7
Interpolation and Approximation

In Numerical Analysis one often encounters the situation that instead of a function $f : \mathbf{R} \to \mathbf{R}$, only a few discrete function values $f(t_i)$ are given, and maybe derivatives $f^{(j)}(t_i)$ at finitely many points t_i. This is the case, for example, when the function f is given in the form of experimental data. Also, most methods for solving differential equations calculate a solution $f(t)$ (including its derivative) only at finitely many positions. Historically, this problem occurred in the computation of additional function values between tabulated ones. Nowadays, one of the most important applications occurs in computer graphics, known by the abbreviations CAD (**C**omputer-**A**ided **D**esign) and CAGD (**C**omputer-**A**ided **G**eometric **D**esign).

If one is interested in total behavior of the function, then, from the given data

$$f^{(j)}(t_i) \quad \text{for} \quad i = 0, \ldots, n \quad \text{and} \quad j = 0, \ldots, c_i \,,$$

one should construct a function φ which differs as little as possible from the original function f. In addition, the function φ should be simple to evaluate, like, for example, (piecewise) polynomials, trigonometric, exponential, or rational functions. A first obvious requirement regarding the function φ is the *interpolation property*: φ should coincide with the function f at the *nodes* (sometimes also called *knots*) t_i,

$$\varphi^{(j)}(t_i) = f^{(j)}(t_i) \quad \text{for all} \quad i, j \,.$$

The values $f^{(j)}(t_i)$ are called *node values*. If we compare the two functions φ_1 and φ_2 in Figure 7.1, then both obviously satisfy the interpolation condition at given values $f(t_i)$. Nevertheless, we would prefer φ_1. In addition

P. Deuflhard et al., *Numerical Analysis in Modern Scientific Computing*

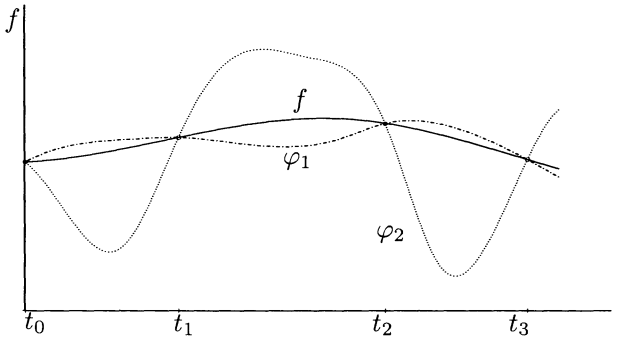

Figure 7.1. Various interpolating functions for f.

to the interpolation property, we therefore require also the *approximation property*: φ should differ as little as possible from f with respect to a norm $\|\cdot\|$ in a suitable function space,

$$\|\varphi - f\| \quad \text{“small.”}$$

7.1 Classical Polynomial Interpolation

We start with the simplest case, when only the values

$$f_i := f(t_i) \quad \text{for} \quad i = 0, \ldots, n$$

are given at the pairwise distinct nodes $t_0, \ldots, t_n$. We now seek a polynomial $P \in \mathbf{P}_n$ of degree $\deg P \leq n$,

$$P(t) = a_n t^n + a_{n-1} t^{n-1} + \cdots + a_1 t + a_0 \quad \text{with} \quad a_0, \ldots, a_n \in \mathbf{R},$$

which interpolates f at the $n+1$ nodes $t_0, \ldots, t_n$, i.e.,

$$P(t_i) = f_i \quad \text{for} \quad i = 0, \ldots, n. \tag{7.1}$$

7.1.1 Uniqueness and Condition Number

The following argument shows that the $n+1$ unknown coefficients $a_0, \ldots, a_n$ are uniquely determined by the condition (7.1): If $P, Q \in \mathbf{P}_n$ are two interpolating polynomials with $P(t_i) = Q(t_i)$ for $i = 0, \ldots, n$, then $P - Q$ is a polynomial of at most n^{th} degree with the $n+1$ roots $t_0, \ldots, t_n$, and is therefore the null-polynomial. But the rule

$$\mathbf{P}_n \to \mathbf{R}^{n+1}, \quad P \mapsto (P(t_0), \ldots, P(t_n))$$

is also a linear mapping between the two $(n+1)$-dimensional real vector spaces $\mathbf{P}_n$ and $\mathbf{R}^{n+1}$, so that injectivity already implies surjectivity. We have therefore proven the following theorem.

Theorem 7.1 *Suppose $n+1$ nodes (t_i, f_i) for $i = 0, \ldots, n$ are given with pairwise distinct nodes $t_0, \ldots, t_n$, then there exists a unique interpolating polynomial $P \in \mathbf{P}_n$, i.e., $P(t_i) = f_i$ for $i = 0, \ldots, n$.*

The unique polynomial P, which is given by theorem 7.1, is called *interpolating polynomial* of f for the pairwise distinct nodes $t_0, \ldots, t_n$, and it is denoted by

$$P = P(f \mid t_0, \ldots, t_n) \; .$$

In order actually to compute the interpolating polynomial, we have to choose a basis of the space of polynomials $\mathbf{P}_n$. In order to illustrate this, we first give two classical representations. If we write P as above in *coefficient representation*

$$P(t) = a_n t^n + a_{n-1} t^{n-1} + \cdots + a_1 t + a_0 \, ,$$

i.e., with respect to the *monomial basis* $\{1, t, \ldots, t^n\}$ of $\mathbf{P}_n$, then the interpolation conditions $P(t_i) = f_i$ can be formulated as a linear system

$$\underbrace{\begin{bmatrix} 1 & t_0 & t_0^2 & \cdots & t_0^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & t_n & t_n^2 & \cdots & t_n^n \end{bmatrix}}_{=: V_n} \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f_0 \\ \vdots \\ f_n \end{pmatrix} \; .$$

The matrix V_n is called *Vandermonde matrix*. For the determinant of V_n, we have

$$\det V_n = \prod_{i=0}^{n} \prod_{j=i+1}^{n} (t_i - t_j) \, ,$$

which is proven in virtually every linear algebra textbook (see ,e.g., [61]). It is different from zero exactly when the nodes $t_0, \ldots, t_n$ are pairwise distinct (in agreement with Theorem 7.1). However, the solution of the system requires an excessive amount of computational effort when compared with the methods that will be discussed below.

In addition, the Vandermonde matrices are almost singular in higher dimensions n. Gaussian elimination *without* pivoting is recommended for its solution, because pivoting strategies may perturb the structure of the matrix (compare [51]). For special nodes, the above Vandermonde matrix can easily be inverted analytically. In Section 7.2 we shall encounter an example of this.

An alternative basis for the representation of the interpolation polynomial is formed by the *Lagrange polynomials* $L_0, \ldots, L_n$. They are defined as the uniquely determined interpolation polynomials $L_i \in \mathbf{P}_n$ with

$$L_i(t_j) = \delta_{ij} \; .$$

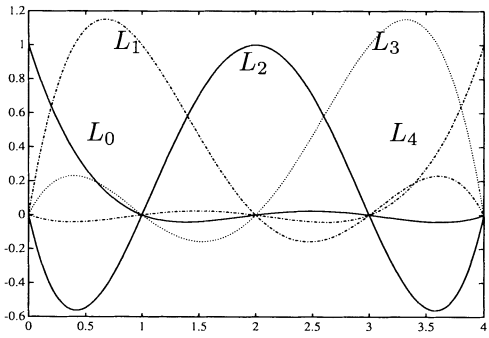

Figure 7.2. Lagrange polynomials L_i for $n = 4$ and equidistant nodes t_i.

The corresponding explicit form (compare Figure 7.2) is

$$L_i(t) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{t - t_j}{t_i - t_j} .$$

The interpolating polynomial P for arbitrary nodes $f_0, \ldots, f_n$ can easily be built from the Lagrange polynomials by superposition: With

$$P(t) := \sum_{i=0}^{n} f_i L_i(t) , \tag{7.2}$$

we obviously have

$$P(t_j) = \sum_{i=0}^{n} f_i L_i(t_j) = \sum_{i=0}^{n} f_i \delta_{ij} = f_j .$$

Remark 7.2 The above statement can also be phrased as follows: The Lagrange polynomials form an orthogonal basis of $\mathbf{P}_n$ with respect to the scalar product

$$\langle P, Q \rangle := \sum_{i=0}^{n} P(t_i) Q(t_i)$$

for $P, Q \in \mathbf{P}_n$. Let $\langle P, L_i \rangle = P(t_i)$, then, obviously,

$$P = \sum_{i=0}^{n} \langle P, L_i \rangle L_i = \sum_{i=0}^{n} P(t_i) L_i = \sum_{i=0}^{n} f_i L_i .$$

For practical purposes, the Lagrange representation (7.2) is computationally too costly; however, in many theoretical questions it is advantageous. An example of this is the determination of the condition number of the interpolation problem.

Theorem 7.3 *Let $a \le t_0 < \cdots < t_n \le b$ be pairwise distinct nodes, and let L_{in} be the corresponding Lagrange polynomials. Then the absolute condition number κ_{abs} of the polynomial interpolation*

$$\phi = P(\cdot \mid t_0, \ldots, t_n) : C[a,b] \longrightarrow \mathbf{P}_n$$

with respect to the supremum-norm is the Lebesgue constant

$$\kappa_{\text{abs}} = \Lambda_n := \max_{t \in [a,b]} \sum_{i=0}^{n} |L_{in}(t)|$$

for the nodes $t_0, \ldots, t_n$.

Proof. The polynomial interpolation is linear, i.e., $\phi'(f)(g) = \phi(g)$. We have to show that $\|\phi'\| = \Lambda_n$. For every continuous function $f \in C[a,b]$, we have

$$\begin{aligned} |\phi(f)(t)| &= |\sum_{i=0}^{n} f(t_i) L_{in}(t)| \le \sum_{i=0}^{n} |f(t_i)|\, |L_{in}(t)| \\ &= \|f\|_\infty \max_{t \in [a,b]} \sum_{i=0}^{n} |L_{in}(t)|\,, \end{aligned}$$

and thus $\kappa_{\text{abs}} \le \Lambda_n$. For the opposite direction, we construct a function $g \in C[a,b]$ such that

$$|\phi(g)(\tau)| = \|g\|_\infty \max_{t \in [a,b]} \sum_{i=0}^{n} |L_{in}(t)|$$

for a $\tau \in [a,b]$. For this let $\tau \in [a,b]$ be the place where the maximum is attained, i.e.,

$$\sum_{i=0}^{n} |L_{in}(\tau)| = \max_{t \in [a,b]} \sum_{i=0}^{n} |L_{in}(t)|\,,$$

and let $g \in C[a,b]$ be a function with $\|g\|_\infty = 1$ and $g(t_i) = \operatorname{sgn} L_i(\tau)$, e.g., the piecewise linear interpolation function corresponding to the points $(t_i, \operatorname{sgn} L_i(\tau))$. Then, as desired

$$|\phi(g)(\tau)| = \sum_{i=0}^{n} |L_{in}(\tau)| = \|g\|_\infty \max_{t \in [a,b]} \sum_{i=0}^{n} |L_{in}(t)|\,,$$

and thus $\kappa_{\text{abs}} \ge \Lambda_n$, and altogether $\kappa_{\text{abs}} = \Lambda_n$. □

One easily computes that the Lebesgue constant Λ_n is invariant under affine transformations (see Exercise 7.1), and therefore depends only on the *relative position* of the nodes t_i with respect to each other. In Table 7.1, Λ_n is given for equidistant nodes in dependence of n. Obviously, Λ_n grows rapidly beyond all reasonable bounds. However, this is not true for

any choice of nodes. For comparison, Table 7.1 also shows the Lebesgue constants for the *Chebyshev nodes* (see Section 7.1.4)

$$t_i = \cos\left(\frac{2i+1}{2n+2}\pi\right) \quad \text{for} \quad i = 0, \ldots, n$$

(where the maximum was taken over $[-1, 1]$). They grow only very slowly.

Table 7.1. Lebesgue constant Λ_n for equidistant and for Chebyshev nodes.

n	Λ_n for equidistant nodes	Λ_n for Chebyshev nodes
5	3.106292	2.104398
10	29.890695	2.489430
15	512.052451	2.727778
20	10986.533993	2.900825

7.1.2 Hermite Interpolation and Divided Differences

If one is only interested in the interpolating polynomial P at a single position t, then the recursive computation of $P(t)$ turns out to be the most effective method. It is based on the following simple observation, the *Aitken lemma.*

Lemma 7.4 *The interpolating polynomial* $P = P(f \mid t_0, \ldots, t_n)$ *satisfies the recurrence relation*

$$P(f \mid t_0, \ldots, t_n) = \frac{(t_0 - t)P(f \mid t_1, \ldots, t_n) - (t_n - t)P(f \mid t_0, \ldots, t_{n-1})}{t_0 - t_n}. \tag{7.3}$$

Proof. Let $\varphi(t)$ be defined as the expression on the right-hand side of (7.3). Then $\varphi \in \mathbf{P}_n$, and

$$\varphi(t_i) = \frac{(t_0 - t_i)f_i - (t_n - t_i)f_i}{t_0 - t_n} = f_i \quad \text{for} \quad i = 1, \ldots, n-1.$$

Similarly, it is simple to conclude that $\varphi(t_0) = f_0$ and $\varphi(t_n) = f_n$, and the statement therefore follows. □

The interpolation polynomials for only one single node are nothing else than the constants

$$P(f \mid t_i) = f_i \quad \text{for} \quad i = 0, \ldots, n.$$

If we simplify the notation for fixed t by

$$P_{ik} := P(f \mid t_{i-k}, \ldots, t_i)(t) \quad \text{for} \quad i \geq k,$$

then the value $P_{nn} = P(f \mid t_0, \ldots, t_n)(t)$ can be computed according to the *Neville scheme*

$$\begin{array}{ccccccc} P_{00} & & & & & & \\ & \searrow & & & & & \\ P_{10} & \to & P_{11} & & & & \\ \vdots & & & \ddots & & & \\ P_{n-1,0} & \to & \cdots & \to & P_{n-1,n-1} & & \\ & & & & & \searrow & \\ P_{n0} & \to & \cdots & \to & P_{n,n-1} & \to & P_{nn} \end{array}$$

given by

$$\begin{aligned} P_{i0} &= f_i \text{ for } i = 0, \ldots, n \\ P_{ik} &= P_{i,k-1} + \frac{t - t_i}{t_i - t_{i-k}}(P_{i,k-1} - P_{i-1,k-1}) \text{ for } i \geq k. \end{aligned} \tag{7.4}$$

Example 7.5 Computation of $\sin\ 62^o$ from the nodes

$$(50^o, \sin\ 50^o), (55^o, \sin\ 55^o), \ldots, (70^o, \sin\ 70^o)$$

by the Aitken-Neville algorithm.

t_i	$\sin\ t_i$				
50^o	0.7660444				
55^o	0.8191520	0.8935027			
60^o	0.8660254	0.8847748	0.8830292		
65^o	0.9063078	0.8821384	0.8829293	0.8829493	
70^o	0.9396926	0.8862768	0.8829661	0.8829465	0.8829476

The recursive structure of the interpolation polynomials according to Lemma 7.4 can also be utilized for the determination of the entire polynomial $P(f \mid t_0, \ldots, t_n)$. This is also true for the generalized interpolation problem, where besides function values $f(t_i)$, also derivatives at the nodes are given, the *Hermite interpolation*. For this we introduce the following practical notation. With

$$a = t_0 \leq t_1 \leq \cdots \leq t_n = b$$

we allow for the occurrence of multiple nodes in $\Delta := \{t_i\}_{i=0,\ldots,n}$. If at a node t_i, the value $f(t_i)$ and the derivatives $f'(t_i), \ldots, f^{(k)}(t_i)$ are given up to an order k, then t_i shall occur $(k+1)$-times in the sequence Δ. The same nodes are enumerated from the left to the right by

$$d_i := \max\{j \mid t_i = t_{i-j}\},$$

e.g.,

t_i	t_0	$=$	t_1	$<$	t_2	$=$	t_3	$=$	t_4	$<$	t_5	$<$	t_6	$=$	t_7
d_i	0		1		0		1		2		0		0		1

By defining now for $i = 0, \ldots, n$ the linear mappings

$$\mu_i : C^n[a,b] \to \mathbf{R}\,, \quad \mu_i(f) := f^{(d_i)}(t_i)\,,$$

then the problem of the *Hermite interpolation* can be phrased as follows: Find a polynomial $P \in \mathbf{P}_n$ such that

$$\mu_i(P) = \mu_i(f) \quad \text{for all} \quad i = 0, \ldots, n\,. \tag{7.5}$$

The solution $P = P(f \mid t_0, \ldots, t_n) \in \mathbf{P}_n$ of the interpolation problem (7.5) is called *Hermite interpolation* of f at the nodes $t_0, \ldots, t_n$. Existence and uniqueness follows as in Theorem 7.1.

Theorem 7.6 *For each function $f \in C^n[a,b]$ and each monotone sequence*

$$a = t_0 \le t_1 \le \cdots \le t_n = b$$

of (not necessarily distinct) nodes, there exists a unique polynomial $P \in \mathbf{P}_n$ such that

$$\mu_i P = \mu_i f \quad \textit{for all} \quad i = 0, \ldots, n\,.$$

Proof. The mapping

$$\mu : \mathbf{P}_n \to \mathbf{R}^{n+1}\,, \quad P \mapsto (\mu_0 P, \ldots, \mu_n P)$$

is obviously linear and also injective. Now $\mu(P) = 0$ implies that P possesses at least $n+1$ roots (counted with multiplicity), and it is therefore the null-polynomial. Since $\dim \mathbf{P}_n = \dim \mathbf{R}^{n+1} = n+1$, this implies again the existence. □

If all nodes are pairwise distinct, then we recover the Lagrange interpolation

$$P(f \mid t_0, \ldots, t_n) = \sum_{i=0}^{n} f(t_i) L_i\,.$$

If all nodes coincide, $t_0 = t_1 = \cdots = t_n$, then the interpolation polynomial is the Taylor polynomial centered at $t = t_0$,

$$P(f \mid t_0, \ldots, t_n)(t) = \sum_{j=0}^{n} \frac{(t-t_0)^j}{j!} f^{(j)}(t_0)\,, \tag{7.6}$$

also called the *Taylor interpolation*.

