universität
wien

# DIPLOMARBEIT

Titel der Diplomarbeit

## QUASI-INTERPOLATION
## AND
## SPLINE-TYPE-SPACES

verfasst von

Nicole Susanne Trabauer

angestrebter akademischer Grad

Magistra der Naturwissenschaften (Mag.rer.nat)

| | |
|---|---|
| Matrikel-Nummer: | 0000667 |
| Studienkennzahl lt. Studienblatt: | A 405 |
| Studienrichtung lt. Studienblatt: | Mathematik |
| Betreuer: | Univ.-Prof. Dr. Hans Georg Feichtinger |

Wien, im April 2013

# CONTENTS

# INTRODUCTION

Approximation Theory is one of the oldest branches of Numerical Mathematics. The historical evolution of the methods and results of Approximation Theory traces back to Leonhard Euler [1] in 1777. [2]

Approximation Theory determines how functions can be best approximated by simpler or more tractable functions. The ambition is to make the approximation as close as possible to the effective function. It also explores the size and properties of the error introduced thereby.
Polynomials and rational functions turn out to be the most natural instrument to achieve a fairly good approximation with Building blocks which are too complicated.

The goal of this thesis is to analyse and compare two common methods of Approximation Theory, namely Interpolation and Quasi-interpolation.

The first chapter of this thesis deals with the interpolation of a function by different interpolating polynomials, for instance Lagrange polynomials, Newton Divided Difference Formula, piecewise linear polynomials, cubic splines and B-Splines.

The second chapter is concerned with Spline-type spaces, respectively principal-shift invariant spaces, and the Riesz basic sequences generating them. Within the Spline-type spaces the various defintions of the quasi-interpolant and its

[1] Leonhard Euler (1707-1783) was a Swiss mathematician, who made enormous improvements in lots of mathematical domains - such as analytic geometry, trigonometry, geometry, calculus and number theory.

[2] To read more about the historical background see [31].

characteristics can be discussed. An iterative procedure indicates exactly the correspondence between interpolating and quasi-interpolating methods, and a special case of functions, the radial basis functions, with its qualities are demonstrated.

In the third chapter some experiments in MATLAB are described in order to provide the reader with insight on both approximation methods. A few examples show the differences, the advantages and the disadvantages of both methods. The reader should see that in several cases it is even better to use quasi-interpolation than interpolation. Although the error estimates for quasi-interpolation are not as strong as similar interpolation, it may be advantageous to apply nevertheless quasi-interpolants instead of interpolating methods, because no linear system of equations has to be solved.

I devote this thesis to my beloved parents Josef and Susanne Trabauer. I want to thank them for their love, endurance, patience and their mental and financial assistance.
Furthermore I want to thank my sister Manuela and my brother Bernhard for their confidence and their trust in me.

Finally I want to thank all my friends for their assistance, help and support.

Special thanks goes to my advisor Prof. Dr. Hans Georg Feichtinger for his patience, supervision and his help.

"Gratitude is not only the greatest of virtues, but the parent of all others."
(M.T. Cicero)

# CHAPTER 1

# INTERPOLATION

The first commencements in Interpolation Theory started around 300 BC in the ancient Babylon and Greece. By using not only linear but also more complex interpolation methods they predicted the position of the sun, moon and the familiar planets. The beneficiaries were the farmers, who could calculate their planting strategies of their crops on these predictions.
In early-medieval China and India the first utilization of interpolation theory dates back to around 600 AC. The astronom Liù Zhuó worked with a second order interpolation formula and produced the “Imperial Standard Calendar”.
The word “interpolation” itself has been used first by J. Wallis in 1655.
Interpolation Theory reached the Western countries only after a great revolution in scientific thinking. The theories of Newton influenced the advancement of mathematics including the Interpolation Theory a lot. (“There is no single person who did so much for this field, as for so many others, as Newton.”)[1]

Throughout history interpolation has always been a very important issue in physics, astronomy and mathematics. The application areas concerning mathematics were and still are Numerical Analysis and Approximation Theory. In the mathematical subfield of numerical analysis, interpolation is a method to approximate functions which are known only at a finite set of points.

[1] To read more about the historical background see [20].

Given a sequence of $n+1$ pairwise distinct numbers $x_0, x_1, ..., x_n$ (called **nodes**) and $n+1$ corresponding values $y_0, y_1, ..., y_n$. The problem consists of finding a function $f$ so that

$$f(x_i) = y_i \quad \text{for} \quad i = 0, 1, ..., n \quad .$$

A pair $(x_i, y_i)$ is then called a **data point** and $f$ is called an **interpolant** for all the data points.

The quality of the interpolation not only depends on the chosen interpolation method, but also on the class of interpolating functions.
A very elementary class of interpolating functions are the polynomials. Their big advantage is the approximation of any continuous functions in a bounded interval with arbitrarily small error. But if the data points are given in a wide interval, the polynomials tend to oscillate near the ends of these interval. Therefore polynomials are only used over narrow intervals and larger intervals are subdivided in smaller ones.

This operation methode leads to the interpolation by piecewise polynomials. In that class the **splines** are of extreme importance, because they have very strong smoothness and excellent approximation properties.
Originally splines were developed for ship-building. The challenge was to draw a smooth curve through a set of points. To solve that problem the architects placed metal weights (*nodes*) at the control points, and bend a thin metal or wooden beam (*spline*) through the weights. So the influence of each weight was greatest at the point of contact, and diminished smoothly along the spline. To get more control over a certain region, more weights were added.

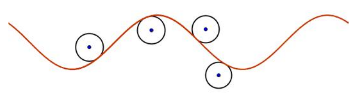


Figure 1.1: Spline and Weights

## 1.1 Lagrange-Interpolation

Lagrange-Interpolation was first discovered by Edward Waring in 1779, and 16 years later Lagrange proposed the same formula.
Nowadays Lagrange-Interpolation is a very well-established classical technique in Interpolation Theory.

Given a set of $n+1$ data points $(x_i, y_i)$, the problem is to find a polynomial $p_n(x) \in \mathbb{P}_n$ such that

$$p_n(x) = a_0 + a_1 x + ... + a_n x^n \quad ,$$

which satisfys the condition

$$p_n(x_i) = y_i \quad \text{for} \quad i = 0, 1, ..., n \quad .$$

**Definition 1.1.** $\mathbb{P}_n$ *denotes the vector space (over* $\mathbb{R}$*) of polynomials of degree* $\leq n$ *in one variable. Its dimension over* $\mathbb{R}$ *is* $(n+1)$*.*

**Theorem 1.2.** *For any pairwise distinct points* $x_0, x_1, ..., x_n$ *and* $n+1$ *corresponding values* $y_0, y_1, ..., y_n$*, there is a unique polynomial* $p_n \in \mathbb{P}_n$ *such that* $p_n(x_i) = y_i$ *for* $i = 0, 1, ..., n$*.*

*Proof.*
**Uniqueness:** To prove uniqueness, suppose that $q_n$ is an arbitrary interpolating polynomial of degree $\leq n$ with $q_n(x_i) = y_i$ for $i = 0, 1, ..., n$. Then the difference polynomial $p_n - q_n$ is of degree $\leq n$ and vanishes at $n+1$ distinct points $x_i$. So $p_n - q_n$ has to be the null polynomial. Therefore, $p_n \equiv q_n$.
**Existence:** By using a more constructive approach, the existence can be proved. If $\{L_i\}_{i=0}^{n}$ is a basis for $\mathbb{P}_n$, then the polynomial $p_n$ is of the form

$$p_n(x) = \sum_{i=0}^{n} b_i L_i(x)$$

with the property that

$$p_n(x_i) = \sum_{j=0}^{n} b_j L_j(x_j) = y_i, \quad i = 0, 1, ..., n. \tag{1.1}$$

If the definition of the polynomials $L_i \in \mathbb{P}_n$ is

$$L_i(x) = \prod_{\substack{j=0,\\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j} \quad i = 0, 1, ..., n \tag{1.2}$$

then $L_i(x) = \delta_{ij}$ [2] and from (1.1) immediately follows that $b_i = y_i$.
So the polynomials $\{L_i : i = 0, 1, ..., n\}$ form a basis for $\mathbb{P}_n$. Hence the interpolating polynomial exists and has the following form

$$p_n(x) = \sum_{i=0}^{n} y_i L_i(x) \quad . \tag{1.3}$$

$\square$

Formula (1.2) bears the name **Lagrange polynomials**. Frequently (1.3) is written

$$p_n(x) = \sum_{i=0}^{n} \frac{\omega(x)}{(x - x_i)\omega'(x_i)} y_i$$

with $\omega(x) = \prod\limits_{i=0}^{n}(x - x_i)$.

**Example 1.3.** *Find the Lagrange polynomials of the Sinus-Function through the points* $(-\pi, 0)$, $(-\frac{\pi}{2}, -1)$, $(0, 0)$, $(\frac{\pi}{2}, 1)$ *and* $(\pi, 0)$ *!*

$$L_0(x) = \frac{(x+\frac{\pi}{2})\cdot(x-0)\cdot(x-\frac{\pi}{2})\cdot(x-\pi)}{(-\pi+\frac{\pi}{2})\cdot(-\pi-0)(-\pi-\frac{\pi}{2})\cdot(-\pi-\pi)} = \frac{x\cdot(x-\pi)\cdot(2x+\pi)\cdot(2x-\pi)}{6\pi^4}$$

$$L_1(x) = \frac{(x+\pi)\cdot(x-0)\cdot(x-\frac{\pi}{2})\cdot(x-\pi)}{(-\frac{\pi}{2}+\pi)\cdot(-\frac{\pi}{2}-0)(-\frac{\pi}{2}-\frac{\pi}{2})\cdot(-\frac{\pi}{2}-\pi)} = \frac{4x\cdot(x+\pi)\cdot(2x-\pi)\cdot(\pi-x)}{3\pi^4} \quad etc.$$

[2] $\delta_{ij}$ is called the **Kronecker-Delta**. $\delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$

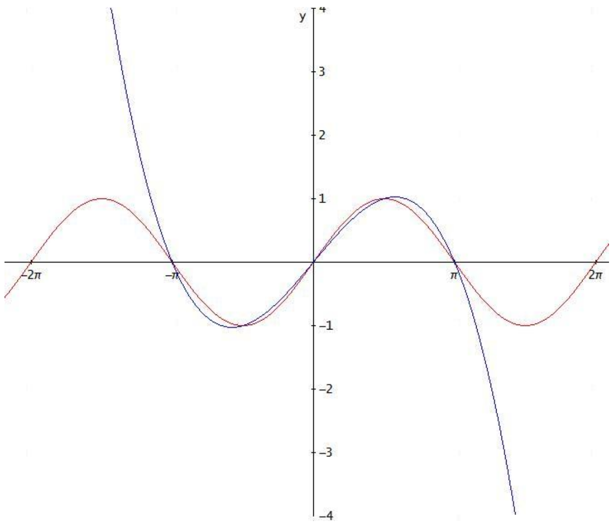



Figure 1.2: Interpolating Polynomial for the Sinus-Function

Lagrange Interpolation is simple, but there exist several disadvantages. Firstly the individual Lagrange polynomials are very complicated especially in the case of numerical calculations. Secondly they depend on the location of parameter values and therefore all of them have to be recomputed whenever any one of the parameter values is being changed.

## 1.2 Newton-Interpolation

Although the Lagrange polynomials solve the interpolation problem completely, there are lots of situations in which a different representation of the interpolation polynomial is more useful.
For practical applications and to minimize the computational cost the Newton-Interpolation is more convenient.

Given again a set of $n+1$ distinct data points $(x_i, y_i)$ $(i = 0, 1, ..., n)$, Newton searched for a series of unknown coefficients $a_0, a_1, ..., a_n$ so that the interpolation polynomial satisfies $p_n(x_i) = f(x_i) = y_i$ and can be written as

$$p_n(x) = a_0 + a_1(x-x_0) + a_2(x-x_0)(x-x_1) + ... + a_n(x-x_0) \cdot ... \cdot (x-x_{n-1}). \quad (1.4)$$

The unknown coefficients $a_i$ $(i = 0, 1, ...n)$ can be determined by solving (1.4) recursively.:

$$\begin{array}{lcl} y_0 = p_n(x_0) = a_0 & \rightarrow & a_0 = y_0 = f[x_0] \\ y_1 = p_n(x_1) = a_0 + a_1(x_1 - x_0) & \rightarrow & a_1 = \frac{y_1 - y_0}{x_1 - x_0} = f[x_0, x_1] \end{array}$$

$f[x_0, x_1]$ is called the *first-order divided difference* of $f$.
The same procedure is used to find the *second-order divided difference.*:

$$\begin{array}{l} y_2 = p_n(x_2) = a_0 + a_1(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) \\ \rightarrow \quad a_2 = \frac{\frac{y_2 - y_0}{x_2 - x_0} - \frac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_1} = f[x_0, x_1, x_2] \end{array}$$

Generally the *n-th order divided difference* is denoted by

$$a_n = f[x_0, x_1, ..., x_n].$$

With this additional information, the interpolating polynomial (1.4) obtains the following form

$$p_n(x) = f[x_0] + f[x_0, x_1](x-x_0) + ... + f[x_0, x_1, ..., x_n](x-x_0) \cdot (x-x_1) \cdot ... \cdot (x-x_{n-1}).$$

This equation is called the **Newton Divided Difference Formula**.

With the aid of the divided differences, it is possible to evaluate the coefficients of the interpolating polynomial. Therefore the divided differences are written in form of a table

$$
\begin{array}{c|ccccc}
x_0 & f(x_0) & & & & \\
 & & f\left[x_0, x_1\right] & & & \\
x_1 & f(x_1) & & f\left[x_0, x_1, x_2\right] & & \\
 & & f\left[x_1, x_2\right] & & & \\
x_2 & f(x_2) & & f\left[x_1, x_2, x_3\right] & \ddots & \\
 & & f\left[x_2, x_3\right] & & & f\left[x_0, x_1, ..., x_n\right] \quad , \\
x_3 & f(x_3) & & \vdots & & \\
\vdots & \vdots & f\left[x_{n-1}, x_n\right] & & & \\
x_n & f(x_n) & & & &
\end{array}
$$

and they are computed by the recursive formula

$$
f\left[x_k, x_{k+1}, ..., x_{k+j}\right] = \frac{f\left[x_{k+1}, ..., x_{k+j}\right] - f\left[x_k, ..., x_{k+j-1}\right]}{x_{k+j} - x_k}
$$

(for $j = 2, ..., n$ and $k = 0, 1, ..., n-j$) with the initial condition $f(x_0) = f\left[x_0\right]$.

