# Chapter 4
# Interpolation and Approximation

*Far better an approximate answer to the right question,*
*which is often vague,*
*than an exact answer to the wrong question,*
*which can always be made more precise.*
—John W. Tukey

## 4.1 The Interpolation Problem

### 4.1.1 Introduction

Polynomials are used as the basic means of approximation in nearly all areas of numerical analysis. We have encountered in Sec. 3.3.4 the problem of interpolating the values of a function $f(x)$ in *n equidistant* points by a polynomial $p(x) \in \mathcal{P}_n$.[126] We have also studied application of polynomial approximations to numerical differentiation and integration. It is de facto so, although the polynomials are invisible in the derivations of formulas by operator methods. In the following sections we shall go deeper into the nonequidistant polynomial interpolation problem.

*Let $a = x_1 < x_2 < \cdots < x_n = b$ be a grid of distinct points $x_i$. Find a polynomial $p \in \mathcal{P}_n$ such that*

$$p(x_i) = f(x_i), \quad i = 1 : n. \tag{4.1.1}$$

By Theorem 3.3.4, the interpolation polynomial *p is uniquely determined.* This theorem is general, although the rest of Sec. 3.3 dealt with interpolation polynomials in the equidistant case only and their application to numerical differentiation. Note that the formulation and the solution of this problem are *independent of the ordering of the points $x_i$.*

[126] Recall the definition of $\mathcal{P}_n$ as the space of polynomials in one variable of degree *less than n*; the dimension of the linear space $\mathcal{P}_n$ is $n$.

### 4.1.2 Bases for Polynomial Interpolation

A set of polynomials $\{p_1(x), p_2(x), \ldots, p_n(x)\}$ such that any polynomial $p \in \mathcal{P}_n$ can be expressed as a linear combination

$$p(x) = \sum_{j=1}^{n} c_j p_j(x)$$

is called a basis in $\mathcal{P}_n$. The column vector $c = (c_1, c_2, \ldots, c_n)^T$ can be viewed as a *coordinate vector* of $p$ in the space $\mathcal{P}_n$, with respect to this basis. The interpolation problem (4.1.1) leads to a linear system of equations

$$c_1 p_1(x_i) + c_2 p_2(x_i) + \cdots + c_n p_n(x_i) = f(x_i), \quad i = 1 : n. \tag{4.1.2}$$

If we introduce the matrix

$$P_n = [p_j(x_i)]_{i,j=1}^{n}, \tag{4.1.3}$$

and the column vector $f = (f(x_1), f(x_2), \ldots, f(x_n))^T$, then the linear system becomes

$$P_n c = f. \tag{4.1.4}$$

Mathematically, the choice of basis (for a finite-dimensional space) makes no difference. Computationally, working with *rounded values of the coefficients*, the choice of basis can make a great difference. If the purpose is to compute derivatives or integrals of the interpolation polynomial, the power basis or the **shifted power basis**, where $p_j(x) = (x-c)^{j-1}$, i.e.,

$$p(x) = \sum_{j=1}^{n} c_j (x - c)^{j-1},$$

is convenient although not always the best. If a shifted power basis is to be used for polynomial approximation on an interval $[a, b]$, it is often best to choose $c = (a + b)/2$, i.e., equal to the midpoint of the interval.

For the **power basis** $p_j(x) = x^{j-1}$, the coefficients of the interpolation polynomial are given by the solution of the linear system $V_n^T c = f$, where $V_n$ is the Vandermonde matrix

$$V_n = [x_j^{i-1}]_{i,j=1}^{n} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{pmatrix}. \tag{4.1.5}$$

By Theorem 3.3.4 this matrix is nonsingular, since the Vandermonde determinant equals (see (3.3.12))

$$\det(V_n) = \prod_{1 \le i < j \le n} (x_i - x_j).$$

Let $\{p_1(x), p_2(x), \ldots, p_n(x)\}$ and $\{q_1(x), q_2(x), \ldots, q_n(x)\}$ be two bases for $\mathcal{P}_n$. Then the $q_j$ must be linear combinations of the $p_k$, $k = 1 : n$. This can be expressed in vector-matrix form:

$$\big(q_1(x), q_2(x), \ldots, q_n(x)\big) = \big(p_1(x), p_2(x), \ldots, p_n(x)\big) S \quad \forall x, \tag{4.1.6}$$

where $S$ is a constant matrix. *$S$ must be nonsingular*, since if $S$ were singular, then there would exist a nontrivial vector $v$ such that $Sv = 0$, hence

$$(q_1(x), q_2(x), \ldots, q_n(x))v = (p_1(x), p_2(x), \ldots, p_n(x))Sv = 0 \quad \forall x,$$

and $(q_1(x), q_2(x), \ldots, q_n(x))$ would thus not be a basis.

Let $P_n = [p_j(x_i)]_{i,j=1}^n$ and $Q_n = [q_j(x_i)]_{i,j=1}^n$. By putting $x = x_i$, $i = 1 : m$, into (4.1.6), we see that $Q_n = P_n S$, and $Q_n$ is nonsingular for every basis. If we set $p(x) = \sum d_j q_j(x)$, the system (4.1.2) becomes for this basis $Q_n d = f$, and then

$$P_n c = f = Q_n d = P_n S d, \qquad c = P_n^{-1} f = Sd. \tag{4.1.7}$$

The matrix $S$ for the transformation between representations is thus like a coordinate transformation in geometry. Matrix representations of various common bases transformations are given by Gander [130].

The power basis has a bad reputation which is related to the ill-conditioning of the corresponding Vandermonde matrix; see Sec. 4.1.3. There are other bases in $\mathcal{P}_n$ which are often more advantageous to use. By a **triangle family** of polynomials we mean a sequence of polynomials

$$\begin{aligned} q_1(x) &= s_{11}, \\ q_2(x) &= s_{12} + s_{22}x, \\ q_3(x) &= s_{13} + s_{23}x + s_{33}x^2, \\ &\cdots \\ q_n(x) &= s_{1n} + s_{2n}x + s_{3n}x^2 + \cdots + s_{nn}x^{n-1}, \end{aligned} \tag{4.1.8}$$

where $s_{jj} \neq 0$ for all $j$. Note that the coefficients form a lower triangular matrix $S$.

Conversely, for any $j$, $p_j(x) = x^{j-1}$ can be expressed recursively and uniquely as linear combinations of $q_1(x), \ldots, q_j(x)$. We obtain a triangular scheme also for the inverse transformation:

$$\begin{aligned} 1 &= t_{11}q_1(x), \\ x &= t_{12}q_1 + t_{22}q_2, \\ x^2 &= s_{13}q_1 + t_{23}q_2 + t_{33}q_3, \\ &\cdots \\ x^n &= t_{1n}q_1 + t_{2n}q_2 + t_{3n}q_3 + \cdots + t_{nn}q_n, \end{aligned} \tag{4.1.9}$$

where $t_{jj} \neq 0$ for all $j$, and the coefficients form a lower triangular matrix $T = S^{-1}$. Thus *every triangle family is a basis for $\mathcal{P}_m$*. (Recall the well-known fact that the inverse of a triangular matrix with nonzero diagonal exists and is triangular.) Among interesting triangle families are the shifted power basis $(x - c)^j$, the Chebyshev polynomials $T_j(x)$, and many other families of orthogonal polynomials.

A triangle family which is often very convenient for solving the interpolation problem is the family of **Newton polynomials**

$$p_1(x) = 1, \qquad p_j(x) = (x - x_1)(x - x_2)\ldots(x - x_{j-1}), \quad j = 2 : n, \tag{4.1.10}$$

which has unit leading coefficients. Since $p_j(x_k) = 0$, if $k < j$ we obtain, using the representation

$$p(x) = c_1 p_1 + c_2 p_2(x) + c_3 p_3(x) + \cdots + c_n p_n(x), \tag{4.1.11}$$

lower triangular system $Lc = f$ for the coefficients, where

$$L = \begin{pmatrix} 1 & & & & \\ 1 & (x_2 - x_1) & & & \\ 1 & (x_3 - x_1) & (x_3 - x_1)(x_3 - x_2) & & \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & (x_n - x_1) & (x_n - x_1)(x_n - x_2) & \cdots & \prod_{j=1}^{n-1}(x_n - x_j) \end{pmatrix}. \tag{4.1.12}$$

Hence the coefficients can be computed by forward substitution. In the next section we shall see how this basis leads to Newton's interpolation formula. This is one of the best interpolation formulas with respect to flexibility, computational economy, and numerical stability.

If a polynomial $p(x)$ is given in the form (4.1.11), then it can be evaluated using only $n$ multiplications and $2n$ additions for a given numerical value $x$ using the nested form

$$\begin{aligned} p(x) = (\cdots (c_n(x - x_{n-1}) + c_{n-1})(x - x_{n-2}) \\ + \cdots + c_3)(x - x_2) + c_2)(x - x_1) + c_1. \end{aligned}$$

This can be evaluated by a recursion formula similar to Horner's rule (see Sec. 1.2.2).

Another basis of $\mathcal{P}_n$ that has many advantages is the Lagrange basis of polynomials $\ell_j(x)$. If $x_i$, $i = 1 : n$, are distinct interpolation points, these are

$$\ell_j(x) = \prod_{\substack{i=1 \\ i \neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)}, \quad j = 1 : n. \tag{4.1.13}$$

These bases polynomials of degree $n - 1$ satisfy

$$\ell_j(x_i) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{4.1.14}$$

From this property directly follows **Lagrange's interpolation formula**, which directly displays the solution of the interpolation problem for $n$ distinct points.[127]

**Theorem 4.1.1.**

*The unique interpolation polynomial $p \in \mathcal{P}_n$ interpolating the function $f$ at the distinct points $x_i$, $i = 1 : n$, can be written*

$$p(x) = \sum_{j=1}^{n} f(x_j) \prod_{\substack{i=1 \\ i \neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)}. \tag{4.1.15}$$

[127] Lagrange published this formula in 1794.

It is an easy exercise to show that by l'Hôpital's rule the Lagrange polynomials can be written

$$\ell_j(x) = \frac{\Phi_n(x)}{(x - x_j)\Phi_n'(x_j)}, \qquad \Phi_n(x) = \prod_{i=1}^{n}(x - x_i). \tag{4.1.16}$$

This property characterizes what, in a more general context, is known as a **cardinal basis**.

Lagrange's interpolation formula has been widely regarded as being more suitable for deriving theoretical results than for practical computation. However, in Sec. 4.2.3 two modified forms of Lagrange's interpolation formula will be given, which are also very attractive computationally.

A natural extension of the interpolation problem is to determine a polynomial $p(x) = \sum_{j=1}^{n} c_j p_j(x) \in \mathcal{P}_n$ that, in some sense, best fits to the data $(x_i, f(x_i))$, $i = 1 : m$, where $m > n$. Since the number of data points is larger than the number of parameters, the corresponding linear system $Pc = f$ is overdetermined and can typically be satisfied only approximately. Overdetermination can be used to attain two different types of **smoothing**:

(i) to reduce the effect of random or other irregular errors in the values of the function;

(ii) to give the polynomial a smoother behavior between the grid points.

In least squares approximation one determines the coefficient vector $c$ so that the sum of squared residuals

$$S(c) = \sum_{i=1}^{m}(p(x_i) - f(x_i))^2 \tag{4.1.17}$$

is minimized; see Sec. 1.4. This can in many applications be motivated by statistical arguments; see the Gauss–Markov theorem (Theorem 1.4.1). It also leads to rather simple computations. The conditions for the minimization are

$$\frac{\partial S(c)}{\partial c_k} = 2\sum_{i=1}^{m} p_k(x_i)\big(p(x_i) - f(x_i)\big) = 0, \quad k = 1 : n.$$

A stable method for discrete least squares polynomial approximation, based on using a basis of orthogonal polynomials, will be given in Sec. 4.5.5.

We mention here that a large part of the theory of polynomial interpolation and approximation is valid also for *generalized polynomials*

$$u(x) = \sum_{k=1}^{n} a_k u_k(x),$$

where $\{u_1, u_2, \ldots, u_n\}$ are continuous real-valued functions that form a **Chebyshev system** on a closed finite interval $[a, b]$; see Sec. 4.5.7.

### 4.1.3 Conditioning of Polynomial Interpolation

Let $p \in \mathcal{P}_n$ be the unique polynomial that interpolates the function values $f$ at the distinct points $x_j$, $j = 1 : n$. Consider the problem to determine the value of $p$ at the fixed point $\tilde{x}$.

With the terminology of Sec. 2.4.3 the input is the vector $f = (f_1, \ldots, f_n)^T \in \mathbf{R}^n$ and the output is the scalar $p(\tilde{x})$.

Using the Lagrange basis (4.1.13) we have

$$p(\tilde{x}) = \sum_{j=1}^{n} f_j \ell_j(\tilde{x}), \qquad \ell_j(\tilde{x}) = \prod_{\substack{i=1 \\ i \neq j}}^{n} \frac{(\tilde{x} - x_i)}{(x_j - x_i)}.$$

If the data are perturbed by $\Delta f$, where $|\Delta f| \leq \epsilon$, then

$$|\Delta p(\tilde{x})| \leq \epsilon \sum_{j=1}^{n} |\ell_j(\tilde{x})| \tag{4.1.18}$$

gives an upper bound for the perturbation in $p(\tilde{x})$.

The Lebesgue constant for polynomial interpolation at distinct points $x_i \in [a, b]$, $i = 1 : n$, is defined as

$$\Lambda_n = \max_{a \leq x \leq b} \sum_{j=1}^{n} |\ell_j(x)|. \tag{4.1.19}$$

From (4.1.18) it follows that $\Lambda_n$ can be interpreted as the condition number for the interpolation problem with $\tilde{x} \in [a, b]$. We have the inequality

$$\max_{a \leq x \leq b} |p(x)| \leq \Lambda_n \max_{a \leq x \leq b} |f(x)|,$$

where equality can be obtained for $f(x)$ piecewise linear and such that $f(x_i) = \operatorname{sign} \ell_i(x_i)$.

For equally spaced points $\Lambda_n$ grows at a rate proportional to $2^n/(n \log n)$; see Cheney and Light [67, Chap. 3]. For the Chebyshev points in $[-1, 1]$, on the other hand,

$$\Lambda_n \leq \frac{2}{\pi} \log(n) + 1.$$

The basis consisting of the Lagrange polynomials for Chebyshev nodes is optimally conditioned among all Lagrangian bases. It is indeed optimal among all polynomial bases in the sense of attaining the optimal growth rate $O(\log n)$; see Gautschi [144].

Usually the interpolation problem is solved in two steps. First, a suitable basis $p_j(x)$, $j = 1 : n$, for the space $\mathcal{P}_n$ is chosen and the coefficients $c_j$ in the expansion

$$p(x) = \sum_{j=1}^{n} c_j p_j(x) \tag{4.1.20}$$

are determined. Second, the right-hand side of (4.1.20) is evaluated at some point $x \in [a, b]$. Then we also need to consider the condition number $\kappa(P_n)$ of the matrix $P_n$ which maps the coefficient vector $c = (c_1, \ldots, c_n)^T$ to $f$.

For the power basis $P_n = V_n^T$. Many bounds and asymptotic estimates of $\kappa(V_n)$ are known; see [147, Sec. 1.3.2]; [199, Sec. 22.1]. For equidistant points $x_i = -1 + 2(i - 1)/(n - 1)$ on $[-1, 1]$, it holds that

$$\kappa_\infty(V_n) = \|V_n^{-1}\|_\infty \|V_n\|_\infty \sim \pi^{-1} e^{\pi/4} (3.1)^n; \tag{4.1.21}$$

e.g., $\kappa_\infty(V_{20}) \approx 1.05 \cdot 10^9$. Other point distributions are even worse. For the harmonic points $x_i = 1/i$, $i = 1 : n$, we have $\kappa_\infty(V_n) > n^{n+1}$, which is faster than exponential growth! Surprisingly, some Vandermonde systems, which are so ill-conditioned that Gaussian elimination with pivoting fails to produce a single correct digit, can be solved to full relative accuracy by a fast algorithm given in Sec. 4.2.5.

For the **Chebyshev points** on $[-1, 1]$

$$x_i = \cos\Big(\frac{2i-1}{n}\frac{\pi}{2}\Big), \quad i = 1 : n, \tag{4.1.22}$$

i.e., the zeros of $T_{n-1}(x)$, the Vandermonde matrix is much better conditioned and

$$\kappa_\infty(V) \sim 0.253^{3/4}(1+\sqrt{2})^n. \tag{4.1.23}$$

In contrast to the power basis the condition of bases of orthogonal polynomials exhibits only polynomial growth in $n$. For Chebyshev polynomials $p_j = T_j$ on $[-1, 1]$ it holds that

$$\kappa(P_n) \le \sqrt{2}\,n,$$

which is a big improvement on the power basis.

It should be stressed that $\kappa(P_n)$ measures only the sensitivity of the coefficients $c_i$ in the polynomial $p(x) = \sum_{j=1}^n c_j x^{j-1}$ to perturbations in the given data $f_i$. It is possible that even when these coefficients are inaccurately determined, the interpolation polynomial $p(x)$ does still reproduce the true interpolation polynomial well. For further discussion of these points, see Sec. 4.2.5 and [176].

## Review Questions

**4.1.1** The interpolation problem in $\mathcal{P}_n$ leads to a linear system $V^T c = f$, where $V$ is a Vandermonde matrix. Write down the expression for the element $v_{ij}$.

**4.1.2** What is meant by the method of undetermined coefficients? Give an example.

**4.1.3** What is meant by a triangle family $q_1(x), q_2(x), \ldots, q_n(x)$ of polynomials? Are all such families a basis for $\mathcal{P}_n$?

**4.1.4** What property characterizes a cardinal basis for $\mathcal{P}_n$?

**4.1.5** What good effects can be achieved by using overdetermination in polynomial interpolation?

**4.1.6** How is the Lebesgue constant defined and what is its significance for the conditioning of the polynomial interpolation problem?

## Problems and Computer Exercises

**4.1.1** (a) Study experimentally interpolation in $\mathcal{P}_n$, $n = 2 : 2 : 16$, for $f(x) = (3+x)^{-1}$, $x \in [-1, 1]$. Use the linear system $V^T c = f$ and the power basis. Study both

equidistant points and Chebyshev points

$$x_i = -1 + 2\frac{i-1}{n-1}, \qquad x_i = \cos\Big(\frac{2i-1}{n}\frac{\pi}{2}\Big), \quad i = 1:n,$$

respectively. Plot the error curve $y = |f(x) - p(x)|$ in semilogarithmic scale. For the larger values of $n$, also conduct experiments to illuminate the effects of random perturbations of the function values on the values of $p(x)$.

(b) Also do a few experiments with a random vector $f$, for $n = 16$ and $n = 8$, in order to compare the grid data and the order of magnitude of $p(x)$ between the grid points.

**4.1.2** *A warning for polynomial extrapolation of empirical functions, or . . .?*

(a) Write a program $c = polyapp(x, y, n)$ that finds the coefficient vector $c$ for a polynomial in $p \in \mathcal{P}_n$, in a shifted power basis, such that $y_i \approx p(x_i)$, $i = 1:m$, $m \geq n$, in the least squares sense, or study a program that does almost this.

(b) The following data show the development of the Swedish gross domestic production (GDP), quoted from a table made by a group associated with the Swedish Employer's Confederation. (The data are expressed in prices of 1985 and scaled so that the value for 1950 is 100.)

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1950 | 1955 | 1960 | 1965 | 1970 | 1975 | 1980 | 1985 | 1990 |
| 100.0 | 117.7 | 139.3 | 179.3 | 219.3 | 249.1 | 267.5 | 291.5 | 326.4 |
| 1952 | 1957 | 1962 | 1967 | 1972 | 1977 | 1982 | 1987 | 1992 |
| 104.5 | 124.6 | 153.5 | 189.2 | 226.4 | 247.7 | 270.2 | 307.6 | 316.6 |

(c) For the upper pairs of data, compute and plot $p(x), x \in [1950, 2000]$ (say). Mark the given data points. Do this for $m = 9$, and for (say) $n = 9$, and then for $n = 8:-2:2$. Store the results so that comparisons can be made afterward.

(d) Do the same for the lower pairs of data. Organize the plots so that interesting comparisons become convenient. How well are the data points of one of the sets interpolated by the results from the other set?

(e) Make forecasts for 1995 and 2000 with both data sets. Then, use a reduced data set, e.g., for the years 1982 and earlier (so that $m = 7$), and compare the forecasts for 1987 and 1992 with the given data. (Isn't it a reasonable test for every suggested forecast model to study its ability to predict the present from the past?)

(f) See if you obtain better results with the logarithms of the GDP values.

## 4.2 Interpolation Formulas and Algorithms

### 4.2.1 Newton's Interpolation Formula

Newton's interpolation formula uses the triangle family of Newton polynomials (4.1.10). Let $p \in \mathcal{P}_n$ be the unique polynomial interpolating a given function $f(x)$ at $n$ distinct real

or complex points $x_1, x_2, \ldots, x_n$. Suppose that an expansion

$$f(x) = c_1 + c_2(x - x_1) + \cdots + c_n(x - x_1)(x - x_2)\cdots(x - x_{n-1}) + A_n(x)(x - x_1)(x - x_2)\cdots(x - x_n) \tag{4.2.1}$$

holds, where $A_n(x)$ is the coefficient of the remainder term. If $f \in \mathcal{P}_n$, we know from Sec. 4.1.2 that such a formula holds with $A_n(x) \equiv 0$. We shall see that it is correct in general.

For $x = x_1$ we get $c_1 = f(x_1)$. Further,

$$[x_1, x]f = c_2 + c_3(x - x_2) + \cdots + c_n(x - x_2)\cdots(x - x_{n-1}) + A_n(x)(x - x_2)\cdots(x - x_n),$$

where we have set

$$[x_1, x]f = \frac{f(x) - f(x_1)}{x - x_1}.$$

This shows that $c_2 = [x_1, x_2]f$.