Remark 7.7 An important application is the *cubic Hermite interpolation*, where function values f_0, f_1 and derivatives f_0', f_1' are given at two nodes t_0, t_1. According to Theorem 7.6, this determines uniquely a cubic polynomial $P \in \mathbf{P}_3$. If the *Hermite polynomials* $H_0^3, \ldots, H_3^3 \in \mathbf{P}_3$ are defined by

$$H_0^3(t_0) = 1, \quad \frac{d}{dt} H_0^3(t_0) = 0, \quad H_0^3(t_1) = 0, \quad \frac{d}{dt} H_0^3(t_1) = 0\,,$$

$$H_1^3(t_0) = 0, \quad \frac{d}{dt}H_1^3(t_0) = 1, \quad H_1^3(t_1) = 0, \quad \frac{d}{dt}H_1^3(t_1) = 0\,,$$
$$H_2^3(t_0) = 0, \quad \frac{d}{dt}H_2^3(t_0) = 0, \quad H_2^3(t_1) = 1, \quad \frac{d}{dt}H_2^3(t_1) = 0\,,$$
$$H_3^3(t_0) = 0, \quad \frac{d}{dt}H_3^3(t_0) = 0, \quad H_3^3(t_1) = 0, \quad \frac{d}{dt}H_3^3(t_1) = 1\,,$$

then the polynomials

$$\{H_0^3(t), H_1^3(t), H_2^3(t), H_3^3(t)\}$$

form a basis of $\mathbf{P}_3$, the *cubic Hermite basis*, with respect to the nodes t_0, t_1. The Hermite polynomial corresponding to the values $\{f_0, f_0', f_1, f_1'\}$ is thus formally given by

$$P(t) = f_0 H_0^3(t) + f_0' H_1^3(t) + f_1 H_2^3(t) + f_1' H_3^3(t)\,.$$

If an entire series $t_0, \ldots, t_n$ of nodes is given, with function values f_i and derivatives f_i', then on each interval $[t_i, t_{i+1}]$, we can consider the cubic Hermite interpolation and join these polynomials at the nodes. Because of the data, it is clear that this piecewise defined function is C^1. This kind of interpolation is called *locally cubic Hermite interpolation*. We have already seen an application in Section 4.4.2. When computing a solution curve by a tangent continuation method we obtain solution points (x_i, λ_i) together with slopes x_i'. In order to get an impression of the entire solution curve from this discrete information, we had connected the points by their locally cubic Hermite interpolation.

Similar to the Aitken lemma, the following recurrence relation is valid for two distinct nodes $t_i \neq t_j$.

Lemma 7.8 *If $t_i \neq t_j$, then the Hermite interpolation polynomial $P = P(f \mid t_0, \ldots, t_n)$ satisfies*

$$P = \frac{(t_i - t)P(f \mid t_1, \ldots, \widehat{t_j}, \ldots, t_n) - (t_j - t)P(f \mid t_1, \ldots, \widehat{t_i}, \ldots, t_n)}{t_i - t_j}\,,$$

where $\widehat{\ }$ indicates that the corresponding node is omitted. ("has to lift its hat").

Proof. Verification of the interpolation property by inserting the definitions. □

For the representation of the interpolation polynomial, we use the *Newton basis* $\omega_0, \ldots, \omega_n$ of the space of polynomials $\mathbf{P}_n$:

$$\omega_i(t) := \prod_{j=0}^{i-1}(t - t_i)\ , \quad \omega_i \in \mathbf{P}_i\,.$$

The coefficients with respect to this basis are the *divided differences*, which we now define.

Definition 7.9 The leading coefficient a_n of the interpolation polynomial

$$P(f \mid t_0, \ldots, t_n)(t) = a_n t^n + a_{n-1} t^{n-1} + \cdots + a_0$$

of f corresponding to the (not necessarily distinct) nodes $t_0 \le t_1 \le \cdots \le t_n$ is called n^{th} *divided difference* of f at $t_0, \ldots, t_n$, and it is denoted by

$$[t_0, \ldots, t_n] f := a_n \ .$$

Theorem 7.10 *For each function $f \in C^n$ and for given (not necessarily distinct) nodes $t_0 \le \cdots \le t_n$, the interpolation polynomial $P(f \mid t_0, \ldots, t_n)$ of f at $t_0, \ldots, t_n$ is given by*

$$P := \sum_{i=0}^{n} [t_0, \ldots, t_i] f \cdot \omega_i \ .$$

If $f \in C^{n+1}$, then

$$f(t) = P(t) + [t_0, \ldots, t_n, t] f \cdot \omega_{n+1}(t) \,. \tag{7.7}$$

Proof. We show the first statement by induction over n. The statement is trivial for $n = 0$. Thus let $n > 0$, and let

$$P_{n-1} := P(f \mid t_0, \ldots, t_{n-1}) = \sum_{i=0}^{n-1} [t_0, \ldots, t_i] f \cdot \omega_i$$

be the interpolation polynomial of f at $t_0, \ldots, t_{n-1}$. Then the interpolation polynomial $P_n = P(f \mid t_0, \ldots, t_n)$ of f at $t_0, \ldots, t_n$ can be written in the form

$$\begin{aligned} P_n(t) &= [t_0, \ldots, t_n] f \cdot t^n + a_{n-1} t^{n-1} + \cdots + a_0 \\ &= [t_0, \ldots, t_n] f \cdot \omega_n(t) + Q_{n-1}(t) \,, \end{aligned}$$

with a polynomial $Q_{n-1} \in \mathbf{P}_{n-1}$. But

$$Q_{n-1} = P_n - [t_0, \ldots, t_n] f \cdot \omega_n$$

obviously satisfies the interpolation conditions for $t_0, \ldots, t_{n-1}$, so that

$$Q_{n-1} = P_{n-1} = \sum_{i=0}^{n-1} [t_0, \ldots, t_i] f \cdot \omega_i \,.$$

This proves the first statement. In particular, it follows that

$$P_n + [t_0, \ldots, t_n, t] f \cdot \omega_{n+1}$$

interpolates the function f at the nodes $t_0, \ldots, t_n$ and t, which proves (7.7). □

From the properties of the Hermite interpolation one can immediately deduce the following statements about the divided differences of f.

Lemma 7.11 *The divided differences $[t_0, \ldots, t_n]f$ satisfy the following properties ($f \in C^n$):*

(i) $[t_0, \ldots, t_n]P = 0$ *for all* $P \in \mathbf{P}_{n-1}$.

(ii) *For multiple nodes* $t_0 = \cdots = t_n$,

$$[t_0, \ldots, t_n]f = f^{(n)}(t_0)/n! \,. \tag{7.8}$$

(iii) *The following recurrence relation holds for* $t_i \neq t_j$:

$$[t_0, \ldots, t_n]f = \frac{[t_0, \ldots, \widehat{t_i}, \ldots, t_n]f - [t_0, \ldots, \widehat{t_j}, \ldots, t_n]f}{t_j - t_i} \,. \tag{7.9}$$

Proof. (i) is true, because the n^{th} coefficient of a polynomial of degree less than or equal to $n-1$ vanishes. (ii) follows from the Taylor interpolation (7.6) and (iii) from Lemma 7.8 and the uniqueness of the leading coefficient. □

With properties (ii) and (iii), the divided differences can be computed recursively from the function values and derivatives $f^{(j)}(t_i)$ of f at the nodes t_i. We also need the recurrence relation in the proof of the following theorem, which states a surprising interpretation of the divided differences: The n^{th} divided difference of a function $f \in C^n$ with respect to the nodes $t_0, \ldots, t_n$ is the integral of the n^{th} derivative over the n-dimensional standard simplex

$$\Sigma^n := \left\{ s = (s_0, \ldots, s_n) \in \mathbf{R}^{n+1} \;\middle|\; \sum_{i=0}^{n} s_i = 1 \;\text{ and }\; s_i \geq 0 \right\} .$$

Theorem 7.12 *Hermite-Genocchi formula. The n^{th} divided difference of a n-times continuously differentiable function $f \in C^n$ satisfies*

$$[t_0, \ldots, t_n]f = \int_{\Sigma^n} f^{(n)}\Big(\sum_{i=0}^{n} s_i t_i\Big)\, ds \,. \tag{7.10}$$

Proof. We prove the formula by induction over n. The statement is trivial for $n = 0$. The induction step from n to $n+1$ is as follows: If all nodes coincide, then the statement follows from (7.8). We can therefore assume without loss of generality that $t_0 \neq t_{n+1}$. We then have

$$
\begin{aligned}
&\int_{\sum_{i=0}^{n+1} s_i=1} f^{(n+1)}\Big(\sum_{i=0}^{n} s_i t_i\Big)\, ds \\
&\quad = \int_{\sum_{i=1}^{n+1} s_i \le 1} f^{(n+1)}\Big(t_0 + \sum_{i=1}^{n} s_i(t_i - t_0)\Big)\, ds \\
&\quad = \int_{\sum_{i=1}^{n} s_i \le 1} \int_{s_0=0}^{1-\sum_{i=1}^{n} s_i} f^{(n+1)}\Big(t_0 + \sum_{i=1}^{n} s_i(t_i - t_0) + s_{n+1}(t_{n+1} - t_0)\Big)\, ds \\
&\quad = \frac{1}{t_{n+1} - t_0} \int_{\sum_{i=1}^{n} s_i \le 1} \Big\{ f^{(n)}\Big(t_{n+1} + \sum_{i=1}^{n} s_i(t_i - t_{n+1})\Big) - \\
&\qquad\qquad f^{(n)}\Big(t_0 + \sum_{i=1}^{n} s_i(t_i - t_0)\Big) \Big\}\, ds \\
&\quad = \frac{1}{t_{n+1} - t_0}\big([t_1, \ldots, t_{n+1}]f - [t_0, \ldots, t_n]f\big) \\
&\quad = [t_0, \ldots, t_{n+1}]f
\end{aligned}
$$

□

Corollary 7.13 *Let $g : \mathbf{R}^{n+1} \to \mathbf{R}$ be the mapping, which is given by the n^{th} divided difference of a function $f \in C^n$ with*

$$g(t_0, \ldots, t_n) := [t_0, \ldots, t_n]f\,.$$

Then g is continuous in its arguments t_i. Furthermore, for all nodes $t_0 \le \cdots \le t_n$, there exists a $\tau \in [t_0, t_n]$ such that

$$[t_0, \ldots, t_n]f = \frac{f^{(n)}(\tau)}{n!}\,. \tag{7.11}$$

Proof. The integral representation (7.10) yields immediately the continuity; and (7.11) follows from the integral mean-value theorem, because the volume of the n-dimensional standard simplex is $\mathrm{vol}(\Sigma^n) = 1/n!$. □

For pairwise distinct nodes $t_0 < \cdots < t_n$, the divided differences can be arranged similar to the Neville-scheme because of the recurrence relation (7.9).

$$
\begin{array}{lcccccccc}
f_0 & = & [t_0]f & & & & & & \\
 & & & \searrow & & & & & \\
f_1 & = & [t_1]f & \to & [t_0,t_1]f & & & & \\
\vdots & & & & & \ddots & & & \\
f_{n-1} & = & [t_{n-1}]f & \to & \cdots & \to & [t_0,\ldots,t_{n-1}]f & & \\
 & & & & & & & \searrow & \\
f_n & = & [t_n]f & \to & \cdots & \to & [t_1,\ldots,t_n]f & \to & [t_0,\ldots,t_n]f
\end{array}
$$

Example 7.14 We compute the interpolation polynomial corresponding to the values

t_i	0	1	2	3
f_i	1	2	0	1

by employing Newton's divided differences,

$$
\begin{array}{llll}
f[t_0]=1 & & & \\
f[t_1]=2 & f[t_0,t_1]=1 & & \\
f[t_2]=0 & f[t_1,t_2]=-2 & f[t_0,t_1,t_2]=-3/2 & \\
f[t_3]=1 & f[t_2,t_3]=1 & f[t_1,t_2,t_3]=3/2 & f[t_0,t_1,t_2,t_3]=1\,,
\end{array}
$$

i.e.,

$$(\alpha_0,\alpha_1,\alpha_2,\alpha_3)=(1,1,-3/2,1)\,.$$

The interpolation polynomial is therefore

$$
\begin{aligned}
P(t) &= 1+1(t-0)+(-3/2)(t-0)(t-1)+1(t-0)(t-1)(t-2)\\
&= t^3-4.5\,t^2+4.5\,t+1\,.
\end{aligned}
$$

A further important property of the divided differences is the following *Leibniz formula.*

Lemma 7.15 *Let $g,h\in C^n$, and let $t_0\le t_1\le\cdots\le t_n$ be an arbitrary sequence of nodes. Then*

$$[t_0,\ldots,t_n]gh=\sum_{i=0}^{n}[t_0,\ldots,t_i]g\cdot[t_i,\ldots,t_n]h\,.$$

Proof. First suppose that the nodes $t_0,\ldots,t_n$ are pairwise distinct. Set

$$\omega_i(t):=\prod_{k=0}^{i-1}(t-t_k)\quad\text{and}\quad\bar\omega_j(t):=\prod_{l=j+1}^{n}(t-t_l)\,.$$

Then, according to Theorem 7.10, the interpolation polynomials $P,Q\in\mathbf{P}_n$ of g, respectively, h are given by

$$P=\sum_{i=0}^{n}[t_0,\ldots,t_i]g\cdot\omega_i\quad\text{and}\quad Q=\sum_{j=0}^{n}[t_j,\ldots,t_n]h\cdot\bar\omega_j\ .$$

The product

$$PQ = \sum_{i,j=0}^{n} [t_0, \ldots, t_i]g \; [t_j, \ldots, t_n]h \cdot \omega_i \bar{\omega}_j$$

thus interpolates the function $f := gh$ in $t_0, \ldots, t_n$. Since $\omega_i(t_k)\bar{\omega}_j(t_k) = 0$ for all k and $i > j$, it follows that

$$F := \sum_{\substack{i,j=0 \\ i \le j}}^{n} [t_0, \ldots, t_i]g \; [t_j, \ldots, t_n]h \cdot \omega_i \bar{\omega}_j \in \mathbf{P}_n$$

is the interpolation polynomial of gh in $t_0, \ldots, t_n$. As claimed, the leading coefficient is

$$\sum_{i=0}^{n} [t_0, \ldots, t_i]g \; [t_i, \ldots, t_n]h \, .$$

For arbitrary, not necessarily distinct nodes t_i, the statement now follows from the continuity of the divided differences at the nodes t_i. □

7.1.3 *Approximation Error*

We shall now turn our attention toward the second requirement, namely, the *approximation property*, and analyze to what extent the polynomials $P(f \mid t_0, \ldots, t_n)$ approximate the original function f. By employing previously derived properties of the divided differences, it is simple to find a representation of the approximation error.

Theorem 7.16 *Suppose that $f \in C^{n+1}$. Then for the approximation error of the Hermite interpolation $P(f \mid t_0, \ldots, t_n)$ and $t_i, t \in [a, b]$ we have*

$$f(t) - P(f \mid t_0, \ldots, t_n)(t) = \frac{f^{(n+1)}(\tau)}{(n+1)!} \omega_{n+1}(t) \tag{7.12}$$

for some $\tau = \tau(t) \in]a, b[$.

Proof. According to theorem 7.10 and theorem 7.13, we have for $P := P(f \mid t_0, \ldots, t_n)$ that

$$f(t) - P(t) = [t_0, \ldots, t_n, t]f \cdot \omega_{n+1}(t) = \frac{f^{(n+1)}(\tau)}{(n+1)!} \omega_{n+1}(t)$$

for some $\tau \in]a, b[$. □

Example 7.17 In the case of the Taylor interpolation (7.6), i.e., $t_0 = \cdots = t_n$, the error formula (7.12) is just the Lagrange remainder of the Taylor expansion

$$f(t) - P(f \mid t_0, \ldots, t_n)(t) = \frac{f^{(n+1)}(\tau)}{(n+1)!} (t - t_0)^{n+1} \, .$$

If we consider the class of functions

$$\mathcal{F} := \left\{ f \in C^{n+1}[a,b] \;\middle|\; \sup_{\tau \in [a,b]} |f^{n+1}(\tau)| \leq M(n+1)! \right\}$$

for a constant $M > 0$, then the approximation error obviously depends crucially on the choice of the nodes $t_0, \ldots, t_n$ via the expression

$$\omega_{n+1}(t) = (t - t_0) \cdots (t - t_n). \tag{7.13}$$

In the next section (see Example 7.20) we will show that, in the case of pairwise distinct nodes, the expression (7.13) can be minimized on an interval $[a,b]$,

$$\max_{t \in [a,b]} |\omega_{n+1}(t)| = \min,$$

by choosing again the Chebyshev nodes for the nodes t_i. We now turn our attention to the question of whether the polynomial interpolation satisfies the approximation property. For merely continuous functions $f \in C[a,b]$, and the supremum-norm $\|f\| = \sup_{t \in [a,b]} f(t)$, the approximation error can in principle grow beyond all bounds. More precisely, according to Faber, for each sequence $\{T_k\}$ of sets of nodes $T_k = \{t_{k,0}, \ldots, t_{k,n_k}\} \subset [a,b]$, there exists a continuous function $f \in C[a,b]$ such that the sequence $\{P_k\}$ of the interpolation polynomials, which belong to the T_k, does not converge uniformly to f.