This formula implicates that for each nonnegative integer $i$, the divided difference $f\left[x_0, x_1, ..., x_i\right]$ only depends on the interpolation points $x_0, x_1, ..., x_i$ and the value of $f(x)$ at these points.
Hence the coefficients $a_0, a_1, ..., a_n$ do not change when some new data points will be added. That is the most important advantage of Newton Interpolation.

**Example 1.4.** *Find the interpolating polynomial through the points* $(-2, 0)$, $(-1, 1)$, $(0, 0)$, $(1, 1)$ *and* $(2, 0)$ *by using divided differences!*

| $x_i$ | $f(x_i)$ | $f\left[x_i, x_{i+1}\right]$ | $f\left[x_i, ..., x_{i+2}\right]$ | $f\left[x_i, ..., x_{i+3}\right]$ | $f\left[x_i, ..., x_{i+4}\right]$ |
|---|---|---|---|---|---|
| -2 | 0 | | | | |
| | | 1 | | | |
| -1 | 1 | | -1 | | |
| | | -1 | | $\frac{2}{3}$ | |
| 0 | 0 | | 1 | | $-\frac{1}{3}$ |
| | | 1 | | $-\frac{2}{3}$ | |
| 1 | 1 | | -1 | | |
| | | -1 | | | |
| 2 | 0 | | | | |

*The coefficients of the interpolating polynomial are given by the first entries of each column. Thus $p_n(x)$ is*

$$\begin{aligned} p_n(x) &= 0 + 1 \cdot (x+2) + (-1) \cdot (x+2)(x+1) + \tfrac{2}{3} \cdot (x+2)(x+1)(x-0) \\ &\quad +(-\tfrac{1}{3}) \cdot (x+2)(x+1)(x-0)(x-1) \\ &= \tfrac{x^4-4x^2}{3} \quad . \end{aligned}$$

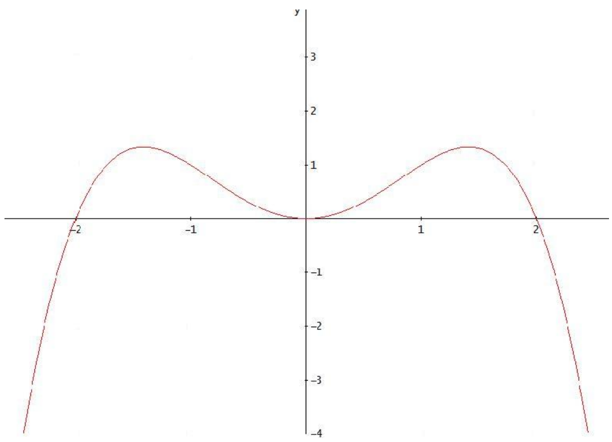



Figure 1.3: Interpolating Polynomial

Suppose that the given data points $(x_i, y_i)$ (for $i = 0, 1, ..., n$) correlate to a real-valued function $f(x)$ defined on an interval $[a, b]$. If $p_n(x)$ is the interpolating polynomial then

$$f(x) = p_n(x) + f\left[x_0, ..., x_n, x\right](x - x_0) \cdot ... \cdot (x - x_n).$$

Consequentially the interpolation error is given by

$$f(x) - p_n(x) = f\left[x_0, ..., x_n, x\right](x - x_0) \cdot ... \cdot (x - x_n).$$

**Theorem 1.5.** [3] *Suppose that the function* $f : [a, b] \to \mathbb{R}$ *is* $(n+1)$*-times continuously differentiable.*

*(i) If* $x_0, ..., x_n \in [a, b]$, $0 \leq i \leq n$, *then*

$$f\left[x_0, ..., x_i\right] = \frac{f^{(i)}(\xi)}{i!} \quad \textit{when} \quad \xi \in [a, b] \, .$$

*(ii) The error of the interpolation polynomial* $p_n(x)$ *to* $f$ *at* $x_0, ..., x_n \in [a, b]$ *can be written as*

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \cdot (x - x_0) \cdot ... \cdot (x - x_n) \quad \textit{for} \quad \xi \in [a, b] \, .$$

**Corollary 1.6.** *If* $x, x_0, ..., x_n \in [a, b]$, *then*

$$|f(x) - p_n(x)| \leq \frac{\parallel f^{(n+1)} \parallel_\infty}{(n+1)!} \cdot (x - x_0) \cdot ... \cdot (x - x_n). \tag{1.5}$$

[3] To prove Theorem 1.5. see [21] page 138-139.

## 1.3 Piecewise Interpolation

Since polynomial interpolation has good approximation properties on narrow intervals, it sometimes failures on wide ones. This yields to piecewise interpolation, in particular to spline interpolation.

### 1.3.1 Piecewise Linear Interpolation

Before spline interpolation will be embraced, take a look at the simplest case of piecewise linear interpolation.

**Definition 1.7.**

*(i) A **grid** on $[a, b]$ is a set $\Delta = \{x_0, x_1, ..., x_n\}$ satisfying*

$$a = x_0 < x_1 < x_2 < ... < x_{n-1} < x_n = b \quad .$$

*$x_0, x_1, ..., x_n$ are called the **nodes** of $\Delta$, and $h := max\{|x_{j+1} - x_j| \; |j = 0, ..., n-1\}$ is called **mesh size** of $\Delta$.*
*A grid $\Delta = \{x_0, x_1, ..., x_n\}$ is called **equispaced** if*

$$x_i = a + i \cdot h \quad for \quad i = 0, 1, ..., n \quad ;$$

*the mesh size is then $h = \frac{b-a}{n}$.*

*(ii) A function $p : [a, b] \to \mathbb{R}$ is called **piecewise linear** over a gird $\Delta$ if it is continuous and agrees on each interval $[x_i, x_{i+1}]$ with a linear polynomial.*

*(iii) On the space of continuous, real-valued functions $f$ defined on the interval $[a, b]$, one defines the norms*

$$\| f \|_\infty := sup\{|f(x)||x \in [a, b]\} \text{ [4]}$$

*and*

$$\| f \|_2 := \sqrt{\int_a^b f(x)^2 dx} \quad \text{[5]} \quad .$$

[4] $\| f \|_\infty$ is called the **supremum norm**.
[5] $\| f \|_2$ is called the **euclidean norm**.

The choice of the grid $\Delta$ for piecewise interpolation is very simple, because it must be the set of the interpolating points. Therefore the interpolation condition

$$S(x_i) = f(x_i) \quad \text{for} \quad i = 0, 1, ..., n$$

guarantees continuity in the interval $[a, b]$, and there exists a unique interpolant

$$S(x) = f(x_i) + f\left[x_i, x_{i+1}\right](x - x_i) \quad \forall x \in [x_i, x_{i+1}]\,.$$

The approximation is more accurate, if the mesh size $h$ is small (i.e., if the data points are closely spaced).

**Theorem 1.8.** *Let $S(x)$ be a piecewise linear interpolating function on the grid $\Delta = \{x_1, ..., x_n\}$ with mesh size $h$ over $[a, b]$. If the function $f(x)$ to be interpolated is twice continuously differentiable, then*

$$|f(x) - S(x)| \leq \frac{h^2}{8} \parallel f'' \parallel_\infty \quad \forall x \in [a, b]\,.$$

*Proof.* For $x \in [x_i, x_{i+1}]$, the relation

$$f(x) - S(x) = f\left[x_i, x_{i+1}, x\right](x - x_i)(x - x_{i+1})$$

holds.
Because of equation (1.5) the bound for the error is

$$|f(x) - S(x)| \leq \frac{1}{2} \parallel f'' \parallel_\infty \cdot |(x - x_i)(x - x_{i+1})| \quad .$$

The maximum of the quadratic function $(x - x_i)(x - x_{i+1})$ over the interval $[x_i, x_{i+1}]$ is incurred at the midpoint

$$x = \frac{x_i + x_{i+1}}{2} \quad ,$$

therefore following inequality is essential

$$(x - x_i)(x - x_{i+1}) \leq \frac{h^2}{4} \quad .$$

So the error will be bounded anywhere on the interval $[a, b]$ by

$$|f(x) - S(x)| \leq \frac{h^2}{8} \parallel f'' \parallel_\infty \quad .$$

□

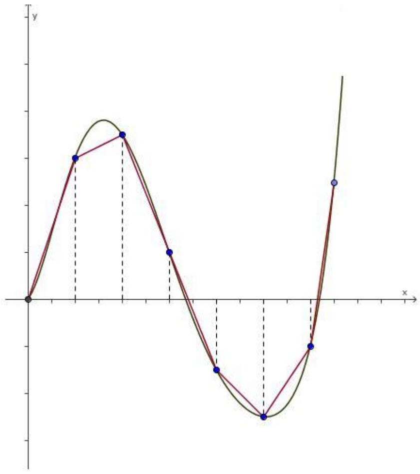



Figure 1.4: Piecewise Linear Interpolation

### 1.3.2 Cubic Splines

A special class of piecewise polynomials represent the splines, which have strong smoothness and excellent approximation properties.

**Definition 1.9.** *Let $x_0, x_1, ..., x_n$ be $n+1$ distinct nodes and $\Delta = \{x_0, x_1, ..., x_n\}$ be a grid on $[a, b]$. A* ***spline*** *of order $k$ over $\Delta$ is a function $S : [a, b] \to \mathbb{R}$ such that*

$$S_{|[x_i,x_{i+1}]} \in \mathbb{P}_k \quad \textit{for} \quad i = 0, 1, ..., n-1$$

*and*

$$S \in C^{k-1}[a, b] . \tag{1.6}$$

*$S_{k,\Delta}$ denotes the space of splines $S$ over $\Delta$ with dim($S_{k,\Delta}$)=n+k.*

Apparently, any polynomial on $[a, b]$ of degree $k$ is a spline; although the main idea behind spline interpolation is to assemble polynomials, since the given function $f(x)$ is represented by a different polynomial over each subinterval. Unfortunately, there can occure discontinuity in its $k$-th derivative at the internal nodes $x_1, x_2, ..., x_{n-1}$. But the result of equation (1.6) is

$$S_{i-1}^{(j)}(x_i) = S_i^{(j)}(x_i) \quad \text{for} \quad i = 1, ..., n-1; \quad j = 0, ..., k-1,$$

where $S_i = S_{|[x_i,x_{i+1}]}$.

So a spline can be represented as

$$S(x) = \sum_{i=0}^{k} S_i(x - x_i)^i \quad \text{if} \quad x \in [x_i, x_{i+1}] .$$

The simplest form of spline interpolation are *linear splines* ($k = 1$), which are equivalent to linear interpolation; but the most familiar ones are **cubic splines** ($k = 3$).

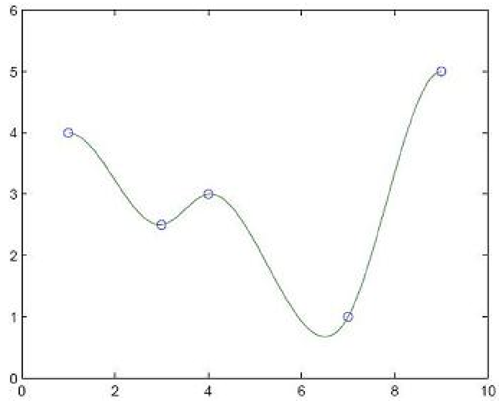



Figure 1.5: Cubic Spline

**Definition 1.10.** *Let $\Delta = \{x_0, x_1, ..., x_n\}$ be a grid over $[a, b]$ with spacings*

$$h_i = x_{i+1} - x_i \quad for \quad i = 0, 1, ..., n-1.$$

*Let $f(x)$ be a function defined on an interval $[a, b]$, and let $x_0, ..., x_n$ be $n+1$ distinct nodes with*

$$a = x_0 < x_1 < ... < x_{n-1} < x_n = b.$$

*A **cubic spline** is a piecewise polynomial S(x) that satisfies the following conditions:*

*(i) On each interval $[x_i, x_{i+1}]$ $(i = 0, 1, ..., n-1)$*

$$S(x) = S_i(x) \quad ,$$

*where $S_i(x)$ is a cubic spline.*

*(ii) The interpolation condition $S(x_i) = f(x_i)$ $(i = 0, 1, ..., n)$ holds.*

*(iii) S(x) is twice continuously differentiable on $(a, b)$.*

*(iv) Either one of the following boundary conditions is satisfied:*

- *$S''(a) = S''(b) = 0$, then the spline is called **natural cubic spline**.*
- *$S'(a) = f'(a)$ and $S'(b) = f'(b)$, then the spline is called **clamped cubic spline**.*

Clamped cubic splines approximate the function $f(x)$ more accurate, because they comprehend more information about $f(x)$. However, if the values of $f'(x)$ are not receivable, then other boundary conditions like the natural ones are applied.

To construct a cubic spline, it is clear from Definition 1.10. that the piecewise polynomial is of the form

$$S(x) = S_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i. \tag{1.7}$$

To get the value of $S(x)$ in each subinterval $[x_i, x_{i+1}]$, it is necessary to calculate for every subinterval a different cubic polynomial.

