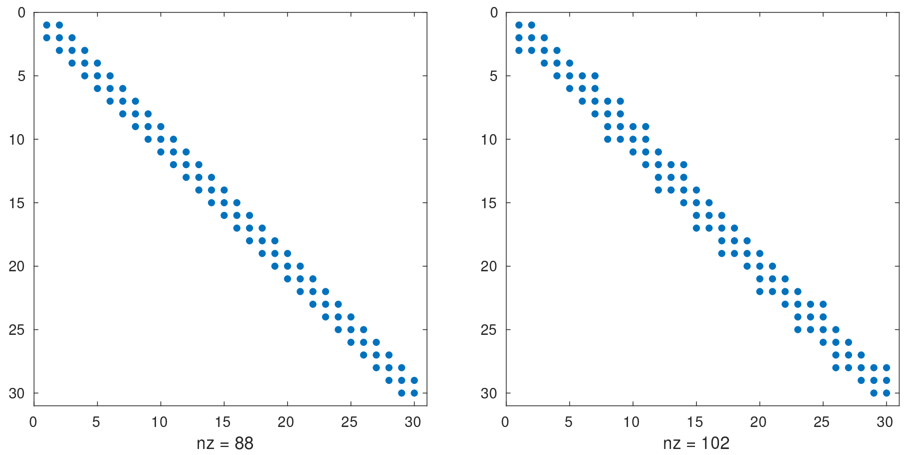
$$\mathbf{N_V} = \begin{pmatrix} N_{1,p}(u_1) & \dots & N_{1,p}(u_{n_V}) \\ \vdots & \ddots & \vdots \\ N_{n,p}(u_1) & \dots & N_{n,p}(u_{n_V}) \end{pmatrix}, \quad \mathbf{M_V} = \begin{pmatrix} M_{1,q}(v_1) & \dots & M_{1,q}(v_{n_V}) \\ \vdots & \ddots & \vdots \\ M_{m,q}(v_1) & \dots & M_{m,q}(v_{n_V}) \end{pmatrix}. \tag{7.4}$$

7.2 Collocation

In chapter 5 we introduced the matrix $\mathbf{N}$ as the collocation matrix. In (7.2) we introduced the matrix $\mathbf{N_I}$ to be the interpolation matrix. In fact we see that $\mathbf{N} = \mathbf{N_I}$. As we have already discussed, interpolation is a collocation method for the identity operator, and hence interpolation is a special case of collocation. From this, it becomes clear that a stiffness matrix $\mathbf{L} \in \mathbb{R}^{n\times n}$, with entries $\{\mathcal{L}_{i,j}\}_{i,j=1}^{n}$, is in fact a more general type of collocation matrix, and that $\mathbf{L} = \mathbf{N}$ if $\mathcal{L} = I$.

When implementing numerical methods, it is important to exploit the sparsity of matrices. From interpolation we know that the matrix $\mathbf{N}$ is sparse and has a bandwidth of only $p+1$, since each basis function is only non-zero on so many knot spans. The matrix $\mathbf{N}$ consists of B-splines evaluated at collocation points and the matrix $\mathbf{L}$ matrix will, for most problems, consist of some derivative of the B-spline function evaluated at collocation points. Since we know that the derivative of a zero function is zero, we know for certain that the collocation matrix $\mathbf{L}$ is always sparse and has *at most* a bandwidth of $p+1$ for one dimensional problems. This can of course vary between problem, but it can be valuable to figure this out before hand to save computation costs both in the assembly process, and in the linear system solver.

Figure 7.1a shows the matrix structure of the stiffness matrix obtained when solving the one dimensional Poisson problem with homogeneous Dirichlet boundary using the collocation scheme from equation (6.21) for $n = 32$. Figure 7.1b shows the matrix structure from the same problem, but now with Neumann conditions on the right hand side boundary.



(a) Dirichlet boundary for $n = 32$.

(b) Neumann boundary for $n = 32$.

Figure 7.1: Sparse matrix structure for the stiffness matrix obtained from one dimensional collocation methods. The number of non zero elements, marked by blue points, are 88 and 102.

7.3 A general recipe

A numerical spline approximation method consists of two main steps.

1. Determine the spline degrees p, q and knot vectors $\Xi, \mathcal{H}$.

2. Determine B-spline coefficients $\mathbf{Z}$ from given data according to some formula.

For tensor product spline surfaces in two dimensions, calculated by equation 4.6, the coefficients $\mathbf{Z}$ is pre-defined, and they were introduced as control points. The same holds for spline curves in one dimension. For the Variation diminishing spline approximation, the control points are also pre-determined, but are now a function value at the knot averages. However, for general spline interpolation the coefficients are the solution of a linear set of equations 5.15. Collocation methods also requires a linear system to be solved for finding the coefficients, but the assembly process used to construct the linear system is different from the interpolation case.