7.1.4 Min-Max Property of Chebyshev Polynomials

In the previous chapter we have repeatedly mentioned the Chebyshev nodes, for which the polynomial interpolation has particularly nice properties (such as bounded condition and optimal approximation of f over the class $\mathcal{F}$). The Chebyshev nodes are the roots of the Chebyshev polynomials, which we already encountered in Chapter 6.1.1. In the investigation of the approximation error, as well as in the condition analysis of the polynomial interpolation, we were lead to the following approximation problem: Find the polynomial $P_n \in \mathbf{P}_n$ of degree $\deg P_n = n$ with leading coefficient 1, and the smallest supremum-norm over the interval $[a,b]$, i.e.,

$$\max_{t \in [a,b]} |P_n(t)| = \min. \tag{7.14}$$

We shall now see that the *Chebyshev polynomials* T_n, which we already encountered as orthogonal polynomials with respect to the weight function $\omega(x) = (1 - x^2)^{-\frac{1}{2}}$ over $[-1,1]$, solve this *min-max problem* (up to a scalar factor and an affine transformation). In order to show this, we first reduce the problem to the interval $[-1,1]$, which is suitable for the Chebyshev polynomials, with the help of the affine mapping

$$\begin{aligned} x : [a,b] &\xrightarrow{\cong} [-1,1] \\ t &\longmapsto x = x(t) = 2\,\frac{t-a}{b-a} - 1 = \frac{2\,t-a-b}{b-a}, \end{aligned}$$

whose inverse mapping is $t : [-1,1] \xrightarrow{\cong} [a,b]$,

$$t = t(x) = \frac{1-x}{2}a + \frac{1+x}{2}b\,.$$

If $P_n \in \mathbf{P}_n$ with $\deg P = n$ and leading coefficient 1 is the solution of the min-max problem

$$\max_{x\in[-1,1]} |P_n(x)| = \min\,,$$

then $\hat{P}_n(t) := P_n(t(x))$ is the solution of the original problem (7.14) with leading coefficient $2^n/(b-a)^n$.

In Example 6.3, we introduced the Chebyshev polynomials (see Figure 7.3) via

$$T_n(x) = \cos(n \arccos x) \quad \text{for} \quad x \in [-1,1]$$

and more generally for $x \in \mathbf{R}$ via the three-term recurrence relation

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)\,,\; T_0(x) = 1\,,\; T_1(x) = x\,.$$

The following properties of the Chebyshev polynomials are either obvious

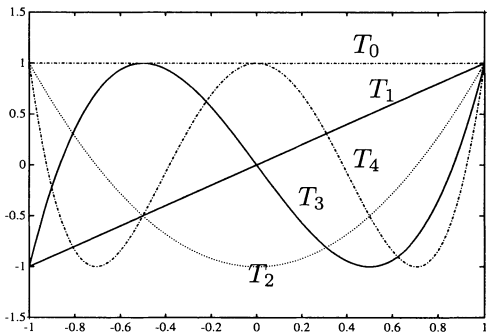

Figure 7.3. Chebyshev polynomials $T_0, \ldots, T_4$.

or easily verified. In particular we can directly give the roots $x_1, \ldots, x_n$ of $T_n(x)$, which are real and simple according to theorem 6.5 (see property 7 below).

Remark 7.18

1. The Chebyshev polynomials have integer coefficients.
2. The leading coefficient of T_n is $a_n = 2^{n-1}$.
3. T_n is an even function if n is even, and an odd one if n is odd.
4. $T_n(1) = 1, \quad T_n(-1) = (-1)^n$.
5. $|T_n(x)| \leq 1$ for $x \in [-1, 1]$.
6. $|T_n(x)|$ takes on the value 1 at the *Chebyshev abscissas* $\bar{x}_k = \cos(k\pi/n)$, i.e.,

$$|T_n(x)| = 1 \iff x = \bar{x}_k = \cos\frac{k\pi}{n} \quad \text{for a } k = 0, \ldots, n.$$

7. The roots of $T_n(x)$ are

$$x_k := \cos\left(\frac{2k-1}{2n}\pi\right) \quad \text{for} \quad k = 1, \ldots, n\,.$$

8. We have

$$T_k(x) = \begin{cases} \cos(k \arccos x) & , \quad \text{if} \quad -1 \leq x \leq 1 \\ \cosh(k\text{arccosh}x) & , \quad \text{if} \quad x \geq 1 \\ (-1)^k \cosh(k\text{arccosh}(-x)) & , \quad \text{if} \quad x \leq -1 \end{cases}.$$

9. The Chebyshev polynomials have the global representation

$$T_k(x) = \frac{1}{2}((x + \sqrt{x^2-1})^k + (x - \sqrt{x^2-1})^k) \quad \text{for} \quad x \in \mathbf{R}\,.$$

Properties 8 and 9 are most easily checked by verifying that they satisfy the three-term recurrence (including the starting values). The min-max property of the Chebyshev polynomials follows from the intermediate-value theorem:

Theorem 7.19 *Every polynomial $P_n \in \mathbf{P}_n$ with leading coefficient $a_n \neq 0$ attains a value of absolute value $\geq |a_n|/2^{n-1}$ in the interval $[-1, 1]$. In particular the Chebyshev polynomials $T_n(x)$ are minimal with respect to the maximum-norm $\|f\|_\infty = \max_{x\in[-1,1]} |f(x)|$ among the polynomials of degree n with leading coefficient 2^{n-1}.*

Proof. Let $P_n \in \mathbf{P}_n$ be a polynomial with leading coefficient $a_n = 2^{n-1}$ and $|P_n(x)| < 1$ for $x \in [-1, 1]$. Then $T_n - P_n$ is a polynomial of degree less then or equal to $n-1$. At the Chebyshev abscissas $\bar{x}_k := \cos\frac{k\pi}{n}$ we have

$$\begin{array}{llll} T_n(\bar{x}_{2k}) = 1, & P_n(\bar{x}_{2k}) < 1 & \Longrightarrow & P_n(\bar{x}_{2k}) - T_n(\bar{x}_{2k}) < 0 \\ T_n(\bar{x}_{2k+1}) = -1, & P_n(\bar{x}_{2k+1}) > -1 & \Longrightarrow & P_n(\bar{x}_{2k+1}) - T_n(\bar{x}_{2k+1}) > 0\,; \end{array}$$

i.e., at the $n+1$ Chebyshev abscissae, the difference $T_n - P_n$ is alternating between positive and negative and has therefore at least n roots in $[-1, 1]$,

which contradicts $0 \neq T_n - P_n \in \mathbf{P}_{n-1}$. As a consequence, for each polynomial $P_n \in \mathbf{P}_n$ with leading coefficient $a_n = 2^{n-1}$ there must exist an $x \in [-1,1]$ such that $|P_n(x)| \geq 1$. For an arbitrary polynomial $P_n \in \mathbf{P}_n$ with leading coefficient $a_n \neq 0$, the statement follows from the fact that $\tilde{P}_n := \frac{2^{n-1}}{a_n} P_n$ is a polynomial with leading coefficient $\tilde{a}_n = 2^{n-1}$. □

Example 7.20 When minimizing the approximation error of the polynomial interpolation, we try to find the nodes $t_0, \ldots, t_n \in [a,b]$, which solve the min-max problem

$$\max_{t\in[a,b]} |\omega(t)| = \max_{t\in[a,b]} |(t-t_0)\cdots(t-t_n)| = \min \ .$$

To put it differently, the goal is to determine the normalized polynomial $\omega(t) \in \mathbf{P}_{n+1}$ with the real roots $t_0, \ldots, t_n$, for which $\max_{t\in[a,b]} |\omega(t)| = \min$. According to theorem 7.19, for the interval $[a,b] = [-1,1]$, this is just the $(n+1)^{\text{st}}$ Chebyshev polynomial $\omega(t) = T_{n+1}(t)$, whose roots

$$t_i = \cos\left(\frac{2i+1}{2n+2}\pi\right) \quad \text{for} \quad i = 0, \ldots, n$$

are just the *Chebyshev nodes.*

We shall now derive a second property of the Chebyshev polynomials, which we will be useful in Chapter 8.

Theorem 7.21 *Let $[a,b]$ be an arbitrary interval, and let $t_0 \notin [a,b]$. Then the modified Chebyshev polynomial*

$$\hat{T}_n(t) := \frac{T_n(x(t))}{T_n(x(t_0))} \quad \textit{with} \quad x(t) := 2\frac{t-a}{b-a} - 1$$

is minimal with respect to the maximum-norm $\|f\|_\infty = \max_{t\in[a,b]} |f(t)|$ among the polynomials $P_n \in \mathbf{P}_n$ with $P_n(t_0) = 1$.

Proof. Since all roots of $T_n(x(t))$ lie in $[a,b]$, we have $c := T_n(x(t_0)) \neq 0$ and $\hat{T}_n$ is well defined. Furthermore $\hat{T}_n(t_0) = 1$, and $|\hat{T}_n(t)| \leq |c|^{-1}$ for all $t \in [a,b]$. Suppose now that there is a polynomial $P_n \in \mathbf{P}_n$ such that $P_n(t_0) = 1$, and $|P_n(t)| < |c|^{-1}$ for all $t \in [a,b]$, then t_0 is a root of the difference $\hat{T}_n - P_n$, i.e.,

$$\hat{T}_n(t) - P_n(t) = Q_{n-1}(t)(t-t_0) \ \text{ for a polynomial } Q_{n-1} \in \mathbf{P}_{n-1}.$$

As in the proof of theorem 7.19, Q_{n-1} changes sign at the Chebyshev abscissae $t_k = t(\bar{x}_k)$ for $k = 0, \ldots, n$ and has therefore at least n distinct roots in $[a,b]$. This contradicts $0 \neq Q_{n-1} \in \mathbf{P}_{n-1}$. □

7.2 Trigonometric Interpolation

In this section we shall interpolate periodic functions by trigonometric polynomials, i.e., linear combinations of trigonometric functions. Next to the polynomials, this class is one of the most important as far as interpolation is concerned, not least because there is an extremely effective algorithm for the solution of the interpolation problem, the *fast Fourier transform*.

In Example 6.20, we have already encountered the algorithm of Goertzel and Reinsch for the evaluation of a trigonometric polynomial

$$f_n(t) = \frac{a_0}{2} + \sum_{j=1}^{n}(a_j \cos jt + b_j \sin jt) = \sum_{j=-n}^{n} c_j e^{ijt} \ .$$

First, we define the N-dimensional spaces of trigonometric polynomials, real as well as complex. In the real case it is necessary to distinguish between the cases of even and odd N. The following considerations will therefore be carried out mostly in the complex case only.

Definition 7.22 By $\mathbf{T}_C^N$ we denote the N-dimensional space of the complex trigonometric polynomials

$$\phi_N(t) = \sum_{j=0}^{N-1} c_j e^{ijt} \quad \text{with} \quad c_j \in \mathbf{C}$$

of degree $N-1$. The N-dimensional spaces $\mathbf{T}_R^N$ contain all real trigonometric polynomials $\phi_N(t)$ of the form

$$\phi_{2n+1}(t) = \frac{a_0}{2} + \sum_{j=1}^{n}(a_j \cos jt + b_j \sin jt), \tag{7.15}$$

for odd $N = 2n+1$, respectively,

$$\phi_{2n}(t) = \frac{a_0}{2} + \sum_{j=1}^{n-1}(a_j \cos jt + b_j \sin jt) + \frac{a_n}{2} \cos nt, \tag{7.16}$$

for even $N = 2n$, where $a_j, b_j \in \mathbf{R}$.

The thus defined trigonometric polynomials are apparently 2π-periodic, i.e., $\varphi(t+2\pi) = \varphi(t)$. Hence, considering interpolation, we choose N distinct nodes

$$0 \le t_0 < t_1 < \cdots < t_{N-1} < 2\pi$$

in the half-open interval $[0, 2\pi[$. Given nodal values $f_0, \ldots, f_{N-1}$ we look for trigonometric polynomials $\phi_N \in T_C^N$, respectively, T_R^N satisfying

$$\phi_N(t_j) = f_j \quad \text{for} \quad j = 0, \ldots, N-1 \,. \tag{7.17}$$

Here, the nodes f_j are of course complex in one case and real in the other. The complex trigonometric interpolation may be easily derived from the

standard (complex) polynomial interpolation since we have the bijection

$$\begin{array}{ccc} T_C^N & \longrightarrow & \mathbf{P}_{N-1} \\ \phi(t) = \sum_{h=0}^{N-1} c_j e^{ijt} & \longmapsto & P(\omega) = \sum_{j=0}^{N-1} c_j \omega^j \,. \end{array}$$

Hence, there is a unique polynomial $\phi_N \in T_C^N$ satisfying the interpolation condition (7.17). For equidistant nodes

$$t_j = 2\pi j/N\,,\ j = 0, \ldots, N-1\,,$$

to which we shall restrict ourselves, we introduce the N^{th} unit roots $\omega_j := e^{it_j} = e^{2\pi ij/N}$. Thus for the coefficients c_j of the complex trigonometric interpolation $\phi_N \in \mathbf{T}_C^N$, we obtain the Vandermonde system:

$$\underbrace{\begin{bmatrix} 1 & \omega_0 & \omega_0^2 & \ldots & \omega_0^{N-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega_{N-1} & \omega_{N-1}^2 & \ldots & \omega_{N-1}^{N-1} \end{bmatrix}}_{=: V_{N-1}} \begin{pmatrix} c_0 \\ \vdots \\ c_{N-1} \end{pmatrix} = \begin{pmatrix} f_0 \\ \vdots \\ f_{N-1} \end{pmatrix}.$$

Using the complex solution $\phi_N \in T_C^N$ of a *real* interpolation problem, $f_j \in \mathbf{R}$, we can compute the real solution $\phi_N \in T_R^N$ by the following lemma.

Lemma 7.23 *Let*

$$\phi_N(t) = \sum_{j=0}^{N-1} c_j e^{ijt} \in T_C^N$$

be the solution of the interpolation problem (7.17) with real nodes $f_j \in \mathbf{R}$ and equidistant nodes $t_j = 2\pi j/N$. Then the real trigonometric polynomial $\psi_N \in T_R^N$, given by the coefficients

$$a_j = 2\Re c_j = c_j + c_{N-j} \quad \textit{and} \quad b_j = -2\Im c_j = i(c_j - c_{N-j})$$

from the representations (7.15), respectively, (7.16), also satisfies the interpolation conditions, i.e., $\varphi_N(t_j) = f_j$.

Proof. We evaluate the polynomial at the N equidistant nodes t_k. Since $e^{2\pi i(N-j)/N} = e^{-2\pi ij/N}$, we have

$$\phi_N(t_k) = \sum_{j=0}^{N-1} c_j e^{ijt_k} = \overline{\phi_N(t_k)} = \sum_{j=0}^{N-1} \bar{c}_j e^{-ijt_k} = \sum_{j=0}^{N-1} \bar{c}_{N-j} e^{ijt_k}\,.$$

The unique interpolation property therefore implies that $c_j = \bar{c}_{N-j}$. In particular, c_0 is real and for even $N = 2n$ also c_{2n}. For odd $N = 2n+1$, we obtain

$$\phi_N(t_k) \;=\; c_0 + \sum_{j=1}^{2n} c_j e^{ijt_k} = c_0 + \sum_{j=1}^{n} \left(c_j e^{ijt_k} + \bar{c}_j e^{-ijt_k}\right)$$

$$= c_0 + \sum_{j=1}^{n} 2\Re(c_j e^{ijt_k}) = c_0 + \sum_{j=1}^{n} (2\Re c_j \cos jt - 2\Im c_j \sin jt) .$$

Hence, the real trigonometric polynomial with the coefficients

$$a_j = 2\Re c_j = c_j + \bar{c}_j = c_j + c_{N-j}$$

and

$$b_j = -2\Im c_j = i(c_j - \bar{c}_j) = i(c_j - c_{N-j})$$

solves the interpolation problem. For even $N = 2n$, the statement follows similarly. □

Lemma 7.23 shows the existence of a real trigonometric interpolating polynomial. However, since the interpolation problem is linear and the numbers of nodes and coefficients coincide, this also implies uniqueness.

In the case at hand, the Vandermonde matrix V_{N-1} can easily be inverted analytically. In order to demonstrate this, we shall first show the orthonormality of the basis functions $\psi_j(t) := e^{ijt}$ with respect to the following scalar product, which is given by the equidistant nodes t_j,

$$\langle f, g \rangle := \frac{1}{N} \sum_{j=0}^{N-1} f(t_j)\overline{g(t_j)} . \tag{7.18}$$

Lemma 7.24 *For the N^{th} unit roots $\omega_j := e^{2\pi ij/N}$ we have*

$$\sum_{j=0}^{N-1} \omega_j^k \omega_j^{-l} = N\delta_{kl} .$$

In particular, the functions $\psi_j(t) = e^{ijt}$ are orthonormal with respect to the scalar product (7.18), i.e., $\langle \psi_k, \psi_l \rangle = \delta_{kl}$.

Proof. The statement is obviously equivalent to

$$\sum_{j=0}^{N-1} \omega_k^j = N\delta_{0k} .$$

(Observe that $\omega_j^k = \omega_k^j$.) Now the N^{th} unit roots ω_k are solutions of the equation

$$0 = \omega^N - 1 = (\omega - 1)(\omega^{N-1} + \omega^{N-2} + \cdots + 1) = (\omega - 1) \sum_{j=0}^{N-1} \omega^j .$$

If $k \neq 0$, then $\omega_k \neq 1$ and therefore $\sum_{j=0}^{N-1} \omega_k^j = 0$. In the other case we obviously have $\sum_{j=0}^{N-1} \omega_0^j = \sum_{j=0}^{N-1} 1 = N$. □

With the help of this orthogonality relation, we can easily give the solution of the interpolation problem.