The aim is to obtain the coefficients $a_i$, $b_i$, $c_i$ and $d_i$. Therefore all characteristics of a cubic spline have to be used.

$$\begin{array}{lcllcl} S_i(x_i) & = & d_i & = & y_i \\ S_i(x_{i+1}) & = & a_i h_i^3 + b_i h_i^2 + c_i h_i + d_i & = & y_{i+1} \\ S_i'(x_i) & = & c_i & & \\ S_i'(x_{i+1}) & = & 3a_i h_i^2 + 2b_i h_i + c_i & & \\ S_i''(x_i) & = & 2b_i & = & y_i'' \\ S_i''(x_{i+1}) & = & 6a_i h_i + 2b_i & = & y_{i+1}'' \end{array} \tag{1.8}$$

Some translations of (1.8) lead to the coefficients

$$\boxed{\begin{array}{l} a_i = \frac{1}{6h_i}(y_{i+1}'' - y_i'') \\[2ex] b_i = \frac{1}{2}y_i'' \\[2ex] c_i = \frac{1}{h_i}(y_{i+1} - y_i) - \frac{1}{6}h_i(y_{i+1}'' + 2y_i'') \\[2ex] d_i = y_i \end{array}} \tag{1.9}$$

By inserting the coefficients $a_i$, $b_i$, $c_i$ and $d_i$ in equation (1.7), substituting the index $i$ by $i-1$ and some further translations, one receives the following equation

$$h_{i-1}y_{i-1}'' + 2(h_{i-1} + h_i)y_i'' + h_i y_{i+1}'' - \frac{6}{h_i}(y_{i+1} - y_i) + \frac{6}{h_{i-1}}(y_i - y_{i-1}) = 0. \tag{1.10}$$

All interior nodes $x_i$ $(i = 1, 2, ..., n-1)$ must achieve constraint (1.10). This equation leads to a system of $(n-1)$ linear equations for the $(n-1)$ unknown values $y''_1, ..., y''_{n-1}$ of a natural cubic spline ($y''_0 = y''_n = 0$). For instance, the linear system of equations for $n = 5$ is:

| $y''_1$ | $y''_2$ | $y''_3$ | $y''_4$ | 1 |
|---|---|---|---|---|
| $2(h_0+h_1)$ | $h_1$ | | | $\frac{6}{h_0}(y_1-y_0) - \frac{6}{h_1}(y_2-y_1) + h_0 y''_0$ |
| $h_1$ | $2(h_1+h_2)$ | $h_2$ | | $\frac{6}{h_1}(y_2-y_1) - \frac{6}{h_2}(y_3-y_2)$ |
| | $h_2$ | $2(h_2+h_3)$ | $h_3$ | $\frac{6}{h_2}(y_3-y_2) - \frac{6}{h_3}(y_4-y_3)$ |
| | | $h_3$ | $2(h_3+h_4)$ | $\frac{6}{h_3}(y_4-y_3) - \frac{6}{h_4}(y_5-y_4) + h_4 y''_5$ |

For equispaced grids (i.e. $x_i = a + i \cdot h$ for $i = 0, 1, ..., n$) the system of equations is going to be simplified.

| $y''_1$ | $y''_2$ | $y''_3$ | $y''_4$ | 1 |
|---|---|---|---|---|
| 4 | 1 | | | $-\frac{6}{h^2}(y_2 - 2y_1 + y_0) + y''_0$ |
| 1 | 4 | 1 | | $-\frac{6}{h^2}(y_3 - 2y_2 + y_1)$ |
| | 1 | 4 | 1 | $-\frac{6}{h^2}(y_4 - 2y_3 + y_2)$ |
| | | 1 | 4 | $-\frac{6}{h^2}(y_5 - 2y_4 + y_3) + y''_5$ |

So to obtain a cubic spline the system $Ax = b$ has to be solved, where the matrix A is symmetric, tridiagonal [6] and strictly diagonally dominant[7]. The Gaussian elimination algorithm yields a unique solution for both boundary conditions.

[6] A $(n \times n)$ matrix $A$ is called **tridiagonal** if $a_{ij} = 0$ for $i > j+1$ or $j > i+1$.

[7] A matrix $A = (a_{ij})$ is **strictly diagonally dominant** if $|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|$ $\forall i = 1, ..., n$.

**Theorem 1.11.** [8] *Let $x_0, x_1, ..., x_n$ be $n+1$ distinct nodes in the interval $[a,b]$, where*

$$a = x_0 < x_1 < ... < x_{n-1} < x_n = b \quad ,$$

*and let $f(x)$ be a function defined on $[a,b]$. Then $f(x)$ has a unique cubic spline $S(x)$ that is defined on the nodes and satisfies the natural (clamped) boundary condition $S''(a) = S''(b) = 0$ ($S'(a) = f'(a)$ and $S'(b) = f'(b)$).*

**Example 1.12.** *Represent the natural cubic spline for the following data points and only for comparison show the interpolating polynomial $P_{14}(x)$ too.*

| $x_k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $y_k$ | 7 | 6 | 4 | 4 | 5 | 4 | 2 | 3 | 5 | 7 | 6 | 4 | 4 | 5 | 7 |

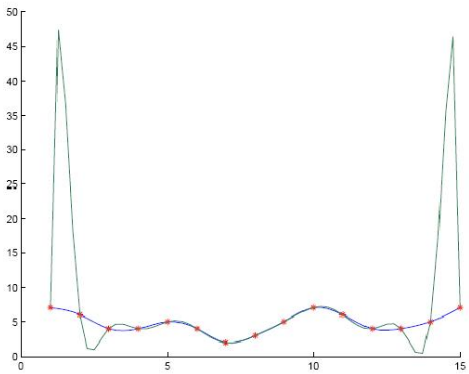



Figure 1.6: Natural Cubic Spline and $P_{14}(x)$

After knowing how to construct a cubic spline, it is time for one main property which was not mentioned yet.

**Corollary 1.13.** *Let $f(x) \in C^2([a,b])$, and let $S(x)$ be the natural cubic spline interpolating $f(x)$. Then*

$$\int_b^a [S''(x)]^2 \, dx \leq \int_b^a [f''(x)]^2 \, dx \quad , \tag{1.11}$$

*where equality holds if and only if $f(x) = S(x)$.*

[8] To prove Theorem 1.11. see [6] page 145-148.

*Proof.* It is ordinary that following equation holds

$$f(x) = S(x) + (f(x) - S(x)).$$

Therefore one gets

$$\begin{aligned}\int_b^a [f''(x)]^2\,dx &= \int_b^a [S''(x)]^2\,dx + 2\cdot\int_b^a S''(x)\cdot[f''(x)-S''(x)]\,dx \\ &+ \underbrace{\int_b^a [f''(x)-S''(x)]^2\,dx}_{\geq 0} \quad .\end{aligned}$$

Using integration by parts leads to

$$\begin{aligned}\int_b^a S''(x)\cdot[f''(x)-S''(x)]\,dx &= S''(x)\,[f'(x)-S'(x)]\Big|_a^b + \\ &- \int_b^a S'''(x)\cdot[f'(x)-S'(x)]\,dx \quad .\end{aligned}$$

The first term $S''(x)\,[f'(x)-S'(x)]\Big|_a^b$ vanishes because of the natural boundary conditions. Since $S'''(x)$ is constant in every subinterval $[x_i, x_{i+1}]$ (i.e. $S'''(x) = S_i'''(x)$), the second interval has the value

$$\int_b^a S'''(x)\cdot[f'(x)-S'(x)]\,dx = \sum_{i=0}^{n-1} S_i'''(x)\cdot[f(x)-S(x)]\Big|_{x_i}^{x_{i+1}} \quad .$$

The function $f(x)$ and the natural cubic spline $S(x)$ agree at all the nodes, and consequently all the terms of the sum above are zero.
Thus

$$\int_b^a [f''(x)]^2\,dx \geq \int_b^a [S''(x)]^2\,dx \quad .$$

□

The main property (1.11) of cubic splines is called the *minimum norm property*, which means that the cubic splines minimize the energy principle in mechanics. Historical constraint (1.11) was the reason for mathematician to grapple with the theory of splines.

**Remark 1.14.** *The total curvature of a function $f(x)$ is*

$$\kappa(x) = \frac{|f''(x)|}{(1 + [f'(x)]^2)^{\frac{3}{2}}} \approx |f''(x)| \quad .$$

*Hence Corollary 1.13. can be interpreted in the following geometrical way: The cubic spline has a total curvature that is at least as large as that of the original function $f(x)$. Moreover $S(x)$ is approximately optimal over all interpolating functions, because it has minimal total curvature.*

Concerning the error estimate for cubic splines, following result holds.:

**Theorem 1.15.** *Let $\Delta = \{x_0, x_1, ..., x_n\}$ be a grid over $[a, b]$ with spacing $h = \max_i |x_{i+1} - x_i|$ (for i=0,1,...,n-1). Let $S(x)$ be the cubic spline interpolating the function $f(x) \in C^4([a, b])$, then*

$$\| f(x) - S(x) \|_\infty \leq \frac{5}{384} h^4 \| f^{(4)} \|_\infty \quad .$$

### 1.3.3 B-Splines

B-splines are a very good instance of computing splines to fit given data points, because they form a basis for the vector space of splines $S_{k,\Delta}$. The letter B is based on the fact that these splines form a basis and have bell-shaped graphs.

**Definition 1.16.** *Let $t_0 \leq t_1 \leq ... \leq t_n$ be a sequence of distinct nodes. For $x \in \mathbb{R}$ and $i = 0, ..., n-k-1$ the* ***B-splines*** *$B_{i,k}(x)$ of degree $k$ are defined*

$$B_{i,0}(x) := \begin{cases} 1 & \text{if } x \in [t_i, t_{i+1}[ \\ 0 & \text{otherwise} \end{cases}$$

$$B_{i,k}(x) := \omega_{i,k}(x)B_{i,k-1}(x) + (1 - \omega_{i+1,k}(x))B_{i+1,k-1}(x) \quad \text{for} \quad k \geq 1, \tag{1.12}$$

*for*

$$\omega_{i,j}(x) := \begin{cases} \frac{x-t_i}{t_{i+j}-t_i} & \text{if } t_i < t_{i+j} \\ 0 & \text{otherwise} \end{cases} .$$

Hence, if the nodes are coincident (i.e. $t_i = t_{i+k+1}$) one has $B_{i,k} \equiv 0$. Apparently formula (1.12) indicates the following properties for B-splines.:

**Theorem 1.17.** *Let $B_{i,k}(x)$ be a B-spline, then*

*(i)* $B_{i,k}(x)$ *is a piecewise polynomial of degree $k$.*

*(ii)* $B_{i,k}(x) = 0$ *for $x \notin [t_i, t_{i+k+1}]$, so supp $B_{i,k} = [t_i, t_{i+k+1}]$. Thus, each B-spline has compact support* [9].

*(iii)* $B_{i,k}(x) > 0$ *for $x \in ]t_i, t_{i+k+1}[$; $B_{i,k}(t_i) = 0$ unless $t_i = t_{i+1} = t_{i+2} = ... = t_{i+k} < t_{i+k+1}$, and then $B_{i,k}(t_i) = 1$.*

[9] The **support** of a function $f(x) : \mathbb{C} \to \mathbb{R}$ is supp(f)=$\overline{\{x \in \mathbb{R} | f(x) \neq 0\}}$.
f(x) has *compact support* if supp(f) is a compact set.

*(iv) Let $[a, b]$ be an interval such that $t_k \leq a$ and $t_{n-k} \geq b$. Then*

$$\sum_{i=0}^{n-k-1} B_{i,k}(x) = 1 \quad \forall x \in [a, b]\,.$$

*Therefore, the B-splines form a partition of unity* [10] *on $[a, b]$.*

*(v) Let $x \in ]t_i, t_{i+k+1}[$. Then $B_{i,k}(x) = 1$ if and only if $x = t_{i+1} = ... = t_{i+k}$.*

*(vi) $B_{i,k}(x)$ is right-continuous (and even right-infinitely differentiable), for all $x \in \mathbb{R}$.*

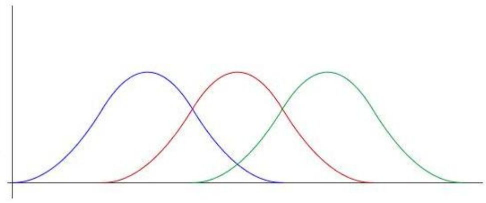


Figure 1.7: B-Splines $B_{i,2}(x)$[11]

*Proof.* Properties (i), (ii), (iii), (v) and (vi) are true for $k = 0$, and by induction on $k$ they also become clear for $B_{i,k}$.
So only property (iv) once again has to be proved by induction on $k$; $k = 0$ achieves (iv).
Let $x \in [a, b]$. Then it exists $j$ with $k \leq j \leq n-k-1$, such that $x \in [t_j, t_{j+1}[$.
If $x = t_j$ and $B_{j,k}(x) = 1$ one preserves immediately (iv).
In the other cases, it follows towards (ii)

$$\sum_{i=0}^{n-k-1} B_{i,k}(x) = \sum_{i=j-k}^{j} B_{i,k}(x).$$

Consequently the recursive formula yields to

$$\sum_{i=j-k}^{j} B_{i,k}(x) = \sum_{i=j-k}^{j} \omega_{i,k}(x) B_{i,k-1}(x) + \sum_{i=j-k}^{j} (1 - \omega_{i+1,k}(x)) B_{i+1,k-1}(x) \quad .$$

[10] A **partition of unity** of a topological set $X$ is a set of continuous functions $\{\chi_i\}_{i \in I}$ ($\chi_i : X \to [0, 1]$) such that $\sum_{i \in I} \chi_i(x) = 1$ for all $x \in X$.

[11] The B-splines $B_{i,2}(x)$ are built from three arcs of parabolas and two half-lines with junction $C^1$ at the knotes.