Chapter 8

Numerical Experiments

The first goal for this chapter is to test and verify the implementation of our 1D collocation solver. We consider two test problems taken from [13] and provide solution plots for each of them. The problems consist of a second order differential equation with a mix of Dirichlet and Neumann bondary conditions. The second goal of this chapter is to test the convergence rates discussed in section 6.5, at least for the Greville points. We will also test an alternative choice of collocation points, namely the knot maxima, and look at how they perform in comparison with Greville points.

The following properties are used for all experiments in this chapter:

- Open and uniform knot vectors
- The parameter space $\hat{\Omega} = [0,1]$
- The physical space $\Omega = [0,1]$
- Linear map $x_L : \hat{\Omega} \rightarrow \Omega$ as described in (4.11)

8.1 1D source problem with Dirichlet

The first test example we study is the following source problem

$$\begin{cases} -u'' + u' + u = f(x), \quad \forall x \in (0,1) \\ u(0) = u(1) = 0, \end{cases} \tag{8.1}$$

where the source function is $f(x) = (1+4\pi^2)\sin(2\pi x) + 2\pi\cos(2\pi x)$. This problem admits the exact solution

$$u = \sin(2\pi x). \tag{8.2}$$

This problem is solved by the collocation scheme for homogeneous Dirichlet problems described in equation (6.20). The linear differential operator is now defined by

$$\mathcal{L} = -\frac{d^2}{dx^2} + \frac{d}{dx} + I \tag{8.3}$$

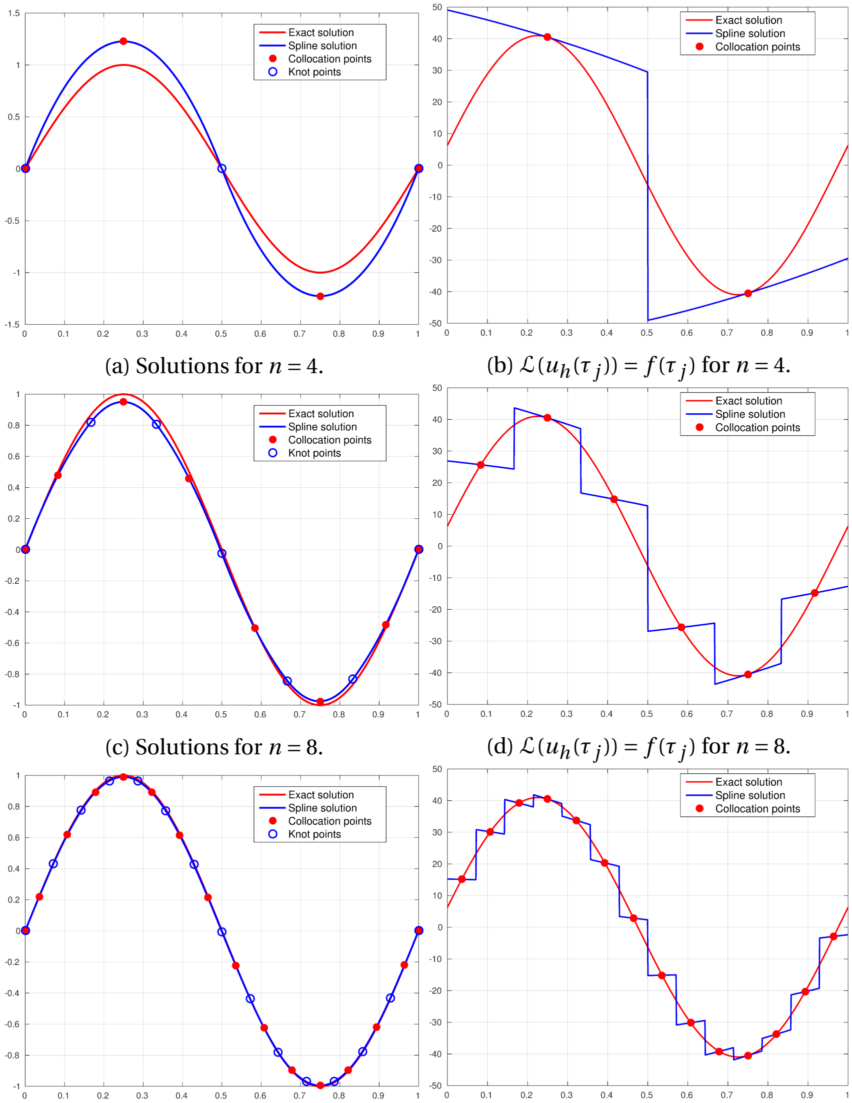
and hence the stiffness matrix **L** has the entries

$$\mathcal{L}_{ij} = -\frac{d^2}{dx^2} N_{i,p}\left(\hat{\tau}_j\right) + \frac{d}{dx} N_{i,p}\left(\hat{\tau}_j\right) + N_{i,p}\left(\hat{\tau}_j\right). \tag{8.4}$$