We now define **divided differences**[128] for $k > 1$, by the recursion

$$[x_1, \ldots, x_{k-1}x_k, x]f = \frac{[x_1, \ldots, x_{k-1}, x]f - [x_1, \ldots, x_{k-1}, x_k]f}{x - x_k}. \tag{4.2.2}$$

We obtain, for $k = 2$,

$$[x_1, x_2, x]f = c_3 + c_4(x - x_3) + \cdots + c_n(x - x_3)\cdots(x - x_{n-1}) + A_n(x)(x - x_3)\cdots(x - x_n),$$

and $c_3 = [x_1, x_2, x_3]f$. By induction it follows that

$$c_k = [x_1, \ldots, x_{k-1}, x_k]f, \quad k = 1 : n; \tag{4.2.3}$$

i.e., $c_k$ in (4.2.1) equals the $(k-1)$th divided difference of $f$. Further,

$$A_n(x) = [x_1, x_2, \ldots, x_n, x]f$$

for the coefficient of the remainder term.

We are now ready to state **Newton's interpolation formula** with exact remainder.

**Theorem 4.2.1.**

*The unique interpolation polynomial $p \in \mathcal{P}_n$ such that $p(x_i) = f(x_i)$, $i = 1 : n$, where the $x_i$ are distinct points, can be written as*

$$p(x) = \sum_{k=1}^{n} c_k \Phi_{k-1}(x), \tag{4.2.4}$$

*where $c_k = [x_1, x_2, \ldots, x_k]f$ is the divided difference and*

$$\Phi_0 = 1, \qquad \Phi_k(x) = \Phi_{k-1}(x)(x - x_k), \quad k = 1 : n. \tag{4.2.5}$$

[128] We prefer the modern notation $[\ldots]f$ to the older notations $f[\ldots]$ or $f(\ldots)$, since it emphasizes that $[\ldots]$ is an operator. Note that the interpretation $[x]f = f(x)$ is consistent with this.

*The formula*

$$f(x) = \sum_{k=1}^{n} [x_1, x_2, \ldots, x_k] f \; \Phi_{k-1}(x) + [x_1, x_2, \ldots, x_n, x] f \; \Phi_n(x) \tag{4.2.6}$$

*is an identity, i.e., the exact remainder equals*

$$f(x) - p(x) = [x_1, x_2, \ldots, x_n, x] f \; \Phi_n(x). \tag{4.2.7}$$

*These formulas are valid also for complex $x_i$ and $x$.*

***Proof.*** For $f \in \mathcal{P}_n$ we know that (4.2.1) is correct, hence we can trust the coefficients $c_k = -[x_1, \ldots, x_k] f$. Moreover, since $p(x_k) = f(x_k)$, $k = 1 : n$, it follows that $[x_1, x_2, \ldots, x_k] p = [x_1, x_2, \ldots, x_k] f$, and hence (4.2.4) holds.

We prove the identity (4.2.6) by induction. For $n = 1$ it is true because the right-hand side becomes $f(x_1) + [x_1, x] f \cdot (x - x_1) = f(x)$, which equals the left-hand side. Next suppose that it is true for $n = m$. The difference between the right-hand side for $n = m + 1$ and $n = m$ is

$$\begin{aligned}
&\big([x_1, \ldots, x_{m+1}] f - [x_1, \ldots, x_m, x] f\big) \, \Phi_m(x) + [x_1, \ldots, x_{m+1}, x] f \; \Phi_{m+1}(x) \\
&= \big([x_1, \ldots, x_{m+1}] f - [x_1, \ldots, x_m, x] f + [x_1, \ldots, x_{m+1}, x] f \; (x - x_{m+1})\big) \Phi_m(x) \\
&= \big([x_1, \ldots, x_{m+1}, x] f \; (x_{m+1} - x) + [x_1, \ldots, x_{m+1}, x] f \; (x - x_{m+1})\big) \Phi_m(x) = 0.
\end{aligned}$$

Hence the conjecture is true for $n = m + 1$. □

Note that to obtain the interpolation polynomial of the next higher degree with Newton's formula, we need only add a term similar to the last term, but involving a new divided difference of one higher order.

In particular, if $f \in \mathcal{P}_n$, then it follows from (4.2.7) that

$$[x_1, x_2, \ldots, x_n, x] f = 0 \quad \forall x.$$

For $x = x_{n+1}$, this equation is, by Theorem 3.3.4 the only nontrivial relation of the form $\sum_{j=1}^{n+1} a_j f(x_j) = 0$ that holds for all $f \in \mathcal{P}_n$.

**Theorem 4.2.2.**

*Let $x_i$, $i = 1 : n$, be pairwise distinct interpolation points. For every $n$, the divided difference $[x_1, x_2, \ldots, x_n] f$ is the unique coefficient of $x^{n-1}$ in the interpolation polynomial $p \in \mathcal{P}_n$.*[129] *Further, we have*

$$[x_1, x_2, \ldots, x_n] f = \sum_{j=1}^{m} f(x_j) \prod_{\substack{i=1 \\ i \neq j}}^{n} \frac{1}{(x_j - x_i)}. \tag{4.2.8}$$

***Proof.*** The first statement follows from (4.2.3). The right-hand side of (4.2.8) is the coefficients of $x^{n-1}$ obtained by using Lagrange's interpolation formula, Theorem 4.1.1. □

[129] Some authors take this as the *definition* of the divided difference.

It follows from (4.2.8) that the divided differences are symmetric functions of its arguments and continuous functions of its arguments, as long as the points $x_i$ are distinct and $f(x)$ is continuous.

Assume $k > i$. By the definition of divided differences,

$$[x_{i+1}, \ldots, x_{k-1}, x_k, x]f = \frac{[x_{i+1}, \ldots, x_{k-1}, x]f - [x_{i+1}, \ldots, x_{k-1}, x_k]f}{x - x_k}.$$

Now set $x = x_i$ and use the symmetry property (Theorem 4.2.2). We obtain the formula

$$[x_i, x_{i+1}, \ldots, x_{k-1}, x_k]f = \frac{[x_i, x_{i+1}, \ldots, x_{k-1}]f - [x_{i+1}, \ldots, x_{k-1}, x_k]f}{x_i - x_k}. \tag{4.2.9}$$

This formula can be used recursively to compute the divided differences. The computation is conveniently arranged in the table below for $n = 5$ (recall that $[x_i]f = f(x_i)$).

$$\begin{array}{llllll}
x_1 & [x_1]f & & & & \\
 & & [x_1, x_2]f & & & \\
x_2 & [x_2]f & & [x_1, x_2, x_3]f & & \\
 & & [x_2, x_3]f & & [x_1, x_2, x_3, x_4]f & \\
x_3 & [x_3]f & & [x_2, x_3, x_4]f & & [x_1, x_2, x_3, x_4, x_5]f \\
 & & [x_3, x_4]f & & [x_2, x_3, x_4, x_5]f & \\
x_4 & [x_4]f & & [x_3, x_4, x_5]f & & \\
 & & [x_4, x_5]f & & & \\
x_5 & [x_5]f & & & &
\end{array}$$

This table is called a **divided-difference table**. Each entry in the table is computed from the two entries above and below in the previous column. Hence the complete table can be constructed, for example, column by column or diagonal by diagonal.

The exact remainder term in Theorem 4.2.1 is not directly useful since unknown value $f(x)$ occurs in the divided difference $[x_1, \ldots, x_n, x]f$. We now derive another expression for the remainder term. In the following, $\text{int}\,(x, x_1, \ldots, x_n)$ denotes the smallest interval that contains the points $x$ and $x_1, \ldots, x_n$.

**Theorem 4.2.3** (*The Remainder Term for Interpolation*).

*Let $f$ be a given real function, with $f^{(n)}$ continuous in* $\text{int}\,(x, x_1, x_2, \ldots, x_n)$. *Denote by $p$ the polynomial of degree $n-1$ for which $p(x_i) = f(x_i)$, $i = 1 : n$. Then*

$$f(x) - p(x) = [x_1, x_2, \ldots, x_n, x]f\ \Phi_n(x) = \frac{f^{(n)}(\xi_x)}{n!}\Phi_n(x), \tag{4.2.10}$$

$\Phi_n(x) = \prod_{i=1}^{n}(x - x_i)$, *for some point* $\xi_x \in \text{int}\,(x, x_1, x_2, \ldots, x_n)$, *and*

$$[x_1, x_2, \ldots, x_n, x_{n+1}]f = \frac{f^{(n)}(\xi)}{n!}, \quad \xi \in \text{int}\,(x_1, \ldots, x_{n+1}). \tag{4.2.11}$$

***Proof.*** Following a proof due to Cauchy, we introduce a new variable $z$ and set

$$G(z) = f(z) - p(z) - [x_1, x_2, \ldots, x_n, x]f\ \Phi_n(z).$$

We know by Theorem 4.2.1 that

$$f(x) - p(x) = [x_1, x_2, \ldots, x_n, x]f\ \Phi_n(x), \tag{4.2.12}$$

hence $G(x) = 0$. Then $G(z) = 0$ for $z = x, x_1, x_2 \ldots, x_n$. From repeated use of Rolle's theorem it follows that there exists a point $\xi_x \in \text{int}\,(x, x_1, x_2, \ldots, x_n)$ such that $G^{(n)}(\xi_x) = 0$. Since $p^{(n)}(z) = 0$ and $\Phi_n^{(n)}(z) = n!$ for all $z$, $G^{(n)}(z) = f^{(n)}(z) - [x_1, x_2, \ldots, x_n, x]f\, n!$. If we now put $z = \xi_x$, we obtain

$$[x_1, x_2, \ldots, x_n, x]f = \frac{f^{(n)}(\xi_x)}{n!}. \tag{4.2.13}$$

Put this into the definition of $G(z)$, and set $z = x$. Since $G(x) = 0$, the first statement follows. The second statement follows from (4.2.13) for $x = x_{n+1}$. □

Notice the similarity to the remainder term in Taylor's formula. Notice also that the right-hand side of (4.2.10) is zero at the grid points—as it should be.

In the proof of this theorem we have assumed that $x_i, x$, and $f(x)$ are *real*. In Sec. 4.3.1 we shall derive a remainder term for the general case that $x_i$ are complex interpolation points and also consider the case when the points are allowed to coincide.

**Theorem 4.2.4.**

*For equidistant points $x_i = x_1 + (i-1)h$, $f_i = f(x_i)$, it holds that*

$$[x_i, x_{i+1}, \ldots, x_{i+k}]f = \frac{\Delta^k f_i}{h^k k!}. \tag{4.2.14}$$

***Proof.*** The proof is obtained by induction, with the use of (4.2.9). The details are left to the reader. □

We have noted above that in the notation for the equidistant case $\nabla^k f_n \approx h^k f^{(k)}$, while in the divided difference notation $f[x_n, x_{n-1}, \ldots, x_{n-k}] \approx f^{(k)}/k!$. For the basis functions of the interpolation formulas we have, respectively,

$$\binom{x}{k} = O(1), \qquad (x - x_n)(x - x_{n-1}) \cdots (x - x_{n-k+1}) = O(h^k),$$

provided that $x - x_{n-j} = O(h)$, $j = 0 : k-1$.

We are now in a position to give a short proof of the important formula (3.3.4) that we now formulate as a theorem.

**Theorem 4.2.5.**

*Assume that $f \in C^k$. Then*

$$\Delta^k f(x) = h^k f^{(k)}(\zeta), \quad \zeta \in [x, x + kh]. \tag{4.2.15}$$

*If $f \in \mathcal{P}_k$, then $\Delta^k f(x) = 0$. Analogous results hold, mutatis mutandis, for backward and central differences.*

***Proof.*** Combine the result in Theorem 4.2.4 with (4.2.11), after making appropriate substitutions. □

**Example 4.2.1.**

Suppose we want to compute by linear interpolation the value $f(x)$ at a point $x = x_0 + \theta h$, $h = x_1 - x_0$. Since $\theta(1-\theta)$ takes on its maximum value $1/4$ for $\theta = 1/2$, it follows from (4.2.10) that for $0 \le \theta \le 1$ the remainder satisfies

$$|f(x) - p(x)| \le h^2 M_2/8, \quad M_2 = \max_{x \in \text{int}\,[x_0, x_1]} |f''(x)|. \tag{4.2.16}$$

If the values $f_0$ and $f_1$ are given to $t$ correct decimal digits, then the roundoff error in evaluating

$$p(x) = (1-\theta) f_0 + \theta f_1, \quad 0 \le \theta \le 1,$$

is bounded by $\frac{1}{2}10^{-t}$. Further, if $\frac{h^2 M_2}{8} \le \frac{1}{2} \cdot 10^{-t}$, then the *total error* in $p(x)$ is bounded by $10^{-t}$.

This analysis motivates the rule of thumb that linear interpolation suffices if $|\Delta^2 f_n|/8$ is a tolerable truncation error.

To form the Newton interpolation polynomial we only need one diagonal of the divided-difference table. It is not necessary to store the entire table. The following algorithm replaces (overwrites) the given function values

$$f_i = f(x_i) = [x_1] f, \quad i = 1 : n,$$

by the downward diagonal of divided differences of the divided-difference table,

$$[x_1] f, [x_1, x_2] f, \ldots, [x_1, x_2, \ldots, x_n] f.$$

At step $j$ the $j$th column of the table is computed as follows:

```
% Compute downward diagonal of the divided-difference
% table. Function values are overwritten
for j = 1:n-1
   for i = n:(-1):j+1
      f(i) = (f(i) - f(j-1))/(x(i) - x(i-j));
   end
end
```

Note that to avoid overwriting data needed later it is necessary to proceed from the bottom of each column. The algorithm uses $n(n-1)/2$ divisions and $n(n-1)$ subtractions.

Newton's interpolation formula has the advantage that if it is not known in advance how many interpolation points are needed to achieve the required accuracy, then one interpolation point can be added at a time. The following *progressive algorithm* computes the divided-difference table one diagonal at a time. In the $j$th step, $j = 2 : n$, the entries

$$[x_{j-1}, x_j] f, \ldots, [x_1, x_2, \ldots, x_j] f$$

on the upward diagonal of the divided-difference table overwrite the function values $f_{j-1}, \ldots, f_1$.

```
% Compute upward diagonal of the divided-difference
% table. Function values are overwritten
for j = 2:n
   for i = j:(-1):2
      f(i-1) = (f(i) - f(i-1))/(x(j) - x(i-1));
   end
end
```

By Theorem 4.2.1 the Newton polynomial has the form

$$p(x) = c_1 + \sum_{j=2}^{n} c_j \prod_{i=1}^{j-1}(x - x_i), \quad c_j = [x_1, \ldots, x_j]f. \tag{4.2.17}$$

Substituting $z$ for $x$ we have, by a simple generalization of Horner's rule, that $p(z) = b_1(z)$, where

$$b_n = c_n, \quad b_i(z) = b_{i+1}(z)(z - x_i) + c_i, \quad i = n-1 : -1 : 1. \tag{4.2.18}$$

This recursion can be used algebraically or to evaluate $p(x)$ for a given numerical value $x = z$. It is straightforward to show that in the latter case the computed result is the exact value corresponding to slightly perturbed divided differences; cf. Problem 2.3.6.

The auxiliary quantities $b_n(z), \ldots, b_2(z)$ are of independent interest, since

$$p(x) = b_1 + (x - z)\left(b_2 + \sum_{j=2}^{n-1} b_{j+1}\phi_{j-1}(x)\right). \tag{4.2.19}$$

Hence they give the coefficients of the quotient polynomial in synthetic division of $p(x)$ with $(x - z)$. The proof of this result is left as an exercise. Derivatives of a Newton polynomial can be evaluated by repeated applications of the Horner scheme.

**Example 4.2.2.**

Compute the interpolation polynomial for the following table:

$$\begin{array}{lllll}
x_1 = 1 & \mathbf{0} & & & \\
 & & \mathbf{2} & & \\
x_2 = 2 & 2 & & \mathbf{1} & \\
 & & 5 & & \mathbf{0} \\
x_3 = 4 & 12 & & 1 & \\
 & & 8 & & \\
x_4 = 5 & 20 & & &
\end{array}$$

(the entries appearing in the Newton forward interpolation formula are boldface). We get

$$\begin{aligned} p(x) &= 0 + 2(x-1) + 1(x-1)(x-2) + 0(x-1)(x-2)(x-4) \\ &= x^2 - x. \end{aligned}$$

Note that for these particular data the unique interpolation polynomial in $\mathcal{P}_4$ actually belongs to the subspace $\mathcal{P}_3$.

**Example 4.2.3.**

Set $f(x; z) = 1/(z - x)$; $x$ is the variable, $z$ is a parameter, and both may be complex. The following elementary, though remarkable, expansion can be proved directly by induction.

$$\begin{aligned}\frac{1}{z-x} &= \frac{1}{z-x_1} + \frac{x-x_1}{(z-x_1)(z-x_2)} + \cdots + \frac{(x-x_1)(x-x_2)\cdots(x-x_{n-1})}{(z-x_1)(z-x_2)\cdots(z-x_n)} \\ &\quad + \frac{(x-x_1)\cdots(x-x_n)}{(z-x_1)\cdots(z-x_n)(z-x)} \\ &= \sum_{j=1}^{n} \frac{\Phi_{j-1}(x)}{\Phi_j(z)} + \frac{\Phi_n(x)}{\Phi_n(z)(z-x)}. \end{aligned} \tag{4.2.20}$$

When we match this with Newton's interpolation formula we find that

$$[x_1, x_2, \ldots, x_j] f(x; z) = \frac{1}{\Phi_j(z)}, \tag{4.2.21}$$

$$[x_1, x_2, \ldots, x_j, x] f(x; z) = \frac{1}{\Phi_j(z)(z-x)}. \tag{4.2.22}$$

These formulas can also be proved by induction, by working algebraically with $1/(z-x)$ in the divided-difference table (Problem 4.2.4). See also Problem 3.3.3 (a) for the equidistant case.

This is more than a particular example. Since $1/(z - x)$ is the kernel of Cauchy's integral (and several other integral representations), this expansion is easily generalized to arbitrary analytic functions; see Sec. 4.3.2.

An interesting feature is that these formulas do *not* require that the points $x_i$ be distinct. (They are consistent with the extension to nondistinct points that will be made in Sec. 4.3.1.) Everything is continuous except if $z = x_i$, $i = 1 : n$, or, of course, if $z = x$. If all the $x_i$ coincide, we obtain a geometric series with a remainder.

For given interpolation points the divided differences in Newton's interpolation formula depend on the ordering in which the points $x_i$ are introduced. Mathematically all orderings give the same unique interpolation polynomial. But *the condition number for the coefficients c in the Newton polynomial can depend strongly on the ordering of the interpolation points*. For simple everyday interpolation problems the increasing order $x_1 < x_2 < \cdots < x_n$ will give satisfactory results.

If the point $\tilde{x}$ where the polynomial is to be evaluated is known, then an ordering such that

$$|\tilde{x} - x_1| \le |\tilde{x} - x_2| \le \cdots \le |\tilde{x} - x_n|$$

can be recommended. In the equidistant case this corresponds to using Stirling's interpolation formula (3.3.39). In case convergence is slow and an interpolation polynomial of high

order has to be used, another suitable choice is the **Leja ordering**, defined by

$$x_1 = \max_{1\le i\le n} |x_i|, \qquad \prod_{k=1}^{j-1} |x_j - x_k| = \max_{i\ge j} \prod_{k=1}^{j-1} |x_i - x_k|, \quad j = 2 : n-1. \tag{4.2.23}$$

Let $\mathcal{K}$ be a compact set in the complex plane with a connected complement. Any sequence of points $\xi_1, \xi_2, \ldots$ which satisfies the conditions

$$|\xi_1| = \max_{\xi\in\mathcal{K}} |\xi|, \qquad \prod_{k=1}^{j-1} |\xi_j - \xi_k| = \max_{\xi\in\mathcal{K}} \prod_{k=1}^{j-1} |\xi - \xi_k|, \quad j = 2, 3, \ldots, \tag{4.2.24}$$

are **Leja points** for $\mathcal{K}$. The points may not be uniquely defined by (4.2.24). For a real interval $[a, b]$ the Leja points are distributed similarly to the Chebyshev points. The main advantage of the Leja points is that it is easy to add new Leja points successively to an already computed sequence of Leja points; see Reichel [299].

### 4.2.2 Inverse Interpolation

It often happens that one has a sequence of pairs $\{(x_i, y_i)\}$ and wants to determine a point where $y(x) = c$. We saw an example in the simulation of the motion of a ball (Sec. 1.5.2), where the landing point was computed by linear inverse interpolation.

In general a natural approach is to reverse the roles of $x$ and $y$, i.e., to compute the inverse function $x(y)$ for $y = c$ by means of Newton's interpolation formula with the divided differences $[y_i, y_{i+1}, \ldots, y_{i+j}]x$. This is called **inverse interpolation**. It is convenient to order the points so that

$$\cdots < y_5 < y_3 < y_1 < c < y_2 < y_4 < \cdots.$$

This approach is successful if the function $x(y)$ is suitable for local approximation by a polynomial.

Sometimes, however, the function $y(x)$ is much better suited for local approximation by a polynomial than the inverse function $x(y)$. Then we can instead, for some $m$, solve the following equation:

$$y_1 + [x_1, x_2]y \cdot (x - x_1) + \sum_{j=2}^{n-1} [x_1, x_2, \ldots, x_{j+1}]y\, \Phi_j(x) = c. \tag{4.2.25}$$

Again it is convenient to order the points so that the root $\alpha$ comes in the middle, for example,

$$\cdots < x_5 < x_3 < x_1 < \alpha < x_2 < x_4 < \cdots.$$

Suppose that $x_i - x_1 = O(h)$, $i > 1$, where $h$ is some small parameter in the context (usually some step size). Then

$$\Phi_j(x) = O(h^j), \qquad \Phi_j'(x) = O(h^{j-1}).$$

The divided differences are $O(1)$, and we assume that $[x_1, x_2]y$ is bounded away from zero. Then the terms of the sum decrease like $h^j$.