Some terms can be combined, so that

$$\begin{aligned}\sum_{i=j-k}^{j} B_{i,k}(x) &= \omega_{j-k,k}(x)B_{j-k,k-1}(x) + \sum_{i=j+1-k}^{j} B_{i,k-1}(x) + \\ &+ (1-\omega_{j+1,k}(x))B_{j+1,k-1}(x) \quad .\end{aligned}$$

Due to property (ii) it is $B_{j-k,k-1}(x) = 0$ and $B_{j+1,k-1}(x) = 0$, because $x \in [t_j, t_{j+1}[$. So the following result holds

$$\sum_{i=j+1-k}^{j} B_{i,k-1}(x) = \sum_{i=0}^{n-(k-1)-1} B_{i,k-1}(x) = 1 \quad .$$

□

**Remark 1.18.** *If $t_{n-k} = t = ... = t_n = b$ and $B_{n-k-1,k}(b) = 1$, then the B-spline $B_{n-k-1,k}(x)$ is left-continuous at b.*

**Corollary 1.19.** *For all $k \geq 0$ and all $x \in \mathbb{R}$, $B_{i,k}(x)$ is right-differentiable and*

$$B'_{i,k}(x) = k\left[\frac{B_{i,k-1}(x)}{t_{i+k}-t_i} - \frac{B_{i+1,k-1}(x)}{t_{i+k+1}-t_{i+1}}\right] \quad . \tag{1.13}$$

*Proof.* By using induction on $k$, formula (1.13) is true for $k = 0$.
The common case is shown by differentiate definition (1.12):

$$\begin{aligned}B'_{i,k}(x) &= \tfrac{1}{t_{i+k}-t_i}B_{i,k-1}(x) + \tfrac{x-t_i}{t_{i+k}-t_i}B'_{i,k-1}(x) + \\ &+ \tfrac{(-1)}{t_{i+k+1}-t_{i+1}}B_{i+1,k-1}(x) + \tfrac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}}B'_{i+1,k-1}(x) \quad .\end{aligned}$$

Now formula (1.13) is going to be inserted for $B'_{i,k-1}(x)$ and $B'_{i+1,k-1}(x)$, so one gets

$$\begin{aligned}B'_{i,k}(x) &= \tfrac{B_{i,k-1}(x)}{t_{i+k}-t_i} - \tfrac{B_{i+1,k-1}(x)}{t_{i+k+1}-t_{i+1}} + (k-1)\left[\tfrac{x-t_i}{t_{i+k}-t_i}\left(\tfrac{B_{i,k-2}(x)}{t_{i+k-1}-t_i} - \tfrac{B_{i+1,k-2}(x)}{t_{i+k}-t_{i+1}}\right) + \right. \\ &\left. + \tfrac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}}\left(\tfrac{B_{i+1,k-2}(x)}{t_{i+k}-t_{i+1}} - \tfrac{B_{i+2,k-2}(x)}{t_{i+k+1}-t_{i+2}}\right)\right] \quad .\end{aligned}$$

$$\begin{aligned}\Rightarrow B'_{i,k}(x) \quad &= \quad \tfrac{B_{i,k-1}(x)}{t_{i+k}-t_i} - \tfrac{B_{i+1,k-1}(x)}{t_{i+k+1}-t_{i+1}} + \tfrac{k-1}{t_{i+k}-t_i}\tfrac{x-t_i}{t_{i+k-1}-t_i}B_{i,k-2}(x)+ \\ & \quad +\tfrac{k-1}{t_{i+k}-t_{i+1}}\left(\tfrac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}} - \tfrac{x-t_i}{t_{i+k-1}-t_i}\right)B_{i+1,k-2}(x)+ \qquad (1.14) \\ & \quad +(-1)\cdot\tfrac{k-1}{t_{i+k+1}-t_{i+1}}\tfrac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+2}}B_{i+2,k-2}(x)\end{aligned}$$

To receive a more elementary form of (1.14), it is important to know that

$$\begin{aligned}\tfrac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}} - \tfrac{x-t_i}{t_{i+k}-t_i} \quad &= \quad \tfrac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}} - \tfrac{t_{i+k+1}-t_{i+1}}{t_{i+k+1}-t_{i+1}} + \tfrac{t_{i+k}-t_i}{t_{i+k}-t_i} - \tfrac{x-t_i}{t_{i+k}-t_i} \\ &= \quad \tfrac{t_{i+k}-x}{t_{i+k}-t_i} - \tfrac{x-t_{i+1}}{t_{i+k+1}-t_{i+1}} \quad .\end{aligned}$$

This additional information and the correct arrangement of the terms lead to

$$\begin{aligned}B'_{i,k}(x) \quad &= \quad \tfrac{B_{i,k-1}(x)}{t_{i+k}-t_i} - \tfrac{B_{i+1,k-1}(x)}{t_{i+k+1}-t_{i+1}} + \tfrac{k-1}{t_{i+k}-t_i}\left[\tfrac{x-t_i}{t_{i+k-1}-t_i}B_{i,k-2}(x)+\right. \\ & \quad \left.+\tfrac{t_{i+k}-x}{t_{i+k}-t_{i+1}}B_{i+1,k-2}(x)\right] - \tfrac{k-1}{t_{i+k+1}-t_{i+1}}\left[\tfrac{x-t_{i+1}}{t_{i+k}-t_{i+1}}B_{i+1,k-2}(x)+\right. \\ & \quad \left.+\ \tfrac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+2}}B_{i+2,k-2}(x)\right] \quad .\end{aligned} \qquad (1.15)$$

Now Definition 1.16. is used to get formula (1.13)

$$B'_{i,k}(x) = k\left[\frac{B_{i,k-1}(x)}{t_{i+k}-t_i} - \frac{B_{i+1,k-1}(x)}{t_{i+k+1}-t_{i+1}}\right] \quad .$$

□

To be able to construct the required basis for $S_{k,\Delta}$, it is necessary to examine a new sequence $\bar{\Delta} = \{\tau_i\}_{i=0,...,l+k}$ of points in $[a, b]$.

$$
\begin{array}{lccccccccccc}
\Delta: & a & = & t_0 & < & t_1 & < & \dots & < & t_n & = & b \\
 & & & \| & & \| & & & & \| & & \\
\bar{\Delta}: & \tau_0 = \dots & = & \tau_k & < & \tau_{k+1} & < & \dots & < & \tau_l & = & \dots = \tau_{l+k}
\end{array}
$$

The points $\tau_i$ $(i = 0, ..., l + k)$ are called **knots**, and they are uniquely determined by the sequence $t_0, ..., t_n$.
A knot $\tau_i$ is of **multiplicity** $r$, if $\tau_i = \tau_{i+1} = \ldots = \tau_{i+r-1}$. Hence the knots $\tau_k$ and $\tau_l$ in the new gird $\bar{\Delta}$ are of multiplicity $k + 1$.

It is easy to show that $l = n + k$ holds, which complies with the dimension of the vector space $S_{k,\Delta}$.
Consequently the $l$ B-splines $B_{i,k}(x)$ $(i = 0, ..., l - 1)$ of the grid $\bar{\Delta}$ form the new basis of $S_{k,\Delta}$.

**Theorem 1.20.** [12] *If all the knots are of multiplicity $\leq k + 1$, then the B-splines $B_{i,k}(x)$ $(i = 0, ..., l - 1)$ constitute a basis* [13] *for the vector space $S_{k,\Delta}$.*

A part and parcel of the proof of Theorem 1.20. is the following Lemma.:

**Lemma 1.21.** [14] *For $t \in [a, b]$ following result*

$$(x - t)^k = \sum_{i=0}^{l-1} \psi_{i,k}(t) B_{i,k}(x)$$

*with $\psi_{i,k}(t) = \prod_{j=1}^{k} (\tau_{i+j} - t)$ and $\psi_{i,0}(t) = 1$ holds.*

Due to Theorem 1.20. any spline $S(x) \in S_{k,\Delta}$ can be uniquely written as

$$S(x) = \sum_{i=0}^{n-1} d_i B_{i,k}(x) \quad .$$

The coefficients $d_i$ are going to be calculated by **De Boor's algorithm**[15], which was developed by De Boor in 1972 and is numerically stable.

[12] To prove Theorem 1.20. see [25] page 21 and [28] page 163-165.
[13] A **basis** of a vector space V is a linearly independent subset of V that spans V.
[14] To prove Lemma 1.21. see [25] page 21-22.
[15] To read more about **De Boor's algorithm** see [3] page 63-65.

**Corollary 1.22.** *Let $x \geq \tau_k$, then*

$$S(x) = \sum_i d_i B_{i,k}(x) = \sum_i d_i^{(1)} B_{i,k-1}(x) = \ldots = \sum_i d_i^{(k)} B_{i,0}(x) \quad ,$$

*with*

$$d_i^{j+1} = \omega_{i,k-j}(x) d_i^{(j)} + (1 - \omega_{i,k-j}(x)) d_{i-1}^{(j)}$$

*holds.*

A special case of the B-splines are the cubic splines - already mentioned in Section 1.3.2 .

**Example 1.23.** *B-splines are a very useful tool in Approximation Theory. An application range of the B-splines are for instance the* ***Bernstein polynomials***[16]*, which are used to build Bézier curves*[17]*.*

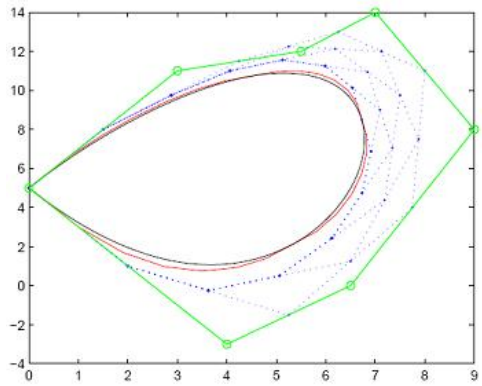



Figure 1.8: Bézier curve

[16]The **Bernstein polynomials** over the interval $[0, 1]$ are defined as $B_i^k(X) = \binom{k}{i}(1-X)^{k-i}X^i$ $(0 \leq i \leq k)$.

[17]To read more about Bézier curves see [23] and [25].

# CHAPTER 2

# QUASI-INTERPOLATION

The last preceding chapter delivered insight into different methods of Interpolation Theory. Over the past years another well-known application area of Approximation Theory became more and more important. Quasi-interpolation schemes are nowadays well-established among approximation theoretican, but unfortunately have found small support in practical employments.

Isaac Schoenberg [1] and Carl de Boor [2] did a lot of research on this area. ("The fundamental papers by Schoenberg form a monument in the history of the subject as well as its inauguration."[3]) Hence the quasi-interpolant is often known as Schoenberg-operator by various authors.

Given a sequence of $n+1$ nodes $x_0, x_1, ..., x_n$ and $n+1$ corresponding values $y_0, y_1, ..., y_n$. The problem consists of finding a function $Qf$ so that

$$Qf(x_i) \approx y_i \quad \text{for} \quad i = 0, 1, ..., n \quad .$$

$Qf$ is then called **quasi-interpolant operator** for all the data points $(x_i, y_i)$.

[1] Isaac Schoenberg (1903-1990) was a Romanian mathematician. He is deemed to be the father of splines.("Schoenberg's more than 40 papers on splines after 1960 gave much impetus to the rapid development of the field.")

[2] Carl de Boor (born 1937) is a German-American mathematician.

[3] To read more about the biographical abstract of Issac Schoenberg see [2].

In opposition to the interpolating function $f(x)$, which fits all the given data points, the quasi-interpolating function $Qf(x)$ does not necessarily match function values at all the nodes exactly.
The flexibility and the simplicity of a quasi-interpolant make it much easier to evaluate the approximant directly, without solving any linear systems of equations. This actuality is the biggest advantage of Quasi-interpolation schemes.

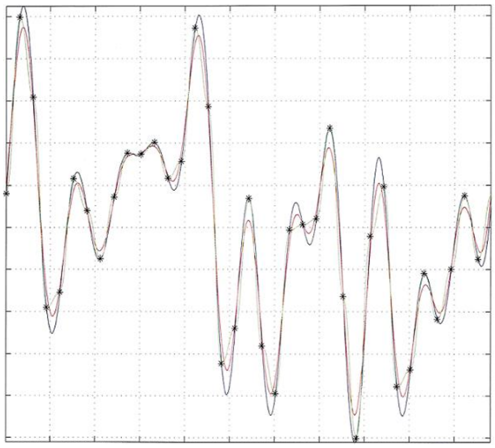


Figure 2.1: Interpolating function and Quasi-interpolating function

The earliest appearance of quasi-interpolating functions is perhaps Bernstein's [4] approximation, which uses Bernstein polynomials

$$B_i^k(x) = \binom{k}{i} x^i (1-x)^{k-i}$$

to build a quasi-interpolant of an univariate function $f(x)$ on $[0, 1]$ via

$$\sum_{i=0}^{k} f(\frac{i}{k}) B_i^k(x), \quad x \in [0, 1] \quad .$$

Nowadays this method is well-known under the names of Bézier and De Casteljau [5] and used in Computer Aided Geometric Design.

[4] Sergei Natanovich Bernstein (1880-1968) was a Russian and Soviet mathematician.
[5] Paul De Casteljau (born 1930) is a French physicist and mathematician.

## 2.1 Spline-type space

Popular methods like interpolation and least squares approximation produce a spline function from given discrete data points. Therefore both of these approximation schemes resolve a linear system of equations with as many unkowns as the dimension of the spline space. Hence both methods are not suitable for real-time processing of large streams of data.

Quasi-interpolation is a good methodology that allows to obtain from given functions or discrete data (generally regular samples of a continuous function) smooth functions that observe the general behaviour of the underlying continuous function.

First and foremost some notations and basic facts, which are an important issue within the Approximation Theory of shift-invariant spaces, for instance interpolation theory, radial basis approximation, wavelets and sampling theory.

**Definition 2.1.** *A space $V^p$ is called **shift-invariant space** for a lattice $\Lambda = \mathbb{Z}^d$ if it is a closed linear subspace of $L^p(\mathbb{R}^d)$* [6] *$(1 \leq p < \infty)$, which is invariant under all translations, i.e. if for every $\lambda \in \Lambda$ and $f \in V^p$ implies $f(\cdot - \lambda) \in V^p$.*

Whenever the shifts of a single function $f \in L^p(\mathbb{R}^d)$ $(1 \leq p < \infty)$ generate the $L^p$-closure of all finite linear combinations of translates of $f$, $V^p$ is called **principal shift-invariant space**.

A lot of research on this domain has been done by [10], [11], [7] and [26].