We have solved this problem on the Greville abscissae for $n = 4, 8, 16$ using quadratic splines. The resulting plots are shown in figure 8.1. The plots to the left show the solutions while the plots to the right show the collocation conditions

$$\mathcal{L}(u_h(\tau_j)) = f(\tau_j) \quad \text{for} \quad i = 2, \ldots, n-1, \tag{8.5}$$

where the collocation points τ_j are the image of collocation points in physical space. These conditions, together with the boundary conditions, are fulfilled in each case. We can see that $\mathcal{L}(u_h(\xi))$ is a piecewise linear function. It is linear on each knot span, and discontinuous at each knot. This is due to the fact that we use splines of degree $p = 2$. Since p is even one can also notice how the collocation points are placed in the center of the knot spans.



(a) Solutions for $n = 4$.

(b) $\mathcal{L}(u_h(\tau_j)) = f(\tau_j)$ for $n = 4$.

(c) Solutions for $n = 8$.

(d) $\mathcal{L}(u_h(\tau_j)) = f(\tau_j)$ for $n = 8$.

(e) Solutions for $n = 16$.

(f) $\mathcal{L}(u_h(\tau_j)) = f(\tau_j)$ for $n = 16$.

Figure 8.1: 1D source problem with Dirichlet boundary conditions solved using 1D collocation method, Greville collocation points and quadratic splines.

8.2 1D source problem with Dirichlet-Neumann

The next test example is also a source problem. We now have a different source function and a Neumann condition on the right hand side, hence a different solution. The problem is given as

$$\begin{cases} -u'' + u' + u = f(x), & \forall x \in (0,1) \\ u(0) = u'(1) = 0, & \end{cases} \tag{8.6}$$

where the source function is $f(x) = (1+4\pi^2)\cos(2\pi x) - 2\pi\sin(2\pi x) - 1$. The problem admits the exact solution