Writing the equation in the form $x = x_1 + F(x)$, where

$$F(x) \equiv \frac{(c - y_1) - \sum_{j=2}^{n-1} [x_1, x_2, \ldots, x_{j+1}]y \, \Phi_j(x)}{[x_1, x_2]y}, \tag{4.2.26}$$

we can use the *iteration* $x^{k+1} = x_1 + F(x^k)$ to determine $x$. We ignore the sum to get the first guess $x^0$; this means the same as linear inverse interpolation. We stop when $x^k$ and $x^{k-1}$ are sufficiently close. A more careful termination criterion will be suggested in Chapter 6, where the effect on the result of errors like the interpolation error is also discussed. From the discussion of fixed-point iteration in Sec. 1.1.1, we conclude that the iteration converges with linear ratio equal to

$$F'(x) \approx \frac{\Phi_2'(x)[x_1, x_2, x_3]y}{[x_1, x_2]y} = O(h).$$

Thus, if $h$ is small enough, the iterations converge rapidly. If more than two iterations are needed, Aitken acceleration (Sec. 3.4.2) may be practical.

### 4.2.3 Barycentric Lagrange Interpolation

Lagrange's interpolation formula was introduced in Sec. 4.1.2. For distinct real interpolation points $x_i$, $i = 1 : n$, this formula uses the cardinal basis of $\mathcal{P}_n$ consisting of the Lagrange polynomials of degree $n - 1$:

$$\ell_j(x) = \prod_{\substack{i=1 \\ i \neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)}, \quad j = 1 : n, \tag{4.2.27}$$

with the property that

$$\ell_j(x_i) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

Quite often it is asserted that the Lagrange form is a bad choice for practical computations,[130] since for each new value of $x$ the functions $\ell_j(x)$ have to be recomputed at a cost $\mathcal{O}(n^2)$. Further, adding a new data point $x_{n+1}$, $f_{n+1}$ will require a new computation from scratch. Therefore, it is concluded that Lagrange's formula is not as efficient as Newton's interpolation formula.

The above assertions are, however, not well-founded. *The Lagrange representation can easily be rewritten in two more attractive forms which are both eminently suitable for computation*; see Rutishauser [312] and Berrut and Trefethen [24]. The key idea is to take out the common factor $\Phi_n(x)$ in (4.1.15) and introduce the **support coefficients**

$$w_j = \frac{1}{\prod_{\substack{i=1 \\ i \neq j}} (x_j - x_i)}, \quad j = 1 : n. \tag{4.2.28}$$

[130] "For actual numerical interpolation Lagrange's formula is, however, as a rule not very suitable." (Steffensen [330, p. 25].)

The Lagrange polynomials can then be written as

$$\ell_j(x) = \Phi_n(x)\frac{w_j}{x - x_j}.$$

Taking out the common factor gives the **modified form** of Lagrange's interpolation formula:

$$p(x) = \Phi_n(x)\sum_{j=1}^{n}\frac{w_j}{x - x_j}f(x_j). \tag{4.2.29}$$

Here $w_j$ depend only on the given points $x_j$, $j = 1 : n$, and can be computed in $2(n-1)$ flops. This is slightly more than the $\frac{3}{2}n(n-1)$ flops required to compute the divided differences for Newton's interpolation formula. Then, to evaluate $p(x)$ from (4.2.29) for a new value of $x$ we only need to compute $\Phi_n(x)$ and $w_j/(x - x_j)$, $j = 1 : n$, which just requires $\mathcal{O}(n)$ operations.

The product factor $\Phi_n(x)$ in (4.2.29) can be eliminated as follows. Since the interpolation formula is exact for $f(x) \equiv 1$, we have

$$1 = \Phi_n(x)\sum_{j=1}^{n}\frac{w_j}{x - x_j}.$$

Substituting this in (4.2.29),

$$p(x) = \frac{\displaystyle\sum_{j=1}^{n}\frac{w_j}{x - x_j}f(x_j)}{\displaystyle\sum_{j=1}^{n}\frac{w_j}{x - x_j}} \quad \text{if } x \neq x_j, \quad j = 1 : n, \tag{4.2.30}$$

which is the **barycentric form** of Lagrange's interpolation formula. This expresses the value $p(x)$ as a weighted mean of the values $f_i$. (Note that the coefficients $w_j/(x - x_j)$ need not be positive, so the term "barycentric" is not quite appropriate.)

The barycentric formula (4.2.30) has a beautiful symmetric form and is "eminently suitable for machine computation" (Henrici [195, p. 237]). Unlike Newton's interpolation formula, it does not depend on how the nodes are ordered. The numerical stability of the two modified Lagrange interpolation formulas is, contrary to what is often stated, very good. Note that the interpolation property of $p(x)$ is preserved even if the coefficients $w_i$ are perturbed, but then $p(x)$ is usually no longer a polynomial but a rational function.

There seems to be a stability problem for the formula (4.2.30) when $x$ is very close to one of the interpolation points $x_i$. In this case $w_i/(x - x_i)$ will be very large and not accurately computed because of the cancellation in the denominator. But this is in fact no problem, since there will be exactly the same error in the denominator. Further, in case $\Delta_i = \mathrm{fl}(x - x_i)$ is exactly zero, we can simply put $\Delta_i = u$, where $u$ is the unit roundoff; see Theorem 2.2.2.

The barycentric form of Lagrange's interpolation formula can be as efficiently updated as Newton's formula. Suppose the support coefficients $w_i^{(k-1)}$, $i = 1 : k-1$, for the points $x_1, \ldots, x_{k-1}$ are known. Adding the point $x_k$, the first $k-1$ new support coefficients can be calculated from

$$w_i^{(k)} = w_i^{(k-1)}/(x_i - x_k), \quad i = 1 : k-1,$$

using $(k-1)$ divisions and subtractions. Finally we have $w_k^{(k)} = 1/\prod_{i=1}^{k-1}(x_k - x_i)$. The computation of the support coefficients is summarized in the following program:

```
% Compute support coefficients
w(1) = 1;
for k = 2:n
   w(k) = 1;
   for i = 1:k-1
      w(i) = w(i)/(x(i) - x(k));
      w(k) = w(k)/(x(k) - x(i));
   end
end
```

*Note that the support coefficients $w_i$ do not depend on the function to be interpolated.* Once they are known interpolating a new function $f$ only requires $\mathcal{O}(n)$ operations. This contrasts favorably with Newton's interpolation formula, which requires the calculation of a new table of divided differences for each new function.

Suppose that we use interpolation points in an interval $[a, b]$ of length $2C$. Then as $n \to \infty$ the scale of the weights will grow or decay exponentially at the rate $C^{-n}$. If $n$ is large or $C$ is far from 1, the result may underflow or overflow even in IEEE double precision. In such cases there may be a need to rescale the support coefficients.

The computation of the support coefficients can be done in only $n(n-1)$ flops by using the relation (see Problem 4.2.5 and [318, Sec. 3.2.1]) $\sum_{i=1}^{n} w_i = 0$, $n > 1$, to compute $w_n = \sum_{i=1}^{n-1} w_i$. But using this identity destroys the symmetry and can lead to stability problems for large $n$. Serious cancellation in the sum will occur whenever $\max_i |w_i|$ is much larger than $|w_n|$. Hence the use of this identity is not recommended in general.

**Theorem 4.2.6** (*Higham* [200]).

*Assume that $x_i$, $f_i$, and $x$ are floating-point numbers. Then the computed value $\tilde{p}(x)$ of the interpolation polynomial using the modified Lagrange formula* (4.2.29) *satisfies*

$$\tilde{p}(x) = \Phi_n(x) \sum_{i=1}^{n} \frac{w_i}{x - x_i} f(x_i) \prod_{j=1}^{5(n+1)} (1 + \delta_{ij})^{\pm 1}, \tag{4.2.31}$$

*where $|\delta_{ij}| \le u$.*

*Thus the formula* (4.2.29) *computes the exact value of an interpolating polynomial corresponding to slightly perturbed function values $f(x_i)$. Hence this formula is* backward stable *in the sense of Definition* 2.4.10.

The barycentric formula is not backward stable. A forward error bound similar to that for the modified formula but containing an extra term proportional to $\sum_{j=1}^{n} |\ell_j(x)|$ can be shown. Hence the barycentric formula can be significantly less accurate than the modified Lagrange formula (4.2.29) only for a poor choice of interpolation points with a large Lebesgue constant $\Lambda_n$.

For various important distributions of interpolating points, it is possible to compute the support coefficients $w_i$ analytically.

**Example 4.2.4.**

For interpolation at the equidistant points $x_1$, $x_i = x_1 + (i-1)h$, $i = 2:n$, the support coefficients are

$$\begin{aligned} w_i &= 1\Big/\left((x_i - x_1)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)\right) \\ &= \frac{(-1)^{n-i}}{\left(h^{n-1}(i-1)!\,(n-i)!\right)} \\ &= \frac{(-1)^{n-i}}{h^{n-1}(n-1)!}\binom{n-1}{i}. \end{aligned}$$

In the barycentric formula (4.2.30) a common factor in the coefficients $w_i$ cancels and we may use instead the modified support coefficients

$$w_i^* = (-1)^{i+1}\binom{n-1}{i}. \tag{4.2.32}$$

For a given $n$ these can be evaluated in only $2n$ operations using the recursion

$$w_1^* = n-1, \qquad w_i^* = w_{i-1}^*\frac{n-i}{i}, \quad i = 2:n.$$

**Example 4.2.5.**

Explicit support coefficients are also known for the Chebyshev points of the first and second kind on $[-1, 1]$. For the Chebyshev points of the first kind they are

$$w_i = (-1)^i \sin\frac{(2i-1)}{n}\frac{\pi}{2}, \qquad x_i = \cos\frac{(2i-1)}{n}\frac{\pi}{2}, \quad i = 1:n. \tag{4.2.33}$$

For the Chebyshev points of the second kind they are

$$w_i = (-1)^i\delta_j, \qquad x_i = \cos\frac{(i-1)}{(n-1)}\pi, \quad i = 1:n, \tag{4.2.34}$$

where $\delta_j = 1/2$ if $i = 1$ or $i = n$, and 1 otherwise. Note that all but two weights are equal. This will be considered from another point of view in Sec. 4.6.

For an interval $[a, b]$ the Chebyshev points can be generated by a linear transformation. The corresponding weight $w_i$ then gets multiplied by $2^n(b-a)^n$. But this factor cancels out in the barycentric formula, and there is no need to include it. Indeed, by *not* doing so the risk of overflow or underflow, when $|b-a|$ is far from 1 and $n$ is large, is avoided.

The two examples above show that with equidistant or Chebyshev points only $\mathcal{O}(n)$ operations are needed to get the weights $w_i$. For these cases the barycentric formula seems superior to all other interpolation formulas.

Lagrange's interpolation formula can be used to compute the inverse of the Vandermonde matrix $V$ in (4.1.5) in $\mathcal{O}(n^2)$ operations. If we set $V^{-1} = W = (w_{ij})_{i,j=1}^n$, then $WV = I$, the $i$th row of which can be written

$$\sum_{j=1}^{n} w_{ij}x_k^j = \delta_{ik}, \quad k = 1:n.$$

This is an interpolation problem that is solved by the Lagrange basis polynomial

$$\ell_i(x) = \prod_{\substack{k=1 \\ k\neq i}}^{n} \frac{(x - x_k)}{(x_i - x_k)} = \sum_{j=1}^{n} w_{ij} x^j, \quad j = 1 : n. \tag{4.2.35}$$

Clearly $V$ is nonsingular if and only if the points $x_i$ are distinct.

The elements $w_{ij}$ in inverse Vandermonde matrix $W = V^{-1}$ can be computed as follows: First compute the coefficients of the polynomial

$$\Phi_n(x) = (x - x_1)(x - x_2) \cdots (x - x_n) = \sum_{j=1}^{n+1} a_j x^{j-1}.$$

This can be done by the following MATLAB script:

```
% Compute inverse Vandermonde matrix
a(1) = -x(1); a(2) = 1;
for k = 2:n
   a(k+1) = 1;
   for j = k:-1:2
      a(j) = a(j-1) - x(k)*a(j);
   end
   a(1) = -x(k)*a(1);
end
```

Then compute the coefficients of the polynomials

$$q_i(x) = \Phi_n(x)/(x - x_i), \quad i = 1 : n,$$

by synthetic division (see Sec. 1.2.2). By (4.2.35) the $n^2$ elements in $W$ equal the coefficients of the Lagrange polynomials

$$\ell_i(x) = q_i(x)/q_i(x_i), \quad i = 1 : n.$$

Here the scalars $q_i(x_i)$ are computed by Horner's rule. The cost of computing the $n^2$ elements in $W$ by this algorithm is only $6n^2$ flops.

### 4.2.4 Iterative Linear Interpolation

There are other recursive algorithms for interpolation. Of interest are those based on *successive linear interpolations*. The basic formula is given in the following theorem.

**Theorem 4.2.7.**

*Assume that the two polynomials $p_{n-1}(x)$ and $q_{n-1}(x)$, both in $\mathcal{P}_{n-1}$, interpolate $f(x)$ at the points $x_1, \ldots, x_{n-1}$ and $x_2, \ldots, x_n$, respectively. If the n points $x_1, x_2, \ldots, x_{n-1}, x_n$ are distinct, then*

$$p_n(x) = \frac{(x - x_1)q_{n-1}(x) - (x - x_n)p_{n-1}(x)}{x_n - x_1}$$

*is the unique polynomial in $\mathcal{P}_n$ that interpolates $f(x)$ at the m points $x_1, x_2, \ldots, x_{n-1}, x_n$.*

***Proof.*** Since $q_{n-1}(x)$ and $p_{n-1}(x)$ both interpolate $f(x)$ at the points $x_2, \ldots, x_{n-1}$ and

$$\frac{(x - x_1) - (x - x_n)}{x_n - x_1} = 1,$$

it follows that $p_n(x)$ also interpolates $f(x)$ at these points. Further, $p_n(x_1) = p_{n-1}(x_1)$ and hence interpolates $f(x)$ at $x_1$. A similar argument shows that $p_n(x)$ interpolates $f(x)$ at $x = x_n$. Hence $p_n(x)$ is the unique polynomial interpolating $f(x)$ at the distinct points $x_1, x_2, \ldots, x_n$. □

A variety of schemes use Theorem 4.2.7 to construct successively higher order interpolation polynomials. Denote by $P_{j,j+1,\ldots,k}(x)$, $k > j$, the polynomial interpolating $f(x)$ at the points $x_j, x_{j+1}, \ldots, x_k$. The calculations in **Neville's** algorithm can be arranged in a triangular table.

$$\begin{array}{cc|cccccc}
x_1 & f(x_1) & & & & & & \\
x_2 & f(x_2) & P_{1,2}(x) & & & & & \\
x_3 & f(x_3) & P_{2,3}(x) & P_{1,2,3}(x) & & & & \\
x_4 & f(x_4) & P_{3,4}(x) & P_{2,3,4}(x) & P_{1,2,3,4}(x) & & & \\
\vdots & \vdots & \vdots & \vdots & \vdots & & & \\
x_k & f(x_k) & P_{k-1,k}(x) & P_{k-2,k-1,k}(x) & P_{k-3,k-2,k-1,k}(x) & \ldots & P_{1,2,3,\ldots,k} &
\end{array}$$

Any entry in this table is obtained as a linear combination of the entries to the left and diagonally above in the preceding column.

Note that it is easy to add a new interpolation point in this scheme. To proceed only the last row needs to be retained. This is convenient in applications where the function values are generated sequentially and it is not known in advance how many values are to be generated.

Neville's algorithm is used, for example, in repeated Richardson extrapolation (see Sec. 3.4.6), where polynomial extrapolation to $x = 0$ is to be carried out. Another use of Neville's algorithm is in iterative inverse interpolation; see Isaacson and Keller [208, Chapter 6.2].

If it is known in advance that a fixed number $k$ of points are to be used, then one can instead generate the table column by column. When one column has been evaluated the preceding may be discarded.

**Aitken's** algorithm uses another sequence of interpolants, as indicated in the table below.

$$\begin{array}{cc|cccccc}
x_1 & f(x_1) & & & & & & \\
x_2 & f(x_2) & P_{1,2}(x) & & & & & \\
x_3 & f(x_3) & P_{1,3}(x) & P_{1,2,3}(x) & & & & \\
x_4 & f(x_4) & P_{1,4}(x) & P_{1,2,4}(x) & P_{1,2,3,4}(x) & & & \\
\vdots & \vdots & \vdots & \vdots & & & & \\
x_k & f(x_k) & P_{1,k}(x) & P_{1,2,k}(x) & P_{1,2,3,k}(x) & \ldots & P_{1,2,3,\ldots,k} &
\end{array}$$

For a fixed number $k$ of points this table can be generated column by column. To add a new point the upper diagonal $f(x_1)$, $P_{1,2}(x)$, $P_{1,2,3}(x)$, $\ldots$, $P_{1,2,\ldots,k}(x)$ needs to be saved. The basic difference between these two procedures is that in Aitken's the interpolants in any row use points with subscripts near 1, whereas Neville's algorithm uses rows with subscripts nearest $n$.

Neville's and Aitken's algorithms can easily be used in the case of multiple interpolation points also. The modification is similar to that in Newton's interpolation method.

### 4.2.5 Fast Algorithms for Vandermonde Systems

Given distinct scalars $x_1, x_2, \ldots, x_n$, the coefficients for the interpolating polynomial in the power basis $p = \sum_{i=1}^{n} a_i x^{i-1}$ are given by the solution to the linear system

$$V^T a = f, \tag{4.2.36}$$

where $V$ is the Vandermonde matrix

$$V = V(x_1, x_2, \ldots, x_n) = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & \cdots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{pmatrix}. \tag{4.2.37}$$

Vandermonde systems are appropriate to use for polynomial interpolation when the result is to be expressed in the monomial basis. Often the Newton basis is a better choice. As we shall see, even for solving the Vandermonde system (4.2.36) an algorithm based on Newton's interpolation formula is a much better choice than Gaussian elimination.

Newton's interpolation formula gives the interpolation polynomial in the form

$$p(x) = c_1 p_1(x) + c_2 p_2(x) + \cdots + c_n p_n(x),$$

where the basis polynomials are

$$p_1(x) = 1, \qquad p_k(x) = (x - x_1) \cdots (x - x_{k-1}), \quad k = 2 : n.$$

Here $c_j = [x_1, \ldots, x_{j-1}]f$ and the divided differences can be recursively computed as described in Sec. 4.2.1. Then the coefficient vector $a$ of $p(x)$ in the power basis

$$p(x) = a_1 + a_2 x + \cdots + a_n x^{n-1}$$

can be computed by Horner's rule. Note that the matrix $V^T$ is never formed and only storage for a few vectors is needed. It is easily verified that the operation count is $\frac{5}{2}n(n+1)$ flops.

The related **primal Vandermonde system**

$$Vy = b \tag{4.2.38}$$

arises in related problems of determining approximation of linear functionals.

**Example 4.2.6.**

We shall find a formula for integrals of the form $I(f) = \int_0^1 x^{-1/2} f(x)\,dx$ that is exact for $f \in \mathcal{P}_n$ and uses values $f(x_i)$, $i = 1:n$. Such integrals need a special treatment because of the singularity at $x = 0$. Set $\mu_j = \int_0^1 x^{-1/2} x^{j-1}\,dx$ and introduce the row vector $\mu^T = (\mu_1, \mu_2, \ldots, \mu_n)$. We have that $f(x) \approx p(x) = \sum_{i=1}^n c_j x^{j-1}$, where $V^T c = f$, and

$$I(f) \approx \int_0^1 x^{-1/2} p(x)\,dx = \sum_{i=1}^n c_j \mu_j = \mu^T V_n^{-T} f, \tag{4.2.39}$$

where $V_n$ is the Vandermonde basis. This can be written $I(f) = a^T f$, where the coefficient vector is $a = V_n^{-1}\mu$, i.e., $a$ is the solution to the primal Vandermonde system $Va = \mu$.

Clearly the approach in Example 4.2.6 can be used for any linear functional. We would also like to have a stable and efficient method for solving the primal system. One possibility would be to use the algorithm given in Sec. 4.2.3 which computes the inverse $V^{-1}$ in about $6n^2$ operations, and then form the product $V^{-1}b = y$.

We shall now derive a more efficient and accurate algorithm for solving primal Vandermonde systems. We start by expressing the solution of the dual problem in terms of a matrix factorization. Using the power basis the unique polynomial satisfying the interpolation conditions $p(x_i) = f_i$, $i = 1:n$, is

$$p(x) = (1, x, \ldots, x^{n-1})a,$$

where the coefficient vector $a$ satisfies the linear system $V^T a = f$.

To derive a corresponding algorithm for solving primal Vandermonde systems the above algorithm can be interpreted as a factorization of the matrix $(V^T)^{-1}$ into a product of diagonal and lower bidiagonal matrices. Let

$$D_k = \operatorname{diag}(1, \ldots, 1, (x_{k+1} - x_1), \ldots, (x_n - x_{n-k})) \tag{4.2.40}$$

and define the matrices

$$L_k(x) = \begin{pmatrix} I_{k-1} & 0 \\ 0 & B_{n-k+1}(x) \end{pmatrix}, \quad k = 1:n-1, \tag{4.2.41}$$

where

$$B_p(x) = \begin{pmatrix} 1 & & & \\ -x & 1 & & \\ & \ddots & \ddots & \\ & & -x & 1 \end{pmatrix} \in \mathbf{R}^{p\times p}. \tag{4.2.42}$$

Then the dual Vandermonde algorithm can be written in matrix terms as $c = U^T f, a = L^T c$, where

$$U^T = D_{n-1}^{-1} L_{n-1}(1) \cdots D_1^{-1} L_1(1), \tag{4.2.43}$$

$$L^T = L_1^T(x_1) L_2^T(x_2) \cdots L_{n-1}^T(x_{n-1}). \tag{4.2.44}$$

Since $a = V^{-T} f = L^T U^T f$, we have $V^{-T} = L^T U^T$.