**Definition 2.2.** *For a given Hilbert space $H$ a sequence $\{\varphi_k\}_{k=1}^{\infty}$ is called a **Riesz basis** or a **Riesz basic sequence** if there exist constants $A, B > 0$ such that for all $c \in \ell^2$ :*[7]

$$A \cdot \|c\|_{\ell^2}^2 \leq \|\sum_{k=1}^{\infty} c_k \varphi_k\|^2 \leq B \cdot \|c\|_{\ell^2}^2 \tag{2.1}$$

In reference to the Linear Algebra a sequence $\{\varphi_k\}_{k=1}^{\infty}$ is a Riesz basis if and only if the corresponding Gram matrix, whose entries are the scalar products $(\langle \varphi_k, \varphi_{k'} \rangle)_{k,k'}$, is invertible on $(\ell^2, \|\cdot\|_2)$. This can be used to prove Lemma 2.3.:

[6] The Lebesgue space $L^p$ is the completion of the continuous functions with compact support using the $L^p$-norm.

[7] $\ell^2 = \left\{ \{x_n\}_{n=1}^{\infty} \mid \|x\|_{\ell^2}^2 := \sum_{n=1}^{\infty} |x_n|^2 < \infty \right\}$ is a Hilbert space.

**Lemma 2.3.** *A countable family $\{\varphi_k\}_{k=1}^{\infty}$ in $H$ is a Riesz basic sequence if and only if there exists some orthonormal family $\{\psi_k\}_{k=1}^{\infty}$ and an invertible mapping $T : H \to H$ such that $\varphi_k = T\psi_k$.*

**Remark 2.4.** *It is well justified to call the quotient of the optimal values for $A$ and $B$ the **Riesz condition number** of the Riesz basic sequence, i.e.*

$$\kappa = \frac{B}{A}.$$

**Lemma 2.5.** *If $\Lambda$ is a lattice in $\mathbb{R}^d$ and $\varphi \in C_c(\mathbb{R}^d)$ [8] the familiy of functions such that $\{T_\lambda \varphi\}_{\lambda\in\Lambda}$ [9] is a Riesz basis, then the following statements are equivalent for any $1 \leq p < \infty$:*

*(i)* $f \in L^p(\mathbb{R}^d)$ *and* $f = \sum_{\lambda\in\Lambda} c_\lambda T_\lambda \varphi$

*(ii)* $f = \sum_{\lambda\in\Lambda} c_\lambda T_\lambda \varphi$ *with* $c_\lambda \in \ell^p$

*(iii)* $f$ *is in the* $L^p$*-closure of the finite linear combinations of the form* $\sum_{k=1}^{K} c_{\lambda_k} T_{\lambda_k} \varphi$.

*Proof.* To show the correctness of those statements, let $p = 2$ and $\varphi_\lambda = T_\lambda \varphi$. The general case requires more detailed knowledge of function spaces, in particular Wiener Amalgam spaces[10].
$(ii) \Rightarrow (iii)$: If $c \in \ell^2$, then the finite partial sums of the form

$$\sum_{\lambda\in F} c_\lambda \varphi_\lambda = \sum_{k=1}^{K} c_{\lambda_k} \varphi_{\lambda_k} \quad ,$$

for $F \subseteq \Lambda$ is finite, form a Cauchy sequence[11], because for $K_1 < K_2$

$$\|\sum_{k=1}^{K_1} c_{\lambda_k}\varphi_{\lambda_k} - \sum_{k=1}^{K_2} c_{\lambda_k}\varphi_{\lambda_k}\|^2 \leq \|\sum_{k=K_1+1}^{K_2} c_{\lambda_k}\varphi_{\lambda_k}\|^2 \leq B \cdot \underbrace{\sum_{k=K_1+1}^{K_2} |c_{\lambda_k}|^2}_{\substack{\to 0 \\ \text{for } \mathrm{K}_1, K_2 \to \infty}}$$

[8] $C_c(\mathbb{R}^d) = \{f : \mathbb{R}^d \to \mathbb{C} | \quad \text{f is complex-valued, continuous and has compact support}\}$

[9] The **translation operator T** moves the graph of the function $f$ by the vector $x$ to another position, i.e. $T_x : T_x f(z) = [T_x f](z) = f(z - x)$

[10] The **Wiener Amalgam space** with local component X, a normed space with $\|\cdot\|_X$, and global component $L^p_m$, a weighted $L^p$ space with non-negative weight $m$, is defined by $W(X, L^p_m) := \left\{f : \left(\int_{\mathbb{R}^d} \|f(\cdot)\bar{g}(\cdot - x)\|_X^p m^p(x)dx\right)^{1/p} < \infty\right\}$, where g is a continuously differentiable, compactly supported function.

[11] A sequence of real numbers $\{x_n\}_n$ is a **Cauchy sequence** if for every $\epsilon > 0$ there exists $N > 0$ such that $|x_m - x_n| < \epsilon \quad \forall m, n > N$.

using the upper estimate in (2.1).
Consequently the completenes of the Hilbert space $L^2$ yields the convergence to $f = \sum_{k=1}^{\infty} c_{\lambda_k} T_{\lambda_k}\varphi$, i.e.

$$\|\sum_{k=1}^{K} c_{\lambda_k} T_{\lambda_k}\varphi - f\|_{\ell^2} \to 0 \quad \text{for} \quad K \to \infty.$$

$(i) \Rightarrow (ii)$: $L^2$-convergence of the series $\sum_{\lambda\in\Lambda} c_\lambda T_\lambda \varphi$ implies that the corresponding sequence $\{c_\lambda\}_{\lambda\in\Lambda}$ is in $\ell^2$.
Finally assume that $f_n \overset{L^2}{\to} f$ with $f_n = \sum_{k=1}^{\infty} c_k^{(n)}\varphi_k$. Since $f_n$ is a Cauchy sequence by (2.1), then $\{c^{(n)}\}$ is a Cauchy sequence in $\ell^2$, again by the lower estimate in (2.1).
Since $\ell^2$ is complete there exists $c \in \ell^2(\Lambda)$ such that $c = \lim_{n\to\infty} c^{(n)}$ in $\ell^2$, and hence $f \equiv \sum_{k=1}^{\infty} c_k\varphi_k$.

□

Lemma 2.6. shows a nice extra property for $\sum_{k\in\mathbb{Z}^d} c_k\varphi_k(x)$:

**Lemma 2.6.** ***Unique representation property***
*If $\sum_{k\in\mathbb{Z}^d} c_k\varphi_k(x) = f(x) = \sum_{k\in\mathbb{Z}^d} d_k\varphi_k(x)$, then $c_k = d_k \quad \forall k \in \mathbb{Z}^d$.*

*Proof.* $\sum_{k\in\mathbb{Z}^d}(c_k - d_k)\varphi_k(x) = 0$
$\{\varphi_k\}_{k\in\mathbb{Z}^d}$ is a Riesz basic sequence and therefore by (2.1)

$$A \cdot \|c-d\|^2 \le \underbrace{\|\sum_{k\in\mathbb{Z}^d}(c_k - d_k)\varphi_k\|^2}_{=0} \quad .$$

This implies $c = d$ and consequently $c_k = d_k \quad \forall k \in \mathbb{Z}^d$. □

Hence the **Spline-type space** for a function $f$ can be defined as the $L^p$-closure of all finite linear combinations of translates of $f$. An important special case, obtained by the choice $\Lambda = a\mathbb{Z}^d$, is given by Definition 2.7.:

**Definition 2.7.** *Given $a > 0$ and a function $\varphi \in L^2(\mathbb{R}^d)$. $V_{\varphi,a}$ is called **Spline-type space** generated by the pair $(\varphi, a)$ if the family $(T_{ak}\varphi)_{k\in\mathbb{Z}^d}$ is a Riesz basis for $V_{\varphi,a}$, its closed linear span in $L^2(\mathbb{R}^d)$.*

$V_{\varphi,a}$ is translation invariant along the lattice $a\mathbb{Z}^d$. Based on the circumstances that those spaces are generated by a single function and its translates, they

are sometimes called principal shift-invariant spaces.
$(T_{ak}\varphi)_{k\in\mathbb{Z}^d}$ accomplishes the Riesz basis property, and therefore

$$V_{\varphi,a} = \left\{ \sum_{k\in\mathbb{Z}^d} c_k T_{ak}\varphi \mid c \in \ell^2(\mathbb{Z}^d) \right\}.$$

Before discussing the verification of the Riesz basis property, one needs the following definition first.:

**Definition 2.8.** *The* **convolution** $f * g$ *of* $f$ *with* $g$ *is defined to be*

$$(f * g)(x) = \int_{\mathbb{R}} f(y)g(x-y)dy \quad .$$

The Riesz basic sequence condition of $(T_{ak}\varphi)_{k\in\mathbb{Z}^d}$ is demonstrated by

$$\langle T_{ak}\varphi, T_{ak'}\varphi\rangle = \langle \varphi, T_{-ak}T_{ak'}\varphi\rangle = \langle \varphi, T_{ak'-ak}\varphi\rangle .$$

That shows that the corresponding Gram matrix $T_{ak'-ak}$ is circulant with entries being the sampling values of the autocorrelation function $\varphi * \varphi^*$ with $\varphi^*(x) = \bar{\varphi}(-x)$. Hence $(T_{ak}\varphi)_{k\in\mathbb{Z}^d}$ is a Riesz basic sequence if and only if this circulant matrix is invertible.

**Remark 2.9.** *The B-splines form a Riesz basic sequence and consequently they are a good choice for the family* $(T_{ak}\varphi)_{k\in\mathbb{Z}^d}$.

For any Riesz basic sequence $\{\varphi_k\}_{k\in\mathbb{Z}^d}$ spanning a closed subspace $H_0 \subseteq H$ there exists another uniquely determined Riesz basic sequence $\{\tilde{\varphi}_k\}_{k\in\mathbb{Z}^d}$ for $H_0$ such that

$$\langle \varphi_k, \tilde{\varphi}_{k'}\rangle = \delta_{k,k'},$$

called the **biorthogonal Riesz basic sequence**. $\tilde{\varphi} := \tilde{\varphi}_0$ is the *dual atom* of $\varphi$.

It is useful because it allows to determine the coefficients of $f = \sum_{k\in\mathbb{Z}^d} c_k\varphi_k$ through

$$c_k = \langle f, \tilde{\varphi}_k\rangle .$$

More generally, for $f \in H$, one has

$$P_{H_0}f = \sum_{k\in\mathbb{Z}^d} \langle f, \tilde{\varphi}_k\rangle \varphi_k.$$

If $\varphi_k = T_k\varphi$ then one can show that $\tilde{\varphi}_k = T_k\tilde{\varphi}$ holds.

**Lemma 2.10. Biorthogonality criteria**
*For each Riesz basic sequence* $(T_k\varphi)$ *the unique biorthogonal*[12] *basis is of the form* $(T_k\tilde{\varphi})$ *for* $V_\varphi$.

*Proof.* If $\varphi(x) = \varphi(-x)$ one can show that $\tilde{\varphi}(x) = \tilde{\varphi}(-x)$.
First of all the statement $\tilde{\varphi}_k = \widetilde{T_k\varphi} = T_k\tilde{\varphi}$ for $\tilde{\varphi} := \tilde{\varphi}_0$ has to be proved.

$$\langle T_{ak}\tilde{\varphi}, T_{ak'}\varphi\rangle = \left\langle \tilde{\varphi}_0, T_{a(k'-k)}\varphi \right\rangle = \begin{cases} 1 & \text{if } k = k' \\ 0 & \text{if } k \neq k' \end{cases}.$$

By the uniqueness of the biorthogonal system one gets $\tilde{\varphi}_k = \widetilde{T_k\varphi} = T_k\tilde{\varphi}$. Currently the orthogonal projection $P$ of the function $f$ can be calculated by

$$\begin{aligned} P_\varphi f(x) &= \textstyle\sum_k \langle f, \widetilde{\varphi_k}\rangle \varphi_k(x) &&= \textstyle\sum_k \langle f, T_k\tilde{\varphi}\rangle T_k\varphi(x) \\ &= \textstyle\sum_k f * \tilde{\varphi}(k)\varphi(x-k) &&= Q_\varphi(f * \tilde{\varphi}) \end{aligned}$$

with $f * \tilde{\varphi}(k)$ being the convolution of $f$ and $\tilde{\varphi}$, restricted to $\mathbb{Z}^d$. □

Now it is necessary to discuss how one can verify the Riesz basic property and how $\tilde{\varphi}$ is determined.

**Definition 2.11.** *The* **Fourier transform** *of a complex-valued function* $f(x)$, $x \in \mathbb{R}^d$, *is defined formally by*

$$(\mathcal{F}f)(s) = \hat{f}(s) = \int_{\mathbb{R}^d} f(x)e^{-2\pi isx}dx.$$

**Lemma 2.12.** *Let* $\Lambda = (a\mathbb{Z}^d)^\perp = \frac{1}{a}\mathbb{Z}^d$, *then* $(T_{ak})_{k\in\mathbb{Z}^d}$ *is a Riesz basic sequence if and only if there exists* $A', B' > 0$ *such that*

$$0 < A' \leq \sum_{\lambda\in\frac{1}{a}\mathbb{Z}^d} |T_\lambda\varphi|^2(s) = \sum_{n\in\mathbb{Z}^d} |\varphi|^2(s - n/a) \leq B' < \infty \quad \forall s \in \mathbb{R}^d.$$

In most cases this condition is easily satisfied if $a$ is large enough, hence $\frac{1}{a}$ is small.

[12] Two sequences $\{v_i\}_i$ and $\{u_i\}_i$ in a Hilbert space H are **biorthogonal** if $\langle v_i, u_j\rangle = \delta_{ij}$.

**Lemma 2.13.** *If* $(T_{ak}\varphi)$ *generates a Riesz basic sequence, then* $(T_{ak}\tilde{\varphi})$ *is a Riesz basic sequence too.*

$\tilde{\varphi} = \tilde{\varphi}_0$ is characterized by its Fourier transform as follows

$$\mathcal{F}(\tilde{\varphi}_0)(s) = \frac{\hat{\varphi}(s)}{\sum_{n\in\mathbb{Z}^d} |\hat{\varphi}(s-n/a)|^2} \quad .$$

## 2.2 Quasi-interpolant

In the application area of Approximation Theory there does not exist a unique definition of the quasi-interpolant. Quasi-interpolation operators are used and defined by various mathematician in many different ways.