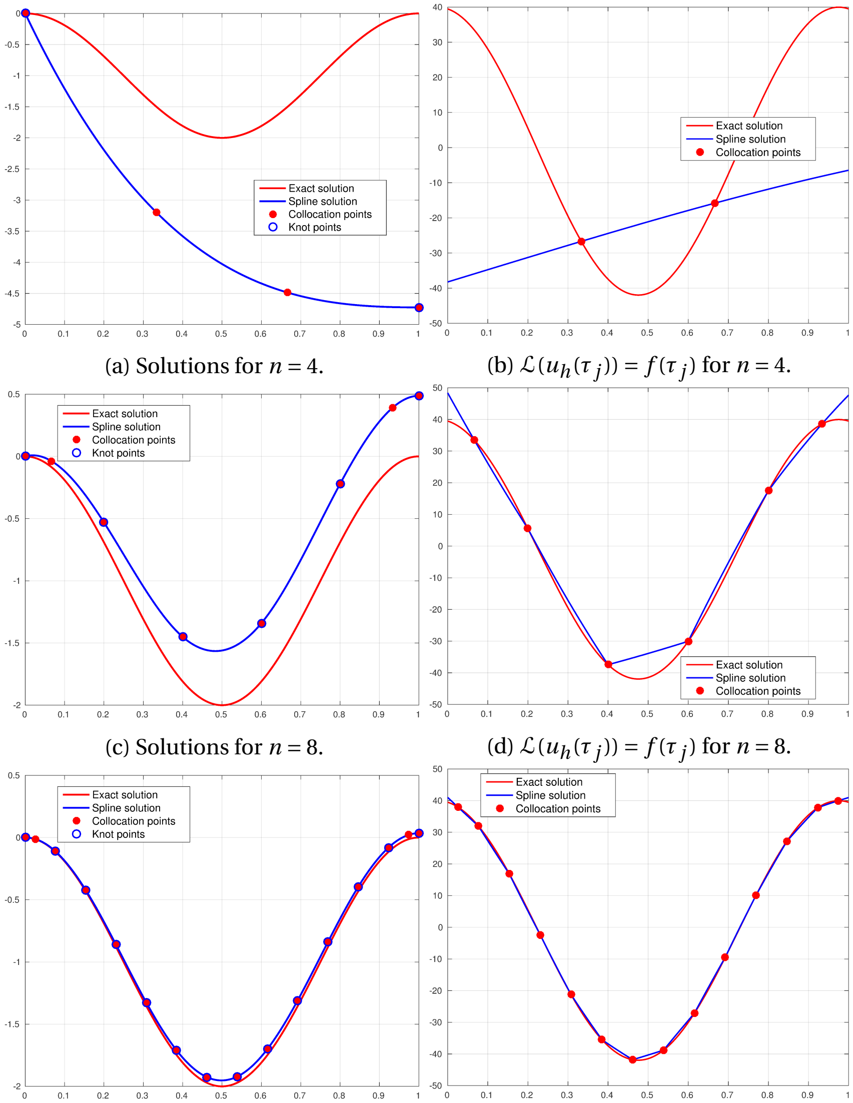
$$u = \cos(2\pi x) - 1. \tag{8.7}$$

This problem is now solved by the mixed boundary collocation scheme

$$(\mathbf{L^T} - C_b\mathbf{B_n})\mathbf{d} = \mathbf{f} - u_a\mathbf{b_1} - \frac{c_b}{N'_{n,p}(\hat{b})}\mathbf{b_n}, \tag{8.8}$$

where $u_a = c_b = 0$. The linear operator $\mathcal{L}$, and hence the stiffness matrix $\mathbf{L}$ is unchanged from the Dirichlet case.

The plots in figure 8.2 shows solutions of the problem for $n = 4, 8, 16$ using qubic splines. The collocation conditions (8.5) are fulfilled on the interior in each case, as seen in the right hand side plots. The Neumann boundary also seems to be fulfilled since the solution tends towards zero when $\xi \to 1$. The solution is not a good fit for $n = 4$ but it converges towards the exact solution as n becomes larger. In this case, $\mathcal{L}(u_h(\xi))$ is a linear function on each element with C^0-continuity at knot points due to the fact that $p = 3$. One can also notice that most collocation points are place placed at the knots, because of the odd spline degree.



(a) Solutions for $n = 4$.

(b) $\mathcal{L}(u_h(\tau_j)) = f(\tau_j)$ for $n = 4$.

(c) Solutions for $n = 8$.

(d) $\mathcal{L}(u_h(\tau_j)) = f(\tau_j)$ for $n = 8$.

(e) Solutions for $n = 16$.

(f) $\mathcal{L}(u_h(\tau_j)) = f(\tau_j)$ for $n = 16$.

Figure 8.2: The 1D source problem with Dirichlet-Neumann boundary conditions solved using 1D collocation method, Greville collocation points and qubic splines.

8.3 Results on Convergence

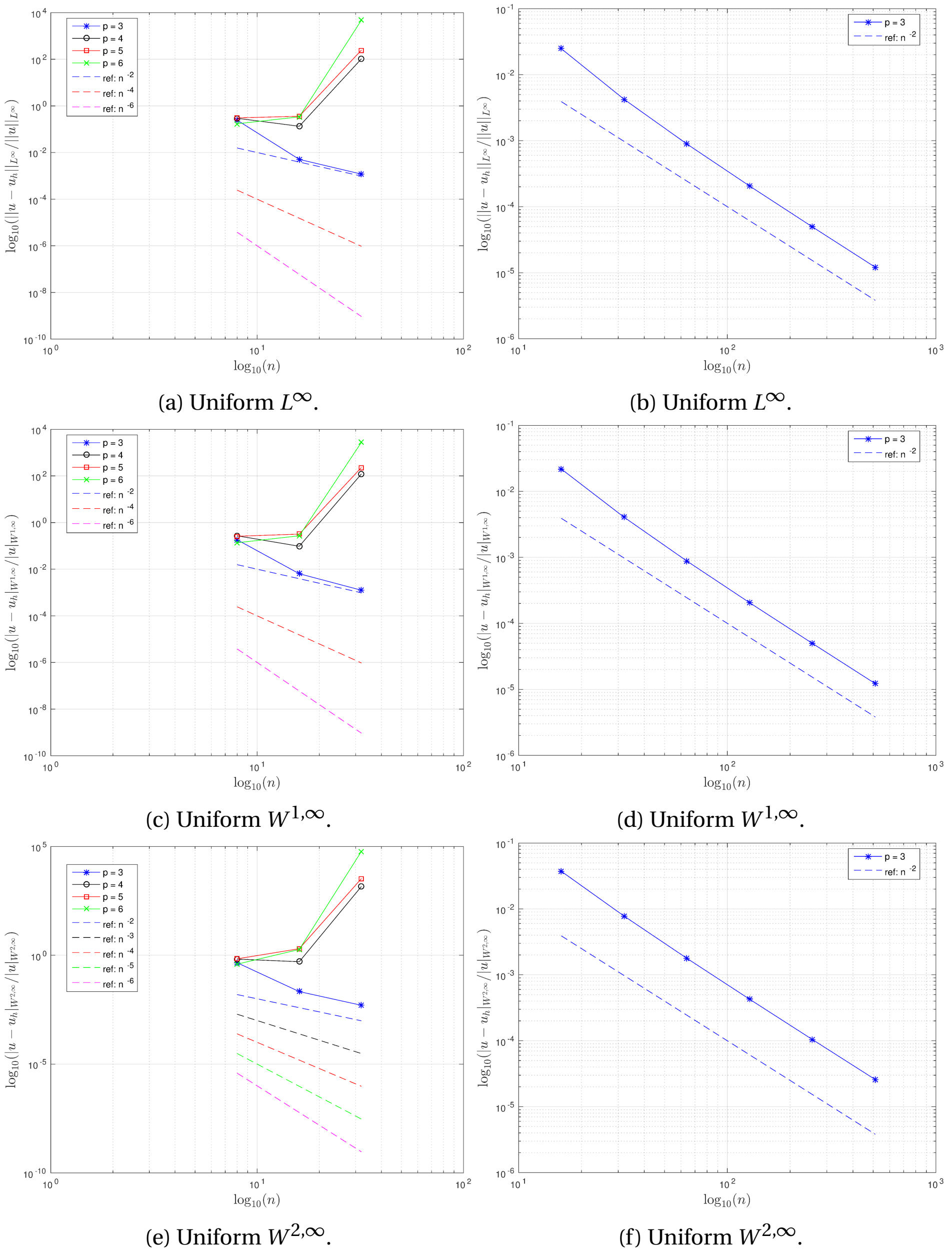
We are now moving forward to test convergence rates. The two test examples are now being tested systematically. We solve for different spline degrees p and different number of basis functions n and compare the numerical solutions u_h to the analytical solutions u using the L^∞-norm (6.42) and the seminorms $W^{1,\infty}$ (6.43) and $W^{2,\infty}$ (6.44). We choose to use these norms since they are the ones used by [13], and we have not included results for $p = 2$, since this case is not covered by the convergence theory given in [13].