**ALGORITHM 4.1.** *Fast Dual Vandermonde Solver.*

```
function a = dvand(x,f);
  n = length(x);
  a = f;
  % Compute divided differences of f(1:n).
  for k = 1:n-1
     for j = n:(-1):k+1
        a(j) = (a(j) - a(j-1))/(x(j) - x(j-k));
     end
  end
  % Compute coefficients using Horner's scheme.
  for k = n-1:(-1):1
     for j = k:n-1
        a(j) = a(j) - x(k)*a(j+1);
     end
  end
```

We can now obtain a fast algorithm for solving a primal Vandermonde system $Vy = b$ as follows. Transposing the matrix factorization of $V^{-T}$ gives $V^{-1} = UL$. Hence $y = V^{-1}b = U(Lb)$ and the solution to the primal system can be computed from $d = Lb$, $y = Ud$. Transposing (4.2.43)–(4.2.44) gives

$$L = L_{n-1}(x_{n-1}) \cdots L_2(x_2) L_1(x_1),$$

$$U = L_1^T(1) D_1^{-1} \cdots L_{n-1}^T(1) D_{n-1}^{-1}.$$

This leads to an algorithm for solving primal Vandermonde systems. The operation count and storage requirement of this are the same as for Algorithm 4.1.

**ALGORITHM 4.2.** *Fast Primal Vandermonde Solver.*

```
function y = pvand(x,b);
n = length(x);
  y = b;
  for k = 1:n-1
     for j = n:(-1):k+1
        y(j) = y(j) - x(k)*y(j-1);
     end
  end
  for k = n-1:(-1):1
     for j = k+1:n
        y(j) = y(j)/(x(j) - x(j-k));
      end
    for j = k:n -1
        y(j) = y(j) - y(j+1);
    end
  end
```

The given algorithms can be generalized to confluent Vandermonde matrices (see Example 4.3.1).

The above two algorithms are not only fast. They also give almost full relative accuracy in the solution of some Vandermonde systems which are so ill-conditioned that Gaussian elimination with complete pivoting fails to produce a single correct digit. This was first observed by Björck and Pereyra [31], from which the following example is taken.

**Example 4.2.7.**

Consider a primal Vandermonde system $V_n y = b$, with

$$x_i = 1/(i+2), \qquad b_i = 1/2^{i-1}, \quad i = 1:n.$$

The exact solution can be shown to be

$$y_i = (-1)^{i-1}\binom{n}{i}(1+i/2)^{n-1}.$$

Let $\bar{y}_i$ be the solution computed by the primal Vandermonde algorithm and take as a measure of the relative error $e_n = \max_{1\le i\le n} |y_i - \bar{y}_i|/|y_i|$. Using a hexadecimal floating-point arithmetic with $u = 16^{-13} = 2.22 \cdot 10^{-16}$, the following results were obtained.

| $n$ | 5 | 10 | 15 | 20 | 25 |
|---|---|---|---|---|---|
| $e_n/u$ | 4 | 5 | 10 | 54 | 81 |

The computed solution has small componentwise relative error which is remarkable since, for example, $\kappa(V_{10}) = 9 \cdot 10^{13}$.

A forward error analysis given by Higham [198] explains the surprisingly favorable results. If the points are positive and monotonically ordered,

$$0 < x_1 < x_2 < \cdots < x_n, \tag{4.2.45}$$

then the error in the solution $\bar{a}$ of a Vandermonde system $Vy = b$ computed by the primal algorithm can be bounded as

$$|\bar{a} - a| \le 5u|V^{-1}|\,|b| + O(u^2). \tag{4.2.46}$$

If the components of the right-hand side satisfy $(-1)^n b_i \ge 0$, then $|V^{-1}|\,|b| = |V^{-1}b|$, and this bound reduces to

$$|\bar{a} - a| \le 5u|a| + O(u^2); \tag{4.2.47}$$

i.e., the solution is computed with small relative error independent of the conditioning of $V$. A similar result holds for the dual algorithm. These good results can be shown to be related to the fact that when (4.2.45) holds, the matrix $V(x_1, x_2, \ldots, x_n)$ is **totally positive**, i.e., the determinant of every square's submatrix of $V$ is positive; see [135, 42, 95].

Fast Björck–Pereyra-type algorithms for **Vandermonde-like** matrices of the form

$$P = (p_i(\alpha_j))_{i,j=1}^{n},$$

where $p_i(x)$, $i = 1 : n$, are basis polynomials in $\mathcal{P}_n$ that satisfy a three-term recurrence relation, have also been developed; see Higham [199, Sec. 22.2]. Cauchy linear systems $Cx = b$, where the elements of $C$ have the special form

$$c_{ij} = 1/(x_i + y_j), \quad 1 \le i, j \le n,$$

can also be solved with an $O(n^2)$ Björck–Pereyra-type algorithm; see Boros, Kailath, and Olshevsky [42].

### 4.2.6 The Runge Phenomenon

The remainder term in interpolation is, according to Theorem 4.2.3, equal to

$$\frac{1}{n!}\prod_{i=1}^{n}(x - x_i)\, f^{(n)}(\xi_x).$$

Here $\xi_x$ depends on $x$, but one can say that the error curve behaves for the most part like a polynomial curve $y = c\prod_{i=1}^{n}(x - x_i)$. A similar curve is also typical for error curves arising from least squares approximation. This contrasts sharply with the error curve for Taylor approximation, whose behavior is described approximatively by $y = c(x - x_0)^n$.

It is natural to ask what the optimal placing of the interpolation points $x_1, \ldots, x_n$ should be in order to minimize the maximum magnitude of $\Phi_n(x) = \prod_{i=1}^{n}(x - x_i)$ in the interval in which the formula is to be used. For the interval $[-1, 1]$ the answer is given directly by the minimax property (Lemma 3.2.4) of the Chebyshev polynomials—choose

$$\Phi_n(x) = T_n(x)/2^{n-1}.$$

Thus the interpolation points should be taken as the zeros of $T_n(x)$. (In the case of an interval $[a, b]$ one makes the linear substitution $x = \frac{1}{2}(a + b) + \frac{1}{2}(b - a)t$.)

**Example 4.2.8.**

Use the same notations as before. For $f(x) = x^n$ the interpolation error becomes $f(x) - p(x) = \Phi_n(x)$, because $f^{(n)}(x)/n! \equiv 1$. Figure 4.2.1 shows the interpolation error with $n$ equidistant points on $[-1, 1]$ and with $n$ Chebyshev points on the same interval, i.e.,

$$x_i = -1 + 2\frac{i-1}{n-1}, \qquad x_i = \cos\Big(\frac{2i-1}{n}\frac{\pi}{2}\Big),$$

respectively, for $n = 6$ and $n = 12$. The behavior of the error curves as shown in Figure 4.2.1 is rather typical for functions where $f^{(n)}(x)$ is slowly varying. Also note that the error increases rapidly when $x$ leaves the interval $\text{int}(x_1, x_2, \ldots, x_n)$. In the equidistant case, the error is also quite large in the outer parts of the interval.

Equidistant interpolation can give rise to convergence difficulties when the number of interpolation points becomes large. This difficulty is often referred to as **Runge's phenomenon**, and we illustrate it in the following example. A more advanced discussion is given in Sec. 4.3.5, by means of complex analysis.

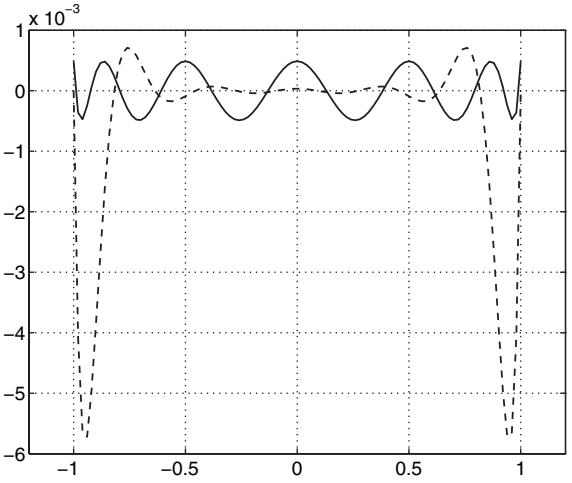



**Figure 4.2.1.** *Error of interpolation in $\mathcal{P}_n$ for $f(x) = x^n$, using $n = 12$: Chebyshev points (solid line) and equidistant points (dashed line).*

**Example 4.2.9.**

The graph of the function

$$f = \frac{1}{1+25x^2} = \frac{i}{2}\Big(\frac{1}{i+5x} + \frac{1}{i-5x}\Big),$$

where $i = \sqrt{-1}$, is the continuous curve shown in Figure 4.2.2, and is approximated in two different ways by a polynomial of degree 10 in $[-1, 1]$. The dashed curve has been

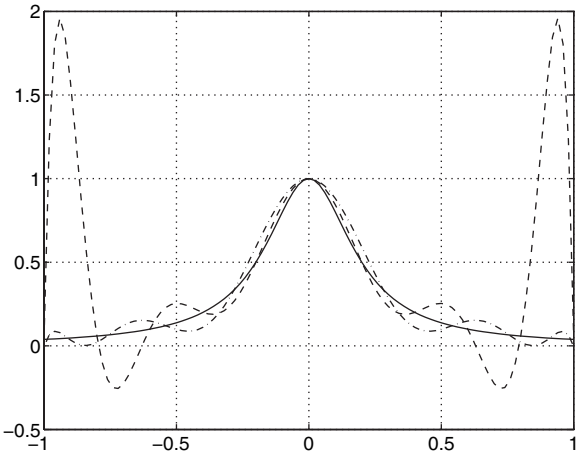



**Figure 4.2.2.** *Polynomial interpolation of $1/(1+25x^2)$ in two ways using 11 points: equidistant points (dashed curve), Chebyshev abscissae (dashed-dotted curve).*

determined by interpolation on the equidistant grid with $m = 11$ points

$$x_i = -1 + 2(i-1)/(m-1), \quad i = 1 : m. \tag{4.2.48}$$

The dash-dot curve has been determined by interpolation at the Chebyshev points

$$x_i = \cos\Big(\frac{2i-1}{m}\frac{\pi}{2}\Big), \quad i = 1 : m. \tag{4.2.49}$$

The graph of the polynomial obtained from the equidistant grid has—unlike the graph of $f$—a disturbing course between the grid points. The agreement with $f$ near the ends of the interval is especially bad, while near the center of the interval $[-\frac{1}{5}, \frac{1}{5}]$ the agreement is fairly good. Such behavior is typical of *equidistant interpolation of fairly high degree*, and can be explained theoretically.

The polynomial obtained from interpolation at Chebyshev points agrees much better with $f$, but still is not good. The function $f$ is not at all suited for approximation by one polynomial *over the entire interval*. One would get a much better result using approximation with piecewise polynomials; see Sec. 4.4.

Notice that the difference between the values of the two polynomials is much smaller at the grid points of the equidistant grid than in certain points between the grid points, especially in the outer parts of the interval. This intimates that the values which one gets by equidistant interpolation with a polynomial of high degree can be very sensitive to disturbances in the given values of the function. Put another way, *equidistant interpolation using polynomials of high degree is in some cases an ill-conditioned problem, especially in the outer parts of the interval* $[x_1, x_m]$. The effect is even worse if one extrapolates—i.e., if one computes values of the polynomial outside the grid. But equidistant interpolation works well near the center of the interval.

Even with equidistant data one can often get a more well behaved curve by—instead of interpolating—fitting a polynomial of lower degree using the method of least squares. Generally, if one chooses $n < 2\sqrt{m}$, then the polynomial fit is quite well-conditioned, but higher values of $n$ should be avoided.[131]

If one intends to approximate a function in $[-1, 1]$ and one can choose the points at which the function is computed or measured, then one should choose the Chebyshev points. Using these points, interpolation is a fairly *well-conditioned* problem in the entire interval. The risk of disturbing surprises between the grid points is insignificant. One can also conveniently fit a polynomial of lower degree than $n-1$, if one wishes to smooth errors in measurement; see Sec. 4.5.5.

Example 4.2.9 shows *how important it is to study the course of the approximating curve* $p^*(x)$ *between the points which are used in the calculation before one accepts the approximation*. When one uses procedures for approximation for which one does not have a complete theoretical analysis, one should make an *experimental perturbational calculation*. In the above case such a calculation would very probably reveal that the interpolation polynomial reacts quite strongly if the values of the function are disturbed by small amounts, say $\pm 10^{-3}$. This would give a basis for rejecting the unpleasant dashed curve in the example, even if one knew nothing more about the function than its values at the equidistant grid points.

[131] This fact is related to the shape of the so-called Gram polynomials; see Sec. 4.5.5.

## Review Questions

**4.2.1** Prove the theorem which says that the interpolation problem for polynomials has a unique solution.

**4.2.2** When is linear interpolation sufficient?

**4.2.3** Derive Newton's interpolation formula.

**4.2.4** Derive Newton's interpolation formula for the equidistant case, starting from Newton's general interpolation formula. How is this formula easily remembered?

**4.2.5** Derive the Lagrange interpolation formula. Show how it can be rewritten in barycentric form. When is the latter form more efficient to use?

**4.2.6** Neville's and Aitken's interpolation algorithms both perform successive linear interpolation. What is the difference between these?

**4.2.7** The fast algorithm in Sec. 4.2.5 for solving primal Vandermonde systems can give surprisingly accurate results provided that the points $x_1, x_2, \ldots, x_n$ satisfy certain conditions. Which?

**4.2.8** (a) Why is it important to study the course of the approximating curve $p^*(x)$ between the points which are used in the calculation before one accepts the approximation?

(b) What is a good choice of interpolation points in the interval $[a, b]$, if one wants to get a small error?

## Problems and Computer Exercises

**4.2.1** (a) Compute $f(3)$ by quadratic interpolation in the following table:

| $x$ | 1 | 2 | 4 | 5 |
|---|---|---|---|---|
| $f(x)$ | 0 | 2 | 12 | 21 |

Use the points 1, 2, and 4, and the points 2, 4, and 5, and compare the results.

(b) Compute $f(3)$ by cubic interpolation.

**4.2.2** Compute $f(0)$ using one of the interpolation formulas treated above on the following table:

| $x$ | 0.1 | 0.2 | 0.4 | 0.8 |
|---|---|---|---|---|
| $f(x)$ | 0.64987 | 0.62055 | 0.56074 | 0.43609 |

The interpolation formula is here used for *extrapolation*. Use also Richardson extrapolation and compare the results.

**4.2.3** Work out the details of Example 4.2.3 (about divided differences for $1/(z - x)$).

**4.2.4** (a) Consider the two polynomials $p(x)$ and $q(x)$, both in $\mathcal{P}_n$, which interpolate $f(x)$ at the points $x_1, \ldots, x_n$, and $x_2, \ldots, x_{n+1}$, respectively. Assume that $\{x_i\}_{i=1}^{n+1}$ is an increasing sequence, and that $f^{(n)}(x)$ has constant sign in the interval $[x_1, x_{n+1}]$.

Show that $f(x)$ is contained between $p(x)$ and $q(x)$ for all $x \in [x_1, x_{n+1}]$.
(b) Suppose that $f(x) = f_1(x) - f_2(x)$, where both $f_1^{(n)}(x)$ and $f_2^{(n)}(x)$ have the same constant sign in $[x_1, x_{n+1}]$. Formulate and prove a kind of generalization of the result in (a).

**4.2.5** Using the barycentric formula (4.2.29) the interpolation polynomial can be written as

$$p(x) = \sum_{i=1}^{n} w_i f(x_i) \prod_{\substack{j=1 \\ j \neq i}}^{m} (x - x_j).$$

Show by taking $f(x) \equiv 1$ and equating the coefficients for $x^{n-1}$ on both sides that the support coefficients satisfy $\sum_{i=1}^{n} w_i = 0$.

# 4.3 Generalizations and Applications

## 4.3.1 Hermite Interpolation

Newton's interpolation formula can also be used in generalized interpolation problems, where one or more derivatives are matched at the interpolation points. This problem is known as **Hermite interpolation**.[132]

**Theorem 4.3.1.**

*Let $\{z_i\}_{i=1}^{m}$ be $m$ distinct real or complex points. Let $f(z)$ be a given real- or complex-valued function that is defined and has derivatives up to order $r_{i-1}$ ($r_i \geq 1$) at $z_i$. The* **Hermite interpolation problem***, to find a polynomial $p(z)$ of degree $\leq n-1$, where $n = \sum_{i=1}^{m} r_i$ such that $p(z)$ and its first $r_i - 1$ derivatives agree with those of $f(z)$ at $z_i$, i.e.,*

$$p^{(j)}(z_i) = f^{(j)}(z_i), \quad j = 0 : r_i - 1, \quad i = 1 : m, \tag{4.3.1}$$

*is uniquely solvable. We use here the notation $f^{(0)}(z)$ for $f(z)$.*

***Proof.*** Note that (4.3.1) are precisely $n$ conditions on $p(z)$. The conditions can be expressed by a system of $n$ linear equations for the coefficients $p$, with respect to some basis. This has a unique solution for any right-hand side, unless the corresponding homogeneous problem has a nontrivial solution. Suppose that a polynomial $p \in \mathcal{P}_n$ comes from such a solution of the homogeneous problem, that is,

$$p^{(j)}(z_i) = 0, \quad i = 1 : m, \quad j = 0 : r_i - 1.$$

Then $z_i$ must be a zero of multiplicity $r_i$ of $p(x)$, hence $p(z)$ must have at least $\sum r_i = n$ zeros (counting the multiplicities). But this is impossible, because the degree of $p$ is less than $n$. This contradiction proves the theorem. □

Hermite interpolation can be viewed as the result of passages to the limit in interpolation at $n$ points, where for $i = 1 : m$, $r_i$ interpolation points coalesce into the point $z_i$. We

[132] Charles Hermite (1822–1901), a French mathematician, made important contributions to number theory, orthogonal polynomials, and elliptic functions.

say that the point $z_i$ has **multiplicity** $r_i$. For example, the Taylor polynomial in $\mathcal{P}_n$,

$$p(z) = \sum_{j=0}^{n-1} \frac{f^{(j)}(z_1)}{j!}(z - z_1)^j, \tag{4.3.2}$$

interpolates $f(z)$ at the point $z_1$ with multiplicity $n$ (or $z_1$ is repeated $n$ times).

**Example 4.3.1.**

Consider the problem of finding a polynomial $p(x) \in \mathcal{P}_4$ that interpolates the function $f$ and its first derivative $f'$ at the two points $z_1$ and $z_2$, and also its second derivative at $z_1$. In the notations of Sec. 4.1.1 the linear system for the coefficient vector $c$ becomes $V^T c = f$, where $f = (f(z_1),\, f'(z_1),\, f''(z_1),\, f(z_2),\, f'(z_2))^T$, and

$$V = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ z_1 & 1 & 0 & z_2 & 1 \\ z_1^2 & 2z_1 & 2 & z_2^2 & 2z_2 \\ z_1^3 & 3z_1^2 & 6z_1 & z_2^3 & 3z_2^2 \\ z_1^4 & 4z_1^3 & 12z_1^2 & z_2^4 & 4z_2^3 \end{pmatrix} \tag{4.3.3}$$

is a **confluent Vandermonde matrix**. Note that the second, third, and fifth column of $V$ is obtained by "differentiating" the previous column. From Theorem 4.3.1 we conclude that such confluent Vandermonde matrices are nonsingular. We remark that the fast algorithms in Sec. 4.2.5 can be generalized to confluent Vandermone systems; see [29].

For the determinant of the general confluent Vandermonde matrix one can show that

$$\det(V) = \prod_{j=1}^{m} \prod_{n=0}^{r_j - 1} n! \prod_{i<j} (z_i - z_j)^{r_i r_j}. \tag{4.3.4}$$

When there are gaps in the sequence of derivatives that are known at a point the interpolation problem is called **Birkhoff interpolation** or **lacunary** interpolation. Such a problem may not have a solution, as is illustrated by the following example.

**Example 4.3.2.**

Find a polynomial $p \in \mathcal{P}_3$ that interpolates the data $f = (f(-1),\, f'(0),\, f(1))^T$. The new feature is that $f(0)$ is missing. If we use the power basis, then we obtain the linear system

$$M_{\mathbf{p}} = \begin{pmatrix} 1 & -1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}.$$

The determinant is evidently zero, so *there is no solution* for most data. An explanation is that $hf' = \mu\delta f$ for all $f \in \mathcal{P}_3$.

Newton's interpolation formula can be generalized to the confluent case quite easily. We will require some continuity and differentiability properties for divided differences.

These can be obtained through an alternative expression for divided differences that we now give.

**Definition 4.3.2.**

*A set $S$ of points in $\mathbf{C}$ is called convex if for any $z, u \in S$, the straight line $\{tz+(1-t)u \mid t \in (0,1)\}$ is also contained in $S$. The* **convex hull** *of a set $S$ in $\mathbf{R}^d$ is the smallest convex subset of $\mathbf{C}$ which contains $S$.*

Let $D$ be the convex hull of the set of points $z, u_1, \ldots, u_n$ in $\mathbf{C}$. Assume that $f$ is defined in $D$ and has that its $n$th derivative exists and is continuous on $D$. Set $u_0(z) = f(z)$ and consider the functions

$$w_k(z) = \int_0^1 \int_0^{t_1} \cdots \int_0^{t_{k-1}} f^{(k)}\big[u_1 + (u_2 - u_1)t_1 + \cdots + (z - u_k)t_k\big]\, dt_k \cdots dt_1, \quad (4.3.5)$$

$k = 1 : n$. The argument of the integrand lies in the convex hull $D$, since

$$\begin{aligned}\zeta_k &= u_1 + (z - u_1)t_1 + \cdots + (z - u_k)t_k \\ &= (1 - t_1)u_1 + (t_1 - t_2)u_2 + \cdots + (t_{k-1} - t_k)u_k + t_k z \\ &= \lambda_1 u_1 + \lambda_2 u_2 + \cdots + \lambda_k u_k + \lambda_{k+1} z,\end{aligned}$$

where $1 \le k \le n$. From $1 \ge t_1 \ge t_2 \ge \cdots \ge t_n \ge 0$, it follows that

$$\sum_{i=1}^{k+1} \lambda_i = 1, \quad \lambda_i \ge 0, \quad i = 1 : k+1.$$

If in (4.3.5) we carry out the integration with respect to $t_k$ and express the right-hand side using $u_{k-1}$ we find that the functions $u_k(z)$ can be defined through the recursion

$$w_k(z) = \frac{w_{k-1}(z) - w_{k-1}(x_k)}{z - u_k}, \quad k = 1 : n, \quad (4.3.6)$$

with $w_0(z) = f(z)$. But this is the same recursion (4.3.5) that was used before to define the divided differences and thus

$$[z, u_1, \ldots, u_k]f = \int_0^1 \int_0^{t_1} \cdots \int_0^{t_{k-1}} f^{(k)}\big[u_1 + (u_2 - u_1)t_1 + \cdots + (z - u_k)t_k\big]\, dt_k \cdots dt_1, \quad (4.3.7)$$

holds for arbitrary distinct points $z, u_1, \ldots, u_n$.