**Definition 2.14.** *Given $a > 0$ and a function $\varphi \in C_c(\mathbb{R}^d)$ the operator $Q$ with*

$$f \mapsto Qf(x) = \sum_{k \in \mathbb{Z}^d} f(ak)\varphi(x - ak)$$

*is called **quasi-interpolant for f**, along the lattice $\Lambda = a\mathbb{Z}^d$, using the atom $\varphi$.*

Sometimes it is useful to adapt the function $\varphi$ to the lattice size.

**Definition 2.15.** *Let $f$ be a continuous function on $\mathbb{R}^d$* [13] *and $h > 0$ then the* **quasi-interpolant** *$Q_h f$ is defined by*

$$Q_h f(x) = \sum_{k \in \mathbb{Z}^d} f(hk)\varphi(\frac{x}{h} - k), \quad x \in \mathbb{R}^d \tag{2.2}$$

*for some prescribed function $\varphi$ on $\mathbb{R}^d$.*

In the general terminology one can see that $Q_h f$ is just a variant of $Qf$, but with $\varphi$ replaced by a dilated version of $\varphi$, namely $\varphi_h(z) := \varphi(\frac{z}{h})$.

In the literature the quasi-interpolant $Q_h f$ is often called *Schoenberg's operator*. $Q_h f$ is a superposition of dilated and shifted versions of $\varphi$, which uses samples of the function $f$ on the fine grid $h\mathbb{Z}^d$ to get a good approximation of $f$.

For appropriate $\varphi$, e.g. B-splines, formula (2.2) describes an approximation to the function $f$ on $\mathbb{R}^d$ from its samples on the fine grid $h\mathbb{Z}^d$.
Among the quasi-interpolants the most important ones are those which arise from BUPUs.

**Definition 2.16.** *Let $\varphi \in C_c(\mathbb{R}^d)$ be a compactly supported function and $a\mathbb{Z}^d$ be a lattice in $\mathbb{R}^d$. A sequence $\Phi = (T_\lambda \varphi)_{\lambda \in a\mathbb{Z}^d}$ is called a* **regular bounded uniform partition of unity** *(**BUPU**) if*

$$\sum_{\lambda \in a\mathbb{Z}^d} \varphi(x - \lambda) \equiv 1.$$

[13] $\mathbb{R}^d$ is the $d-$dimensional Euclidean space.

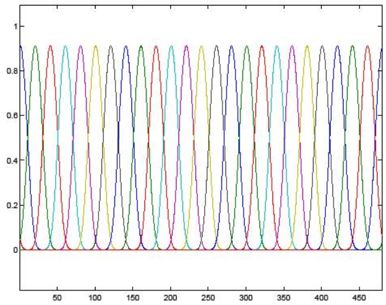



Figure 2.2: Bounded uniform partition of unity (BUPU)

**Remark 2.17.** *For B-splines the advantage of this normalization is that BUPU, i.e.* $Q_h(1) = 1 \quad \forall h > 0$.

**Theorem 2.18.** *Assume that the function* $\varphi \in C_c(\mathbb{R}^d)$ *defines a BUPU, and* $Q_h f$ $(h > 0)$ *is the quasi-interpolant of* $f$.
*Then for all uniformly continuous* [14] *functions* $f$ *one has*

$$\|Q_h f - f\|_\infty \to 0 \quad \textit{for} \quad h \to 0 \quad . \tag{2.3}$$

The proof of statement (2.3) makes use of the *oscillation function* and its properties.

**Definition 2.19.** *For a function* $f$ *on* $\mathbb{R}^d$ *and any* $\delta > 0$*, the function*

$$x \mapsto osc_\delta(f)(x) = \sup_{|y| \le \delta} |T_y f(x) - f(x)| = \sup_{y \in B_\delta(x)} |f(x) - f(x+y)|$$

*is called the* $\delta$***-oscillation function of*** $f$.

**Remark 2.20.** *For any uniformly continuous function* $f$

$$\|osc_\delta f\|_\infty \to 0 \quad \textit{for} \quad \delta \to 0 \quad .$$

*The claim of Remark 2.20. can be verified taking a closer look at the definitions used.*

[14] A real function f is **uniformly continuous** if $\forall x, y \quad \forall \epsilon > 0 \quad \exists \delta > 0$ $|x - y| < \delta \Rightarrow |f(x) - f(y)| < \epsilon$.

Starting the **Proof of Theorem 2.18.**:

*Proof.* To prove the correctness of (2.3) only the following pointwise estimate between $Q_h f(x)$ and $f(x)$ has to be shown

$$|(Q_h f - f)(x)| \leq osc_\delta(f)(x) \quad \forall x \in \mathbb{R}^d, \tag{2.4}$$

because then (2.3) follows from Remark 2.20.
One can obtain the estimate (2.4) following three steps.:

*1.* For $h > 0$ let the sequence $\Psi = (\psi_k)_{k\in\mathbb{Z}^d}$ given by $\psi_k(x) = \varphi(\frac{x}{h} - k)$ be a BUPU, if $\sum_{k\in\mathbb{Z}^d} T_k\varphi \equiv 1$.

$$\sum_{k\in\mathbb{Z}^d} \psi_k(x) = \sum_{k\in\mathbb{Z}^d} \varphi(\frac{x-kh}{h}) = \sum_{k\in\mathbb{Z}^d} \varphi(\underbrace{\frac{x}{h}}_{y=\frac{x}{h}} - k) = \sum_{k\in\mathbb{Z}^d} \varphi(y-k) = \sum_{k\in\mathbb{Z}^d} T_k\varphi(y) = 1.$$

*2.* Without loss of generality supp $\varphi \subseteq B_R(0)$. Hence $\varphi(\frac{x}{h} - k) \neq 0$ if and only if $\frac{x}{h} - k \in B_R(0)$ or $|\frac{x}{h} - k| \leq R \Leftrightarrow |x - hk| \leq R \cdot h \leq \delta$, i.e. in particular supp $\psi_k \subseteq B_\delta(hk)$ if h is small enough, namely $h < \frac{\delta}{R}$.

*3.* Assume that $x \in \mathbb{R}^d$ is fixed. Since $f(x) = f(x) \cdot 1 = \sum_{k\in\mathbb{Z}^d} \psi_k(x) f(x)$, it only has to be estimated

$$|(Q_h f - f)(x)| = |Q_h f(x) - f(x) \underbrace{\sum_{k\in\mathbb{Z}^d} \psi_k(x)}_{=1}| \leq \sum_{k\in\mathbb{Z}^d} |(f(hk) - f(x))| \psi_k(x).$$

Due to *Step 2* the following inequality

$$|f(hk) - f(x)| \leq osc_\delta(f)(x) \quad \text{for} \quad |hk - x| \leq \delta$$

holds, which implies the desired estimate

$$|(Q_h f - f)(x)| \leq \sum_{k\in\mathbb{Z}} osc_\delta(f)(x)\psi_k(x) \leq osc_\delta(f)(x) \quad .$$

Consequently

$$\|Q_h f - f\|_\infty \leq \|\mathrm{osc}_\delta(f)\|_\infty.$$

□

To obtain a generalisation of the estimate (2.3) the choice of the function space is of paramount importance. In signal processing, sampling theory, wavelet theory, etc. a lot of research on Quasi-interpolation in the Feichtinger algebra $S_0(\mathbb{R}^d)$[15], the Fourier algebra, Wiener Amalgam spaces, Sobolev spaces[16], etc. has been done. To obtain a better appreciation of those spaces and the resulting convergence criteria see [12], [13], [14], [34] and [39].

The approximation theorem of Stone-Weierstrass[17] signifies that each continuous function $f$ can be uniformly approximated by polynomials to any degree of accuracy in the uniform sense.
In contrast Bernstein did not claim that the function $f$ should be interpolated by polynomial functions. His idea was to make the intervall small enough, so that a good approximation is possible.

Sometimes it is quite difficult to receive an interpolating polynom because a huge linear equation system has to be solved, and therefore quasi-interpolation methods are a good alternative to Interpolation theory. Quasi-interpolants guarantee smooth approximation and are easier to compute. The big disadvantage is the fact, that the quasi-interpolant do not interpolate correctly.

In order to improve the situation an **iterative procedure** can be described, which applies at least for quaradtic and cubic splines.:
Consider the case $d = 1$; and since one can reduce the dilation to the case $a = 1$, let us assume without loss of generality $a = 1$.
Given a sequence of data $(f(n))_{n\in\mathbb{Z}}$ one can form $f^{(1)} := Q_\varphi f$ as a first approximation, using as a generator the cubic B-spline $\varphi$. Since

$$d^{(1)}(n) := f(n) - Q_\varphi f(n) = f(n) - f^{(1)}(n)$$

may be different from zero, it is plausible to correct this error by adding the function

$$g^{(1)}(x) = \sum_{n\in\mathbb{Z}} d^{(1)}(n) T_n \varphi(x) \quad .$$

[15]The function space $S_0(\mathbb{R}^d)$ includes continuous and integrable functions. Nowadays it is called Segal Algebra or **Feichtinger's algebra**.

[16]Let $\Omega \subseteq \mathbb{R}^N$ be open, $m \in \mathbb{N}$, und $1 \leq p < \infty$.
The vectorspace $W^{m,p}(\Omega) := \{f \in L^p(\Omega) \mid \forall \alpha \text{ mit } 1 \leq |\alpha| \leq m \quad \exists D^\alpha f \in L^p(\Omega)\}$ is called **Sobolev space**.

[17]If $f$ is a continuous real-valued function on an interval $[a,b]$, then for every $\epsilon > 0$ there exists a polynomial function $P$ on $[a,b]$, such that $|f(x) - P(x)| < \epsilon \quad \forall x \in [a,b]$.

Then $f^{(2)} := f^{(1)} + g^{(1)} = Q_\varphi f + g^{(1)}$ provides a better approximation with

$$d^{(2)}(n) = f(n) - f^{(2)}(n)$$

even smaller, in fact $\|d^{(2)}\|_{\ell^2(\mathbb{Z})} < \gamma \|d^{(1)}\|_{\ell^2(\mathbb{Z})}$ for some $\gamma < 1$.
The existence of $\gamma < 1$ stems from the fact that

$$\sum_{n \neq 0} \varphi(n) \leq \gamma \cdot \varphi(0) \quad ,$$

which is certainly valid for cubic and other low order B-splines.
Going on like that, i.e. forming

$$f^{(m+1)} = f^{(m)} + g^{(m)} = f^{(m)}(n) + \sum_{n \in \mathbb{Z}} d^{(m)}(n) T_n \varphi$$

one finds that $\|d^{(m)}\|_{\ell^2(\mathbb{Z})} \leq \gamma^{m-1} \|d^{(1)}\|_{\ell^2(\mathbb{Z})} \to 0$ for $m \to \infty$, i.e. in the limit one has perfect interpolation of the data $(f(n))_{n \in \mathbb{Z}}$.
Since (by induction) one has for each $m \in \mathbb{N}$

$$\|\sum_{n \in \mathbb{Z}} d^{(m)}(n) T_n \varphi\|_{L^2(\mathbb{R})} \leq B' \|d^{(m)}\|_{\ell^2}.$$

The series built in this way is convergent, because

$$\begin{aligned} \|\textstyle\sum_{m=1}^\infty \sum_{n \in \mathbb{Z}} d^{(m)}(n) T_n \varphi\|_{L^2(\mathbb{R})} &\leq \textstyle\sum_{m=1}^\infty \|\sum_{n \in \mathbb{Z}} d^{(m)}(n) T_n \varphi\|_{L^2(\mathbb{R})} \\ &\leq B' \textstyle\sum_{m=1}^\infty \gamma^{m-1} \|d^{(1)}\|_{\ell^2(\mathbb{Z})} \leq B' \frac{1}{1-\gamma} < \infty \quad . \end{aligned}$$

Hence $h = Q_\varphi f + \sum_{n \in \mathbb{Z}} d^{(m)}(n) T_n \varphi$ belongs to $V_\varphi$ and interpolates exactly.

There is however a method similar to the one used for biorthogonality of the Riesz basic sequence to obtain this interpolating spline function as quasi-interpolator, at the cost of replacing $\varphi$ by another generator $\varphi_L$, the **Lagrange interpolator**.

**Lemma 2.21.** *If* $(T_{ak}\varphi)$ *is a Riesz basic sequence, and* $\hat{\varphi}(s) \geq 0\ \forall s \in \mathbb{R}$ *(e.g. cubic B-splines), then* $\Phi(s) = \sum_{n\in\mathbb{Z}} \hat{\varphi}(s - n/a)$ *is free of zeros, and consequently one can define a Lagrange interpolator* $\varphi_L$ *with*

$$\varphi_L(ak) = \delta_{0,ak}$$

*in* $V_{\varphi,a}$ *by its Fourier transform*

$$\mathcal{F}(\varphi_L)(s) = \frac{\hat{\varphi}(s)}{\sum_{n\in\mathbb{Z}^d} \hat{\varphi}(s - n/a)} \quad .$$

*Moreover,* $\varphi_L$ *is uniquely determined within* $V_{\varphi,a}$ *by this property.*

**Remark 2.22.** *The result above turns out to be exactly* $Q_{\varphi_L} f$*.*

In practice the Lagrange interpolator is a frequently used technology in image processing. To gain a bit of more insight see [16].

## 2.3 Radial basis functions

To get good approximation properties the choice of the the generator (function) $\varphi$ is of prime importance.
Buhmann[18] applies quasi-interpolants with **radial basis functions** $\phi$, where $\varphi$ is a finite linear combination of radial basis functions, i.e.

$$\varphi(x) = \sum_{|k|<F} \lambda_k \phi(|x-k|), \quad x \in \mathbb{R}, \quad N > 0.$$

Several examples for radial basis functions are $\phi(x) = x^2 \log x$ (thin-plate splines), $\phi(x) = \sqrt{x^2+c^2}$ for $c > 0$ (multiquadratic functions) or $\phi(x) = \exp^{-\alpha x^2}$ for $\alpha > 0$ (Gaussian functions).