Theorem 7.25 *The coefficients c_j of the trigonometric interpolation corresponding to the N nodes (t_k, f_k) with equidistant nodes $t_k = 2\pi k/N$, i.e.,*

$$\phi_N(t_k) = \sum_{j=0}^{N-1} c_j e^{ijt_k} = \sum_{j=0}^{N-1} c_j \omega_k^j = f_k \quad \text{for } k = 0, \ldots, N-1\,,$$

are given by

$$c_j = \frac{1}{N} \sum_{k=0}^{N-1} f_k \omega_k^{-j} \quad \text{for } j = 0, \ldots, N-1.$$

Proof. We insert the given solution for the coefficients c_j and obtain

$$\begin{aligned} \sum_{j=0}^{N-1} c_j \omega_l^j &= \sum_{j=0}^{N-1} \left(\frac{1}{N} \sum_{k=0}^{N-1} f_k \omega_k^{-j} \right) \omega_l^j \\ &= \frac{1}{N} \sum_{k=0}^{N-1} f_k \sum_{j=0}^{N-1} \omega_k^{-j} \omega_l^j = \frac{1}{N} \sum_{k=0}^{N-1} f_k \delta_{kl} N = f_l\,. \end{aligned}$$

□

Remark 7.26 For odd $N = 2n+1$, and letting $c_{-j} := c_{N-j}$ for $j > 0$, we can rewrite the trigonometric interpolation polynomial in symmetric form:

$$\varphi_{N-1}(t_k) = \sum_{j=0}^{N-1} c_j e^{ijt_k} = \sum_{j=-n}^{n} c_j e^{ijt_k}\,.$$

In this form, it strongly resembles the truncated Fourier series

$$f_n(t) = \sum_{j=-n}^{n} \hat{f}(j) e^{ijt}$$

of a 2π-periodic function $f \in L^2(\mathbf{R})$ with coefficients

$$\hat{f}(j) = (f, e^{ijt}) = \frac{1}{2\pi} \int_0^{2\pi} f(t) e^{-ijt}\, dt\,. \tag{7.19}$$

In fact, the coefficients c_j can be considered as the approximation of the integral in (7.19) by the trapezoidal sum (compare Section 9.2) with respect to the nodes $t_k = 2\pi k/N$. If we insert this approximation

$$\int_0^{2\pi} g(t)\, dt \;\approx\; \frac{2\pi}{N} \sum_{k=0}^{N-1} g(t_k) \tag{7.20}$$

into (7.19), then this yields

$$\hat{f}(j) \approx \frac{1}{N} \sum_{k=0}^{N-1} f_k e^{-ijt_k} = \frac{1}{N} \sum_{k=0}^{N-1} e^{-2\pi ijk/N} f_k = c_j \,. \tag{7.21}$$

Observe that the formula (7.20) is actually exact for trigonometric polynomials $g \in \mathbf{T}_C^N$, and equality in (7.21) holds therefore also for $f \in \mathbf{T}_C^N$. For this reason the isomorphism

$$\mathcal{F}_N : \mathbf{C}^N \to \mathbf{C}^N, \quad (f_j) \mapsto (c_j)$$

with

$$c_j = \frac{1}{N} \sum_{k=0}^{N-1} f_k e^{-2\pi ijk/N} \quad \text{for} \quad j = 0, \ldots, N-1$$

is called a *discrete Fourier transform*. The inverse mapping $\mathcal{F}_N^{-1}$ is

$$f_j = \sum_{k=0}^{N-1} c_k e^{2\pi ijk/N} \quad \text{for} \quad j = 0, \ldots, N-1 \,.$$

The computation of the coefficients c_j from the values f_j (or the other way around) is in principle a matrix-vector multiplication, for which we expect a cost of $O(N^2)$ operations. However, there is an algorithm, which requires only $O(N \log_2 N)$ operations, the *fast Fourier transform* (FFT). It is based on a separate analysis of the expressions for the coefficients c_j for odd, respectively, even indices j, called the *odd even reduction*. This way it is possible to transform the original problem into two similar partial problems of half dimension.

Lemma 7.27 *Let $N = 2M$ be even and $\omega = e^{\pm 2\pi i/N}$. Then the trigonometric sums*

$$\alpha_j = \sum_{k=0}^{N-1} f_k \omega^{kj} \quad \textit{for } j = 0, \ldots, N-1$$

can be computed as follows, where $\xi := \omega^2$ and $l = 0, \ldots, M-1$:

$$\begin{aligned} \alpha_{2l} &= \sum_{k=0}^{M-1} g_k \xi^{kl} \quad \textit{with} \quad g_k = f_k + f_{k+M} \,, \\ \alpha_{2l+1} &= \sum_{k=0}^{M-1} h_k \xi^{kl} \quad \textit{with} \quad h_k = (f_k - f_{k+M})\omega^k \,; \end{aligned}$$

i.e., the computation of the α_j can be reduced to two similar problems of half dimension $M = N/2$.

Proof. In the even case $j = 2l$, and because of $\omega^{Nl} = 1$, it follows that

$$\begin{aligned}\alpha_{2l} &= \sum_{k=0}^{N-1} f_k \omega^{2kl} \\ &= \sum_{k=0}^{N/2-1} \left(f_k \omega^{2kl} + f_{k+N/2}\, \omega^{2(k+N/2)l} \right) \\ &= \sum_{k=0}^{M-1} (f_k + f_{k+M})(\omega^2)^{kl} .\end{aligned}$$

Similarly, because of $\omega^{N/2} = -1$, we obtain for odd indices $j = 2l + 1$,

$$\begin{aligned}\alpha_{2l+1} &= \sum_{k=0}^{N-1} f_k \omega^{k(2l+1)} \\ &= \sum_{k=0}^{N/2-1} \left(f_k \omega^{k(2l+1)} + f_{k+N/2} \omega^{(k+N/2)(2l+1)} \right) \\ &= \sum_{k=0}^{M-1} (f_k - f_{k+M}) \omega^k (\omega^2)^{kl} .\end{aligned}$$

□

The lemma can be applied to the discrete Fourier analysis $(f_k) \mapsto (c_j)$, as well as to the synthesis $(c_j) \mapsto (f_k)$. If the number N of the given points is a power of two $N = 2^p$, $p \in \mathbf{N}$, then we can iterate the process. This algorithm due to W. Cooley and J. W. Tukey [85] is frequently called the *Cooley-Tukey algorithm.* The computation can essentially be carried out on a single vector, if the current number-pairs are overwritten. In the Algorithm 7.28, we simply overwrite the input values $f_0, \ldots, f_{N-1}$. However, here the order is interchanged in each reduction step because of the separation of even and odd indices. We have illustrated this permutation of indices in Table 7.2. We obtain the right indices by reversing the order of the bits in the dual-representation of the indices.

We therefore define a permutation σ,

$$\begin{aligned}\sigma : \{0, \ldots, N-1\} &\longrightarrow \{0, \ldots, N-1\} \\ \sum_{j=0}^{p-1} a_j 2^j &\longmapsto \sum_{j=0}^{p-1} a_{p-1-j} 2^j, \quad a_j \in \{0, 1\},\end{aligned}$$

which represents this operation, and which can be realized on a computer at little cost by a corresponding bit-manipulation.

Table 7.2. Interchange of the indices of the fast Fourier transform for $N = 8$, i.e., $p = 3$.

k	dual	1. reduction	2. reduction	dual
0	000	0	0	000
1	001	2	4	100
2	010	4	2	010
3	011	6	6	110
4	100	1	1	001
5	101	3	5	101
6	110	5	3	011
7	111	7	7	111

Algorithm 7.28 *Fast Fourier transform (FFT).* From given input values $f_0, \ldots, f_{N-1}$ for $N = 2^p$ and $\omega = e^{\pm 2\pi i/N}$ the algorithm computes the transformed values $\alpha_0, \ldots, \alpha_{N-1}$ with $\alpha_j = \sum_{k=0}^{N-1} f_k \omega^{kj}$.

$N_{\text{red}} := N;$
$z := \omega;$
while $N_{\text{red}} > 1$ **do**
  $M_{\text{red}} := N_{\text{red}}/2;$
  for $j := 0$ **to** $N/N_{\text{red}} - 1$ **do**
    $l := jN_{\text{red}};$
    for $k := 0$ **to** $M_{\text{red}} - 1$ **do**
      $a := f_{l+k} + f_{l+k+M_{\text{red}}};$
      $f_{l+k+M_{\text{red}}} := (f_{l+k} - f_{l+k+M_{\text{red}}})z^k;$
      $f_{l+k} := a;$
    end for
  end for
  $N_{\text{red}} := M_{\text{red}};$
  $z := z^2;$
end while
for $k := 0$ **to** $N - 1$ **do**
  $\alpha_{\sigma(k)} := f_k$
end for

In each reduction step, we need $2 \cdot 2^p = 2N$ multiplications, where the evaluation of the exponential function counts for one multiplication (recursive computation of $\cos jx$, $\sin jx$). After $p = \log_2 N$ steps, all $\alpha_0, \ldots, \alpha_{N-1}$ are computed at the cost of $2N \log_2 N$ multiplications.

7.3 Bézier Techniques

The topics, which have so far been presented in this chapter belong to the classical part of Numerical Analysis, as the names Lagrange and Newton indicate. With the increasing importance of computer-aided construction, new ground has recently been broken (i.e., in the last thirty years) in interpolation and approximation theory, which we shall indicate in this section. It is interesting that geometric aspects gain a decisive importance here. A curve or surface has to be represented on a computer in a way that it can be drawn and manipulated quickly. In order to achieve this, parametrizations of the geometric objects are used, whose relevant parameters have geometric meaning.

In this introduction we can only illustrate these considerations in the simplest situations. In particular, we shall restrict ourselves to polynomial curves, i.e., one-dimensional geometric objects. The book by C. de Boor [15] and the newer textbook by G. Farin [30] are recommended to those who want to familiarize themselves in more detail with this area.

We start with a generalization of real-valued polynomials.

Definition 7.29 A *polynomial* (or a *polynomial curve*) *of degree* n in $\mathbf{R}^d$ is a function P of the form

$$P : \mathbf{R} \to \mathbf{R}^d, \quad P(t) = \sum_{i=0}^{n} a_i t^i \quad \text{with} \quad a_0, \ldots, a_n \in \mathbf{R}^d, \quad a_n \neq 0 .$$

The space of polynomials of degree less than or equal to n in $\mathbf{R}^d$ is denoted by $\mathbf{P}_n^d$.

The most interesting cases for us are the curves in space ($d = 3$) or in the plane ($d = 2$). If $\{P_0, \ldots, P_n\}$ is a basis of $\mathbf{P}_n$ and $\{e_1, \ldots, e_d\}$ the standard basis of $\mathbf{R}^d$, then the polynomials

$$\{e_i P_j \mid \; i = 1, \ldots, d \;\text{ and }\; j = 0, \ldots, n\}$$

form a basis of $\mathbf{P}_n^d$. The graph Γ_P of a polynomial $P \in \mathbf{P}_n^d$

$$\Gamma_P : \mathbf{R} \longrightarrow \mathbf{R}^{d+1}, \quad t \longmapsto (t, P(t))$$

can now again be considered as a polynomial $\Gamma_P \in \mathbf{P}_n^{d+1}$. If P is given in coefficient representation

$$P(t) = a_0 + a_1 t + \cdots + a_n t^n ,$$

then

$$\Gamma_P(t) = \begin{pmatrix} 0 \\ a_0 \end{pmatrix} + \begin{pmatrix} 1 \\ a_1 \end{pmatrix} t + \begin{pmatrix} 0 \\ a_2 \end{pmatrix} t^2 + \cdots + \begin{pmatrix} 0 \\ a_n \end{pmatrix} t^n .$$

7.3.1 *Bernstein Polynomials and Bézier Representation*

So far, we have considered three different bases of the space P_n of polynomials of degree less than or equal to n:

(a) Monomial basis $\{1, t, t^2, \ldots, t^n\}$,

(b) Lagrange basis $\{L_0(t), \ldots, L_n(t)\}$,

(c) Newton basis $\{\omega_0(t), \ldots, \omega_n(t)\}$.

The last two bases are already oriented toward interpolation and depend on the nodes $t_0, \ldots, t_n$. The basis polynomials, which we shall now present, apply to two parameters $a, b \in \mathbf{R}$. They are therefore very suitable for the *local* representation of a polynomial. In the following, the closed interval *between* the two points a and b is denoted by $[a, b]$ also when $a > b$, i.e., (compare Definition 7.37)

$$[a, b] := \{x = \lambda a + (1 - \lambda)b \mid \lambda \in [0, 1]\}.$$

The first step consists of an affine transformation onto the unit interval $[0, 1]$,

$$\begin{aligned} [a, b] &\longrightarrow [0, 1] \\ t &\longmapsto \lambda = \lambda(t) := \frac{t - a}{b - a}, \end{aligned} \tag{7.22}$$

with the help of which we can usually restrict our consideration to $[0, 1]$. By virtue of the binomial theorem, we can represent the unit function as

$$1 = ((1 - \lambda) + \lambda)^n = \sum_{i=0}^{n} \binom{n}{i} (1 - \lambda)^{n-i} \lambda^i .$$

The terms of this partition of unity are just the *Bernstein polynomials* with respect to the interval $[0, 1]$. By composing these with the above affine transformation (7.22), we then obtain the Bernstein polynomials with respect to the interval $[a, b]$.

Definition 7.30 The i^{th} *Bernstein polynomial* (compare Figure 7.4) of degree n with respect to the interval $[0, 1]$ is the polynomial $B_i^n \in \mathbf{P}_n$ with

$$B_i^n(\lambda) := \binom{n}{i} (1 - \lambda)^{n-i} \lambda^i ,$$

where $i = 0, \ldots, n$. Similarly, $B_i^n(\cdot\,; a, b) \in \mathbf{P}_n$ with

$$B_i^n(t; a, b) := B_i^n(\lambda(t)) = B_i^n\left(\frac{t - a}{b - a}\right) = \frac{1}{(b - a)^n} \binom{n}{i} (t - a)^i (b - t)^{n-i}$$

is the i^{th} *Bernstein polynomial* of degree n with respect to the interval $[a, b]$.

Instead of $B_i^n(t;a,b)$, we shall in the following often simply write $B_i^n(t)$, if confusion with the Bernstein polynomials $B_i^n(\lambda)$ with respect to $[0,1]$ is impossible. In the following theorem we list the most important properties

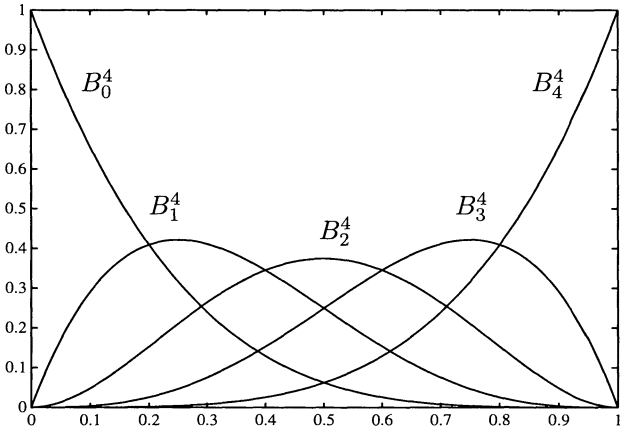

Figure 7.4. Bernstein polynomials for $n = 4$.

of the Bernstein polynomials.

Theorem 7.31 *The Bernstein polynomials $B_i^n(\lambda)$ satisfy the following properties:*

1. $\lambda = 0$ *is a multiplicity* i *root of* B_i^n.
2. $\lambda = 1$ *is a multiplicity* $(n-i)$ *root of* B_i^n.
3. $B_i^n(\lambda) = B_{n-i}^n(1-\lambda)$ *for* $i = 0,\ldots,n$ *(symmetry).*
4. $(1-\lambda)B_0^n = B_0^{n+1}$ *and* $\lambda B_n^n = B_{n+1}^{n+1}$.
5. *The Bernstein polynomials B_i^n are nonnegative on $[0,1]$ and form a partition of unity, i.e.,*

$$B_i^n(\lambda) \geq 0 \quad \text{for} \quad \lambda \in [0,1] \quad \text{and} \quad \sum_{i=0}^{n} B_i^n(\lambda) = 1 \quad \text{for} \quad \lambda \in \mathbf{R}.$$

6. B_i^n *has exactly one maximum value in the interval $[0,1]$, namely, at $\lambda = i/n$.*
7. *The Bernstein polynomials satisfy the recurrence relation*

$$B_i^n(\lambda) = \lambda B_{i-1}^{n-1}(\lambda) + (1-\lambda)B_i^{n-1}(\lambda) \tag{7.23}$$

for $i = 1,\ldots,n$ *and* $\lambda \in \mathbf{R}$.

8. *The Bernstein polynomials form a basis* $\mathcal{B} := \{B_0^n, \ldots, B_n^n\}$ *of* $\mathbf{P}_n$.

Proof. The first five statements are either obvious or can be easily verified. Statement 6 follows from the fact that

$$\frac{d}{d\lambda} B_i^n(\lambda) = \binom{n}{i} (1-\lambda)^{n-i-1} \lambda^{i-1} (i - n\lambda)$$

for $i = 1, \ldots, n$. The recurrence relation (7.23) follows from the definition and the formula

$$\binom{n}{i} = \binom{n-1}{i-1} + \binom{n-1}{i}$$

for the binomial coefficients. For the last statement, we show that the $n+1$ polynomials B_i^n are linearly independent. If

$$0 = \sum_{i=0}^{n} b_i B_i^n(\lambda)\,,$$

then according to 1 and 2,

$$0 = \sum_{i=0}^{n} b_i B_i^n(1) = b_n B_n^n(1) = b_n$$

and therefore inductively $b_0 = \cdots = b_n = 0$. □

Similar statements are of course true for the Bernstein polynomials with respect to the interval $[a, b]$. Here the maximum value of $B_i^n(t; a, b)$ in $[a, b]$ is attained at

$$t = a + \frac{i}{n}(b - a)\,.$$

Remark 7.32 The property that the Bernstein polynomials form a partition of unity is equivalent to the fact that the Bézier points are *affine invariant*. If $\phi : \mathbf{R}^d \to \mathbf{R}^d$ is an affine mapping,

$$\begin{aligned} \phi : \mathbf{R}^d &\longrightarrow \mathbf{R}^d \quad \text{with} \quad A \in \mathrm{Mat}_d(\mathbf{R}) \quad \text{and} \quad v \in \mathbf{R}^d \\ u &\longmapsto Au + v, \end{aligned}$$

then the images $\phi(b_i)$ of the Bézier points b_i of a polynomial $P \in \mathbf{P}_n^d$ are the Bézier points of $\phi \circ P$.

We now know that we can write any polynomial $P \in \mathbf{P}_n^d$ as a linear combination with respect to the Bernstein basis

$$P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b), \quad b_i \in \mathbf{R}^d\,. \tag{7.24}$$

Remark 7.33 The symmetry $B_i^n(\lambda) = B_{n-i}^n(1-\lambda)$ of the Bernstein polynomials yields in particular

$$\sum_{i=0}^{n} b_i B_i^n(t;a,b) = \sum_{i=0}^{n} b_{n-i} B_i^n(t;b,a)\,;$$

i.e., the Bézier coefficients with respect to b, a are just the ones of a, b in reverse order.