First up is to show an example of something that does not work. That is in fact the uniformly distributed collocation points (6.39), the most trivial and naive choice. These points are used when solving (8.4) and the results can be seen in figure 8.3. We can observe that all spline degrees $p > 3$ diverge for increasing n. However, the cubic splines $p = 3$ actually converge with a linear rate. The reason for this is unclear, but we refer to [13] if an answer is sought.

The two last figures present convergence results for the two test problems using both Greville points and knot maxima. The source problem with Dirichlet boundary can be seen in figure 8.4, and the source problem with Dirichlet-Neumann boundary can be seen in figure 8.5.

The plots confirms that in the first two norms an order of convergence p is attained for even degrees, while an order $p-1$ is attained for odd degrees. In the second norm and order of convergence $p-1$ is attained for all degrees.

We would like the reader to compare the plots in this thesis with the plots in [13]. There are some differences in the convergence plots we would like to discuss. The first is that the convergence rate in our plots starts to decrease for large n, at least for the degrees $p = 6$ and $p = 7$. It almost seems like the error hits a minimum and then starts to increase from there. It appears that this minimum value corresponds to a relatively low



(a) Uniform L^∞.

(b) Uniform L^∞.

(c) Uniform $W^{1,\infty}$.

(d) Uniform $W^{1,\infty}$.

(e) Uniform $W^{2,\infty}$.