The quasi-interpolant $Q_h f$ depends on $h$ as well as on $f$, however $\varphi(x)$ is completely independent of the function $f$. Although $Q_h f$ does not need to satisfy any interpolation properties, it should be a proper approximation due to qualities of the function $\varphi$, i.e. $Q_h f \approx f$.
To guarantee the convergence for $d = 1$ of $Q_h f$, $\varphi$ has to satisfy the conditions

$$\sum_{k\in\mathbb{Z}} |\varphi(x-k)| < \infty \quad \text{and} \quad \sum_{k\in\mathbb{Z}} \varphi(x-k) \equiv 1 \quad \text{for all} \quad x \in \mathbb{R}.$$

For though these conditions are relative weak, they will yield to good approximation methods.

**Lemma 2.23.** [19] *Presupposed the radial basis functions are of the form $\phi(x) = \sqrt{x^2+c^2}$ with a parameter $c > 0$ and $\varphi$ is a second divided difference of $\phi$, e.g.*

$$\varphi(x) = \frac{1}{2}\phi(|x-1|) - \phi(|x|) + \frac{1}{2}\phi(|x+1|) \quad ,$$

*then the following polynomial reproduction property*

$$\sum_{k\in\mathbb{Z}} (a+bk)\varphi(x-k) = a + bx \quad \textit{for any} \quad x, a, b \in \mathbb{R}$$

*holds.*

Lemma 2.23. supplies polynomial recovery, which helps to receive a uniform convergence result by suitable Taylor[20] polynomials.

[18] Martin D. Buhmann is a German mathematician.
[19] To prove Lemma 2.16. see [4] page 18-19.
[20] $T(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + ... + \frac{f^n(a)}{n!}(x-a)^n + R_n$ is called **the n-th Taylor polynomial** of $f$ around $a$ with error term $R_n$.

**Theorem 2.24. Convergence Theorem**[21]
*Let the function $f$ be twice differentiable such that $\|f'\|_\infty$ and $\|f''\|_\infty$ are finite. Then for any nonnegative $c$*

$$\|f - Q_h f\| = O(h^2 + c^2 h^2 |\log h|), \quad h \to 0 \quad .$$

It is quite evident that there exists a close relation between convergence and polynomial recovery, because smooth functions can be locally approximated by Taylor polynomials which may be recovered by quasi-interpolation.
Radial basis functions $\varphi$ are good components to approximate linear polynomials by quasi-interpolant just as well as smooth functions of higher order. These results were generalized for arbitrary Hilbert spaces by [4], [5] and [40].

[21] To prove Theorem 2.17. see [4] page 20-21.

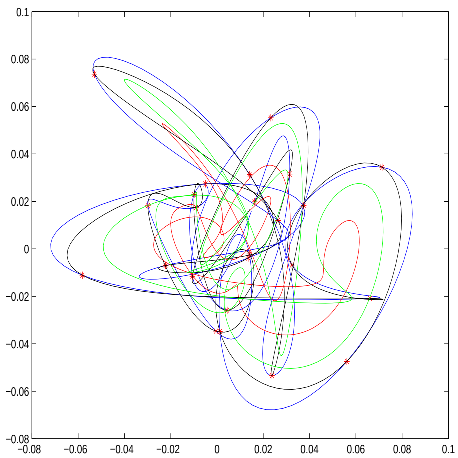
# CHAPTER 3

# INTERPOLATION VS. QUASI-INTERPOLATION

Quasi-interpolating methods compared with interpolating schemes show that quasi-interpolants are a good alternative for approximation.

**Example 3.1.**

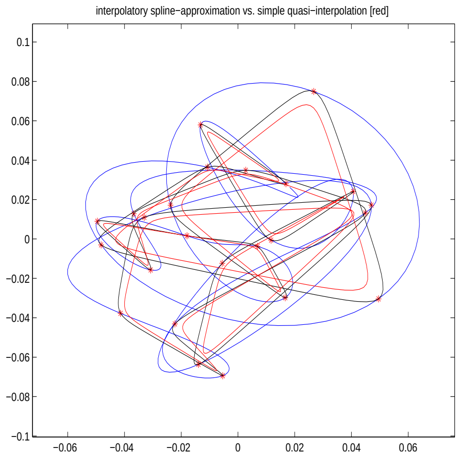
```
xx = lowsign(n,11);
plot(xx);
hold on; plot(xx(1:a:n),'r*');figure(gcf);
xs = xx(1:a:n);
xxbsp = xs*BUPU;
plot(xxbsp,'r'); figure(gcf)
xxbspi = xs*trlbas(bspi,a);
plot(xxbspi,'k');
plot(xs*trlbas(bspai2,a),'g'); figure(gcf)
```



This is a good example to show how difficult it is to find a suitable quasi-interpolant for some data points. The black interpolating function fits the given data set perfectly. Although the red quasi-interpolanting function is smoother, the green quasi-interpolating function draws near the original blue function. But both quasi-interpolants do not fit all the given points. The choice of $\varphi$ and the density of the samples are an important tool for the quasi-interpolant to approach the given points.

**Example 3.2.**

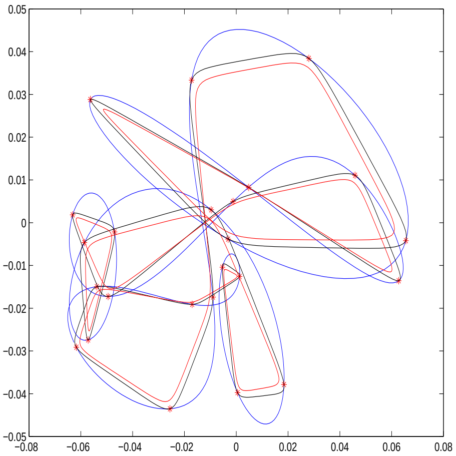
```
xx = real(lowsign(n,20));
n/41;
xx = real(lowsign(n,10));
n/11; n/21;
plotcsa(xx,a);
xx = (lowsign(n,10));
plotcsmpi(xx,a);
plot(xx);
xx1= [xx, xx(1)];
plot(xx1);
hold on; plot(xx1(1:a:n),'r*');
[BUP,bsp] = bupuspline(n,a,a/2,4);
plotc(bsp);
bspi = trl1int(bsp,a); plotc(bspi);
sh = zeros(1,n); sh(1:a:n) = 1;
xxint = ifft( fft(xx.*sh) .* fft(bspi));
plot(xx); hold on; plot(xx(1:a:n),'r*'); plot([xxint,xxint(1)] ,'k '); hold off;
plot(real(xx)); hold on; plot(real(xxint),'k');
plot(real(xx(1:a:n)),'r*'); hold off;
plot(real(xxint),'k'); plot(1:a:n,real(xx(1:a:n)),'r*');
xxbsp = ifft( fft(xx.*sh) .* fft(bsp));
hold on; plot(real(xxbsp),'r'); figure(gcf)
```




This plot indicates the interpolant and the quasi-interpolant of a smooth complex function. The blue curve shows the choice of a smooth signal with maximal frequency 10, so that 21 random complex coefficients are used. The red curve is the cubic quasi-interpolant, which does not match all the points exactly. On the contrary the black curve is the interpolant via spline functions, which matches exactly at the given sampling points.

**Example 3.3.**

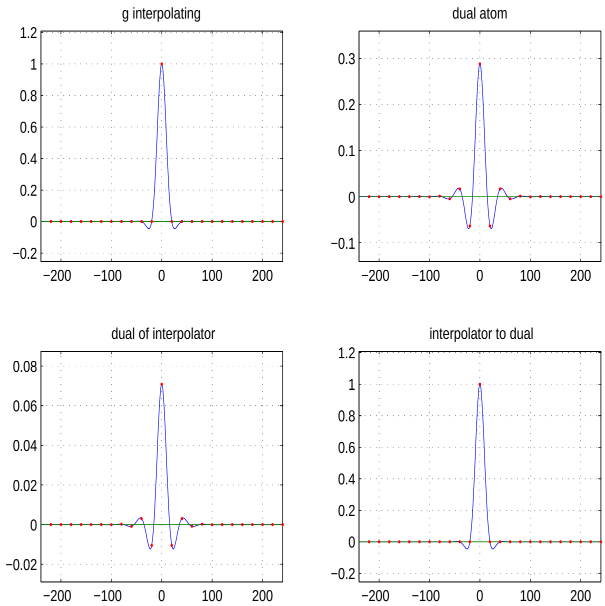
```
maxfr = 8;
xx = (lowsign(n,maxfr));
sh = zeros(1,n); sh(1:a:n) = 1;
[BUPU, bsp ] = bupuspline(n,a,a/2,4);
bspi = trl1int(bsp,a);
hold off; plot(xx);
xp = 1:a:n;
hold on; plot(xx(xp),'r*');
xxint = ifft( fft(xx.*sh) .* fft(bspi));
plot([xxint, xxint(1)], 'k');
xxbsp = ifft( fft(xx.*sh) .* fft(bsp));
plot([xxbsp, xxbsp(1)],'r-'); figure(gcf)
```




Example 3.3. shows that the quasi-interpolant function in red approximates the original function in blue quite exactly. Although the number of sampling points is low, the choice of $\varphi$ is good enough so that in some areas the actual error is nearly zero.

**Example 3.4.**

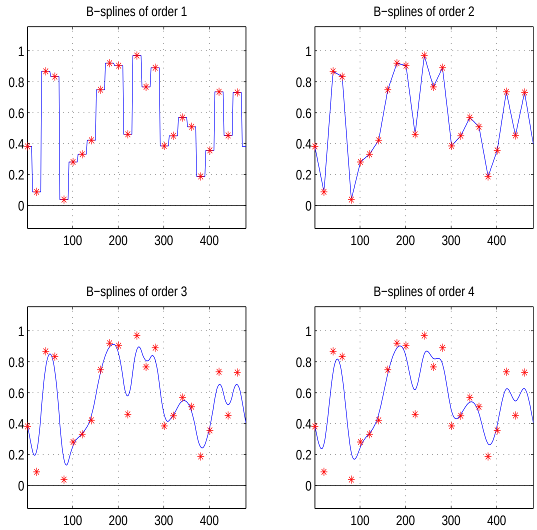
```
function [gi,gd,gdi,gid] = nicoldem1(n,a);
if exist('n') == 0; n = 480; end;
if exist( 'a') ==0; a= 20; end;
g = gaussnk(n);
gi = trl1int(g,a);
gd = trl1du(g,a);
gdi = trl1int(gd,a);
gid = trl1du(gi,a);
subplot(221); plotcsmp(gi,a,'r.'); title('g interpolating');
subplot(223); plotcsmp(gd,a,'r.'); title('dual atom');
subplot(223); plotcsmp(gid,a,'r.'); title('dual of interpolator');
subplot(224); plotcsmp(gdi,a,'r.'); ; title('interpolator to dual');
```



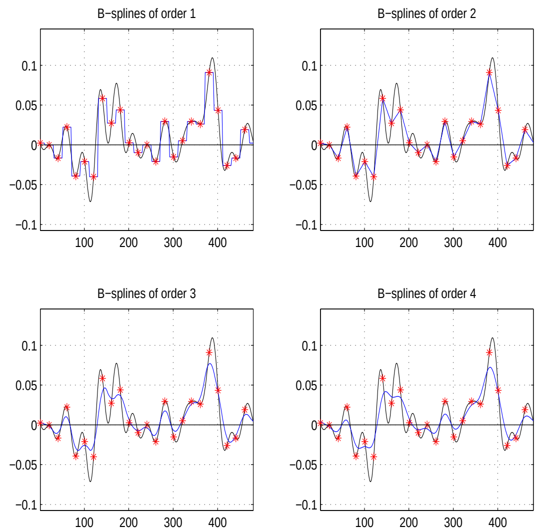

The first row of pictures shows the interpolator of the Gaussian function $g$ and the dual atom of $g$. In the next step the dual of the interpolator is built and the dual of $g$ is interpolated. The pictures of the "dual atom" and the "dual of the interpolator" look alike, but it is just a visual illusion. On closer inspection one sees that the scaling is completely different. On the contrary the pictures of "g interpolating" and "interpolator to dual" are identical.
All these elements received by translations are in the same space.

**Example 3.5.**

The first series of plots shows some random real data, marked with red stars, which should be the given data at the regularly spaced points in $Z_{480}$ at a relative distance of 20 samples apart from each other. After using of B-spline (quasi)-interpolation for B-splines of order $j = 1, 2, 3, 4$, one can see that the B-splines are interpolating the data for $j = 1, 2$, but unfortunately not for $j \geq 2$.

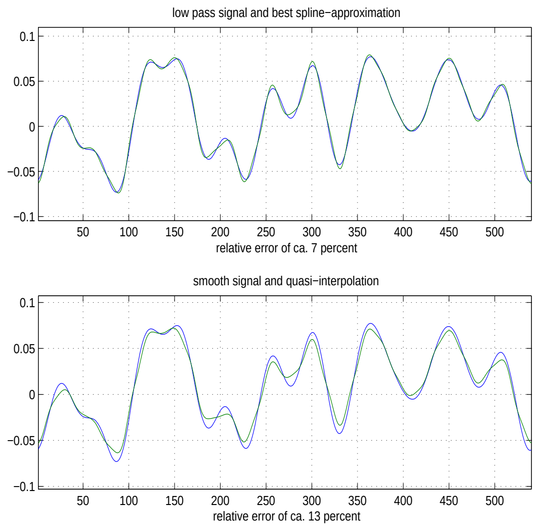



In the next picture the black curve indicates a smooth curve, generally a real valued and band-limited function, with 31 random complex Fourier coeffcients.

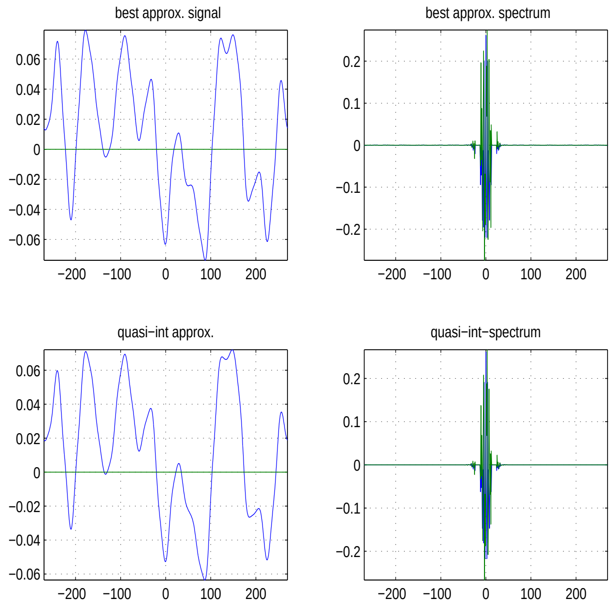


**Example 3.6.**

The picture of Example 3.6. shows that the best spline-approximation, given by the green curve, approximates the original function in blue nearly exactly. The relative error is around 7%. The quasi-interpolating curve in the second window is a very good approximation too. The relative error is arounf 13%.