The coefficients $b_0, \ldots, b_n$ are called *control* or *Bézier points* of P, the corresponding polygonal path a *Bézier polygon.* Because of

$$\lambda = \sum_{i=0}^{n} \frac{i}{n} B_i^n(\lambda) \quad \Longrightarrow \quad t = \sum_{i=0}^{n} \left(a + \frac{i}{n}(b-a)\right) B_i^n(t;a,b)$$

the Bézier points of the polynomial $P(t) = t$ are, just the maxima $b_i = a + \frac{i}{n}(b-a)$ of the Bernstein polynomials. The Bézier representation of the graph Γ_P of a polynomial P as in (7.24) is therefore just

$$\Gamma_P(t) = \begin{pmatrix} t \\ P(t) \end{pmatrix} = \sum_{i=0}^{n} \begin{pmatrix} a + \frac{i}{n}(b-a) \\ b_i \end{pmatrix} B_i^n(t;a,b)\,. \tag{7.25}$$

In Figure 7.5 we have plotted the graph of a cubic polynomial together

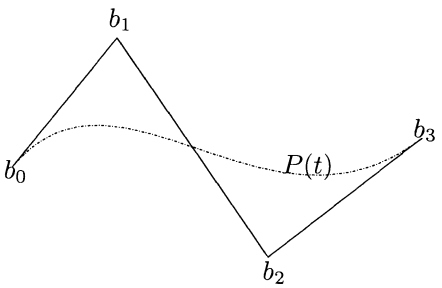

Figure 7.5. Cubic polynomial with its Bézier points.

with its Bézier polygon. It is striking that the shape of the curve is closely related to the shape of the Bézier polygon. In the following we shall more closely investigate this geometric meaning of the Bézier points. First, it is clear from Theorem 7.31 that the beginning and ending points of the polynomial curve and the Bézier polygon coincide. Furthermore, it appears that the tangents at the boundary points also coincide with the straight lines at the end of the Bézier polygon. In order to verify this property, we compute the derivatives of a polynomial in Bézier representation. We shall restrict ourselves to the derivatives of the Bézier representation with respect to the unit interval $[0,1]$. Together with the derivative of the affine

transformation $\lambda(t)$ from $[a,b]$ onto $[0,1]$,

$$\frac{d}{dt}\lambda(t) = \frac{1}{b-a},$$

one immediately obtains the derivatives in the general case also.

Lemma 7.34 *The derivative of the Bernstein polynomials $B_i^n(\lambda)$ with respect to $[0,1]$ satisfies*

$$\frac{d}{d\lambda}B_i^n(\lambda) = \begin{cases} -nB_0^{n-1} & \text{for } i=0 \\ n(B_{i-1}^{n-1}(\lambda) - B_i^{n-1}(\lambda)) & \text{for } i=1,\ldots,n-1 \\ nB_{n-1}^{n-1}(\lambda) & \text{for } i=n\,. \end{cases}$$

Proof. The statement follows from

$$\frac{d}{d\lambda}B_i^n(\lambda) = \binom{n}{i}\left[i(1-\lambda)^{n-i}\lambda^{i-1} - (n-i)(1-\lambda)^{n-i-1}\lambda^i\right]$$

by virtue of the identities of Theorem 7.31. □

Theorem 7.35 *Let $P(\lambda) = \sum_{i=0}^n b_i B_i^n(\lambda)$ be a polynomial in Bézier representation with respect to $[0,1]$. Then the k^{th} derivative of P satisfies*

$$P^{(k)}(\lambda) = \frac{n!}{(n-k)!}\sum_{i=0}^{n-k}\Delta^k b_i B_i^{n-k}(\lambda),$$

where the forward difference operator Δ operates on the lower index, i.e.,

$$\Delta^1 b_i := b_{i+1} - b_i \quad \text{and} \quad \Delta^k b_i := \Delta^{k-1}b_{i+1} - \Delta^{k-1}b_i \quad \text{for} \quad k>1\,.$$

Proof. Induction over k, see Exercise 7.6. □

Corollary 7.36 *For the boundary points $\lambda = 0,1$ one obtains the values*

$$P^{(k)}(0) = \frac{n!}{(n-k)!}\Delta^k b_0 \quad \text{and} \quad P^{(k)}(1) = \frac{n!}{(n-k)!}\Delta^k b_{n-k},$$

thus in particular up to the second derivative,

(a) $P(0) = b_0$ *and* $P(1) = b_n$,

(b) $P'(0) = n(b_1 - b_0)$ *and* $P'(1) = n(b_n - b_{n-1})$,

(c) $P''(0) = n(n-1)(b_2 - 2b_1 + b_0)$ *and* $P''(1) = n(n-1)(b_n - 2b_{n-1} + b_{n-2})$.

Proof. Note that $B_i^{n-k}(0) = \delta_{0,i}$ and $B_i^{n-k}(1) = \delta_{n-k,i}$. □

Corollary 7.36 therefore confirms the geometric observations that we described above. It is important that at a boundary point, the curve is determined up to the k^{th} derivative by the k closest Bézier points. This property will be crucial later on for the purpose of joining several pieces together. The Bézier points are of further geometric importance. In order to describe this, we need the notion of a *convex hull* of a set $A \subset \mathbf{R}^d$, which we shall briefly review.

Definition 7.37 A set $A \subset \mathbf{R}^d$ is called a *convex*, if together with any two points $x, y \in A$, the straight line, which joins them is also contained in A, i.e.,

$$[x, y] := \{\lambda x + (1 - \lambda)y \mid \lambda \in [0, 1]\} \subset A \quad \text{for all} \quad x, y \in A\,.$$

The *convex hull* $\operatorname{co}(A)$ of a set $A \subset \mathbf{R}^d$ is the smallest convex subset of $\mathbf{R}^d$, which contains A. A linear combination of the form

$$x = \sum_{i=1}^{k} \lambda_i x_i, \quad \text{with} \quad x_i \in \mathbf{R}^d, \quad \lambda_i \geq 0 \quad \text{and} \quad \sum_{i=1}^{k} \lambda_i = 1$$

is called *convex combination* of $x_1, \ldots, x_k$.

Remark 7.38 The convex hull $\operatorname{co}(A)$ of $A \subset \mathbf{R}^d$ is the set of all convex combinations of points of A, i.e.,

$$\begin{aligned} \operatorname{co}(A) &= \bigcap \{B \subset \mathbf{R}^d \mid^{\bullet} B \text{ convex with } A \subset B\} \\ &= \left\{ x = \sum_{i=1}^{m} \lambda_i x_i \;\middle|\; m \in \mathbf{N}, \quad x_i \in A, \quad \lambda_i \geq 0, \quad \sum_{i=1}^{m} \lambda_i = 1 \right\}. \end{aligned}$$

The following theorem states that a polynomial curve is always contained in the convex hull of its Bézier points.

Theorem 7.39 *The image $P([a, b])$ of a polynomial $P \in \mathbf{P}_n^d$ in Bernstein representation $P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b)$ with respect to $[a, b]$ is contained in the convex hull of the Bézier points b_i, i.e.,*

$$P(t) \in \operatorname{co}(b_0, \ldots, b_n) \quad \textit{for} \quad t \in [a, b]\,.$$

In particular, the graph of the polynomial for $t \in [a, b]$ is contained in the convex hull of the points $\mathbf{b}_i$.

Proof. On $[a, b]$, the Bernstein polynomials form a nonnegative partition of unity, i.e., $B_i^n(t; a, b) \geq 0$ for $t \in [a, b]$ and $\sum_{i=0}^{n} B_i^n(t) = 1$. Therefore

$$P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b)$$

is a convex combination of the Bézier points $b_0, \ldots, b_n$. The second statement follows from the Bézier representation (7.25) of the graph Γ_P of P.

□

As one can already see in Figure 7.5, for a cubic polynomial $P \in \mathbf{P}_3$, this means that the graph of P for $t \in [a, b]$ is completely contained in the convex hull of the four Bézier points $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$, and $\mathbf{b}_4$. The name control point is explained by the fact that, because of their geometric significance, the points $\mathbf{b}_i$ can be used to control a polynomial curve. Because of Theorem 7.31, at the position $\lambda = i/n$, over which it is plotted, the control point b_i has the greatest "weight" $B_i^n(\lambda)$. This is another reason that the curve between a and b is closely related to the Bézier polygon, as the figure indicates.

7.3.2 De Casteljau Algorithm

Besides the geometric interpretation of the Bézier points, the importance of the Bézier representation rests mainly on the fact that there is an algorithm, which builds on continued convex combinations, and which, besides the function value $P(t)$ at an arbitrary position t, also yields information about the derivatives. Furthermore, the same algorithm can be used to subdivide the Bézier curve into two segments. By repeating this partitioning into segments, the sequence of the Bézier polygons converges extremely fast to the curve (exponentially when dividing the interval into halves), so that this method is very well-suited to effectively plot a curve, e.g., in computer graphics. This construction principle is also tailor-made to control a milling cutter, which can only "remove" material. We start with the definition of the *partial polynomials* of P.

Definition 7.40 Let $P(\lambda) = \sum_{i=0}^{n} b_i B_i^n(\lambda)$ be a polynomial in Bézier representation with respect to $[0, 1]$. Then we define the *partial polynomials* $b_i^k \in \mathbf{P}_k^d$ of P for $i = 0, \ldots, n-k$ by

$$b_i^k(\lambda) := \sum_{j=0}^{k} b_{i+j} B_j^k(\lambda) = \sum_{j=i}^{i+k} b_j B_{j-i}^k(\lambda) .$$

For a polynomial $P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b)$ in Bézier representation with respect to $[a, b]$, the partial polynomials b_i^k are similarly defined by

$$b_i^k(t; a, b) := b_i^k(\lambda(t)) = \sum_{j=0}^{k} b_{i+j} B_j^k(t; a, b) .$$

Thus the partial polynomial $b_i^k \in \mathbf{P}_k^d$ is just the polynomial, which is defined by the Bézier points $b_i, \ldots, b_{i+k}$ (see Figure 7.6). If no confusion arises, then we simply write $b_i^k(t)$ for $b_i^k(t; a, b)$. In particular, $b_0^n(t) = P(t)$ is the starting polynomial, and the $b_i^0(t) = b_i$ are its Bézier points for all $t \in \mathbf{R}$. Furthermore, for all boundary points,

$$b_i^k(a) = b_i \quad \text{and} \quad b_i^k(b) = b_{i+k} .$$

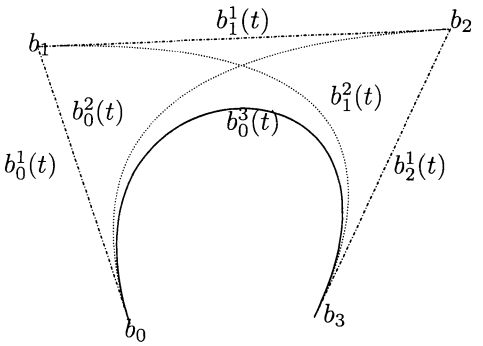

Figure 7.6. Cubic polynomial with its partial polynomials.

Similar to the Aitken lemma, the following recurrence relation is true, which is the base for the algorithm of de Casteljau.

Lemma 7.41 *The partial polynomials* $b_i^k(t)$ *of* $P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b)$ *satisfy the recurrence relation*

$$b_i^k = (1-\lambda)b_i^{k-1} + \lambda b_{i+1}^{k-1} \quad \textit{with} \quad \lambda = \lambda(t) = \frac{t-a}{b-a}$$

for $k = 0, \ldots, n$ *and* $i = 0, \ldots, n-k$.

Proof. We insert the recurrence relation (7.23) into the definition of the partial polynomials b_i^k and obtain

$$\begin{aligned}
b_i^k &= \sum_{j=0}^{k} b_{i+j} B_j^k \\
&= b_i B_0^k + b_{i+k} B_k^k + \sum_{j=1}^{k-1} b_{i+j} B_j^k \\
&= b_i (1-\lambda) B_0^{(k-1)} + b_{i+k} \lambda B_{k-1}^{k-1} + \sum_{j=1}^{k-1} b_{i+j} \left((1-\lambda) B_j^{k-1} + \lambda B_{j-1}^{k-1} \right) \\
&= \sum_{j=0}^{k-1} b_{i+j} (1-\lambda) B_j^{k-1} + \sum_{j=1}^{k} b_{i+j} \lambda B_{j-1}^{k-1} \\
&= (1-\lambda) b_i^{k-1} + \lambda b_{i+1}^{k-1} .
\end{aligned}$$

□

Because of $b_i^0(t) = b_i$, by continued convex combination (which, for $t \notin [a, b]$ is only an affine combination) we can compute the function value $P(t) = b_0^n(t)$ from the Bézier points. The auxiliary points b_i^k can, similar

to the scheme of Neville, be arranged in the *de Casteljau scheme.*

$$\begin{array}{lllllllll}
b_n & = & b_n^0 & & & & & & \\
 & & & \searrow & & & & & \\
b_{n-1} & = & b_{n-1}^0 & \to & b_{n-1}^1 & & & & \\
 & \vdots & & & & \ddots & & & \\
b_1 & = & b_1^0 & \to & \cdots & \to & b_1^{n-1} & & \\
 & & & \searrow & & & & \searrow & \\
b_0 & = & b_0^0 & \to & \cdots & \to & b_0^{n-1} & \to & b_0^n
\end{array}$$

In fact, the derivatives of P are hidden behind the auxiliary points b_i^k of de Casteljau's scheme, as the following theorem shows. Here we again only consider the Bézier representation with respect to the unit interval $[0,1]$.

Theorem 7.42 *Let $P(\lambda) = \sum_{i=0}^n b_i B_i^n(\lambda)$ be a polynomial in Bézier representation with respect to $[0,1]$. Then the derivatives $P^{(k)}(\lambda)$ for $k = 0, \ldots, n$ can be computed from the partial polynomials $b_i^k(\lambda)$ via the relation*

$$P^{(k)}(\lambda) = \frac{n!}{(n-k)!} \Delta^k b_0^{n-k}(\lambda),$$

where $\Delta b_i^k = b_{i+1}^k - b_i^k$.

Proof. The statement follows from Theorem 7.35 and from the fact that the forwards difference operator commutes with the sum:

$$\begin{aligned}
P^{(k)}(\lambda) &= \frac{n!}{(n-k)!} \sum_{i=0}^{n-k} \Delta^k b_i B_i^{n-k}(\lambda) = \frac{n!}{(n-k)!} \Delta^k \sum_{i=0}^{n-k} b_i B_i^{n-k}(\lambda) \\
&= \frac{n!}{(n-k)!} \Delta^k b_0^{n-k}(\lambda) .
\end{aligned}$$

□

Thus the k^{th} derivative $P^{(k)}(\lambda)$ at the position λ is computed from the $(n-k)^{\text{th}}$ column of the de Casteljau scheme. In particular,

$$\begin{aligned}
P(t) &= b_0^n , \\
P'(t) &= n(b_1^{n-1} - b_0^{n-1}) , \\
P''(t) &= n(n-1)(b_2^{n-2} - 2b_1^{n-2} + b_0^{n-2}) .
\end{aligned}$$

So far we have only considered the Bézier representation of a single polynomial with respect to a fixed reference interval. Here the question remains open on how the Bézier points are transformed when we change the reference interval (see Figure 7.7). It would also be interesting to know how to join several pieces of polynomial curves continuously or smoothly (see Figure 7.8). Finally, we would be interested in the possibility of subdividing curves, in the sense that we subdivide the reference interval and

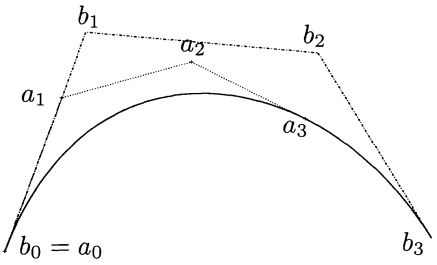

Figure 7.7. Cubic polynomial with two Bézier representations.

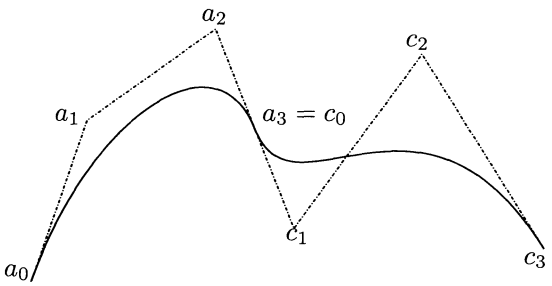

Figure 7.8. Two cubic Bézier curves with C^1-smoothness at $a_3 = c_0$.

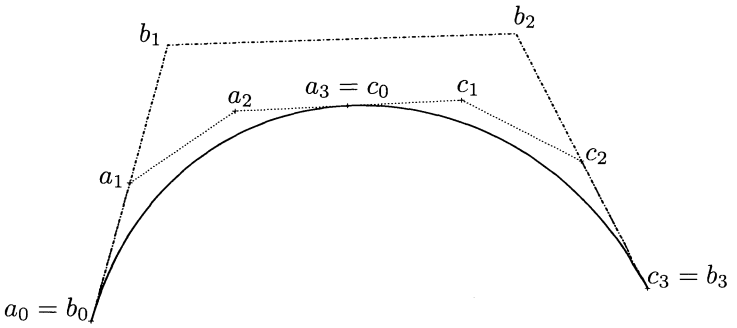

Figure 7.9. Subdivision of a cubic Bézier curve.

compute the Bézier points for the subintervals (see Figure 7.9). According to Theorem 7.39, the curve is contained in the convex hull of the Bézier points. Hence, it is clear that the Bézier polygons approach more and more closely the curve, when the subdivision is refined. These three questions are closely related, which can readily be seen from the figures. We shall see that they can be easily resolved in the context of the Bézier technique. The connecting elements are the partial polynomials. We have already seen in

Corollary 7.36 that, at a boundary point, a Bézier curve P is determined to the k^{th} derivative by the k closest Bézier points. The opposite is also true: The values of P up to the k^{th} derivative at the position $\lambda = 0$ already determine the Bézier points $b_0, \ldots, b_k$. More precisely, this is even true for the partial polynomials $b_0^0(\lambda), \ldots, b_0^k(\lambda)$, as we shall prove in the following lemma.