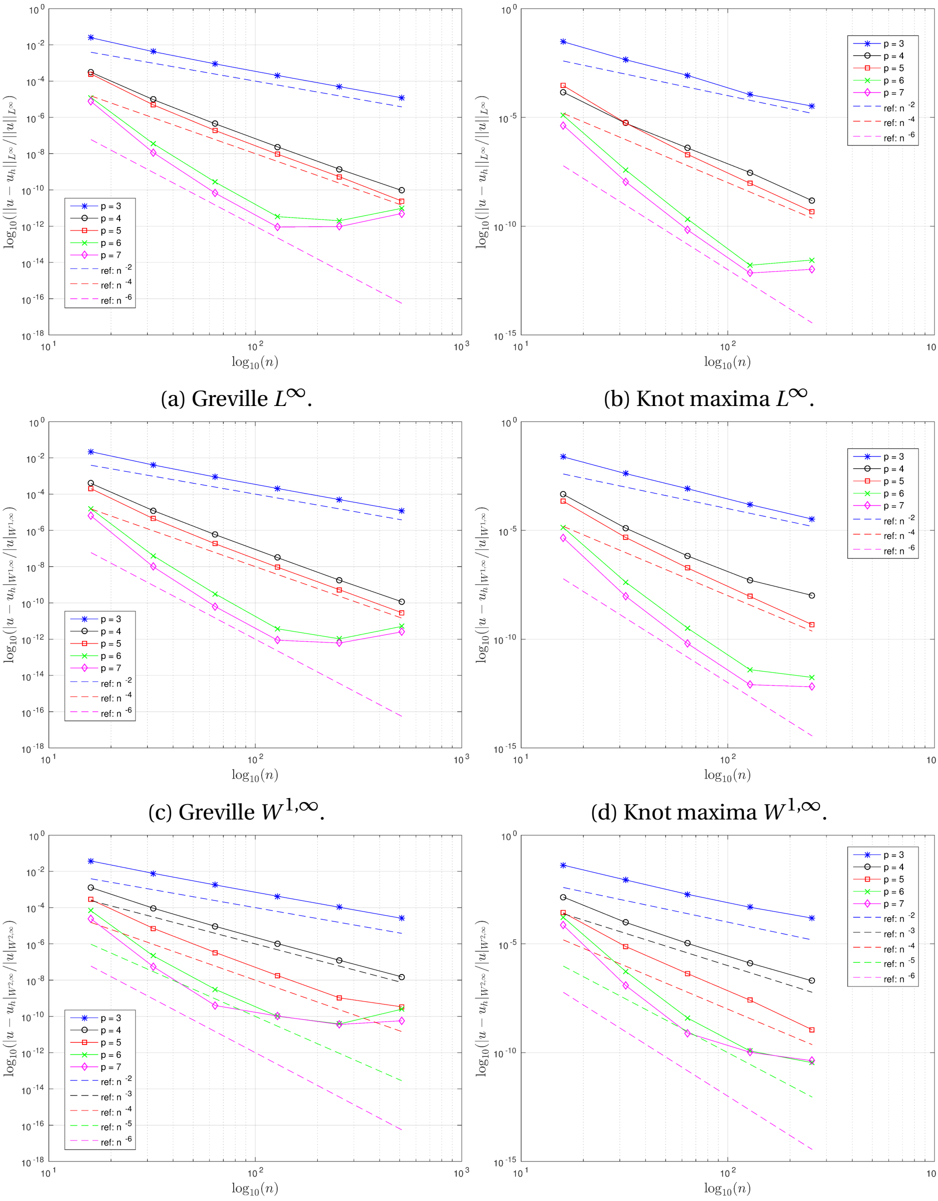
(f) Uniform $W^{2,\infty}$.

Figure 8.3: The 1D source problem with Dirichlet boundary conditions, using uniform collocation points. Relative error in different norms for different p and n. In general these points do not converge, but they do for $p = 3$.

relative error, around the order of 10^{-10} to 10^{-12}. We know that machine precision is accurate up to 10^{-16}, so it does not seem likely to justify the results based on floating points errors either. As any other good scientist, we cannot leave out the possibility of a bug in our code. Of course one would hope that is not the case, and we would like to present other options first.

It is hard to determine exactly why the convergence plots are in the way they are, but it can be valuable to discuss the differences between our collocation method, and the collocation method in [13] and try to present some possible explanations in that sense. The first is that they use Non-Uniform Rational B-Splines (NURBS) as opposed to us who use only uniform and non-rational B-splines. A second thing is that [13] only present the plots for $n \in [32, 128]$. There might be some reason why they did this, but I don't know. However, if our plots was given in the same interval, for the first two norms, they would also seem to be better. This is not the case for the last norm. Either way, we have chosen to include finer grids (larger n) in our plots to see a clearer trend in convergence.

The knot maxima convergence results are shown in the right hand sides. It seems like they perform with the same order of convergence as the Greville points, but a slight difference in the magnitude of errors is observed for certain degrees. When implementing knot maxima we did this in a brute force sense, by searching for the max value in the vector containing the values of the B-spline. The B-splines were evaluated at 1000 elements in the interval $[0, 1]$. Each knot maxima point can therefore potentially by off by an order of 10^{-3}. This can help to explain why the convergence rate is kind of unsteady for the knot maxima, as can be seen in figure 8.4b for $p = 4$ and $p = 5$ for example. An alternative method for finding knot maxima would be to solve equation (6.41) exactly and define the points thereafter. That would perhaps have given smoother convergence results. What is good about our code is that at least the derivatives of solutions are calculated analytically, so the derivative seminorms and the L^{∞}-norm should be equally reliable.



(a) Greville L^{∞}.

(b) Knot maxima L^{∞}.