Notice that the integrand on the right-hand side of (4.3.7) is a continuous function of the variables $z, u_1, \ldots, u_n$ and hence the right-hand side is a continuous function of these variables. Thus, when the $n$th derivative of $f$ exists and is continuous on $D$, (4.3.7) defines the continuous extension of $[z, u_1, \ldots, u_k]f$ to the confluent case.

From the continuity of the divided differences it follows that the remainder term for interpolation given in Theorem 4.2.3 remains valid for Hermitian interpolation. Provided the points $x, z_1, \ldots, z_m$ are real we have

$$f(x) - p(x) = \frac{f^{(n)}(\xi_x)}{n!}\Phi_n(x), \quad \Phi_n(x) = \prod_{i=1}^{m}(x - z_i)^{r_i}, \quad (4.3.8)$$

with $\xi_x \in \text{int}\,(x, z_1, \ldots, z_m)$. From (4.3.7) follows

$$\big|[z, x_1, \ldots, x_k]f\big| \le \max_{z\in D} \big|f^{(n)}(z)\big| \int_0^1 \int_0^{t_1} \cdots \int_0^{t_{n-1}} dt_n \cdots dt_1 \tag{4.3.9}$$

$$= \frac{1}{n!} \max_{z\in D} \big|f^{(n)}(z)\big|, \tag{4.3.10}$$

which can be used to give an upper bound for the remainder in polynomial interpolation for arbitrary interpolation points.

From (4.3.6) it follows that the divided difference $[u_1, \ldots, u_{k+s}, x]f$ is equal to the divided difference of $w_k(x)$ at $[u_{k+1}, \ldots, u_{k+s}, x]f$, so that by (4.3.7) we can write

$$w_{k+s}(z) = \int_0^1 \cdots \int_0^{t_{s-1}} u_k^{(s)}\big[u_{k+1} + (u_{k+2} - u_{k+1})t_1 + \cdots + (z - u_{k+s})t_k\big]\, dt_s \cdots dt_1.$$

If all points $u_{k+1}, \ldots, u_{k+s}$ all tend to the limit $z$ and $w_k(z)$ has a continuous $s$th derivative at $z$, then it holds that

$$w_{k+s}(z) = w_k^{(s)}(z) \int_0^1 \cdots \int_0^{t_{s-1}} dt_s \cdots dt_1 = \frac{w_k^{(s)}(z)}{s!}.$$

It can be shown that if $f \in C^k$, the divided differences belong to $C^{k+1-\max r_i}$, and that the interpolation polynomial has this kind of differentiability with respect to the $u_i$ (*nota bene*, if the "groups" do not coalesce further).

**Example 4.3.3.**

As established above, the usual recurrence formula for divided differences can still be used for the construction of the divided-difference table in case of multiple points. The limit process is just applied to the divided differences, for example,

$$[x_0, x_0]f = \lim_{x_1\to x_0} \frac{f(x_1) - f(x_0)}{x_1 - x_0} = f'(x_0),$$

$$[x_0, x_0, x_1]f = \frac{[x_0, x_0]f - [x_0, x_1]f}{x_0 - x_1} = \frac{f'(x_0) - [x_0, x_1]f}{x_0 - x_1}.$$

For the interpolation problem considered in Example 4.3.1 we construct the generalized divided-difference table, where $x_1 \neq x_0$.

$$\begin{array}{llllll}
x_0 & f_0 & & & & \\
 & & f_0' & & & \\
x_0 & f_0 & & \frac{1}{2}f_0'' & & \\
 & & f_0' & & [x_0, x_0, x_0, x_1]f & \\
x_0 & f_0 & & [x_0, x_0, x_1]f & & [x_0, x_0, x_0, x_1, x_1]f \\
 & & [x_0, x_1]f & & [x_0, x_0, x_1, x_1]f & \\
x_1 & f_1 & & [x_0, x_1, x_1]f & & \\
 & & f_1' & & & \\
x_1 & f_1 & & & &
\end{array}$$

The interpolating polynomial is

$$p(x) = f_0 + (x - x_0) f_0' + (x - x_0)^2 \frac{1}{2} f_0'' + (x - x_0)^3 [x_0, x_0, x_0, x_1] f,$$
$$+ (x - x_0)^3 (x - x_1)[x_0, x_0, x_0, x_1, x_1] f,$$

and the remainder is

$$f(x) - p(x) = [x_0, x_0, x_0, x_1, x_1, x] f (x - x_0)^3 (x - x_1)^2$$
$$= f^{(5)}(\xi_x)(x - x_0)^3 (x - x_1)^2 / 5.$$

An important case of the Hermite interpolation problem is when the given data are $f_i = f(x_i)$, $f_i' = f'(x_i)$, $i = 0, 1$. We can then write the interpolation polynomial as

$$p(x) = f_0 + (x - x_0)[x_0, x_1] f + (x - x_0)(x - x_1)[x_0, x_0, x_1] f$$
$$+ (x - x_0)^2 (x - x_1)[x_0, x_0, x_1, x_1] f.$$

Set $x_1 = x_0 + h$ and $x = x_0 + \theta h$, and denote the remainder $f(x) - p(x)$ by $R_T$. Then one can show (Problem 4.3.1) that

$$\begin{aligned} p(x) &= f_0 + \theta \Delta f_0 + \theta(1 - \theta)(h f_0' - \Delta f_0) \\ &\quad - \theta^2 (1 - \theta) \Big[ (h f_0' - \Delta f_0) + (h f_1' - \Delta f_0) \Big] \\ &= (1 - \theta) f_0 + \theta f_1 + \theta(1 - \theta) \Big[ (1 - \theta)(h f_0' - \Delta f_0) - \theta (h f_1' - \Delta f_0) \Big]. \end{aligned} \tag{4.3.11}$$

For $x \in [x_0, x_1]$ we get the error bound

$$|f(x) - p(x)| \le \frac{1}{384} h^4 \max_{x \in [x_0, x_1]} |f^{(4)}(x)|. \tag{4.3.12}$$

Setting $t = 1/2$, we get the useful approximation formula

$$f\left( \frac{1}{2}(x_0 + x_1) \right) \approx \frac{1}{2}(f_0 + f_1) + \frac{1}{8} h (f_0' - f_1'). \tag{4.3.13}$$

## 4.3.2 Complex Analysis in Polynomial Interpolation

We shall encounter multivalued functions: the logarithm and the square root. For each of these we choose that branch which is positive for large positive values of the argument $z$. They will appear in such contexts that we can then keep them nonambiguous by forbidding $z$ to cross the interval $[-1, 1]$. (We can, however, allow $z$ to approach that interval.)

We first consider the general problem of interpolation of an analytic function, at an arbitrary sequence of nodes $u_1, u_2, \ldots, u_n$ in $\mathbf{C}$. Multiple nodes are allowed. Set

$$\Phi_n(z) = (z - u_1)(z - u_2) \cdots (z - u_n), \quad z, u_j \in \mathbf{C}.$$

Let $\mathcal{D}$ be a simply connected open domain in $\mathbf{C}$ that contains the point $u$ and the nodes. The interpolation problem is to find the polynomial $p^* \in \mathcal{P}_n$, that is determined by the

conditions $p^*(u_j) = f(u_j)$, $j = 1 : n$, or the appropriate Hermite interpolation problem in the case of multiple nodes. We know that $p^*$ depends linearly on $f$. In other words, there exists a linear mapping $L_n$ from some appropriate function space so that $p^* = L_n f$.

Assume that $f$ is an analytic function in the closure of $\mathcal{D}$, perhaps except for a finite number of poles $p$. A pole must not be a node. Recall the elementary identity (4.2.20) in Example 4.2.3,

$$\frac{1}{z-u} = \sum_{j=1}^{n} \frac{\Phi_{j-1}(u)}{\Phi_j(z)} + \frac{\Phi_n(u)}{\Phi_n(z)(z-u)}, \tag{4.3.14}$$

which is valid also for multiple nodes. Introduce the linear operator $K_n$,

$$(K_n f)(u) = \frac{1}{2\pi i} \int_{\partial\mathcal{D}} \frac{\Phi_n(u) f(z)}{\Phi_n(z)(z-u)}\, dz, \tag{4.3.15}$$

multiply the above identity by $f(z)/(2\pi i)$, and integrate along the boundary of $\mathcal{D}$ to get

$$\frac{1}{2\pi i} \int_{\partial\mathcal{D}} \frac{f(z)}{z-u}\, dz = \sum_{j=1}^{n} \Phi_{j-1}(u) \frac{1}{2\pi i} \int_{\partial\mathcal{D}} \frac{f(z)}{\Phi_j(z)}\, dz + (K_n f)(u). \tag{4.3.16}$$

First, assume that $f$ has no poles in $\mathcal{D}$. By applying the residue theorem to the first two integrals, we note that the equation has the same structure as Newton's unique interpolation formula with exact remainder (4.2.6),

$$f(u) = \sum_{j=1}^{n} \Phi_{j-1}(u)[u_1, \ldots, u_j]f + \Phi_n(u)[u_1, u_2, \ldots, u_n, u]f.$$

Matching terms in the last two formulas we obtain

$$(f - L_n f)(u) = (K_n f)(u),$$

and, if $f$ is analytic in $\mathcal{D}$, also the formula for the divided-difference,

$$[u_1, u_2, \ldots, u_j]f = \frac{1}{2\pi i} \int_{\partial\mathcal{D}} \frac{f(z)\, dz}{(z-u_1)\cdots(z-u_j)}. \tag{4.3.17}$$

From this we obtain an upper bound for the divided-difference

$$\left|[u_1, u_2, \ldots, u_j]f\right| = \frac{L}{2\pi} \frac{\max_{z\in\partial\mathcal{D}} |f(z)|}{\min_{z\in\partial\mathcal{D}} |(z-u_1)\cdots(z-u_j)|}, \tag{4.3.18}$$

where $L = \int_{\partial\mathcal{D}} |dz|$.

If there are poles $p \in \mathcal{D}$ with residues $\operatorname{res}_f(p)$, we must add $\sum_p \operatorname{res}_f(p)/(z-p)$ to the left-hand side of (4.3.17) and $\sum_p \operatorname{res}_f(p) \sum_j \Phi_{j-1}(u)/\Phi_j(p)$ to the right-hand side. By (4.3.14) this is, however, equivalent to subtracting

$$\sum_p \operatorname{res}_f(p) \Phi_n(u)/\Phi_n(p)(p-u)$$

from the right-hand side. This yields the following theorem.

**Theorem 4.3.3.**

*Assume that $f$ is analytic in the closure of the open region $\mathcal{D}$, perhaps except for a finite number of poles, $p \in \mathcal{D}$, with residues $\operatorname{res}_f(p)$. $\mathcal{D}$ also contains the interpolation points $u_1, u_2, \ldots, u_n$, as well as the point $u$. Multiple nodes are allowed. The point $u$ and the poles $p$ must not be nodes, and $p \neq u$.*

*Then the interpolation error can be expressed as a complex integral,*

$$(f - L_n f)(u) = (K_n f)(u) - \Phi_n(u) \sum_p \frac{\operatorname{res}_f(p)}{\Phi_n(p)(p-u)}, \tag{4.3.19}$$

*where $K_n$ is defined by* (4.3.15) *and the sum (which may be void) is extended over the poles of $f$ in $\mathcal{D}$.*

This theorem is valid when the interpolation points $u_j$ are in the complex plane, although we shall here mainly apply it to the case when they are located in the interval $[-1, 1]$. An important feature is that this expression for the interpolation error requires no knowledge of $f^{(n)}(z)$.

For the case of *distinct nodes* we have, by the residue theorem,

$$(K_n f)(u) = \sum_{j=1}^{n} \frac{\Phi_n(u)}{(u_j - u)\Phi_n'(u_j)} f(u_j) + f(u), \tag{4.3.20}$$

where the sum, with reversed sign, is Lagrange's form of the interpolation polynomial $L_n f(u)$.

**Example 4.3.4** (*Chebyshev Interpolation*).

In Chebyshev interpolation on the interval $[-1, 1]$, the nodes are the zeros of the Chebyshev polynomials $T_n(u)$, and

$$\Phi_n(u) = 2^{1-n} T_n(u).$$

Recall the notation and results in the discussion of Chebyshev expansions in Sec. 3.2.3.

In this example we assume that $f(u)$ is analytic in $\mathcal{D}$. We shall, by means of the integral $K_n f$, show that it yields almost as accurate an approximation as the first $n$ terms of the Chebyshev expansion of the same function.

Let $\mathcal{D} = \mathcal{E}_R$, where $\mathcal{E}_R$ is the ellipse with foci at $\pm 1$, and where $R$ is the sum of the semiaxis. Let $z \in \partial\mathcal{E}_R$ and $u \in [-1, 1]$. Then $|T_n(u)| \leq 1$, and it can be shown (Problem 4.3.13) that

$$|T_n(z)| \geq \tfrac{1}{2}(R^n - R^{-n}), \qquad |z - u| \geq a_R - 1, \qquad \int_{\partial\mathcal{E}_R} |dz| \leq 2\pi a_R.$$

If we assume that $|f(z)| \leq M_R$ for $z \in \partial\mathcal{E}_R$, a straightforward estimation of the line integral (4.3.20) gives

$$\begin{aligned} |f(u) - (L_n f)(u)| = |(K_n f)(u)| &\leq \frac{1}{2\pi} \frac{2^{1-n} M_R 2\pi a_R}{2^{1-n}\frac{1}{2}(R^n - R^{-n})(a_R - 1)} \\ &\leq \frac{2 M_R a_R R^{-n}}{(1 - R^{-2n})(a_R - 1)}. \end{aligned} \tag{4.3.21}$$

We see that the interpolation error converges at the same exponential rate as the truncation error of the Chebyshev expansion (see Sec. 3.2.3). If $f(z)$ has a singularity arbitrarily close to the interval $[-1, 1]$, then $R \approx 1$, and the exponential convergence rate will be very poor.

This above analysis can be extended to the case of multiple poles by including higher derivatives of $f$. This yields the general Lagrange formula valid also when there are interpolation points of multiplicity greater than one:

$$p(u) = \sum_{i=1}^{m} \left[ \sum_{k=0}^{r_i-1} \frac{1}{k!} \frac{d^k}{du^k} \frac{f(u)}{\prod_{\substack{q=1 \\ q\neq i}}^{m} (u-u_q)^{r_q}} \Bigg|_{u=u_i} (u-u_i)^k \right] \prod_{\substack{j=1 \\ j\neq i}}^{m} (u-u_j)^{r_j}. \tag{4.3.22}$$

Clearly, when $r_i = 1$ for all $i = 1 : m$, this formula equals the usual Lagrange interpolation formula. It can be written in the simpler form

$$p(u) = \sum_{i=1}^{n} \sum_{k=0}^{r_i-1} f^{(k)}(u_i) L_{ik}(u), \tag{4.3.23}$$

where $L_{ik}(u)$ are **generalized Lagrange polynomials**. These can be defined starting from the auxiliary polynomials

$$l_{ik}(u) = \frac{(u-u_i)^k}{k!} \prod_{\substack{j=1 \\ j\neq i}}^{n} \left( \frac{u-u_j}{u_i-u_j} \right)^{r_j}, \quad i = 1 : m, \quad k = 0 : r_i - 1.$$

Set $L_{i,r_i-1} = l_{i,r_i-1}$, $i = 1 : m$, and form recursively

$$L_{ik}(u) = l_{ik}(u) - \sum_{v=k+1}^{r_i-1} l_{ik}^{(v)}(u_i) L_{i,v}(u), \quad k = r_i - 2 : -1 : 0.$$

It can be shown by induction that

$$L_{ik}^{(\sigma)}(u_j) = \begin{cases} 1 & \text{if } i = j \text{ and } k = \sigma, \\ 0 & \text{otherwise.} \end{cases}$$

Hence the $L_{ik}$ are indeed the appropriate polynomials.

**Example 4.3.5.**

When $r_i = 2$, $i = 1 : n$, the Hermite interpolating polynomial is the polynomial of degree less than $n = 2m$, which agrees with $f(u)$ and $f'(u)$ at $u = u_i$, $i = 1 : m$. We have

$$p(u) = \sum_{i=1}^{n} (f(u_i) L_{i0}(u) + f'(u_i) L_{i1}(u)),$$

where $L_{ik}(u)$ can be written in the form

$$L_{i1}(u) = (u-u_i) l_i(u)^2, \qquad L_{i0}(u) = (1 - 2l_i'(u_i)(u-u_i)) l_i(u)^2,$$

and $l_i(u)$, $i = 1 : m$, are the elementary Lagrange polynomials:

$$l_{ik}(u) = \frac{(u-u_i)^k}{k!} \prod_{\substack{j=1 \\ j\neq i}}^{n} \left( \frac{u-u_j}{u_i-u_j} \right)^{r_j}, \quad i = 1 : n, \quad k = 0 : r_i - 1.$$

### 4.3.3 Rational Interpolation

The rational interpolation problem is to determine a rational function

$$r_{m,n}(z) = \frac{P_m(z)}{Q_n(z)} \equiv \frac{\sum_{j=0}^{m} p_j z^j}{\sum_{j=0}^{n} q_j z^j}, \tag{4.3.24}$$

with numerator of degree $m$ and denominator of degree $n$ so that at distinct points $x_0, x_1, \ldots, x_n$ agrees with a function $f$

$$r_{m,n}(x_i) = f_i, \quad i = 0 : N, \quad N = m + n. \tag{4.3.25}$$

Rational approximation is often superior to polynomial approximation in the neighborhood of a point at which the function has a singularity. Since the coefficients can be determined only up to a common nonzero factor, the number of unknown constants in (4.3.24) equals the number of interpolation points $N + 1 = m + n + 1$.

A *necessary condition* for (4.3.25) to hold clearly is that the linearized condition

$$P_m(x_i) - f_i Q_n(x_i) = 0, \quad i = 0 : N, \tag{4.3.26}$$

is satisfied, i.e., for $i = 0 : m + n$,

$$p_0 x_i + p_1 x_i + \cdots + p_m x_i^m - f_i(q_0 x_i + q_1 x_i + \cdots + q_n x_i^n) = 0. \tag{4.3.27}$$

This is a homogeneous linear system of $(m+n+1)$ equations for the $(m+n+2)$ coefficients in $P_m$ and $Q_n$. If we introduce the Vandermonde matrices

$$A = \begin{pmatrix} 1 & x_0 & \ldots & x_0^m \\ 1 & x_1 & \ldots & x_1^m \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & \ldots & x_N^m \end{pmatrix}, \qquad B = \begin{pmatrix} 1 & x_0 & \ldots & x_0^n \\ 1 & x_1 & \ldots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & \ldots & x_N^n \end{pmatrix},$$

this system can be written in matrix form as

$$(\, A \quad FB \,) \begin{pmatrix} p \\ q \end{pmatrix} = 0, \qquad F = \operatorname{diag}(f_0, f_1, \ldots, f_N), \tag{4.3.28}$$

where $p = (p_0, p_1, \ldots, p_m)^T, q = (q_0, q_1, \ldots, q_n)^T$. This is a homogeneous linear system of $N + 1$ equations in $N + 2$ unknowns. Such a system always has a nontrivial solution. Moreover, since $A$ has full column rank, we must have $q \neq 0$ for such a solution.

We note that the rational interpolant is fully determined by the denominator polynomial. By (4.3.26) $P_m(x)$ is the unique polynomial determined by the interpolation conditions

$$P(x_i) = f_i Q(x_i), \quad i = 1 : m.$$

While for polynomial interpolation there is always a unique solution to the interpolation problem, this cannot be guaranteed for rational interpolation, as shown by the example below. A further drawback is that the denominator polynomial $Q(x)$ may turn out to have zeros in the interval of interpolation.

**Example 4.3.6.**

Assume that we want to interpolate the four points

| $x$ | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $y$ | 2 | 3/2 | 4/5 | 1/2 |

by a rational function

$$r(x) = \frac{p_0 + p_1 x + p_2 x^2}{q_0 + q_1 x}.$$

Then we must solve the homogeneous linear system

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix} - \begin{pmatrix} 2 & 0 \\ 3/2 & 3/2 \\ 4/5 & 8/5 \\ 1/2 & 3/2 \end{pmatrix} \begin{pmatrix} q_0 \\ q_1 \end{pmatrix} = 0.$$

Setting $p_2 = 1$ we find the solution $p_0 = 8$, $p_1 = -6$, $q_0 = 4$, $q_1 = -2$. The corresponding rational function

$$r(x) = \frac{8 - 6x + x^2}{4 - 2x} = \frac{(4 - x)(2 - x)}{2(2 - x)}$$

has the common factor $(2 - x)$ and is *reducible* to $f_{2,1} = (4 - x)/2$. The original form is indeterminate $0/0$ at $x = 2$, while the reduced form does not take on the prescribed value at $x = 2$.

As shown in the above example, for given data $(x_0, f_0), \ldots, (x_n, f_n)$ there can be certain points $x_j$ where the given function value $f_j$ cannot be attained. Such a point $x_j$ is called **unattainable**. This can occur only if $x_j$ is a zero of the denominator $Q_n(x)$. From (4.3.26) it also follows that $P_m(x_j) = 0$. Hence the polynomials $P(x)$ and $Q(x)$ have a common factor $(x - x_j)^d$, where $d$ is chosen maximal. The polynomials pair

$$P^*(x) = \frac{P(x)}{(x - x_j)^d}, \qquad Q^*(x) = \frac{Q(x)}{(x - x_j)^d} \tag{4.3.29}$$

then satisfies (4.3.26) for all points $x_k \neq x_j$. Since $d$ was chosen maximal it holds that $Q^*(x_j) \neq 0$, and the value $P^*(x_j)/Q^*(x_j)$ must be finite. But since when $Q_n(x_j) = 0$, (4.3.26) is satisfied for *any choice of* $f_j$, one cannot expect the rational function to interpolate a particular value at $x_j$.