Lemma 7.43 *The partial polynomial $b_0^k(\lambda)$ of a Bézier curve $P(\lambda) = b_0^n(\lambda) = \sum_{i=0}^n b_i B_i^n(\lambda)$ is completely determined by the values of P up to and including the k^{th} derivative at the position $\lambda = 0$.*

Proof. According to Theorem 7.42, the derivatives at the position $\lambda = 0$ satisfy

$$\frac{d^l}{d\lambda^l} b_0^k(0) = \frac{k!}{(k-l)!} \Delta^l b_0 = \frac{(n-l)!\,k!}{(k-l)!\,n!} \frac{d^l}{d\lambda^l} b_0^n(0)$$

for $l = 0, \ldots, k$. The statement follows, because a polynomial is completely determined by all derivatives at one position. □

Together with Corollary 7.36, we obtain the following theorem:

Theorem 7.44 *Let $P(t) = a_0^n(t; a, b)$ and $Q(t) = b_0^n(t; a, c)$ be two Bézier curves with respect to a, b, respectively, a, c. Then the following statements are equivalent:*

(i) *$P(t)$ and $Q(t)$ coincide at the position $t = a$ up to the k^{th} derivative, i.e.,*

$$P^{(l)}(a) = Q^{(l)}(a) \quad \text{for} \quad l = 0, \ldots, k\,.$$

(ii) $a_0^k(t; a, b) = b_0^k(t; a, c)$ *for all* $t \in \mathbf{R}$.

(iii) $a_0^l(t; a, b) = b_0^l(t; a, c)$ *for all* $t \in \mathbf{R}$ *and* $l = 0, \ldots, k$.

(iv) $a_l = b_0^l(b; a, c)$ *for* $l = 0, \ldots, k$.

Proof. We show (i) $\Leftrightarrow$ (ii) $\Rightarrow$ (iii) $\Rightarrow$ (iv) $\Rightarrow$ (ii). According to Corollary 7.36 and Lemma 7.43, the two curves $P(t)$ and $Q(t)$ coincide at the position $t = a$ up to the k^{th} derivative, if and only if they have the same partial polynomials $a_0^k(t; a, b) = b_0^k(t; a, c)$. The two first statements are therefore equivalent. If a_0^k and b_0^k coincide, then so do their partial polynomials a_0^l and b_0^l for $l = 0, \ldots, k$; i.e., (ii) implies (iii). By inserting $t = b$ into (iii), it follows in particular that

$$a_l = a_0^l(1) = a_0^l(b; a, b) = b_0^l(b; a, c)\,,$$

and therefore (iv). Since a polynomial is uniquely determined by its Bézier coefficients, (iv) therefore implies (ii) and thus the equivalence of the four statements. □

With this result in hand, we can easily answer our three questions. As a first corollary we compute the Bézier points that are created when *subdividing* the reference interval. At the same time, this answers the question regarding the change of the reference interval.

Corollary 7.45 *Let*

$$a_0^n(t;a,b) = b_0^n(t;a,c) = c_0^n(t;b,c)$$

be the Bézier representations of a polynomial curve $P(t)$ with respect to the intervals $[a,b]$, $[a,c]$ and $[b,c]$, i.e.,

$$P(t) = \sum_{i=0}^{n} a_i B_i^n(t;a,b) = \sum_{i=0}^{n} b_i B_i^n(t;a,c) = \sum_{i=0}^{n} c_i B_i^n(t;b,c)$$

(see Figure 7.9). Then the Bézier coefficients a_i and c_i of the partial curves can be computed from the Bézier coefficients b_i with respect to the entire interval via

$$a_k = b_0^k(b;a,c) \quad \text{and} \quad c_k = b_k^{n-k}(b;a,c)$$

for $k = 0,\ldots,n$.

Proof. Because a polynomial of degree n is completely determined by its derivatives at one point, the statement follows from Theorem 7.44 for $k = n$ and the symmetry of the Bézier representation, see Remark 7.33. □

Since the curve pieces always lie in the convex hull of their Bézier points, the corresponding Bézier polynomials converge to the curve when continuously subdivided. By employing this method, the evaluation of a polynomial is very stable, since only convex combinations are computed in the algorithm of de Casteljau. In Figure 7.10, we have always divided the reference interval of a Bézier curve of degree 4 in half, and we have plotted the Bézier polygon of the first three subdivisions. After only a few subdivisions, it is almost impossible to distinguish the curve from the polygonal path.

If we do utilize the fact that only the derivatives at one position must coincide, then we can solve the problem of continuously joining two polygonal curves:

Corollary 7.46 *A joined Bézier curve*

$$R(t) = \begin{cases} a_0^n(t;a,b) & \text{if} \quad a \le t < b \\ c_0^n(t;b,c) & \text{if} \quad b \le t \le c \end{cases}$$

is C^k-smooth, if and only if

$$c_l = a_{n-l}^l(c;a,b) \quad \text{for} \quad l = 0,\ldots,k$$

or, equivalently,

$$a_{n-l} = c_0^l(a;b,c) \quad \text{for} \quad l = 0,\ldots,k\,.$$

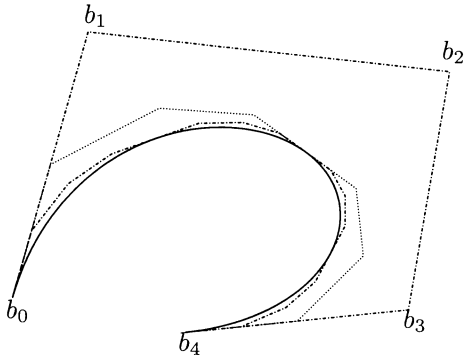

Figure 7.10. Threefold subdivision of a Bézier curve of degree $n = 4$.

Therefore, through the C^k-smoothness, the first $k+1$ Bézier points of the second partial curve are determined by the last $k+1$ Bézier points of the first and vice versa. A polynomial $a_0^n(t; a, b)$ over $[a, b]$ can therefore be continued C^k-smoothly by a polynomial $c_0^n(t; b, c)$ over $[b, c]$, by determining the Bézier points $c_0, \ldots, c_k$ according to Corollary 7.45 by employing the algorithm of de Casteljau, whereas the remaining $c_{k+1}, \ldots, c_n$ can be chosen freely.

In particular, the joined curve $R(t)$ is *continuous*, if and only if

$$a_n = c_0 \,.$$

It is *continuously differentiable*, if and only if, in addition,

$$\begin{aligned} c_1 &= a_{n-1}^1(c; a, b) \\ &= a_{n-1}^1(\lambda) = (1-\lambda)a_{n-1} + \lambda a_n \quad \text{with} \quad \lambda = \frac{c-a}{b-a} \end{aligned}$$

or, equivalently,

$$\begin{aligned} a_{n-1} &= c_0^1(a; b, c) \\ &= c_0^1(\mu) = (1-\mu)c_0 + \mu c_1 \quad \text{with} \quad \mu = \frac{a-b}{c-b} \,. \end{aligned}$$

This implies that

$$a_n = c_0 = \frac{c-b}{c-a} a_{n-1} + \frac{b-a}{c-a} c_1 \,, \tag{7.26}$$

i.e., the point $a_n = c_0$ has to divide the segment $[a_{n-1}, c_1]$ in the proportion $c-b$ to $b-a$. If the pieces of curves fit *C^2-smoothly*, then a_{n-2}, a_{n-1} and a_n describe the same parabola as c_0, c_1 and c_2, namely, with respect to $[a, b]$, respectively, $[b, c]$. According to Corollary 7.46, the Bézier points of this parabola with respect to the entire interval $[a, c]$ are a_{n-2}, d and c_2, where d is the auxiliary point

$$d := a_{n-2}^1(c; a, b) = a_{n-2}^1(\lambda) = c_1^1(a; b, c) = c_1^1(\mu)$$

(see Figure 7.11). Furthermore, according to Corollary 7.46, it follows from

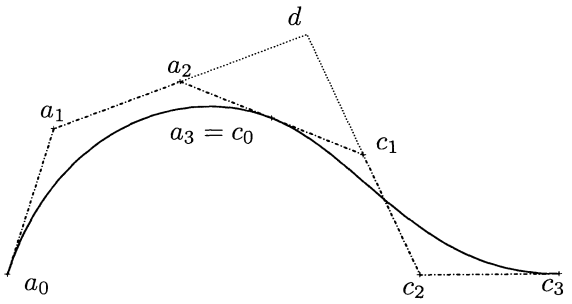

Figure 7.11. Two cubic Bézier curves with C^2-transition at $a_3 = c_0$.

the C^2-smoothness that

$$c_2 = a^2_{n-2}(\lambda) = (1-\lambda)\underbrace{a^1_{n-2}(\lambda)}_{=d} + \lambda \underbrace{a^1_{n-1}(\lambda)}_{=c_1}$$

and

$$a_{n-2} = c^2_0(\mu) = (1-\mu)\underbrace{c^1_0(\mu)}_{=a_{n-1}} + \mu \underbrace{c^1_1(\mu)}_{=d}.$$

The joined curve is therefore C^2-smooth, if and only if there is a point d such that

$$c_2 = (1-\lambda)d + \lambda c_1 \quad \text{and} \quad a_{n-2} = (1-\mu)a_{n-1} + \mu d.$$

The auxiliary point d, the *de Boor point*, will play an important role in the next section in the construction of cubic splines.

7.4 Splines

As we have seen, the classical polynomial interpolation is incapable of solving the approximation problem with a large number of equidistant nodes. Polynomials of high degree tend to oscillate a lot, as the sketches of the Lagrange polynomials indicate (see Figure 7.2). They may thus not only spoil the condition number (small changes of the nodes f_i induce large changes of the interpolation polynomial $P(t)$ at intermediate values $t \neq t_i$), but also lead to large oscillations of the interpolating curve between the nodes. As one can imagine, such oscillations are highly undesirable. One need only think of the induced vibrations of an airfoil formed according to such an interpolation curve. If we require that an interpolating curve passes "as

smooth as possible" through given nodes (t_i, f_i), then it is obvious to locally use polynomials of *lower degree* and to join these at the nodes. As a first possibility, we have encountered the cubic Hermite interpolation in Example 7.7, which was, however, dependent on the special prescription of function values and derivatives at the nodes. A second possibility are the *spline functions*, with which we shall be concerned in this chapter.

7.4.1 Spline Spaces and B-Splines

We start with the definition of k^{th} order splines over a grid $\Delta = \{t_0, \ldots, t_{l+1}\}$ of node points. These functions have proven to be an extremely versatile tool, from interpolation and approximation and the modeling in CAGD to collocation and Galerkin methods for differential equations.

Definition 7.47 Let $\Delta = \{t_0, \ldots, t_{l+1}\}$ be a *grid* of $l+2$ pairwise distinct node points

$$a = t_0 < t_1 < \cdots < t_{l+1} = b\,.$$

A *spline* of degree $k-1$ (order k) with respect to Δ is a function $s \in C^{k-2}[a,b]$, which on each interval $[t_i, t_{i+1}]$ for $i = 0, \ldots, l$ coincides with a polynomial $s_i \in \mathbf{P}_{k-1}$ of degree $\leq k-1$. The space of splines of degree $k-1$ with respect to Δ is denoted by $\mathbf{S}_{k,\Delta}$.

The most important spline functions are the *linear splines* of order $k = 2$ (see Figure 7.12) and the *cubic splines* of order $k = 4$ (see Figure 7.13). The linear splines are the continuous, piecewise linear functions with respect to the intervals $[t_i, t_{i+1}]$. The cubic splines are best suited for the graphic

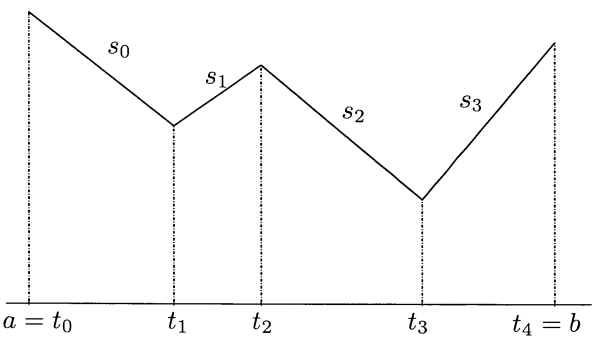

Figure 7.12. Linear splines, order $k = 2$.

representation of curves, since the eye can still recognize discontinuities of

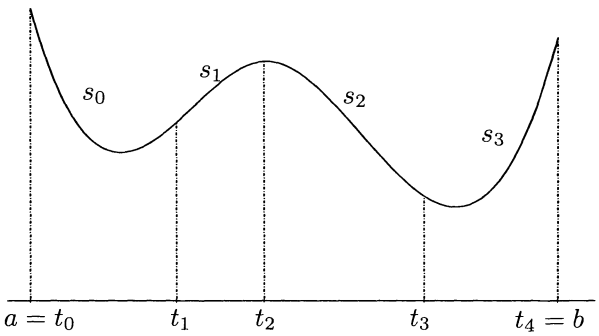

Figure 7.13. Cubic splines, order $k = 4$.

curvature, i.e., of the second derivative. Thus the C^2-smooth cubic splines are recognized as "smooth."

It is obvious that $\mathbf{S}_{k,\Delta}$ is a real vector space, which, in particular, contains all polynomials of degree $\leq k-1$, i.e., $\mathbf{P}_{k-1} \subset \mathbf{S}_{k,\Delta}$. Furthermore, the *truncated powers* of degree k,

$$(t-t_i)_+^k := \begin{cases} (t-t_i)^k & \text{if} \quad t \geq t_i \\ 0 & \text{if} \quad t < t_i \end{cases}$$

are contained in $\mathbf{S}_{k,\Delta}$. Together with the monomials $1, t, \ldots, t^{k-1}$, they form a basis of $\mathbf{S}_{k,\Delta}$, as we shall show in the following theorem:

Theorem 7.48 *The monomials and truncated powers form a basis*

$$\mathcal{B} := \{1, t, \ldots, t^{k-1}, (t-t_1)_+^{k-1}, \ldots, (t-t_l)_+^{k-1}\} \tag{7.27}$$

of the spline space $\mathbf{S}_{k,\Delta}$. *In particular, the dimension of* $\mathbf{S}_{k,\Delta}$ *is*

$$\dim \mathbf{S}_{k,\Delta} = k + l\,.$$

Proof. We first show that one has at most $k+l$ degrees of freedom for the construction of a spline $s \in \mathbf{S}_{k,\Delta}$. On the interval $[t_0, t_1]$, we can choose any polynomial of degree $\leq k-1$; these are k free parameters. Because of the smoothness requirement $s \in C^{k-2}$, the polynomials on the following intervals $[t_1, t_2], \ldots, [t_l, t_{l+1}]$ are determined by their predecessor up to one parameter. Thus, we have another l parameters. Therefore $\dim \mathbf{S}_{k,\Delta} \leq k+l$. The remaining claim is that the $k+l$ functions in $\mathcal{B}$ are linearly independent. To prove this, let

$$s(t) := \sum_{i=0}^{k-1} a_i t^i + \sum_{i=1}^{l} c_i (t-t_i)_+^{k-1} = 0 \quad \text{for all} \quad t \in [a, b]\,.$$

By applying the linear functionals

$$G_i(f) := \frac{1}{(k-1)!}\big(f^{(k-1)}(t_i^+) - f^{(k-1)}(t_i^-)\big)$$

to s (where $f(t^+)$ and $f(t^-)$ denote the right, respectively, left-sided limits), then for all $i = 1, \ldots, l$, it follows that

$$0 = G_i(s) = \underbrace{G_i\Big(\sum_{j=0}^{k-1} a_j t^j\Big)}_{=0} + \sum_{j=1}^{l} c_j \underbrace{G_i(t-t_j)_+^{k-1}}_{=\delta_{ij}} = c_i\,.$$

Thus $s(t) = \sum_{i=0}^{k-1} a_i t^i = 0$ for all $t \in [a,b]$, and therefore also $a_0 = \cdots = a_{k-1} = 0$. □

However, the basis $\mathcal{B}$ of $\mathbf{S}_{k,\Delta}$ given in (7.27) has several disadvantages. For one, the basis elements are not local; e.g., the support of the monomials t^i is the whole of $\mathbf{R}$. Second, the truncated powers are "almost" linearly dependent for close nodes t_i, t_{i+1}. This results in the fact that the evaluation of a spline in the representation

$$s(t) = \sum_{i=0}^{k-1} a_i t^i + \sum_{i=1}^{l} c_i (t-t_i)_+^{k-1}$$

is poorly conditioned with respect to perturbations in the coefficients c_i. Third, the coefficients a_i and c_i have no geometric meaning—unlike the Bézier points b_i. In the following, we shall therefore construct a basis for the spline space $\mathbf{S}_{k,\Delta}$, which has properties as good as the Bernstein basis has for $\mathbf{P}_k$. In order to achieve this, we define recursively the following generalization of the characteristic function $\chi_{[\tau_i,\tau_{i+1}[}$ of an interval and also the "hat function" (see Figure 7.14).