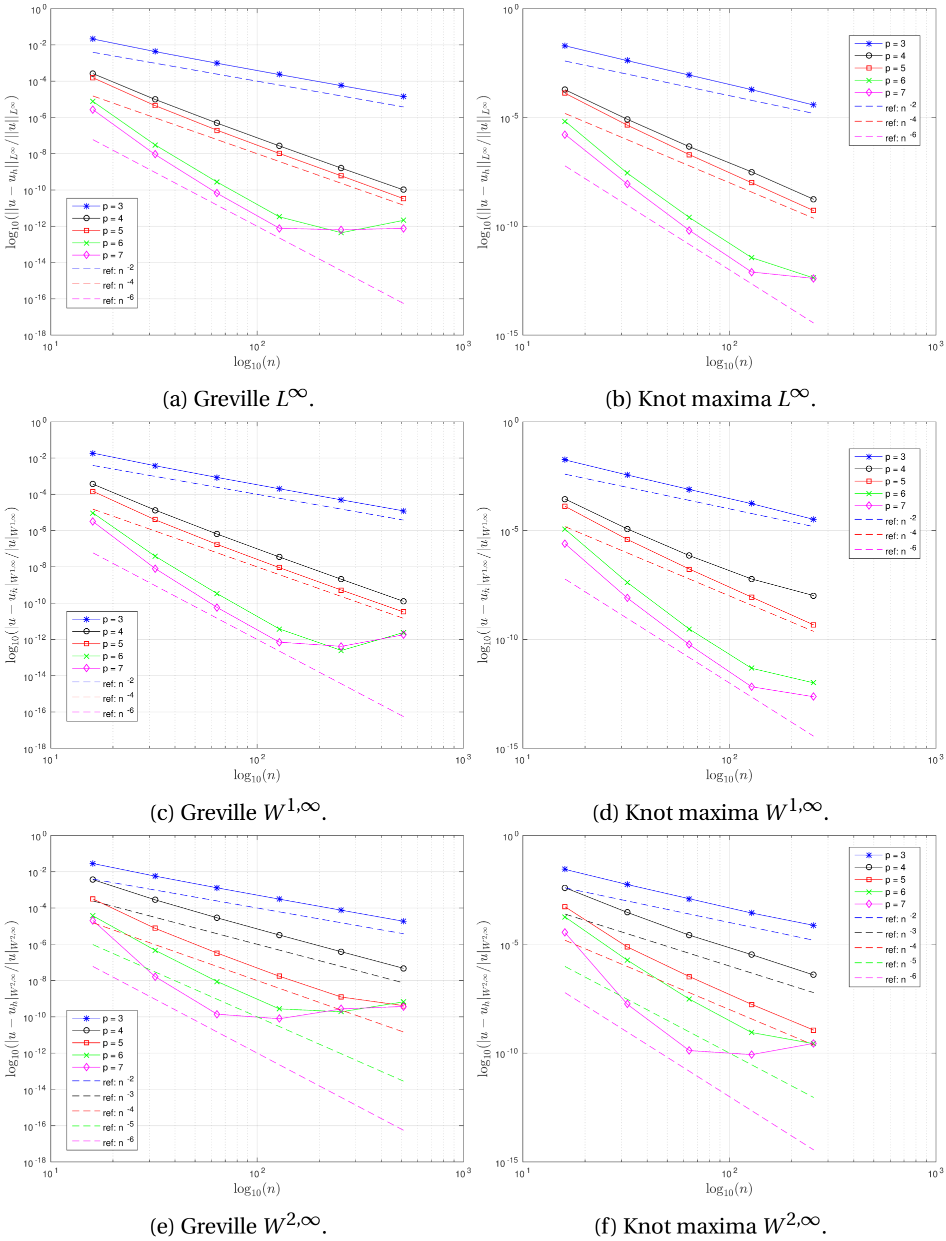
(c) Greville $W^{1,\infty}$.

(d) Knot maxima $W^{1,\infty}$.

(e) Greville $W^{2,\infty}$.

(f) Knot maxima $W^{2,\infty}$.

Figure 8.4: 1D source problem with Dirichlet boundary conditions, using Greville abscissae and knot maxima as collocation points. Relative error in different norms for different p and n.



(a) Greville L^∞.

(b) Knot maxima L^∞.

(c) Greville $W^{1,\infty}$.

(d) Knot maxima $W^{1,\infty}$.

(e) Greville $W^{2,\infty}$.

(f) Knot maxima $W^{2,\infty}$.

Figure 8.5: 1D source problem with Dirichlet-Neumann boundary conditions, using Greville abscissae and knot maxima as collocation points. Relative error in different norms for different p and n.

Chapter 9

Discussion and Conclusion

Our main focus for the convergence results was at collocation points. So what have we figured out in terms of what works and what does not work? For interpolation problems it is clear that the Schoenberg-Whitney nesting conditions are the number one rule to follow when choosing interpolation points. It seems like the Schoenberg-Whitney nesting conditions are also good guidelines for choosing collocation points in collocation methods. When we look at the numerical results, it is evident that there is a correlation between points that works well in interpolations and points that work well in collocation. [13] have provided more reliable results on this topic.

The Greville abscissae was by far the most promising choice of collocation points. An interesting and promising choice was however the knot maxima, giving good convergence rates for the test problems.