If there are unattainable points, then the polynomials defined in (4.3.29) only solve the linearized equations in the attainable points. It can also happen that the polynomials given by the linearized equations have a common factor $(x - x^*)^d$, $d \geq 1$, with $x^* \neq x_i$, $i = 1 : n$. In this case all polynomials of the form

$$P^*(x) = \frac{P(x)}{(x - x_j)^\nu}, \qquad Q^*(x) = \frac{Q(x)}{(x - x_j)^\nu}, \quad \nu = 0 : d,$$

satisfy the linearized system and the matrix $(\,A \quad FB\,)$ in (4.3.28) has at most rank $N + 1 - d$. Conversely we have the following result.

**Theorem 4.3.4.**

*If the rank of the matrix $(\,A \quad FB\,)$ equals $N + 1 - d$, then there exists a unique solution $p, q$ corresponding to polynomials $P^*$ and $Q^*$ with degrees at most $m - d$ and $n - d$. Further, all solutions have the form*

$$r(x) = \frac{s(x)P^*(x)}{s(x)Q^*(x)},$$

*where $s(x)$ is a polynomial of degree at most $d$. A point $x_j$ is unattainable if and only if $Q^*(x_j) = 0$.*

***Proof.*** See Schaback and Werner [313, Theorem 12.1.8]. □

Another complication of rational interpolation is that zeros may occur in $Q_n(x)$, which are not common to $P_m(x)$. These zeros correspond to poles in $r(x)$, which if they lie inside the interval $[\min x_k, \max x_k]$ may cause trouble. In general it is not possible to determine *a priori* if the given data $(x_k, f_k)$ will give rise to such poles. Neither do the coefficients in the representation of $r(x)$ give any hint of such occurrences.

An algorithm similar to Newton's algorithm can be used for finding rational interpolants in continued fraction form. Set $v_0(x) = f(x)$, and use a sequence of substitutions:

$$v_k(x) = v_k(x_k) + \frac{x - x_k}{v_{k+1}(x)}, \quad k = 0, 1, 2, \ldots. \tag{4.3.30}$$

The first two substitutions give

$$f(x) = v_0(x) = v_0(x_0) + \frac{x - x_0}{v_1(x)} = v_0(x_0) + \cfrac{x - x_0}{v_1(x_1) + \cfrac{x - x_1}{v_2(x)}}.$$

In general this gives a continued fraction

$$f(x) = a_0 + \cfrac{x - x_0}{a_1 + \cfrac{x - x_1}{a_2 + \cfrac{x - x_2}{a_3 +}}} \ldots = a_0 + \frac{x - x_1}{a_1 +} \, \frac{x - x_2}{a_2 +} \, \frac{x - x_3}{a_3 +} \ldots, \tag{4.3.31}$$

where $a_k = v_k(x_k)$, and we have used the compact notation introduced in Sec. 3.5.1. This becomes an identity if the expansion is terminated by replacing $a_n$ in the last denominator

by $a_n + (x - x_n)/v_{n+1}(x)$. If we set $x = x_k$, $k \le n$, then the fraction terminates before the residual $(x - x_n)/v_{n+1}(x)$ is introduced. This means that setting $1/v_{k+1} = 0$ will give a rational function which agrees with $f(x)$ at the points $x_i$, $i = 0 : k \le n$, assuming that the constants $a_0, \ldots, a_k$ exist. These continued fractions give a sequence of rational approximations $f_{k,k}$, $f_{k+1,k}$, $k = 0, 1, 2, \ldots$.

Introducing the notation

$$v_k(x) = [x_0, x_1, \ldots, x_{k-1}, x]\phi \tag{4.3.32}$$

we have $a_k = [x_0, x_1, \ldots, x_{k-1}, x_k]\phi$. Then by (4.3.30) we have

$$[x]\phi = f(x), \qquad [x_0, x]\phi = \frac{x - x_0}{[x]\phi - [x_0]\phi} = \frac{x - x_0}{f(x) - f(x_0)},$$
$$[x_0, x_1, x]\phi = \frac{x - x_1}{[x_0, x]\phi - [x_0, x_1]\phi},$$

and in general

$$[x_0, x_1, \ldots, x_{k-1}, x]\phi = \frac{x - x_{k-1}}{[x_0, \ldots, x_{k-2}, x]\phi - [x_0, \ldots, x_{k-2}, x_{k-1}]\phi}. \tag{4.3.33}$$

Therefore, we also have

$$a_k = \frac{x_k - x_{k-1}}{[x_0, \ldots, x_{k-2}, x_k]\phi - [x_0, \ldots, x_{k-2}, x_{k-1}]\phi}. \tag{4.3.34}$$

We call the quantity defined by (4.3.34) the $k$th **inverse divided difference** of $f(x)$. Note that certain inverse differences can become infinite if the denominator vanishes. They are, in general, *symmetrical only in their last two arguments*.

The inverse divided differences of a function $f(x)$ can conveniently be computed recursively and arranged in a table similar to the divided-difference table.

$$\begin{array}{llllll}
x_1 & f(x_1) & [x_1]\phi & & & \\
 & & & [x_1, x_2]\phi & & \\
x_2 & f(x_2) & [x_2]\phi & & [x_1, x_2, x_3]\phi & \\
 & & & [x_2, x_3]\phi & & [x_1, x_2, x_3, x_4]\phi \\
x_3 & f(x_3) & [x_3]\phi & & [x_2, x_3, x_4]\phi & \\
 & & & [x_3, x_4]\phi & & \\
x_4 & f(x_4) & [x_4]\phi & & &
\end{array}$$

Here the upper diagonal elements are the desired coefficients in the expansion (4.3.31).

**Example 4.3.7.**

Assume that we want to interpolate the four points given in Example 4.3.6 and the additional points (4, 6/17). Forming the inverse differences we get the following table.

| $x_i$ | $f_i$ | $\phi_1$ | $\phi_2$ | $\phi_3$ | $\phi_4$ |
|---|---|---|---|---|---|
| 0 | 2 | | | | |
| | | $-2$ | | | |
| 1 | 3/2 | | 3 | | |
| | | 5/3 | | 0 | |
| 2 | 4/5 | | $\infty$ | | $-5$ |
| | | $-2$ | | $-1/5$ | |
| 3 | 1/2 | | $-7$ | | |
| | | $-17/7$ | | | |
| 4 | 6/17 | | | | |

This gives a sequence of rational approximations. If we terminate the expansion

$$f_{2,2} = 2 + \frac{x}{-2+}\,\frac{x-1}{3+}\,\frac{x-2}{0+}\,\frac{x-3}{-5}$$

after $a_3$ we recover the solution of the previous example. Note that the degeneracy of the approximation is shown by the entry $a_3 = 0$. Adding the last fraction gives the (degenerate) approximation

$$f_{2,2} = \frac{2+x}{1+x^2}.$$

It is verified directly that this rational function interpolates all the given points.

Because the inverse differences are not symmetric in all their arguments the **reciprocal differences** are often preferred. These are recursively defined by

$$[x_i]\rho = f_i, \qquad [x_i, x_{i+1}]\rho = \frac{x_i - x_{i+1}}{f_i - f_{i+1}}, \tag{4.3.35}$$

$$[x_i, x_{i+1}, \ldots, x_{i+k}]\rho = \frac{x_i - x_{i+k}}{[x_i, \ldots, x_{i+k-1}]\rho - [x_{i+1}, \ldots, x_{i+k}]\rho + [x_{i+1}, \ldots, x_{i+k-1}]\rho}. \tag{4.3.36}$$

See Hildebrand [201, p. 406]. While this formula is less simple than (4.3.34), the reciprocal differences are *symmetric functions of all their arguments*. The symmetry permits the calculation of the $k$th reciprocal difference from any two $(k-1)$th reciprocal differences having $k-1$ arguments in common, together with the $(k-2)$th reciprocal difference formed with this argument. Using (4.3.36) a reciprocal difference table may be constructed.

The coefficients in the continued fraction (4.3.31) can then be determined by an interpolation formula due to Thiele [350]:

$$a_0 = f_0, \qquad a_1 = [x_0, x_1]\rho, \qquad a_2 = [x_0, x_1, x_2]\rho - f_0,$$
$$a_3 = [x_0, x_1, x_2, x_3]\rho - [x_0, x_1]\rho, \ldots.$$

The formulas using inverse or reciprocal differences are useful if one wants to determine the coefficients of the rational approximation, and use it to compute approximations

for several arguments. If one only wants the value of the rational interpolating function for a single argument, then it is more convenient to use an alternative algorithm of Neville type. This is the case in the $\rho$-algorithm, which is a convergence acceleration procedure using rational interpolation to extrapolate to infinity with the same degree in the numerator and denominator.

If we consider the sequence of rational approximations of degrees $(m, n)$

$$(0,0),\quad (0,1),\quad (1,1),\quad (1,2),\quad (2,2),$$

the following recursive algorithm results (Stoer and Bulirsch [338, Sec. 2.2]):
For $i = 0, 1, 2, \ldots$, set $T_{i,-1} = 0$, $T_{i,0} = f_i$, and

$$T_{ik} = T_{i,k-1} + \frac{T_{i,k-1} - T_{i-1,k-1}}{\dfrac{x - x_{i-k}}{x - x_i}\left[1 - \dfrac{T_{i,k-1} - T_{i-1,k-1}}{T_{i,k-1} - T_{i-1,k-2}}\right] - 1}, \quad 1 \le k \le i. \tag{4.3.37}$$

As in Neville interpolation the calculations can be arranged in a table of the following form.

| $(m,n)$ | $(0,0)$ | $(0,1)$ | $(1,1)$ | $(1,2)$ | $\cdots$ |
|---|---|---|---|---|---|
| | $f_1 = T_{1,0}$ | | | | |
| 0 | | $T_{2,1}$ | | | |
| | $f_2 = T_{2,0}$ | | $T_{3,2}$ | | |
| 0 | | $T_{3,1}$ | | $T_{4,3}$ | |
| | $f_2 = T_{2,0}$ | | $T_{4,2}$ | $\vdots$ | $\ddots$ |
| 0 | | $T_{4,1}$ | $\vdots$ | | |
| | $f_4 = T_{4,0}$ | $\vdots$ | | | |
| $\vdots$ | $\vdots$ | | | | |

Here any entry is determined by a rhombus rule from three entries in the preceding two columns. Note that it is easy to add a new interpolation point in this scheme.

As shown by Berrut and Mittelmann [23], every rational interpolant can be written in barycentric form

$$r(x) = \sum_{k=0}^{N} \frac{u_k}{x - x_k} f_k \Big/ \sum_{k=0}^{N} \frac{u_k}{x - x_k}. \tag{4.3.38}$$

Let $q_k = Q_n(x_k)$, $k = 1 : N$, be the values of the denominator at the nodes. Then the barycentric representation of the denominator $Q_n(x)$ is

$$Q_n(x) = \prod_{i=0}^{N}(x - x_i) \sum_{k=0}^{N} \frac{w_k}{x - x_k} q_k, \quad w_k = 1 \Big/ \prod_{i=0}^{N}(x_k - x_i).$$

Hence $r(x)$ can be written as in (4.3.38), where $u_k = w_k q_k$ is the weight corresponding to the node $x_k$. Since $w_k \neq 0$ for all $k$, it follows that $q_k = 0$ at a node if and only if the corresponding weight equals zero.

The barycentric form has the advantage that the barycentric weights give information about possible unattainable points. However, the determination of the parameter vector $u = (u_0, u_1, \ldots, u_N)^T$ is more complicated. An elimination method for the computation of $u$ in $O(n^3)$ operations is given by Berrut and Mittelmann [23].

### 4.3.4 Multidimensional Interpolation

Polynomial interpolation for functions of several independent variables are generally more difficult than the one-dimensional case. There is in general a lack of uniqueness. In particular, it may not suffice to require that the interpolation points are distinct; see Problem 4.3.9 (b).

As a simple example, consider the problem of interpolating a function given at three distinct points $p_i = (x_i, y_i)$, $i = 1:3$, by a linear function in two variables,

$$p(x, y) = c_1 + c_2 x + c_3 y.$$

This leads to the linear system $Vc = f$, where

$$V = \begin{pmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{pmatrix}, \qquad c = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}, \qquad f = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}.$$

The interpolation problem has exactly one solution if $V$ is nonsingular, i.e., when $\det(V) \neq 0$. But $\frac{1}{2}\det(V)$ is just the area of the triangle with vertices $(x_i, y_i)$, $i = 1:3$. If this area is zero, then the three points lie on a line and the problem has either infinitely many solutions or no solution.

Much of the theory for univariate interpolation can be generalized to multidimensional interpolation problems provided that the function is specified on a Cartesian (tensor) product grid. For simplicity, we first concentrate on functions $f(x, y)$ of two variables, but the extension to more dimensions is not difficult. Assume that we are given function values

$$f_{ij} = f(x_i, y_j), \quad i = 1:n, \quad j = 1:m, \tag{4.3.39}$$

where $x_i$, $i = 1:n$, and $y_j$, $j = 1:m$, are distinct points. We seek a polynomial $p(x, y)$ of degree at most $n-1$ in $x$ and at most $m-1$ in $y$ that interpolates these values. Such a polynomial has the form

$$p(x, y) = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} c_{ij} x^i y^j, \tag{4.3.40}$$

where the $mn$ coefficients $c_{ij}$ are to be determined. Since the number of coefficients equal the number of interpolatory conditions we can expect the polynomial $p(x, y)$ to be uniquely determined. To show this it suffices to show that any polynomial $q(x, y)$ of degree $n-1$ in $x$ and $m-1$ in $y$ that vanishes at the $mn$ distinct points $(x_i, y_j)$, $i = 1:n$, and $y_j$, $j = 1:m$, must vanish identically.

If we want to compute $p(x, y)$ for given values of $x$ and $y$, then we can proceed as follows. For each $j = 1:m$, use univariate interpolation to determine the values $p(x, y_j)$, where $p(x, y)$ is the interpolation polynomial in (4.3.40). Next, the $m$ values $p(x, y_j)$,

$j = 1 : m$, are interpolated using univariate interpolation, which determines $p(x, y)$. Note that since the points $x_i$ and $y_j$ are distinct all univariate interpolation polynomials are uniquely determined. It is clear that we will obtain the same result, whether we interpolate first for $x$ and then for $y$ or vice versa. Clearly this approach can also be used in more than two dimensions.

In many cases we are not satisfied with obtaining $p(x, y)$ for specific values of $x$ and $y$, but want to determine $p(x, y)$ as a polynomial in $x$ and $y$. We can then use the above procedure *algebraically* to derive a Newton formula for a tensor product interpolation problem in two variables. We set $[x_i; y_j]f = f(x_i, y_j)$, and define bivariate divided differences $[x_1, \ldots, x_v; y_1, \ldots, y_\mu]f$, by recurrences, separately for each variable. We start by forming, for each $y_j$, $j = 1 : m$, divided differences with respect to the $x$ variable:

$$\begin{array}{llll} [x_1; y_1]f & [x_1, x_2; y_1]f & \cdots & [x_1, x_2, \ldots, x_n; y_1]f, \\ [x_1; y_2]f & [x_1, x_2; y_2]f & \cdots & [x_1, x_2, \ldots, x_n; y_2]f, \\ \vdots & \vdots & & \vdots \\ [x_1; y_m]f & [x_1, x_2; y_m]f & \cdots & [x_1, x_2, \ldots, x_n; y_m]f. \end{array}$$

These are used to form the Newton polynomials

$$p(x, y_j) = \sum_{i=1}^{n} [x_1, \ldots, x_i; y_j]f \prod_{v=1}^{i-1} (x - x_v), \quad j = 1 : m,$$

which give, for any $x$, the values of the interpolation polynomial $p(x, y)$ for $y_1, \ldots, y_m$. Next we form in each column above the divided differences with respect to $y$:

$$\begin{array}{llll} [x_1; y_1]f & [x_1, x_2; y_1]f & \cdots & [x_1, \ldots, x_n; y_1]f, \\ [x_1; y_1, y_2]f & [x_1, x_2; y_1, y_2]f & \cdots & [x_1, \ldots, x_n; y_1, y_2]f, \\ \vdots & \vdots & & \vdots \\ [x_1; y_1, \ldots, y_m]f & [x_1, x_2; y_1, \ldots, y_m]f & \cdots & [x_1, \ldots, x_n; y_1, \ldots, y_m]f. \end{array}$$

If these $nm$ divided differences are used for Newton interpolation in the $y$ variable, we obtain Newton's interpolation formula in two variables,

$$p(x, y) = \sum_{i=1}^{n} \sum_{j=1}^{m} [x_1, \ldots, x_i; y_1, \ldots, y_j]f \prod_{v=1}^{i-1} (x - x_v) \prod_{\mu=1}^{j-1} (y - y_\mu), \tag{4.3.41}$$

where empty products have the value 1. Note that it is indifferent in which order the divided differences are formed. We could equally well have started to form divided differences with respect to $y$.

Remainder formulas can be derived from the corresponding one-dimensional error formulas; see Isaacson and Keller [208, Sec. 6.6]. For $f$ sufficiently smooth there exist

values $\xi, \xi', \eta, \eta'$ such that

$$R(x, y) = \frac{\partial^n f(\xi, y)}{\partial x^n} \frac{\prod_{\nu=1}^{n}(x - x_\nu)}{n!} + \frac{\partial^m f(x, \eta)}{\partial y^m} \frac{\prod_{\mu=1}^{m}(y - y_\mu)}{m!} \tag{4.3.42}$$

$$- \frac{\partial^{n+m} f(\xi', \eta')}{\partial x^n \partial y^m} \frac{\prod_{\nu=1}^{n}(x - x_\nu)}{n!} \frac{\prod_{\mu=1}^{m}(y - y_\mu)}{m!}. \tag{4.3.43}$$

Lagrange's interpolation formula can also be generalized for the tensor product case. We have

$$p(x, y) = \sum_{i=1}^{n} \sum_{j=1}^{m} f(x_i, y_j) \prod_{\substack{\nu=1 \\ \nu \neq i}}^{n} \frac{(x - x_\nu)}{(x_i - x_\nu)} \prod_{\substack{\mu=1 \\ \mu \neq j}}^{n} \frac{(y - y_\mu)}{(y_j - y_\mu)}. \tag{4.3.44}$$

Clearly $p(x, y)$ assumes the values $f(x_i, y_j)$ for $i = 1 : n$, $j = 1 : m$. As the polynomial is of degree $n - 1$ in $x$ and $m - 1$ in $y$, it must equal the unique interpolating polynomial. Therefore, the remainder must also be the same as for the Newton formula. The Lagrange formula (4.3.44) is easily extended to three and more variables.

The interpolation problem we have treated so far specifies the maximum degree of $p(x, y)$ in $x$ and $y$ separately. Instead we could specify the total degree to be at most $n - 1$. Then the interpolation polynomial must have the form

$$p(x, y) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-i-1} b_{ij} x^i y^j. \tag{4.3.45}$$

There are $\frac{1}{2}n(n + 1)$ coefficients to determine in (4.3.45). We shall show that with the "triangular" array of interpolation points $(x_i, y_j)$, $i + j = 1 : n$, the interpolation polynomial is uniquely determined.

The Newton formula (4.3.41) can be generalized to the case when for each $i, i = 1 : n$, the interpolation given points are $(x_i, y_j)$, $j = 1 : m_i$ with $1 \leq m_i \leq m$, with a slightly more complicated remainder formula. A particularly interesting case is when $m_i = n - i + 1$, i.e., the interpolation points form a triangle. This gives rise to the interpolating polynomial

$$p(x, y) = \sum_{2 \leq i+j \leq n+1} [x_1, \ldots, x_i; y_1, \ldots, y_j] f \prod_{\nu=1}^{i-1}(x - x_\nu) \prod_{\mu=1}^{j-1}(y - y_\mu), \tag{4.3.46}$$

with remainder formula

$$R(x, y) = \sum_{i=1}^{n+1} \frac{\partial^n f(\xi_i, \eta_i)}{\partial x^i \partial y^{n-i}} \frac{\prod_{\nu=1}^{i-1}(x - x_\nu)}{(i - 1)!} \frac{\prod_{\mu=1}^{n-i}(y - y_\mu)}{(n - i)!}. \tag{4.3.47}$$

This formula is due to Biermann [26].

Interpolation formulas for equidistant points $x_i = x_0 + ih$ and $y_j = y_0 + jk$ can readily be obtained from Newton's formula (4.3.41). Using the points $(x_i, y_j)$, $i = 0 : n$, $i = 0 : m$, we get

$$p(x_0 + ph, y + qk) = \sum_{i=0}^{n} \sum_{j=0}^{m} \binom{p}{i} \binom{q}{j} \Delta_x^i \Delta_y^j f(x_0, y_0). \tag{4.3.48}$$

**Example 4.3.8.**

Formulas for equidistant points can also be obtained by using the operator formulation of Taylor's expansion:

$$\begin{aligned} f(x_0+h, y_0+k) &= \exp\left(hD_x + kD_y\right) f(x_0, y_0) \\ &= f_{0,0} + \left(hD_x + kD_y\right) f_{0,0} \\ &\quad + \left(h^2D_x^2 + 2hkD_xD_y + k^2D_y^2\right) f_{0,0} + O(h^3 + k^3). \end{aligned} \tag{4.3.49}$$

An interpolation formula, exact for all functions $p(x, y)$ in (4.3.40) with $m = n = 3$, can be obtained by replacing in Taylor's formula the derivatives by difference approximations valid for quadratic polynomials,

$$\begin{aligned} f(x_0+ph, y_0+qh) \approx\ & f_{0,0} + \frac{1}{2}p(f_{1,0} - f_{-1,0}) + \frac{1}{2}q(f_{0,1} - f_{0,-1}) \\ & + \frac{1}{2}p^2(f_{1,0} - 2f_{0,0} + f_{-1,0}) \\ & + \frac{1}{4}pq(f_{1,1} - f_{1,-1} - f_{-1,1} + f_{-1,-1}) \\ & + \frac{1}{2}q^2(f_{0,1} - 2f_{0,0} + f_{0,-1}). \end{aligned} \tag{4.3.50}$$

This formula uses function values in nine points. (The proof of the expression for approximating the mixed derivative $D_xD_yf_{0,0}$ is left as an exercise; see Problem 4.3.11.)