Definition 7.49 Let $\tau_1 \le \cdots \le \tau_n$ be an arbitrary sequence of nodes. Then the *B-splines* $N_{ik}(t)$ of order k for $k = 1, \ldots, n$ and $i = 1, \ldots, n-k$ are recursively defined by

$$N_{i1}(t) := \chi_{[\tau_i,\tau_{i+1}[}(t) = \begin{cases} 1 & \text{if} \quad \tau_i \le t < \tau_{i+1} \\ 0 & \text{else} \end{cases}\,, \tag{7.28}$$

$$N_{ik}(t) := \frac{t-\tau_i}{\tau_{i+k-1}-\tau_i} N_{i,k-1}(t) + \frac{\tau_{i+k}-t}{\tau_{i+k}-\tau_{i+1}} N_{i+1,k-1}(t)\,. \tag{7.29}$$

Note that the characteristic function in (7.28) vanishes if the nodes coincide, i.e.,

$$N_{i1} = \chi_{[\tau_i,\tau_{i+1}[} = 0 \quad \text{if} \quad \tau_i = \tau_{i+1}\,.$$

The corresponding terms are omitted according to our convention $0/0 = 0$ in the recurrence relation (7.29). Thus, even if the nodes coincide, the B-splines N_{ik} are well-defined by (7.28) and (7.29); furthermore, $N_{ik} = 0$ if

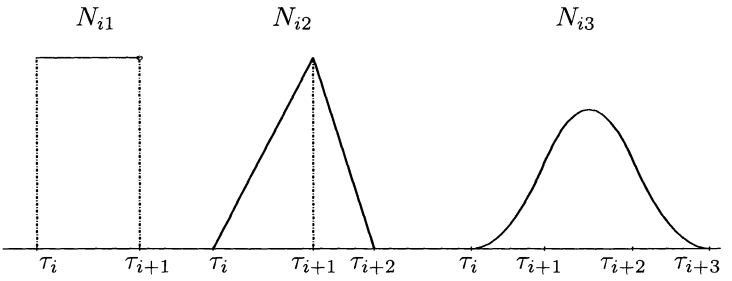

Figure 7.14. B-splines of order $k = 1, 2, 3$.

$\tau_i = \tau_{i+k}$. The following properties are obvious because of the recursive definition.

Remark 7.50 The B-splines satisfy

(a) $\text{supp} N_{ik} \subset [\tau_i, \ldots, \tau_{i+k}]$ (local support),

(b) $N_{ik}(t) \geq 0$ for all $t \in \mathbf{R}$ (nonnegative),

(c) N_{ik} is a piecewise polynomial of degree $\leq k-1$ with respect to the intervals $[\tau_j, \tau_{j+1}]$.

In order to derive further properties, it is convenient to represent the B-splines in closed form. In fact, they can be written as an application of a k^{th} divided difference $[\tau_i, \ldots, \tau_{i+k}]$ to the truncated powers $f(s) = (s-t)_+^{k-1}$.

Lemma 7.51 *If $\tau_i < \tau_{i+k}$, then the B-spline N_{ik} satisfies*

$$N_{ik}(t) = (\tau_{i+k} - \tau_i)[\tau_i, \ldots, \tau_{i+k}](\cdot - t)_+^{k-1} .$$

Proof. For $k = 1$, we obtain for the right-hand side

$$\begin{aligned} (\tau_{i+1} - \tau_i)[\tau_i, \tau_{i+1}](\cdot - t)_+^{k-1} &= (\tau_{i+1} - \tau_i)\frac{(\tau_i - t)_+^0 - (\tau_{i+1} - t)_+^0}{\tau_i - \tau_{i+1}} \\ &= \begin{cases} 1 & , \text{ if } \quad \tau_i \leq t < \tau_{i+1} \\ 0 & , \text{ else} \end{cases} . \end{aligned}$$

Furthermore, by employing the Leibniz formula (Lemma 7.15), it can easily be verified that the right-hand side also satisfies the recurrence relation (7.29). The statement now follows inductively. □

Corollary 7.52 *If τ_j is an m-fold node,i.e.,*

$$\tau_{j-1} < \tau_j = \cdots = \tau_{j+m-1} < \tau_{j+m} ,$$

then, at the position τ_j, N_{ik} is at least $(k-1-m)$-times continuously differentiable. The derivative of N_{ik} satisfies

$$N'_{ik}(t) = (k-1)\left(\frac{N_{i,k-1}(t)}{\tau_{i+k-1}-\tau_i} - \frac{N_{i+k,k-1}(t)}{\tau_{i+k}-\tau_{i+1}}\right).$$

Proof. The first statement follows from the fact that the divided difference $[\tau_i,\dots,\tau_{i+k}]f$ contains at most the $(m-1)^{\text{st}}$ derivative of the function f at the position τ_j. However, the truncated power $f(s) = (s-\tau_j)_+^{k-1}$ is $(k-2)$-times continuously differentiable. The second statement follows from

$$\begin{aligned} N'_{ik}(t) &= -(k-1)(\tau_{i+k}-\tau_i)[\tau_i,\dots,\tau_{i+k}](\cdot - t)_+^{k-2} \\ &= -(k-1)(\tau_{i+k}-\tau_i) \\ &\qquad \left(\frac{[\tau_{i+1},\dots,\tau_{i+k}](\cdot-t)_+^{k-2} - [\tau_i,\dots,\tau_{i+k-1}](\cdot-t)_+^{k-2}}{\tau_{i+k}-\tau_i}\right) \\ &= (k-1)\left(\frac{N_{i,k-1}(t)}{\tau_{i+k-1}-\tau_i} - \frac{N_{i+1,k-1}(t)}{\tau_{i+k}-\tau_{i+1}}\right). \end{aligned}$$

□

We now return to the space $\mathbf{S}_{k,\Delta}$ of splines of order k with respect to the grid $\Delta = \{t_j\}_{j=0,\dots,l+1}$:

$$\Delta : a = t_0 < t_1 < \cdots < t_{l+1} = b\,.$$

For the construction of the desired basis, to Δ we assign the following *extended sequence of nodes* $T = \{\tau_j\}_{j=1,\dots,n+k}$, where the boundary nodes $a = t_0$ and $b = t_{l+1}$ are counted k-times.

$$\begin{array}{lccccccccc} \Delta: & a & = & t_0 & < & t_1 & < \cdots < & t_{l+1} & = & b \\ & & & \| & & \| & & \| & & \\ T: & \tau_1 = \cdots & = & \tau_k & < & \tau_{k+1} & < \cdots < & \tau_{n+1} & = & \cdots = \tau_{n+k} \end{array}$$

Here $n = l + k = \dim \mathbf{S}_{k,\Delta}$ is the dimension of the spline space $\mathbf{S}_{k,\Delta}$. Consider the n B-splines N_{ik} for $i = 1,\dots,n$, which correspond to the extended sequence of nodes $T = \{\tau_j\}$. In the following, we shall see that they form the desired basis of $\mathbf{S}_{k,\Delta}$ (see Figure 7.15). To begin with, it is clear from Corollary 7.52 that the B-splines N_{ik} are indeed splines of order k, i.e.,

$$N_{ik} \in \mathbf{S}_{k,\Delta} \quad \text{for all} \quad i = 1,\dots,n\,.$$

Since the number n coincides with the dimension $n = \dim \mathbf{S}_{k,\Delta}$, it only remains to show that they are linearly independent. For this, we need the following technical statement, which is also known as *Marsden identity*.

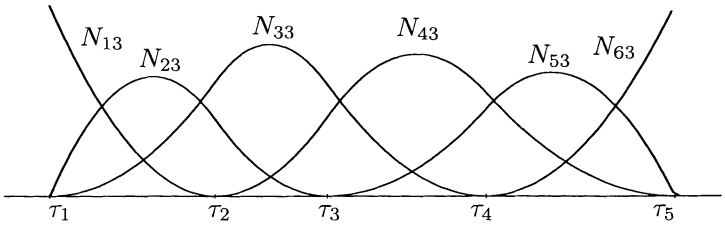

Figure 7.15. B-spline basis of order $k = 3$ (locally quadratic).

Lemma 7.53 *With the above notation, we have for all* $t \in [a, b]$ *and* $s \in \mathbf{R}$ *that*

$$(t-s)^{k-1} = \sum_{i=1}^{n} \varphi_{ik}(s) N_{ik}(t) \quad \textit{with} \quad \varphi_{ik}(s) := \prod_{j=1}^{k-1} (\tau_{i+j} - s)\,.$$

Proof. Because of the recursive definition of the B-splines, the proof is by induction over k. The statement is clear for $k = 1$, because of $1 = \sum_{i=1}^{n} N_{i1}(t)$. Thus let $k > 1$, and suppose that the statement is true for all $l \le k-1$. Insertion of the recurrence relation (7.29) on the right-hand side yields

$$\begin{aligned}
&\sum_{i=1}^{n} \varphi_{ik}(s) N_{ik}(t) \\
&= \sum_{i=2}^{n} \left(\frac{t-\tau_i}{\tau_{i+k-1}-\tau_i} \varphi_{ik}(s) + \frac{\tau_{i+k-1}-t}{\tau_{i+k-1}-\tau_i} \varphi_{i-1,k}(s) \right) N_{i,k-1}(t) \\
&= \sum_{i=2}^{n} \prod_{j=1}^{k-2} (\tau_{i+j} - s) \cdot \\
&\qquad \underbrace{\cdot \left(\frac{t-\tau_i}{\tau_{i+k-1}-\tau_i} (\tau_{i+k-1} - s) + \frac{\tau_{i+k-1}-t}{\tau_{i+k-1}-\tau_i} (\tau_i - s) \right)}_{= t-s} N_{i,k-1}(t) \\
&= (t-s) \sum_{i=2}^{n} \varphi_{i,k-1}(s) N_{i,k-1}(t) \\
&= (t-s)(t-s)^{k-2} = (t-s)^{k-1}\,.
\end{aligned}$$

Here note that the expression, which is "bracketed from below" is the linear interpolation of $t - s$, hence $t - s$ itself. □

Corollary 7.54 *The space* $\mathbf{P}_{k-1}[a, b]$ *of polynomials of degree* $\le k-1$ *over* $[a, b]$ *is contained in the space, which is spanned by the B-splines of order*

k, *i.e.*,

$$\mathbf{P}_{k-1}[a,b] \subset \ \mathit{span}\ (N_{1k},\ldots,N_{nk})\,.$$

In particular,

$$1 = \sum_{i=1}^{n} N_{ik}(t) \quad \mathit{for\ all}\ \ t \in [a,b]\,,$$

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

Proof. For the l^{th} derivative of the function $f(s) := (t-s)^{k-1}$, it follows from the Marsden identity that

$$f^{(l)}(0) = (k-1)\cdots(k-l)(-1)^l t^{k-l-1} = \sum_{i=1}^{n} \varphi_{ik}^{(l)}(0) N_{ik}(t)$$

and therefore, with $m = k-l-1$,

$$t^m = \frac{(-1)^{k-m-1}}{(k-1)\cdots(m+1)} \sum_{i=1}^{n} \varphi_{ik}^{(k-m-1)}(0) N_{ik}(t)\,.$$

The $(k-1^{\text{st}})$ derivative of ϕ_{ik} satisfies

$$\phi_{ik}^{k-1}(s) = \Big(\prod_{j=1}^{k-1}(\tau_{i+j}-s)\Big)^{k-1} = \big((-1)^{k-1}s^{k-1}+\cdots\big)^{k-1} = (-1)^{k-1}(k-1)!$$

and the second statement thus also follows. □

After these preparations, we can now prove the linear independence of the B-splines. They are *locally* independent as the following theorem shows.

Theorem 7.55 *The B-splines N_{ik} are locally linear independent, i.e., if*

$$\sum_{i=1}^{n} c_i N_{ik}(t) = 0 \quad \mathit{for\ all}\ \ t \in]c,d[\subset [a,b]$$

and $]c,d[\cap]\tau_i, \tau_{i+k}[\neq \phi$, *then*

$$c_i = 0\,.$$

Proof. Without loss of generality, we may assume that the open interval $]c,d[$ does not contain any nodes (otherwise we decompose $]c,d[$ into subintervals). According to Corollary 7.54, each polynomial of degree $\leq k-1$ over $]c,d[$ can be represented by the B-splines N_{ik}. However, only $k = \dim \mathbf{P}_{k-1}$ B-splines are different from zero on the interval $]c,d[$. They therefore have to be linearly independent. □

Let us summarize briefly what we have shown: The B-splines N_{ik} of order k with respect to the sequence of nodes $T = \{\tau_j\}$ form a basis

$\mathcal{B} := \{N_{1k}, \ldots, N_{nk}\}$ of the spline space $\mathbf{S}_{k,\Delta}$. They are locally linear independent, are locally supported, and form a positive partition of unity. Each spline $s \in \mathbf{S}_{k,\Delta}$ therefore has a unique representation as a linear combination of the form

$$s = \sum_{i=1}^{n} d_i N_{ik} .$$

The coefficients d_i are called *de Boor points* of s. The function values $s(t)$ are therefore convex combinations of the de Boor points d_i. For the evaluation, we can use the recursive definition of the B-splines N_{ik}, and we can therefore also derive the recurrence relation for the linear combinations themselves, which is given in Exercise 7.9, the *algorithm of de Boor*.

Remark 7.56 By employing the Marsden identity, one can explicitly give the dual basis $\mathcal{B}' = \{\nu_1, \ldots, \nu_n\}$ of the B-spline basis $\mathcal{B}$,

$$\nu_j : \mathbf{S}_{k,\Delta} \to \mathbf{R} \text{ linearly with } \nu_j(N_{ik}) = \delta_{ij} .$$

With this at hand, it can be shown that there is a constant D_k, which depends only on the order k such that

$$D_k \max_{j=1,\ldots,n} |d_j| \leq \| \sum_{j=1}^{n} d_j N_{jk} \|_\infty \leq \max_{j=1,\ldots,n} |d_j| .$$

Here the second inequality follows from the fact that the B-splines form a positive partition of unity. Perturbations in the function values $s(t)$ of the spline $s = \sum_{i=1}^{n} c_i N_{ik}$ and the coefficients can therefore be estimated against each other. In particular, the evaluation of a spline in B-spline representation is *well-conditioned*. Therefore, the basis is also called *well-conditioned*.

7.4.2 Spline Interpolation

We now turn our attention again toward the problem of interpolating a function f, which is given pointwise on a grid $\Delta = \{t_0, \ldots, t_{l+1}\}$,

$$a = t_0 < t_1 < \cdots < t_{l+1} = b .$$

In the linear case $k = 2$, the number $l + 2$ of nodes coincides with the dimension of the spline space $n = \dim \mathbf{S}_{2,\Delta} = l + k$. The linear B-splines N_{i2} with respect to the extended sequence of nodes

$$T = \{\tau_1 = \tau_2 < \cdots < \tau_{n+1} = \tau_{n+2}\} \text{ with } \tau_j = t_{j-2} \text{ for } j = 2, \ldots, n$$

satisfy $N_{i2}(t_j) = \delta_{j+1,i}$. The piecewise linear spline $I_2 f \in \mathbf{S}_{2,\Delta}$, which interpolates f, is therefore uniquely determined with

$$I_2 f = \sum_{i=1}^{n} f(t_{i-1}) N_{i2} .$$

Besides this very simple case of linear spline interpolation, the case $k = 4$ of cubic splines plays the most important role in the applications. In this case, we are missing two conditions to uniquely characterize the interpolating cubic spline $s \in \mathbf{S}_{i,\Delta}$, because

$$\dim \mathbf{S}_{4,\Delta} - \text{number of nodes} = l + k - l - 2 = 2\,.$$

Now the starting idea for the construction of spline functions was to find interpolating curves, which are as "smooth" as possible; we could also say "possibly least curved." The *curvature* of a parametric curve (in the plane)

$$y : [a, b] \to \mathbf{R}, \quad y \in C^2[a, b]$$

at $t \in [a, b]$ is given by

$$\kappa(t) := \frac{y''(t)}{(1 + y'(t)^2)^{3/2}}\,.$$

The absolute value of the curvature is just the reciprocal $1/r$ of the radius r of the osculating circle to the curve at the point $(t, y(t))$ (see Figure 7.16), i.e., the curvature is zero if and only if the osculating circle has the radius

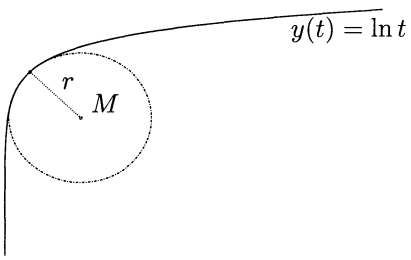

Figure 7.16. Osculating circle of the curve $(t, y(t))$.

∞, hence the curve is straight. In order to simplify this, instead of the curvature, we consider for small $y'(t)$ the reasonable approximation $y''(t)$,

$$\frac{y''(t)}{(1 + y'(t)^2)^{3/2}} \approx y''(t)\,,$$

and measure the curvature of the entire curve by the L_2-norm

$$\|y''\|_2 = \left(\int_a^b y''(t)^2 \, dt \right)^{\frac{1}{2}}$$

of this approximation with respect to $[a, b]$. The interpolating cubic splines, which satisfy the additional properties of Corollary 7.58, minimize this functional.