The quasi-interpolation spectrum shows that the oscillations at the endings get smaller compared to the spectrum of the best spline-approximation, but do not disappear completely.



# MATLAB-FILES

All MATLAB files stem from the NuHAG database. The holder of the copyright of these codes is the NuHAG. I do not claim these codes being my own work. The computations for the experiments are done in cooperation with Hans G. Feichtinger. All functions that are not listed and explained in this chapter here, but appear in the code, are standard MATLAB functions. The codes can be found in the NuHAG database:
http://www.univie.ac.at/nuhag-php/home/index.php

**bupuspline** creates discrete B-spline type BUPUs

- Input:
  - n = signal length
  - gap = lattice constant
  - bas = basis of smoothing spline of order zero
  - ord = order of smoothness (through iterative convolution) default: ord=5
- Output: BUPU = elements of the BUPU rowwise

Usage: [BUPU,bsp] = bupuspline(n,gap,bas,ord)

**lowsign(n,a)** generates a signal of length n (row vector) with maximal (random) frequency r

Usage: xx = lowsign(n,r)

gaussnk creates the canonical discrete gauss function

Input: xx = some signal
n = dimension

Output: g = a canonically discretized gaussian, an eigenvector with eigenvalue 1 of the unitary DFT (that is g=fft(g)/sqrt(n)) (normed with respect to 2-norm)

Usage: g = gaussnk(n)
g = gaussnk(xx)

plotcsa adds samples on a central plot

Input: xx = a complex valued signal
a = sampling distance
col1, col2 = plotting colors

Usage: plotcsa(xx,a,col1,col2)
plotcsa(xx,a,col)
plotcsa(xx,a)

plotcsmp adds sampling markers on a central plot

Input: xx = a real or complex signal
a = vector of sample indexes (can be written in the form 1:step:n or just step as a singular value)
symb1,symb2 = symbols that appear on the plot(x,d,s,o,...)

Usage: plotcsmp(n,a,symb)
plotcsmp(xx,a,symb)

plotcsmpi adds samples on a central plot (also works for irregular sampling)

Input: xx = a real or complex signal
a = vector of sample indexes (can be written in the form 1:step:n or just step as a singular value)
symb1,symb2 = symbols that appear on the plot(x,d,s,o,...)

Usage: plotcsmpi()
plotcmspi(xx)
plotcsmpi(xx,a)
plotcsmpi(xx,a,'*','o')

trlbas the result is the translation basis matrix

Input: wind = a vector
xp = a vector
gap = a number
offset = the starting point of the regular grid

Output: trl = a matrix

Usage: trl = trlbas(wind,gap,offset)
= trlbas(wind,xp); by default offset=1
= trlbas(wind); by default xp=1:length(wind)
= trlbas; starts a demonstration

trl1du the resulting vector atdu is the dual atom
(works for real and complex atoms)

Input: atom, xp = vectors
gap = number
thresh = number

Output: atdu,xes = vectors
dev = scalar product of atdu with atom
condfam = condition number of Riesz basis

Usage: [atdu,xes,dev,condfam] = trl1du(atom,gap,thresh)
= trl1du(atom,gap); by default thresh=$10^{-10}$
= trl1du(atom); by default xp=1:length(atom)
= trl1du; starts a demonstration

trl1int is a Lagrange type interpolating function which tries to solve
the interpolation grid over a grid with step-width gap
(the generating space might be different from the interpolation grid)

Usage: atint = trl1int(atom,[gap])
= trl1int(atom,xp)

# ABSTRACT

In many lectures the readers are introduced in the topic of Interpolation Theory. They learn about the characteristics of Lagrange interpolation, the Newton interpolation and spline interpolation. They get a feeling how interpolating polynomials work and what their advantages are. Nevertheless the reader rarely hears about another very useful approximation method, namely the Quasi-interpolation.

Quasi-interpolation is a good methodology that allows to obtain from given discrete data (generally regular samples of a continuous function) smooth functions that observe the general behaviour of the underlying continuous function, although these functions do not fit the given data exactly.

For the descripton of the quasi-interpolant operator the so-called Spline-type space, respectively a principal shift-invariant space, is quite useful, and the set of translates of some "atom" $\varphi$ form a Riesz basis for such a space. The master thesis informs the reader about the most important characteristics of these Riesz basic sequences.

One difficulty of Quasi-interpolation is the fact, that in the application area of Approximation Theory there does not exist a unique defnition of the quasi-interpolant operator. The different definitions and their qualities are discussed and some theoretical statements are shown and proved.

At the end of this master thesis some practical applications via MATLAB experiments demonstrate the correspondence between Interpolation and Quasi-interpolation. The reader sees that Quasi-interpolation is a good alternative to Interpolation.

# ZUSAMMENFASSUNG

Vorlesungen über die verschiedensten Methoden der Approximationstheorie beinhalten immer das Thema "Interpolation". Man hört von den unterschiedlichsten Arten der Interpolation, beginnend von der Lagrange-Interpolation über die Newton-Interpolation bis hin zur Interpolation via Splines. Man erfährt von den Vorteilen und ebenso von den Nachteilen der einzelnen interpolierenden Polynome; jedoch wird eine doch recht nette andere Methode der Interpolation, nämlich die Quasi-Interpolation, kaum behandelt.

Mittel Quasi-Interpolation erzeugt man eine schöne glatte Kurve, welche alle Eigenschaften der Orginalfunktion beinhaltet, aber leider nicht die gegebenen Daten genau interpoliert. Trifft man eine gute Wahl für das "Atom" $\varphi$, so ist der Approximationsfehler minimalst.

Der Raum, in welchem die Quasi-interpolierenden Funktionen aggieren, ist der Spline-type space, welcher von den Riesz Basen aufgespannt wird. Diese Riesz Basen haben einige besondere Eigenschaften, welche in dieser Diplomarbeit kurz besprochen und behandelt werden.

Die Schwierigkeit der Quasi-Interpolation liegt eher darin, dass es leider keine eindeutige Definition des Quasi-interpolant Operators gibt. Viele Mathematiker verwenden den Begriff "Quasi-Interpolation" in den unterschiedlichsten Zusammenhängen. Diese Diplomarbeit gibt einen kleinen Einblick in die verschiedenen Definitionen und deren theoretischen Aussagen.

Zu Ende hin sieht man anhand praktischer Beispiele via MATLAB den Zusammenhang zwischen Interpolation und Quasi-Interpolation. Man erkennt, dass Quasi-Interpolation eine gute Alternative zur Interpolation darstellt.

# CURRICULUM VITAE

12.5.1982: Geboren in 1090 Wien

1988-1992: Besuch der Volksschule West in Stockerau

1992-2000: Besuch des Bundesrealgymnasiums Stockerau, Matura Abschluss

2000-2013: Studium der Mathematik an der Universität Wien

2010-2013: Vertragslehrer an der HAK-Korneuburg

# INDEX

# BIBLIOGRAPHY

[1] Berrut, J.P. , and Trefethen, L.N. , *Barycentric Lagrange Interpolation*, SIAM Review, Vol. 46, 2004

[2] *Biographical Abstract: Isaac Jacob Schoenberg*, http://www-history.mcs.st-andrews.ac.uk/Biographies/Schoenberg.html, University of St.Andrews, 1998

[3] Boehm, W. , and Prautzsch, H. , *Numerical Methods*, Vieweg, 1993

[4] Buhmann, M.D. , *Radial Basis Functions: Theory and Implementations*, Cambridge University Press, 2003

[5] Buhmann, M.D. , *On Quasi-interpolation with Radial Basis Functions*, Journal of Approximation Theory, Vol. 72, 1993

[6] Burden, R.L. , and Faires, J.D. , *Numerical Analysis*, Brooks Cole, 2004

[7] Christensen, O. , *An introduction to frames and Riesz bases*, Birkhäuser, 2003

[8] Condat, L. , and Blu, T. , *Beyond Interpolation: Optimal Reconstruction by Quasi-Interpolation*, http://bigwww.epfl.ch/preprints/condat0502p.pdf, 2005

[9] Deuflhard, P. , and Hohmann, A. , *Numerische Mathematik 1: Eine algorithmisch orientierte Einführung*, Walter de Gruyter, 1993

[10] De Boor, C. , De Vore, R.A. and Ron, A. , *Approximation from shift-invariant subspaces of* $L^2(\mathbb{R}^d)$, Trans. Am. Math. Soc., Vol. 341(2), 1994

[11] De Boor, C. , De Vore, R.A. and Ron, A. , *The structure of finitely generated shift-invariant spaces in* $L^2(\mathbb{R}^d)$, Journal of Functional Analysis, Vol. 119(1), 1994

[12] Feichtinger, H.G. , and Kaiblinger, N. , *Quasi-Interpolation in the Fourier Algebra*, Journal of Approximation Theory, 2007

[13] Feichtinger, H.G. , and Werther, T. , *Robustness of regular sampling in Sobolev algebras*, Birkhäuser, 2004

[14] Feichtinger, H.G. , *A functional analytic approach to applied analysis* http://www.univie.ac.at/NuHAG/FEICOURS/ws1213/ANGAN1213.pdf, 2012

[15] Gasca, M. , and Sauer, T. , *On the history of multivariate polynomial interpolation*, Computation of Curves and Surfaces, 1990

[16] Getreuer, P. , *Linear Methods for Image Interpolation*, http://www.ipol.im/pub/art/2011/g_lmii/, 2011

[17] *History of Splines* http://www.alias.com/eng/support/studiotools/documentation/Using/About4.html

[18] Hoschek, J. , and Lasser, D. , *Fundamentals of Computer-Aided*

[19] Jüttler, B. , *Skriptum zur Vorlesung Computer Aided Geometric Design*, http://www.ag.jku.at/um/2009cagd-40.pdf, 2009

[20] Meijering, E. , *A Chronology of Interpolation: From Ancient Astronomy to Modern Signal and Image Processing*, Proceedings of the IEEE, Vol. 90, 2002

[21] Neumaier, A. , *Introduction to Numerical Analysis*, Cambridge University Press, 2001

[22] Plonka-Hoch, G. , *Numerik 2 - Numerische Analysis*, http://www.uni-due.de/~hm0029/Numerik2/NumAnalysis_komplett.pdf, 2007

[23] Quarteroni, A. , Sacco, R. , and Saleri, F. , *Numerical Mathematics*, Springer, 2007

[24] Rabut, C. , *Quasi-interpolants and Quasi-interpolation: Important Properties and Propositions of Terminology*, Journal of Approximation Theory, 1994

[25] Risler, J.J. , *Mathematical Methods for CAD*, Cambridge University Press, 1992

[26] Ron, A. , and Shen, Z. , *Frames and stable bases for shift-invariant subspace of* $L^2(\mathbb{R}^d)$, Canadian Journal of Mathematics Vol. 47, 1995

[27] Ron, A. , *Introduction To Numerical Analysis*, http://www.wepapers.com/course_view/1193/Introduction_to_numerical_analysis/Courses/807, 2006

[28] Schichl, H. , *Skriptum zur Vorlesung Numerik 1*, 2000/01 Geometric Design, 1993

[29] Schwarz, H. R. , and Köckler, N. , *Numerische Mathematik*, B.G. Teubner Verlag, 2006

[30] *Section 3: Newton Divided-Difference Interpolating Polynomials*, http://pegasus.cc.ucf.edu/~klee/EGN3420/Notes/Interp_NDD.pdf

[31] Steffens, K. G. , *The History of Approximation Theory: From Euler to Bernstein*, Birkhäuser, 2006

[32] Unser, M. , *Splines: A perfect fit for medical imaging*, Progress in Biomedical Optics and Imaging, 2002

[33] Unser, M. , *Splines: A perfect fit for signal and image processing*, IEEE Signal Processing Magazine, 1999

[34] Werther, T. , *Reconstruction from Irregular Samples with Improved Locality* http://homepage.univie.ac.at/tobias.werther/publications/mthesis.pdf, 1999

[35] *Wikipedia, the free encyclopedia*, http://en.wikipedia.org/wiki/Interpolation

[36] *Wikipedia, the free encyclopedia*, http://en.wikipedia.org/wiki/Wiener_amalgam_space

[37] Wolberg, G. , and Alfy, I. , *Monotonic Cubic Spline Interpolation*, Proceedings of Computer Graphics International, 1999

[38] Wu, ZongMin, *Best quasi-interpolation and scattered data quasi-interpolation*, http://ww1.math.nus.edu.sg/AMC/papers-invited/Wu-Zongmin.pdf, 2005

[39] Wu, Z.M., Liu, J.P. , *Generalized Strang-Fix condition for scattered data quasi-interpolation*, http://wwwx.cs.unc.edu/Research/compgeom/twiki/pub/TModeling/SubdivisionSurfaces/pre04-02.pdf, 2004

[40] Zha, L. , and Feng, R. , *A quasi-interpolation satisfying quadratic polynomial reproduction with radial basis functions*, Journal of Chinese Universities, Vol. 16(4), 2007

**SACHERSCHLIESSUNG**
der
**Fachbereichsbibliothek Mathematik, Statistik, Informatik der Universität Wien**

**Klassifikation**

**Aufstellungsort:** ☒ **HoA** ☐ **HoD**

**MSC2000: 41A05 – 41A15 – 41A30 – 65D05 – 65D07 – 65D10**

**BK: 31.76 – 31.35**

**Schlagwortketten nach RSWK**
Spline-Interpolation – Spline-Approximation (21)

Kontrollvermerk der Fachbibliothek:

**02. Apr. 2013**
Datum

Unterschrift