An important case, to be treated in Sec. 5.4.4, is interpolation formulas in two or more dimensions with function values specified on the vertices and sides of a simplex. These play a fundamental role in the finite element method.

### 4.3.5 Analysis of a Generalized Runge Phenomenon

In this section we make a more detailed theoretical and experimental study of the Runge phenomenon (see Sec. 4.2.6). We then study interpolation at an infinite equidistant point set from the point of view of complex analysis. This interpolation problem, which was studied by Whittaker and others in the beginning of the twentieth century, became revived and modified in the middle of the same century under the name of the **Shannon sampling theorem**, with important applications to communication theory.

It is well known that the Taylor series of an analytic function converges at an exponential rate inside its circle of convergence, while it diverges at an exponential rate outside. The circle of convergence passes through the nearest singularity. We shall see that similar results hold for certain interpolation processes. In general, the domains of convergence are not disks but bounded by level curves of a logarithmic potential, related to the asymptotic distribution of the interpolation points.

For the sake of simplicity, we now confine the discussion to the case when the points of interpolation are located in the standard interval $[-1, 1]$, but we are still interested in the evaluation of the polynomials in the complex domain. Part of the discussion can, however, be generalized to a case when the interpolation points are on an arc in the complex plane.

We shall study interpolation processes which are *regular* in the following sense. Let the nodes $t_{n,j}$, $j = 1 : n$, $n = 1, 2, 3, \ldots$, be such that there exists an increasing continuously differentiable function $q : [a, b] \mapsto [-1, 1]$ such that

$$q(a + (b-a)(j-1)/n) < t_{n,j} \le q(a + (b-a)\,j/n),$$

i.e., one node in each of $n$ subintervals of $[-1, 1]$. More precisely, we assume that $q'(\tau) > 0$, $\tau \in (0, 1)$, while $q'(0)$, $q'(1)$ may be zero. Suppose that $z \notin [-1, 1]$, but $z$ may be close to this interval. Then

$$\begin{aligned} \frac{1}{n} \ln \Phi_n(z) &= \frac{1}{n} \sum_{j=1}^{n} \ln (z - t_{n,j}) \\ &\to \psi(z) := \frac{1}{b-a} \int_a^b \ln (z - q(\tau))\, d\tau, \quad (n \to \infty). \end{aligned} \tag{4.3.51}$$

The crucial factors of the interpolation error are $\Phi_n(u)/\Phi_n(p)$ and $\Phi_n(u)/\Phi_n(z)$. We now obtain a fundamental approximation formula:

$$\begin{aligned} \Phi_n(u)/\Phi_n(z) &= \exp\big(\ln \Phi_n(u) - \ln \Phi_n(z)\big) \\ &= \exp n\big(\psi(u) - \psi(z) + o(1)\big), \quad (n \to \infty). \end{aligned} \tag{4.3.52}$$

If $u$ and $z$ are bounded away from the nodes, the $o(1)$-term is of marginal importance; it is basically $O(1/n)$. For $u \in [-1, 1]$ $\Phi_n(u)$ and $\ln |\Phi_n(u)|$ are oscillatory, and this formula is there to be considered as a one-sided approximate error bound; see Figure 4.3.2 and the more detailed preparatory discussion leading up to Proposition 4.3.6.

We now make a variable transformation in order to be able to utilize results of classical potential theory. Put $q(\tau) = t \in [-1, 1]$, $dt = q'(\tau) d\tau$. Then we can define an asymptotic **node density** for the process,

$$w(t) = \frac{1}{q'_\tau(\tau(t))(b-a)}, \qquad w(t)dt = \frac{d\tau}{b-a}, \tag{4.3.53}$$

where $w$ is the derivative of the inverse function of $q$ divided by $b - a$. Then

$$\psi(z) = \int_{-1}^{1} \ln(z-t) w(t)\, dt, \quad w \in C(-1, 1), \quad w(t) > 0, \quad \int_{-1}^{1} w(t)\, dt = 1, \tag{4.3.54}$$

and $\psi(z)$ is analytic in the whole plane outside the interval $[-\infty, 1]$. Its real part is the **logarithmic potential** of the density $w$ with reversed sign,

$$\frac{1}{n} \ln |\Phi_n(z)| \approx \Re\psi(z) = \int_{-1}^{1} \ln |z - t| w(t)\, dt. \tag{4.3.55}$$

$\Re\psi(z)$ is a harmonic function for all $z = x + iy \notin [-1, 1]$, i.e., it satisfies Laplace's equation $\partial^2 \Re\psi/\partial x^2 + \partial^2 \Re\psi/\partial y^2 = 0$.

The function $\frac{1}{n} \ln |\Phi_n(z)|$ is itself the logarithmic potential of a discrete distribution of equal weights $1/n$ at the nodes $t_{j,n}$, but it is less pleasant to deal with than $\Re\psi(z)$. For example, it becomes $-\infty$ at the nodes while, according to classical results of potential heory, $\Re\psi(z)$ *is continuous and single-valued everywhere, also on the interval* $[-1, 1]$ (see Figure 4.3.1). The imaginary part, however, is not single-valued. It becomes single-valued

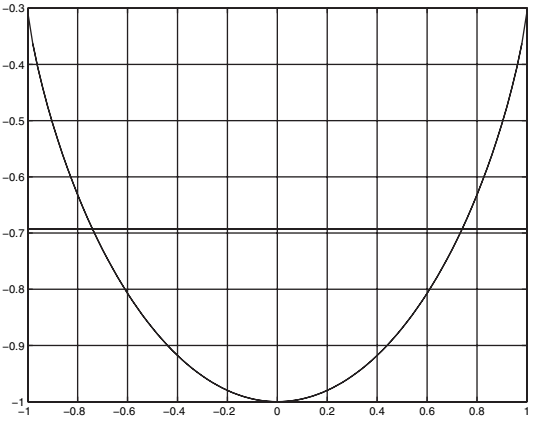



**Figure 4.3.1.** $\Re\psi(u),\ u \in [-1, 1]$, *for the processes based on equidistant nodes and on Chebyshev nodes. For Chebyshev nodes, the convergence properties are the same over the whole interval* $[-1, 1]$, *because* $\Re\psi(u) = -\ln 2 = -0.693$ *for all* $u \in [-1, 1]$. *For equidistant nodes, the curve, which is based on* (4.3.61), *partly explains why there are functions* $f$ *such that the interpolation process diverges fast in the outer parts of* $[-1, 1]$ *and converges fast in the central parts (often faster than Chebyshev interpolation).*

if we cut the complex plane along the real axis from $-\infty$ to 1, but it tends to $+\pi$ or $-\pi$, depending on whether the cut is approached from above or from below.

Another result from classical potential theory that will be useful later reads

$$\psi'(x - 0i) - \psi'(x + 0i) = 2\pi i\, w(x), \quad x \in [-1, 1]. \tag{4.3.56}$$

For $|z| \gg 1$, $\psi(z)$ depends only weakly on the node distribution, and the level curves approach circles since, by (4.3.55),

$$\begin{aligned}\psi(z) &= \int_{-1}^{1} \Big(\ln z + \ln(1 - z^{-1}t)\Big) w(t)\, dt \\ &= \ln z - z^{-1} \int_{-1}^{1} t\, w(t)\, dt - \frac{1}{2} z^{-2} \int_{-1}^{1} t^2 w(t)\, dt \ldots . \end{aligned} \tag{4.3.57}$$

Note that the coefficient of $z^{-1}$ vanishes for a symmetric node distribution.

We have

$$\frac{\partial \Re\psi}{\partial y} = \frac{\Re \partial\psi}{\partial y} = \Re \int_{-1}^{1} \frac{i}{x - t + iy} w(t)\, dt = \int_{-1}^{1} \frac{y}{(x - t)^2 + y^2)} w(t)\, dt > 0$$

if $y > 0$. Then $\Re\psi(x + iy)$ is an increasing function of $y$ for $y > 0$.

$$\frac{\partial \Re\psi}{\partial x} = \frac{\Re \partial\psi}{\partial x} = \Re \int_{-1}^{1} \frac{1}{x - t + iy} w(t)\, dt = \int_{-1}^{1} \frac{x - t}{(x - t)^2 + y^2)} w(t)\, dt > 0$$

if $x > 1$. Then $\Re\psi(x + iy)$ is an increasing function of $x$ for $x > 1$. Similarly $\Re\psi(x + iy)$ is a decreasing function of $y$ for $y < 0$, and a decreasing function of $x$ for $x < -1$.

We define the region

$$\mathcal{D}(v) = \big\{z \in \mathbf{C} : \Re\psi(z) < \Re\psi(v)\big\}. \tag{4.3.58}$$

Set $P^* = \max_{x\in[-1,1]} \Re\psi(x)$, and suppose that $\Re\psi(v) > P^*$. It then follows that $\mathcal{D}(v)$ is a simply connected domain, and the level curve

$$\partial\mathcal{D}(v) = \big\{z : \Re\psi(z) = \Re\psi(v)\big\}$$

encloses $[-1, 1]$. Suppose that $a' > a$ and $a' > P^*$. Then the level curve $\{z : \Re\psi(z) = a\}$ is strictly inside the level curve $\{z : \Re\psi(z) = a'\}$. (The proof of these statements essentially utilizes the minimum principle for harmonic functions and the fact that $\Re\psi(z)$ is a regular harmonic function outside $[-1, 1]$ that grows to $\infty$ with $|z|$.)

Suppose that $f(z)$ is analytic for $|z| = R$, where $R$ is so large that we can set $\psi(z) \approx \ln z$; see (4.3.57). We then obtain, by Theorem 4.3.3, (4.3.20), and (4.3.52),

$$\ln\big|(f - L_n f)(u)\big| \approx n\big(\Re\psi(u) - \ln R + o(1)\big) + \ln M(R). \tag{4.3.59}$$

**Example 4.3.9** (*An Entire Function*).

For $f(z) = \exp z^\alpha$, $\alpha > 0$, $\ln M(R) = R^\alpha$. For a fixed $n$ large enough so that the $o(1)$-term can be neglected, the bound in (4.3.59) has a minimum for $R = (n/\alpha)^{1/\alpha}$. Inserting this into (4.3.59) we obtain

$$\ln\big|(f - L_n f)(u)\big| \le n\Big(\Re\psi(u) - \frac{1}{\alpha}(\ln n + 1 + \ln\alpha) + o(1)\Big),$$

which shows that, in this example, the convergence is faster than exponential, and depends rather weakly on the node distribution.

The following estimate comes as a consequence,

$$\big|(f - L_n f)(u)\big| = \Phi_n(u) n^{-n} e^{-n+o(1)n},$$

but this is no surprise. If $u$ is real and $\alpha = 1$, it follows directly from the remainder term (4.2.10) in interpolation

$$(f - L_n f)(u) = \Phi_n(u) e^{\xi_u}/n!,$$

together with Stirling's formula (3.2.36).

The technique used in this example is, however, very general. It can be used for complex $u$, and for more complicated entire functions.

**Example 4.3.10** (*Functions with Poles*).

We choose $\mathcal{D} = \mathcal{D}(v) = \{z \in \mathbf{C} : \Re\psi(z) < \Re\psi(v)\}$ and assume that $f(z)$ has two conjugate poles, $p$, $\bar{p}$, $\Re\psi(p) < \Re\psi(v)$. (There may be other poles outside $\mathcal{D}(v)$.) Consider the error formula of Theorem 4.3.3. For $n \gg 1$, the ratio of $K_n f$ to the contribution from the poles is $\exp(-n(\Re\psi(v) - \Re\psi(p) + o(1)))$; we can thus neglect $K_n f(u)$. It follows that

$$\begin{aligned}\big|(f - L_n f)(u)\big| &\approx \left|\Phi_n(u) \sum_p \frac{\mathrm{res}_f(p)}{\Phi_n(p)(p-u)}\right| \\ &= \exp n\big(\Re\psi(u) - \Re\psi(p) + o(1)\big) \cdot \left|\frac{2\mathrm{res}_f(p)}{(p-u)}\right|. \end{aligned} \tag{4.3.60}$$

An important conclusion is that *the level curve of the logarithmic potential through the pole $p$ separates the points $u \notin [-1, 1]$, where the interpolation process converges from the points where it diverges.* This **separation statement** holds under more general conditions than we have in this example.

For $u \in [-1, 1]$ there are, however, interpolation processes that may converge in an enumerable point set that is everywhere dense in $[-1, 1]$, even though $\Re\psi(u) > \Re\psi(p)$ in large parts of this interval. It is doubtful if such a process can be regular in the sense defined above, but it can be a subprocess of a regular process.[133] Figure 4.3.3 may give hints how the above separation statement is to be modified in order to make sense also in such a case. The smooth curves of Figure 4.3.3 have been computed by means of (4.3.60) for Runge's example $f(z) = 1/(1 + 25z^2)$ with equidistant nodes; see Example 4.3.11.

We now consider two node distributions.

**Example 4.3.11** (*Equidistant Interpolation*).

In this case we may take $q(\tau) = \tau$, $\tau \in [-1, 1]$, $t \in [-1, 1]$, hence $w(t) \equiv 1/2$. For this **equidistant case** we have, if $z \notin [-\infty, 1]$,

$$\psi(z) = \frac{1}{2}\int_{-1}^{1} \ln(z - t)\, dt = \frac{1}{2}\Big((1 - z)\ln(z - 1) + (1 + z)\ln(z + 1)\Big) - 1.$$

The real part $\Re\psi(z)$ is, however, single-valued and continuous everywhere, as mentioned in the comment after (4.3.55). Some level curves are shown in Figure 4.3.2. Note that the tangent of a curve is discontinuous at the intersection with $[-1.1]$. On the imaginary axis,

$$\Re\psi(iy) = \frac{1}{2}\ln(1 + y^2) + y\left(\text{sign}\,(y)\,\frac{\pi}{2} - \arctan y\right) - 1.$$

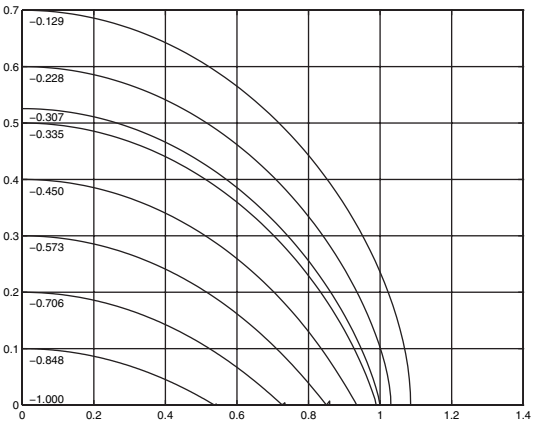



**Figure 4.3.2.** *Some level curves of the logarithmic potential for $w(t) \equiv \frac{1}{2}$, $t \in [-1, 1]$. Due to the symmetry only one quarter of each curve is shown. The value of $\Re\psi(z)$ on a curve is seen to the left close to the curve. It is explained in Example* 4.3.11 *how the curves have been computed.*

[133] For example, a process generated by successive bisections of the interval $[-1, 1]$ can be a subprocess of one of the equidistant processes studied in next example.

When $z \to x \in [-1, 1]$, from any direction, $\Re\psi(z)$ tends to

$$\Re\psi(x) = \frac{1}{2}\big((1-x)\ln(1-x) + (1+x)\ln(1+x)\big) - 1. \tag{4.3.61}$$

The level curve of $\Re\psi(z)$ that passes through the points $\pm 1$ intersects the imaginary axis at the points $\pm iy$, determined by the equation $\Re\psi(iy) = \Re\psi(1) = \ln 2 - 1$, with the root $y = 0.5255$. Theorem 4.3.5 (below) will tell us that $L_n f(x) \to f(x)$ for all $x \in (-1, 1)$, if $f(z)$ is analytic inside and on this contour.

In the classical example of Runge, $f(z) = 1/(1 + 25z^2)$ has poles inside this contour at $z = \pm 0.2i$. The separation statement in Example 4.3.10 told us that the level curve of $\Re\psi(z)$ which passes through these poles will separate between the points, where the interpolation process converges and diverges. Its intersection with the real axis is determined by the equation $\Re\psi(x) = \Re\psi(0.2i) = -0.70571$. The roots are $x = \pm 0.72668$; see also Figure 4.3.2.

The theory based on the logarithmic potential is of asymptotic nature, and one may ask how relevant it is when the number of nodes is of a reasonable size for practical computation. After all, the behavior of the interpolation error between the nodes is quite different from the smooth logarithmic potential. Figure 4.3.3 indicates, however, that the prediction of this theory can be rather realistic when we ask for local maxima of the modulus of the interpolation error on the real axis.

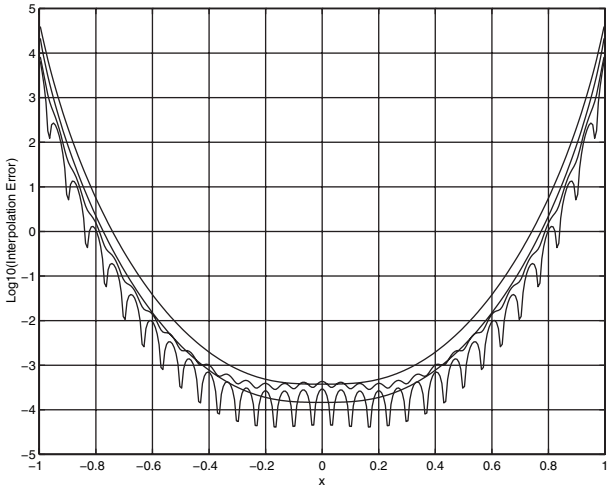



**Figure 4.3.3.** $\log_{10}|(f - L_n f)(u)|$ *for Runge's classical example* $f(u) = 1/(1 + 25u^2)$ *with* 30 equidistant nodes *in* $[-1, 1]$. *The oscillating curves are the empirical interpolation errors (observed at* 300 *equidistant points), for* $u = x$ *in the lower curve and for* $u = x + 0.02i$ *in the upper curve; in both cases* $x \in [-1, 1]$. *The smooth curves are the estimates of these quantities obtained by the logarithmic potential model; see Examples* 4.3.10 *and* 4.3.11.

**Example 4.3.12** (*Chebyshev Interpolation Revisited*).

In this example we have $q(t) = \cos(\pi\tau)$, $\tau \in [-1, 0]$. By (4.3.53)

$$w(t) = \frac{1}{-\pi \sin \pi\tau} = \frac{1}{\pi}(1 - t^2)^{-\frac{1}{2}}.$$

Moreover,[134]

$$\Phi_n(z) = 2^{1-n} T_n(z) = 2^{-n}(s^n + s^{-n}),$$

where $z = \frac{1}{2}(s + s^{-1})$, $s = z + \sqrt{z^2 - 1}$. According to our convention about the choice of branch for the square root $|s| \geq 1$. Hence,

$$\Re\psi(z) = \lim \frac{1}{n} \ln |\Phi_n(z)| - \ln 2 = \ln \frac{|s|}{2} = \ln |z + \sqrt{z^2 - 1}| - \ln 2.$$

As in the previous example, $\Re\psi(z)$ is single-valued and continuous everywhere, while $\psi(z)$ is unique for $z \notin [-\infty, 1]$ only. Therefore, the family of confocal ellipses $\partial\mathcal{E}_R$ with foci at $\pm 1$ are the level curves of $\Re\psi(z)$. In fact, the interior of $\mathcal{E}_R$ equals $\mathcal{D}(\ln R - \ln 2)$. The family includes, as a limit case ($R \to 1$), the interval $[-1, 1]$, on which $\Re\psi(z) = \ln R - \ln 2 \to -\ln 2 = -0.693$. Note that as $z \to \cos\phi \in [-1, 1]$,

$$|z + \sqrt{(z^2 - 1)}| = |\cos\phi + i \sin\phi| = 1, \quad \forall\phi.$$

Some level curves are shown in Figure 4.3.4.

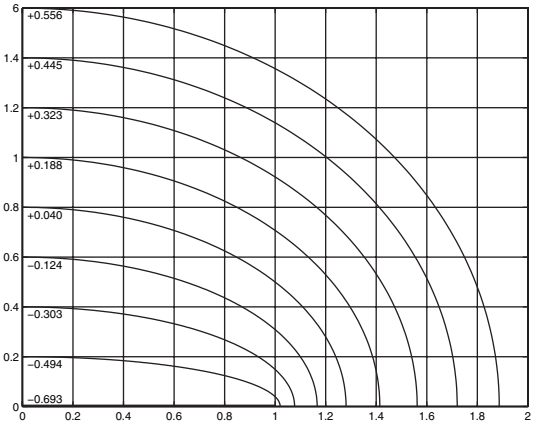



**Figure 4.3.4.** *Some level curves of the logarithmic potential associated with Chebyshev interpolation. They are ellipses with foci at* $\pm 1$. *Due to the symmetry only a quarter is shown of each curve. The value of* $\Re\psi(z)$ *for a curve is seen to the left, close to the curve.*

Note that $\Re\psi(z)$ is smaller in $[-1, 1]$ than anywhere else; this confirms the conclusion in Sec. 4.3.2 that, for any function analytic in a region $\mathcal{D}$ that encloses $[-1, 1]$, Chebyshev interpolation converges at the same exponential rate in (almost) all points $u$ close to $[-1, 1]$.

[134]The complex variable denoted $w$ in Sec. 3.2.3 is now denoted $s$, in order to avoid a collision with the density $w(t)$.

For the classical Runge case, we have

$$|(f - L_n f)(u)| \approx e^{n(\psi(0)-\psi(0.02i))} = e^{n(-0.693+0.494)} = e^{-0.199n}.$$

This is very different from the equidistant case which diverges if $|u| > 0.72668$ but at the central subinterval has, by Figure 4.3.2,

$$|(f - L_n f)(u)| \approx e^{n(-1+0.706)} = e^{-0.294n},$$

which is better than Chebyshev interpolation.