The following list sums up the most important achievements in this thesis:

- We have studied and implemented spline methods that are able to approximate and represent complex and flexible geometries built from curves and surfaces in three dimensional space.
- We have studied and implemented spline interpolation methods that can connect data points in two and three dimensions by curves and surfaces, respectively.
- We have developed and implemented a one dimensional spline collocation method that can potentially solve all linear ordinary differential equations, with Dirichlet or Neumann boundary.
- We have tested the collocation method for various collocation points and presented numerical evidence that some collocation points achieve optimal convergence rates.
- All methods are made adaptive and easy to use in the sense that the user has the option to vary spline degrees, the richness of the spline basis, the linear mapping used, the knot vectors used, and the geometry in physical space to where the numerical solution is mapped and presented.

Bibliography

[1] Philip J Davis. *Interpolation and approximation.* Courier Corporation, 1975.

[2] George M Phillips. *Interpolation and approximation by polynomials*, volume 14. Springer Science & Business Media, 2003.

[3] Endre Süli and David F Mayers. *An introduction to numerical analysis.* Cambridge university press, 2003.

[4] W Gander. Change of basis in polynomial interpolation, 2000.

[5] Carl Runge. Über empirische funktionen und die interpolation zwischen äquidistanten ordinaten. *Zeitschrift für Mathematik und Physik*, 46(224-243):20, 1901.

[6] James F Epperson. On the runge example. *Amer. Math. Monthly*, 94(4):329–341, 1987.

[7] Tom Lyche and Knut Mørken. Spline methods draft. 2011.

[8] J Austin Cottrell, Thomas JR Hughes, and Yuri Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA.* John Wiley & Sons, 2009.

[9] Richard W Johnson. Higher order b-spline collocation at the greville abscissae. *Applied Numerical Mathematics*, 52(1):63–75, 2005.

[10] Choong-Gyoo Lim. *A universal parametrization in b-spline curve and surface interpolation and its performance evaluation.* Louisiana State University and Agricultural & Mechanical College, 1998.

[11] Nicholas Hale and Alex Townsend. Fast and accurate computation of gauss–legendre and gauss–jacobi quadrature nodes and weights. *SIAM Journal on Scientific Computing*, 35(2):A652–A674, 2013.

[12] Cosmin Anitescu, Yue Jia, Yongjie Jessica Zhang, and Timon Rabczuk. An isogeometric collocation method using superconvergent points. *Computer Methods in Applied Mechanics and Engineering*, 284:1073–1097, 2015.

[13] F Auricchio, L Beirao Da Veiga, TJR Hughes, A_ Reali, and G Sangalli. Isogeometric collocation methods. *Mathematical Models and Methods in Applied Sciences*, 20 (11):2075–2107, 2010.

[14] Rong-Qing Jia. Spline interpolation at knot averages. *Constructive Approximation*, 4(1):1–7, 1988.

[15] Stephen Demko. On the existence of interpolation projectors onto spline spaces. *Georgia Institute of Technology, Atlanta, Georgia*, 1983.

[16] Wai Yip Kwok, Robert D Moser, and Javier Jiménez. A critical evaluation of the resolution properties of b-spline and compact finite difference methods. *Journal of Computational Physics*, 174(2):510–551, 2001.

[17] GE Fasshauer. Rbf collocation methods as pseudo-spectral methods, 2005.

[18] David Salomon. *Curves and surfaces for computer graphics.* Springer Science & Business Media, 2007.

[19] Yuri Bazilevs and Thomas JR Hughes. Weak imposition of dirichlet boundary conditions in fluid mechanics. *Computers & Fluids*, 36(1):12–26, 2007.

[20] Dominik Schillinger, John A Evans, Alessandro Reali, Michael A Scott, and Thomas JR Hughes. Isogeometric collocation: Cost comparison with galerkin methods and extension to adaptive hierarchical nurbs discretizations. *Computer Methods in Applied Mechanics and Engineering*, 267:170–232, 2013.

[21] Thomas JR Hughes, John A Cottrell, and Yuri Bazilevs. Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Computer methods in applied mechanics and engineering*, 194(39):4135–4195, 2005.

[22] Yousef Saad. *Iterative methods for sparse linear systems*. Siam, 2003.

[23] A Quarteroni, C Canuto, MY Hussaini, and TA Zang. Spectral methods: Fundamentals in single domains, 2006.

[24] İdris Dağ, Dursun Irk, and Ali Şahin. B-spline collocation methods for numerical solutions of the burgers' equation. *Mathematical Problems in Engineering*, 2005 (5):521–538, 2005.

[25] Nicola Bellomo. *Generalized collocation methods: solutions to nonlinear problems*. Springer Science & Business Media, 2007.

[26] Les Piegl and Wayne Tiller. *The NURBS book*. Springer Science & Business Media, 2012.

[27] Kjetil André Johannessen. An adaptive isogeometric finite element analysis. *Master thesis, Norwegian University of Science and Technology, Norway*, 2009.