Theorem 7.57 *Let s be an interpolating cubic spline of f at the nodes $a = t_0 < \cdots < t_{l+1} = b$, and let $y \in C^2[a,b]$ be an arbitrary interpolating function of f such that*

$$[s(t)''(y(t)' - s(t)')]_{t=a}^{b} = 0\,. \tag{7.30}$$

Then

$$\|s''\|_2 \le \|y''\|_2\,. \tag{7.31}$$

Proof. Trivially, $y'' = s'' + (y'' - s'')$, and, inserted into the right-hand side of (7.31), it follows that

$$\begin{aligned}\int_a^b (y'')^2\,dt &= \int_a^b (s'')^2\,dt + \underbrace{2\int_a^b s''(y''-s'')\,dt}_{(*)} + \underbrace{\int_a^b (y''-s'')^2\,dt}_{\ge 0} \\ &\ge \int_a^b (s'')^2\,dt\,,\end{aligned}$$

if the term (*) vanishes. This holds true under the assumption (7.30), because by partial integration, it follows that

$$\int_a^b s''(y''-s'')\,dt = [s''(y'-s')]_a^b - \int_a^b s'''(y'-s')\,dt\,,$$

where s''' is in general discontinuous at the nodes $t_1, \ldots, t_{n-1}$, and is constant

$$s'''(t) = s_i'''(t) = d_i \quad \text{for} \quad t \in (t_i, t_{i+1})$$

in the interior of the subintervals (the s_i are cubic polynomials). Therefore, under the assumption (7.30), it is true that

$$\begin{aligned}\int_a^b s''(y''-s'')\,dt &= -\sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} d_i(y'-s_i')\,dt \\ &= -\sum_{i=1}^{n} d_i \int_{t_{i-1}}^{t_i} y' - s_i'\,dt \\ &= -\sum_{i=1}^{n} d_i[\underbrace{(y(t_i)-s(t_i))}_{=0} - \underbrace{(y(t_{i-1})-s(t_{i-1}))}_{=0}] \\ &= 0.\end{aligned}$$

□

Corollary 7.58 *In addition to the interpolation conditions $s(t_i) = f(t_i)$, assume that the cubic spline $s \in \mathbf{S}_{4,\Delta}$ satisfies one of the following* boundary conditions*:*

(i) $s'(a) = f'(a)$ *and* $s'(b) = f'(b)$,

(ii) $s''(a) = s''(b) = 0$,

(iii) $s'(a) = s'(b)$ *and* $s''(a) = s''(b)$ *(if* f *is periodic with period* $b - a$*).*

Then there exists a unique solution $s \in \mathbf{S}_{4,\Delta}$*, which satisfies this boundary condition. An arbitrary interpolating function* $y \in C^2[a,b]$*, which satisfies the same boundary condition, furthermore satisfies*

$$\|s''\|_2 \leq \|y''\|_2 .$$

Proof. The requirements are linear in s, and their number coincides with the dimension $n = l+4$ of the spline space $\mathbf{S}_{4,\Delta}$. It is therefore sufficient to show that the trivial spline $s \equiv 0$ is the only solution for the null-function $f \equiv 0$. Since $y \equiv 0$ satisfies all requirements, Theorem 7.57 implies that

$$\|s''\|_2 \leq \|y''\|_2 = 0 . \tag{7.32}$$

Since s'' is continuous, this implies $s'' \equiv 0$; i.e., s is a continuously differentiable, piecewise linear function with $s(t_i) = 0$, and is therefore the null-function. □

The three types (i), (ii), and (iii) are called *complete*, *natural*, and *periodic* cubic spline interpolation. The physical interpretation of the above minimization property (7.32) accounts for the name "spline." If $y(t)$ describes the position of a thin wooden beam, then

$$E = \int_a^b \left(\frac{y''(t)}{(1 + y'(t)^2)^{3/2}} \right)^2 dt$$

measures the "deformation energy" of the beam. Because of Hamilton's principle, the beam takes a position so that this energy is minimized. For small deformations one has approximately

$$E \approx \int_a^b y''(t)^2 \, dt = \|y''\|_2^2 .$$

The interpolating cubic spline $s \in \mathbf{S}_{4,\Delta}$ therefore describes approximately the position of a thin wooden beam, which is fixed at the nodes t_i. In the complete spline interpolation, we have clamped the beam at the boundary nodes with an additional prescription of the slopes. The natural boundary conditions correspond to the situation when the beam is straight outside the interval $[a,b]$. Such thin wooden beams were in fact used as drawing tools and are called "splines."

Note that besides the function values, two additional pieces of information regarding the original function f at the nodes enter in the complete spline interpolation. Thus their approximation properties (particularly at the boundary) are better than the ones of the other types (ii) and (iii). In

fact, the complete interpolating spline $I_4 f \in \mathbf{S}_{4,\Delta}$ approximates a function $f \in C^4[a,b]$ of the order h^4, where

$$h := \max_{i=0,\dots,l} |t_{i+1} - t_i|$$

is the largest distance of the nodes t_i. We state the following related result due to C. A. Hall and W. W. Meyer [48] without proof.

Theorem 7.59 *Let $I_4 f \in \mathbf{S}_{4,\Delta}$ be the complete interpolating spline of a function $f \in C^4[a,b]$ with respect to the nodes t_i with $h := \max_i |t_{i+1} - t_i|$. Then*

$$\|f - I_4 f\|_\infty \le \frac{5}{384} h^4 \|f^{(4)}\|_\infty .$$

Note that this estimate is independent of the position of the nodes t_i.

7.4.3 Computation of Cubic Splines

In the following, we shall derive a system of equations for the cubic interpolating splines. For this purpose, we describe the spline $s \in \mathbf{S}_{4,\Delta}$ by employing the local Bézier representation

$$s(t) = s_i(t) = \sum_{j=0}^{3} b_{3i+j} B_j^3(t; t_i, t_{i+1}) \quad \text{for} \quad t \in [t_i, t_{i+1}] \tag{7.33}$$

of the partial polynomials s_i with respect to the intervals $[t_i, t_{i+1}]$. Here

$$B_j^3(t; t_i, t_{i+1}) = B_j^3\left(\frac{t - t_i}{h_i}\right) \quad \text{with} \quad h_i := t_{i+1} - t_i .$$

The continuity of s enters implicitly into the representation (7.33). By

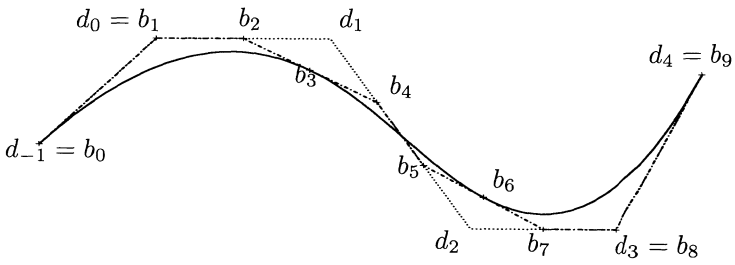

Figure 7.17. Cubic spline with de Boor points d_i and Bézier points b_i.

(7.26), the C^1-smoothness implies that

$$b_{3i} = \frac{h_i}{h_{i-1} + h_i} b_{3i-1} + \frac{h_{i-1}}{h_{i-1} + h_i} b_{3i+1} . \tag{7.34}$$

Furthermore, according to the C^2-smoothness of s, we have shown that there are de Boor points d_i such that

$$\begin{aligned} b_{3i+2} &= -\frac{h_i}{h_{i-1}} d_i + \frac{h_{i-1}+h_i}{h_{i-1}} b_{3i+1} \\ b_{3i-2} &= \frac{h_{i-1}+h_i}{h_i} b_{3i-1} - \frac{h_{i-1}}{h_i} d_i \,. \end{aligned}$$

Graphically, this means that the straight line segment between b_{3i-2} and d_i, respectively, d_i and b_{3i+2} is partitioned at the ratio $h_{i-1} : h_i$ by the Bézier points b_{3i-1}, respectively, b_{3i+1}. The points d_i, b_{3i+1}, b_{3i+2} and d_{i+1} are therefore positioned as shown in Figure 7.17. Taken together, this implies

$$\begin{aligned} b_{3i+1} &= \frac{h_i + h_{i+1}}{h_{i-1}+h_i+h_{i+1}} d_i + \frac{h_{i-1}}{h_{i-1}+h_i+h_{i+1}} d_{i+1} \\ b_{3i-1} &= \frac{h_{i-2} + h_{i-1}}{h_{i-2}+h_{i-1}+h_i} d_i + \frac{h_i}{h_{i-2}+h_{i-1}+h_i} d_{i-1} \end{aligned}$$

If we define at the boundary $h_{-1} := h_{l+1} := 0$ and

$$d_0 := b_1, \quad d_{-1} := b_0, \quad d_{l+1} := b_{3l+2} \quad \text{and} \quad d_{l+2} := b_{3(l+1)}, \tag{7.35}$$

then the Bézier coefficients b_{3i+j}, and thus also the spline s, are completely determined by the $l+4$ points d_{-1} to d_{l+2} and the equations (7.34) to (7.35).

By inserting the interpolation conditions

$$f_i = s(t_i) = b_{3i} \quad \text{for} \quad l = 0, \ldots, l+1,$$

then it follows at the boundary that

$$d_{-1} := f_0 \quad \text{and} \quad d_{l+2} = f_{l+1} \,.$$

The remaining points $d_0, \ldots, d_{l+1}$ of the interpolating spline must solve the following system (proof as an exercise):

$$\begin{bmatrix} 1 & & & & & \\ \alpha_1 & \beta_1 & \gamma_1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \alpha_l & \beta_l & \gamma_l & \\ & & & & 1 & \end{bmatrix} \begin{pmatrix} d_0 \\ \vdots \\ \vdots \\ \vdots \\ d_{l+1} \end{pmatrix} = \begin{pmatrix} b_1 \\ (h_0+h_1) f_1 \\ \vdots \\ (h_{l-1}+h_l) f_l \\ b_{3l+2} \end{pmatrix}$$

with

$$\begin{aligned} \alpha_i &:= \frac{h_i^2}{h_{i-2}+h_{i-1}+h_i} \\ \beta_i &:= \frac{h_i(h_{i-2}+h_{i-1})}{h_{i-2}+h_{i-1}+h_i} + \frac{h_{i-1}(h_i+h_{i+1})}{h_{i-1}+h_i+h_{i+1}} \\ \gamma_i &:= \frac{h_{i-1}^2}{h_{i-1}+h_i+h_{i+1}} \,. \end{aligned}$$

We now only have to determine the Bézier points b_1 and b_{3l+2} from the boundary conditions. We confine ourselves to the first two types. For the *complete spline interpolation*, we obtain from

$$f'_0 = s'(a) = \frac{3}{h_0}(b_1 - b_0) \quad \text{and} \quad f'_{l+1} = s'(b) = \frac{3}{h_l}(b_{3(l+1)} - b_{3l+2})$$

that we have to set

$$b_1 = \frac{h_0}{3} f'_0 + f_0 \quad \text{and} \quad b_{3l+2} = -\frac{h_l}{3} f'_{l+1} + f_{l+1}.$$

For the *natural boundary conditions* we have to choose b_1 and b_{3l+2}, so that $s''(a) = s''(b) = 0$. This is satisfied for

$$b_1 := b_0 = f_0 \quad \text{and} \quad b_{3l+2} := b_{3(l+1)} = f_{l+1}$$

(see Figure 7.18).

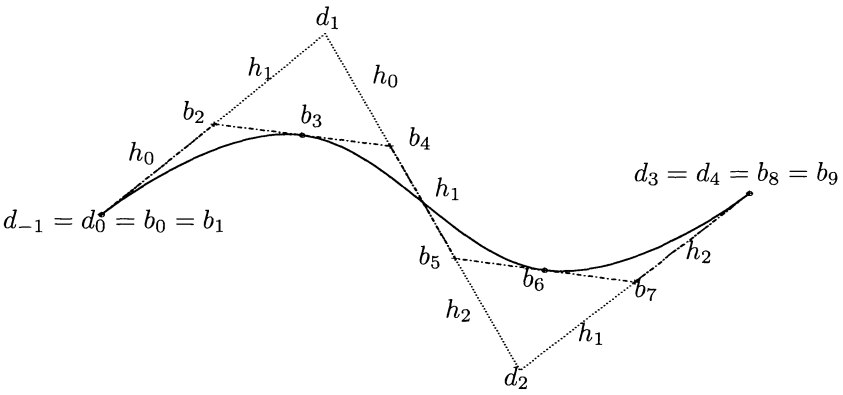

Figure 7.18. Cubic interpolating spline with natural boundary conditions.

Remark 7.60 For an equidistant grid, i.e., $h_i = h$ for all i, we have

$$\alpha_i = \gamma_i = \frac{h}{3} \quad \text{and} \quad \beta_i = \frac{4h}{3} \quad \text{for} \quad i = 2, \ldots, l-1\,.$$

In this case (and also for an almost equidistant grid) the matrix is strictly diagonally dominant, and it can therefore be solved efficiently and in a stable manner by the Gaussian elimination without interchanging columns.

Remark 7.61 The de Boor points d_i are just the B-spline coefficients of the interpolating cubic splines, i.e.,

$$s = \sum_{i=1}^{l+4} d_{i-2} N_{i4}\,,$$

if, as above, N_{i4} are the B-splines for the extended sequence of nodes $T = \{\tau_j\}$.

Exercises

Exercise 7.1 Let $\Lambda_n(K, I)$ denote the Lebesgue constant with respect to the set of nodes K on the interval I.

(a) Let $K = \{t_0, \ldots, t_n\} \subset I = [a, b]$ be pairwise distinct nodes. Suppose that the affine transformation

$$\chi : I \to I_0 = [-1, 1], \quad t \mapsto \frac{2t - a - b}{b - a}$$

of this interval onto the unit interval I_0 maps the set of nodes K onto the set of nodes $K_0 = \chi(K)$. Show that the Lebesgue constant is invariant under this transformation, i.e.,

$$\Lambda_n(K, I) = \Lambda_n(K_0, I_0)\,.$$

(b) Let $K = \{t_0, \ldots, t_n\}$ with $a \le t_0 < t_1 < \cdots < t_n \le b$ be nodes in the interval $I = [a, b]$. Give the affine transformation

$$\chi : [t_0, t_n] \to I$$

on I, which satisfies the property that for $\bar{K} = \chi(K) = \{\bar{t}_0, \ldots, \bar{t}_n\}$:

$$a = \bar{t}_0 < \bar{t}_1 < \cdots < \bar{t}_n = b\,.$$

Show that

$$\Lambda_n(\bar{K}, I) \le \Lambda_n(K, I)\,;$$

i.e., inclusion of the boundary nodes improves the Lebesgue constant.

Exercise 7.2 Consider the class of functions

$$\mathcal{F} := \left\{ f \in C^{n+1}[-1, 1] \;\middle|\; \|f^{(n+1)}\|_\infty \le (n+1)! \right\}\,.$$

For $f \in \mathcal{F}$, let $p_n(f)$ denote the polynomial of degree n of the (Hermite) interpolation for the nodes $K = \{t_0, \ldots, t_n\} \subset I_0 = [-1, 1]$.

(a) Show that

$$\varepsilon_n(K) := \sup_{f \in \mathcal{F}} \|f - p_n(f)\|_\infty = \|\omega_{n+1}\|_\infty\,,$$

where $\omega_{n+1}(t) = (t - t_0) \cdots (t - t_n)$.

(b) Show that $\varepsilon_n(K) \ge 2^{-n}$ and that equality holds if and only if K is the set of the Chebyshev nodes, i.e.,

$$t_j = \cos \frac{2j + 1}{2n + 2} \pi \quad \text{for } j = 0, \ldots, n\,.$$

Exercise 7.3 Count how many computations and how much storage space an economically written program requires for the evaluation of interpolation polynomials on the basis of the Lagrange representation.

Compare with the algorithms of Aitken-Neville and the representation over Newton's divided differences.

Exercise 7.4 Let $a = t_0 < t_1 < \cdots < t_{n-1} < t_n = b$ be a distribution of nodes in the interval $I = [a, b]$. For a continuous function $g \in C(I)$, the *interpolating polygon* $\mathcal{I}g \in C(I)$ is defined by

(a) $\mathcal{I}g(t_i) = g(t_i)$ for $i = 0, \ldots, n$,

(b) $\mathcal{I}g\big|_{[t_i, t_{i+1}]}$ is a polynomial of degree one for $i = 0, \ldots, n-1$.

Show the following:

(a) Any function $g \in C^2(I)$ satisfies

$$\|g - \mathcal{I}g\|_\infty \leq \frac{h^2}{8} \|g''\|_\infty ,$$

where $h = \max\limits_{0 \leq i \leq n-1} (t_{i+1} - t_i)$ is the "grid-width parameter."

(b) The absolute condition of the polygonal interpolation satisfies

$$\kappa_{\text{abs}} = 1 .$$

Discuss and evaluate the difference between this and the polynomial interpolation.

Exercise 7.5 For the approximation of the first derivative of a pointwise given function f, one utilizes the first divided difference

$$(D_h f)(x) := [x, x + h] f .$$

(a) Estimate the approximation error $|D_h f(x) - f'(x)|$ for $f \in C^3$. (Leading order in h for $h \to 0$ is sufficient.)

(b) Instead of $D_h f(x)$, the floating point arithmetic computes $\hat{D}_h f(x)$. Estimate the error $|\hat{D}_h f(x) - D_h f(x)|$ in leading order.

(c) Which h turns out to be optimal, i.e., minimizes the total error?

(d) Test your prediction at $f(x) = e^x$ at the position $x = 1$ with

$$h = 10^{-1},\ 5 \cdot 10^{-2},\ 10^{-2}, \ldots,\ \text{eps} .$$

Exercise 7.6 Show that the derivatives of a polynomial in Bézier representation with respect to the interval $[t_0, t_1]$

$$P(t) = \sum_{i=0}^{n} b_i B_i^n(\lambda), \quad \lambda := \frac{t - t_0}{t_1 - t_0},$$

are given by

$$\frac{d^k}{dt^k}P(t) = \frac{n!}{(n-k)!h^k}\sum_{i=0}^{n-k}\Delta^k b_i B_i^{n-k}(\lambda), \quad h := t_1 - t_0.$$

Exercise 7.7 Find the Bézier representation with respect to $[0,1]$ of the Hermite polynomials H_i^3 for the nodes t_0, t_1, and sketch the Hermite polynomials together with the Bézier polygons.

Exercise 7.8 We have learned three different bases for the space $\mathbf{P}_3$ of polynomials of degree ≤ 3: the monomial basis $\{1, t, t^2, t^3\}$, the Bernstein basis $\{B_0^3(t), B_1^3(t), B_2^3(t), B_3^3(t)\}$ with respect to the interval $[0,1]$, and the Hermite basis $\{H_0^3(t), H_1^3(t), H_2^3(t), H_3^3(t)\}$ for the nodes t_0, t_1. Determine the matrices for the basis changes.

Exercise 7.9 Show that a spline $s = \sum_{i=1}^n d_i N_{ik}$ in B-spline representation with respect to the nodes $\{\tau_i\}$ satisfies the following recurrence relation:

$$s(t) = \sum_{i=l+1}^{n} d_i^l(t) N_{i,k-1}(t)\,.$$

Here the d_i^l are defined by $d_i^0(t) := d_i$ and

$$d_i^l(t) := \begin{cases} \dfrac{t-\tau_i}{\tau_{i+k-l}-\tau_i} d_i^{l-1}(t) + \dfrac{\tau_{i+k-l}-t}{\tau_{i+k-l}-\tau_i} d_{i-1}^{l-1}(t) & \text{if } \tau_{i+k-l} \neq \tau_i \\ 0 & \text{else} \end{cases}$$

for $l > 0$. Show that $s(t) = d_i^{k-1}(t)$ for $t \in [\tau_i, \tau_{i+1}]$. Use this to derive a scheme for the computation of the spline $s(t)$ through continued convex combination of the coefficients d_i (algorithm of de Boor).