Also note that if $z \notin [-1, 1]$, we can, by the definition of $\Re\psi(z)$ as a Riemann sum (see (4.3.51)), find a sequence $\{\epsilon_n\}$ that decreases monotonically to zero such that

$$\frac{1}{n}\big|\ln \Phi_n(z) - \psi(z)\big| < \epsilon_n, \quad z \notin [-1, 1]. \tag{4.3.62}$$

It is conceivable that the same sequence can be used for all $z$ on a curve that does not touch the interval $[-1, 1]$. (This can be proved, but the proof is omitted.)

We can only claim a *one-sided inequality* if we allow that $u \in [-1, 1]$

$$\frac{1}{n}\Re\big(\ln \Phi_n(u) - \psi(u)\big) < \epsilon_n, \quad u \in \mathbf{C}. \tag{4.3.63}$$

(Recall that $\Re \ln \Phi_n(u) = -\infty$ at the nodes.) We can use the same sequence for $z$ and $u$. We can also say that $|\Phi_n(u)|$ *behaves like* $\exp((\Re\psi(u) \pm \delta)n)$ *outside the immediate vicinity of the interpolation points*.

**Theorem 4.3.5.**

*Assume that the nodes are in* $[-1, 1]$*, and that* $[-1, 1]$ *is strictly inside a simply connected domain* $\mathcal{D} \supseteq \mathcal{D}(v)$.

*If* $f(\zeta)$ *is analytic in the closure of* $\mathcal{D}$*, then the interpolation error* $(L_n f)(u) - f(u)$ *converges like an exponential to* 0 *for any* $u \in \mathcal{D}(v)$.

***Proof.*** By Theorem 4.3.3, $f(u) - (L_n f)(u) = I_n(u)$, where

$$I_n(u) = \frac{1}{2\pi i}\int_{\partial\mathcal{D}} \frac{\Phi_n(u) f(z)}{\Phi_n(z)(z - u)}\, dz. \tag{4.3.64}$$

Note that $\Re\psi(z) \geq \Re\psi(v)$, because $\mathcal{D} \supseteq \mathcal{D}(v)$. Then, by (4.3.62) and (4.3.63),

$$\big|\Phi_n(u)/\Phi_n(z)\big| < \exp n\big(\Re\psi(u) - \Re\psi(v) + 2\epsilon_n\big).$$

Let $|f(z)| \leq M$. For any $u \in \mathcal{D}(v)$, we can choose $\delta > 0$ such that $\Re\psi(u) < \Re\psi(v) - 3\delta$, $|z - u| > \delta$. Next, choose $n$ large enough so that $\epsilon_n < \delta$. Then

$$|f(u) - (L_n f)(u)| < \frac{1}{2\pi} M \exp n(-3\delta + 2\delta) \int_{\partial\mathcal{D}(v)} \frac{|dz|}{\delta} \leq \frac{K \exp(-n\delta)}{\delta},$$

where $K = K(v)$ does not depend on $n$, $\delta$, and $u$. Hence, the convergence is exponential. ∎

We shall now state without proof a complement and a kind of converse to Theorem 4.3.5, for functions $f(z)$ that have simple poles in $\mathcal{D}(v)$.

**Proposition 4.3.6.**

*Assume that* $[-1, 1]$ *is strictly inside a domain* $\mathcal{D} \supset \mathcal{D}(v)$*, and that* $f(\zeta)$ *is analytic in the closure of* $\mathcal{D}$*, except for a finite number of simple poles* $p$ *in the interior, all with the same value of* $P(p)$*.*

*Outside the interval* $[-1, 1]$*, the curve* $\partial\mathcal{D}(p)$ *then separates the points, where the sequence* $\{(L_n f)(u)\}$ *converges, from the points, where it diverges. The behavior of* $|(L_n f)(u) - f(u)|$*, when* $u \in \mathcal{D}(v)$*,* $n \gg 1$*, is roughly described by the formula*

$$|(f - L_n f)(u)| \approx K|\Phi_n(u)|e^{(-P(p)\pm\delta)n} / \max_p (1/|p - u|). \tag{4.3.65}$$

There are several interpolation processes with interpolation points in $[-1, 1]$ that converge for all $u \in [-1, 1]$, when the condition of analyticity is replaced by a more modest smoothness assumption, for example, $f \in C^p$. This is the case when the sequence of interpolation points are the zeros of the orthogonal polynomials which belong to a density function that is continuous and strictly positive in $[-1, 1]$. We shall prove the following result.

**Proposition 4.3.7.**

*Consider an interpolation process where the interpolation points have a (perhaps unknown) asymptotic density function* $w(x)$*,* $x \in [-1, 1]$*. Assume that, for some* $k \geq 1$*,*

$$(L_n f - f)(x) \to 0 \quad \forall x \in [-1, 1], \quad \forall f \in C^k[-1, 1]$$

*as* $n \to \infty$*. Then the logarithmic potential* $\Re\psi(x)$ *must be constant in* $[-1, 1]$*, and the density function must be the same as for Chebyshev interpolation, i.e.,* $w(x) = \frac{1}{\pi}(1 - x^2)^{-1/2}$*.*

***Proof.*** Let $f(z)$ be analytic in some neighborhood of $[-1, 1]$, for example, any function with a pole at a point $p$ (arbitrarily) close to this interval. A fortiori, for such a function our interpolation process must converge at all points $u$ in some neighborhood of the interval $[-1, 1]$.

Suppose that $\Re\psi(x)$ is not constant, and let $x_1$, $x_2$ be points such that $\Re\psi(x_1) < \Re\psi(x_2)$. We can then choose the pole $p$ so that $\Re\psi(x_1) + \delta < \Re\psi(p) < \Re\psi(x_2) - \delta$. By Proposition 4.3.6, the process would then diverge at some point $u$ arbitrarily close to $x_2$. This contradiction shows that $\Re\psi(x)$ must be constant in $[-1, 1]$, $\Re\psi(x) = a$ (say).

This gives a Dirichlet problem for the harmonic function $\Re\psi(z)$, $z \notin [-1, 1]$, which has a unique solution, and one can verify that the harmonic function $\Re\psi(z) = a + \Re\ln(z + \sqrt{z^2 - 1})$ satisfies the boundary condition. We must also determine $a$. This is done by means of the behavior as $z \to \infty$. We find that

$$\begin{aligned}\Re\psi(z) &= a + \Re\ln\left(z + z(1 - z^{-2})^{1/2}\right)\\ &= a + \Re\ln\left(2z - O(z^{-1})\right) = a + \Re\ln z + \ln 2 - O(z^{-2}).\end{aligned}$$

This is to be matched with the result of the discussion of the general logarithmic potential in the beginning of Sec. 4.3.5. In our case, where we have a symmetric distribution and

$\int_1^1 w(x)\,dx = 1$, we obtain $\Re\psi(z) = \Re \ln z + O(z^{-2})$. The matching yields $a = -\ln 2$. Finally, by (4.3.56), we obtain after some calculation $w(x) = (1-x^2)^{-1/2}$. □

Our problem is related to a more conventional application of potential theory, namely the problem of finding the electrical charge distribution of a long insulated charged metallic strip through $[-1, 1]$, perpendicular to the $(x, y)$-plane. Such a plate will be equipotential. Suppose that such a distribution is uniquely determined by the total charge.[135] The charge density must then be proportional to the density in our example. This density $w$ corresponds to $q(\tau) = \cos \pi t$, i.e., $w(x) = \frac{1}{\pi}(1-x^2)^{-1/2}$. *A fascinating relationship of electricity to approximation*!

The above discussion is related to the derivations and results concerning the asymptotic distribution of the zeros of orthogonal polynomials, given in the standard monograph Szegö [347].

## Review Questions

**4.3.1** What is meant by Lagrange–Hermite (osculatory) interpolation? Prove the uniqueness result for the Lagrange–Hermite interpolation problem.

**4.3.2** (a) Write down the confluent Vandermonde matrix for the Lagrange–Hermite cubic interpolation problem.

(b) Express the divided difference $[x_0, x_0, x_1, x_1]f$ in terms of $f_0$, $f_0'$, and $f_1, f_1'$.

**4.3.3** What are the *inverse* divided differences of a function $f(x)$ used for? How are they defined? Are they symmetric in all their arguments?

**4.3.4** Give the complex integral formula for the interpolation error of a function that is analytic in a domain. Give the assumptions, and explain your notations. Give the interpolation error for the case with poles in the domain also.

**4.3.5** How is bilinear interpolation performed? What is the order of accuracy?

**4.3.6** What is Chebyshev interpolation, and what can be said about its convergence for analytic functions?

## Problems and Computer Exercises

**4.3.1** (a) Construct the divided-difference scheme for the simplest Lagrange–Hermite interpolation problem, where the given data are $f(x_i)$, $f'(x_i)$, $i = 0, 1$; $x_1 = x_0 + h$. Prove all the formulas concerning this problem that are stated at the end of Sec. 4.3.1.

(b) For $f(x) = (1+x)^{-1}$, $x_0 = 1, x_1 = 1.5$, compute $f(1.25)$ by Lagrange–Hermite interpolation. Compare the error bound and the actual error.

[135] It is unique, but we have not proved this.

(c) Show that for Lagrange–Hermite interpolation

$$|f'(x) - p'(x)| \le \frac{h^3}{72\sqrt{3}}\Big(\max_{x\in[x_0,x_1]} |f^{(iv)}(x)| + O(h|f^{(v)}(x)|)\Big).$$

*Hint:* $\frac{d}{dx}[x_0, x_0, x_1, x_1, x]f = [x_0, x_0, x_1, x_1, x, x]f \le \cdots$.

**4.3.2** Let $p \in \mathcal{P}_6$ be the Lagrange–Hermite interpolation polynomial to the data $x_i$, $y(x_i)$, $y'(x_i)$, $x_i = x_0 + ih$, $i = 1, 2, 3$.

(a) Find the remainder term, and show that the interpolation error for $x \in [x_1, x_3]$ does not exceed $h^6 \max |f^{(6)}(x)|/4860$ in magnitude.

(b) Write a program that computes $p(x_1 + 2jh/k)$, $j = 0 : k$.

*Comment:* This is one of several possible procedures for starting a multistep method for an ordinary differential equation $y' = f(x, y)$. Two steps with an accurate one-step method provide values of $y$, $y'$, and this program then produces starting values ($y$ only) for the multistep method.

**4.3.3** Give a short and complete proof of the uniqueness of the interpolation polynomial for distinct points, by the use of the ideas in the proof of Theorem 4.3.1.

**4.3.4** Derive an approximate formula for $f'(x_0)$ when the values $f(x_{-1})$, $f(x_0)$, $f(x_1)$ are given at three *nonequidistant* points. Give an approximate remainder term. Check the formula and the error estimate on an example of your own choice.

**4.3.5** Consider the problem of finding a polynomial $p \in \mathcal{P}_4$ that interpolates the data $f(1)$, $f(-1)$, $f''(1)$, $f''(-1)$. The new feature is that there are no first derivatives. Show that this problem is uniquely solvable.

*Hint:* Show that using the power basis one obtains a linear system of the form $Mc = f$, where $f = (f_1, f_{-1}, f_1'', f_{-1}'')^T$, and

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 0 & 0 & 2 & 6 \\ 0 & 0 & 2 & -6 \end{pmatrix}$$

with $\det(M) = 48$.

**4.3.6** (a) Given a sequence of function values $f_1, f_2, f_3, \ldots$ at equidistant points $x_j = x_0 + jh$, assume that $\min f_j = f_n$, and let $p(x)$ be the quadratic interpolation polynomial determined by $f_{n-1}$, $f_n$, $f_{n+1}$. Show that

$$\min p(x) = f_n - \frac{(\mu\delta f_n)^2}{2\delta^2 f_n} \quad \text{at} \quad x = x_n - h\frac{\mu\delta f_n}{\delta^2 f_n},$$

and that the error of the minimum value can be bounded by $\max |\Delta^3 f_j|/\sqrt{243}$, where $j$ is in some neighborhood of $n$. Why and how is the estimate of $x$ less accurate?

(b) Write a handy program that includes the search for all local maxima and minima. Sketch or work out improvements of this algorithm, perhaps with ideas of inverse interpolation and with cubic interpolation, and perhaps for nonequidistant data.

**4.3.7** We use the notations and assumptions of Theorem 4.3.3.

Using the representation of the interpolation operator as an integral operator, show that

$$(L_n f)(x) = \frac{1}{2\pi i}\int_{\partial\mathcal{D}} K(x,z)\frac{f(z)}{\Phi(z)}\,dz, \qquad K(x,z) = \frac{\Phi(x)-\Phi(z)}{(x-z)},$$

also if $x \notin \mathcal{D}$. Note that $K(x,z)$ is a polynomial, symmetric in the two variables $x$, $z$.

**4.3.8** If $f \in \mathcal{P}_n$, then $f - L_n f$ is clearly zero. How would you deduce this obvious fact from the integral $(K_n f)(u)$?

*Hint:* Let $\partial D$ be the circle $|z| = R$. Make the standard estimation, and let $R \to \infty$.

**4.3.9** (a) Show the **bilinear interpolation** formula

$$p(x_0+ph, y_0+qk) = (1-p)(1-q)f_{0,0} + p(1-q)f_{1,0} + (1-p)qf_{0,1} + pqf_{1,1} \tag{4.3.66}$$

with error bound

$$\frac{1}{2}\big(p(1-p)h^2|f_{xx}| + q(1-q)k^2|f_{yy}|\big) + O(k^2h^2),$$

where $f_{xx}$ and $f_{yy}$ denote partial derivatives.

(b) Compute by bilinear interpolation $f(0.5, 0.25)$ when

$$f(0,0) = 1, \quad f(1,0) = 2, \quad f(0,1) = 3, \quad f(1,1) = 5.$$

**4.3.10** (a) Consider the interpolation problem: Given $x_i$, $y_i$, $f_i$, $i = 1:6$, find $c = (c_1, c_2, c_3, c_4, c_5, c_6)^T$ so that $p(x_i, y_i; c) = f_i$, $i = 1:6$, where

$$p(x, y; c) = c_1 + c_2x + c_3y + c_4x^2 + c_5xy + c_6y^2.$$

Choose $x_i$, $y_i$, $f_i$, $i = 1:6$, by 18 independent random numbers, solve the linear system $p(x_i, y_i; c) = f_i$, $i = 1:6$, and look at max $|c_i|$. Repeat this (say) 25 times. You have a fair chance to avoid singular cases, or cases where max $|c_i|$ is very large.

(b) Now choose $(x_i, y_i)$ as six distinct points on some circle in $\mathbf{R}^2$, and choose $f_i$ at random. This should theoretically lead to a singular matrix. Explain why, and find experimentally the rank (if your software has convenient commands or routines for that). Find a general geometric characterization of the sextuples of points $(x_i, y_i)$, $i = 1:6$, that lead to singular interpolation problems.

*Hint:* Brush up on your knowledge of conic sections.

**4.3.11** Derive a formula for $f''_{xy}(0,0)$ using $f_{ij}$, $|i| \le 1$, $|j| \le 1$, which is exact for all quadratic functions.

**4.3.12** (Bulirsch and Rutishauser (1968))

(a) The function $\cot x$ has a singularity at $x = 0$. Use values of $\cot x$ for $x = 1°$, $2°, \ldots, 5°$, and rational interpolation of order (2,2) to determine an approximate value of $\cot x$ for $x = 2.5°$, and its error.

(b) Use polynomial interpolation for the same problem. Compare the result with that in (a).

**4.3.13** Check the omitted details of the derivations in Example 4.3.4. Compare the bounds for Chebyshev interpolation and Chebyshev expansion for $R = 1 + k/n$.

**4.3.14** Check the validity of (4.3.56) on the Chebyshev and the equidistant cases. Also show that $\int_{-1}^{1} w(x)\,dx = 1$, and check the statements about the behavior of $P(z)$ for $|z| \gg 1$.

**4.3.15** *A generalization of Runge's example.* Let $f$ be an analytic function for which the poles nearest to $[-1, 1]$ are a pair of complex conjugate poles at an arbitrary place on the imaginary axis. Consider interpolation with nodes in $[-1, 1]$.

(a) Suppose that equidistant interpolation converges at $u = 1$. Is it true that Chebyshev interpolation converges faster at $u = 1$?

(b) Is it true that equidistant interpolation converges faster than Chebyshev interpolation in an environment of $u = 0$?

**4.3.16** (after Meray (1884) and Cheney [66, p. 65])

(a) Let $L_n f$ be the polynomial of degree less than $n$ which interpolates to the function $f(z) = 1/z$ at the $n$th roots of unity. Show that $(L_n f)(z) = z^{n-1}$, and that

$$\lim_{n\to\infty} \max_{|u|=1} |(L_n f - f)(u)| > 0.$$

*Hint:* Solve this directly, without the use of the previous theory.

(b) Modify the theory of Sec. 4.3.2 to the case in (a) with equidistant interpolation points on the unit circle, and make an application to $f(z) = 1/(z - a)$, $a > 0$, $a \neq 1$. Here, $\Phi_n(z) = z^n - 1$. What is $\psi(z)$, $P(z)$? The density function? Check your result by thinking like Faraday. Find out for which values of $a$, $u$, ($|u| \neq 1$, $|u| \neq a$), $(L_n f - f)(u) \to 0$, and estimate the speed of convergence (divergence).

*Hint:* The integral for $\psi(z)$ is a little tricky, but you may find it in a table. There are, however, simpler alternatives to the integral; see the end of Sec. 4.3.2.

(c) What can be said about the cases excluded above, i.e., $|u| = 1$, $|u| = a$? Also look at the case when $|a| = 1$, ($a \neq 1$).

(d) Is the equidistant interpolation on the unit circle identical to the Cauchy–FFT method (with $a = 0$, $R = 1$) for the approximate computation of the coefficients in a power series?

## 4.4 Piecewise Polynomial Interpolation

Interpolating a given function by a single polynomial over its entire range can be an ill-conditioned problem, as illustrated by Runge's phenomenon. On the other hand, polynomials of low degree can give good approximations *locally* in a small interval. In this section we will consider approximation schemes for **piecewise polynomial approximation** with different degrees of global continuity.

With the use of piecewise polynomials, there is no reason to fear equidistant data, as opposed to the situation with higher-degree polynomials. Moreover, if the function to be approximated is badly behaved in a subregion the effect of this can be confined locally, allowing good approximation elsewhere.

In computer graphics and computer aided design (CAD) curves and surfaces have to be represented mathematically, so that they can be manipulated and visualized easily. In 1962 Bézier and de Casteljau, working for the French car companies Renault and Citroën, independently developed Bézier curves for fitting curves and surfaces. Similar work, using bicubic splines, was done in USA at General Motors by Garret Birkhoff and Henry Garabedian [28]. Subsequently, W. J. Gordon of General Motors Research developed the technique of spline blending for fitting smooth surfaces to a rectangular smooth mesh of curves.

Today Bézier curves and spline functions are used extensively in all aircraft and automotive industries. Spline functions can also be used in the numerical treatment of boundary value problems for differential equations. Bézier curves have found use in computer graphics and typography. For example, scalable fonts like PostScript® are stored as piecewise Bézier curves.

### 4.4.1 Bernštein Polynomials and Bézier Curves

In the following we restrict ourselves to considering polynomial curves, i.e., one-dimensional geometric objects. Parametric curves are often used to find the functional form of a curve given geometrically by a set of points $p_i \in \mathbf{R}^d$, $i = 0 : n$.

Let $c(t) \in \mathbf{R}^d$, $t \in [0, 1]$, be a **parametric curve**. In the simplest case, $n = 1$, we take $c(t)$ to be linear,

$$c(t) = (1 - t)p_0 + tp_1,$$

and connecting the two points $p_0$ and $p_1$ so that $p_0 = c(0)$ and $p_1 = c(1)$. For $n > 1$ this will not give a smooth curve and is therefore of limited interest.

We now generalize this approach and take $c(t)$ to be a polynomial of degree $n$,

$$c(t) = \sum_{i=0}^{n} p_i B_i(t), \quad t \in [0, 1],$$

where $B_i(t)$, $i = 0 : n$, are the **Bernštein polynomials**[136] defined by

$$B_i^n(t) = \binom{n}{i} t^i (1 - t)^{n-i}, \quad i = 0 : n. \tag{4.4.1}$$

Using the binomial theorem we have

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

Thus the Bernštein polynomials of degree $n$ are nonnegative on $[0, 1]$ and *give a partition of unity*.

For $n = 3$ the four cubic Bernštein polynomials are

$$B_0^3 = (1 - t)^3, \quad B_1^3 = 3t(1 - t)^2, \quad B_2^3 = 3t^2(1 - t), \quad B_3^3 = t^3. \tag{4.4.2}$$

They are plotted in Figure 4.4.1.

[136] Bernštein introduced the polynomials named after him in 1911.

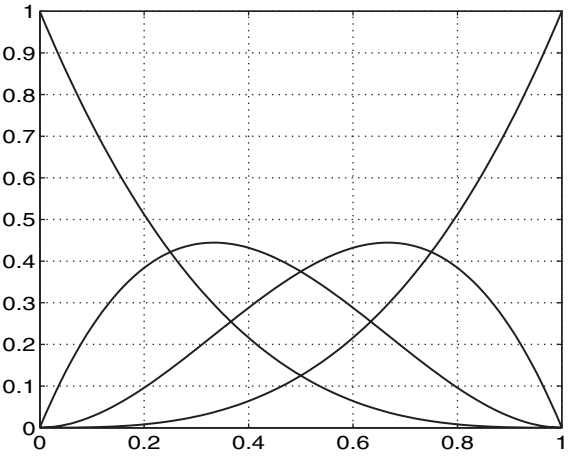



**Figure 4.4.1.** *The four cubic Bernštein polynomials.*

Some important properties of the Bernštein polynomials are given in the following theorem.

**Theorem 4.4.1.**

*The Bernštein polynomials $B_i^n(t)$ have the following properties:*

1. $B_i^n(t) > 0$, $t \in (0, 1)$ *(nonnegativity);*
2. $B_i^n(t) = B_{n-i}^n(1-t)$ *(symmetry);*
3. $B_i^n(t) = 0$ *has a root* $t = 0$ *of multiplicity* $i$ *and a root* $t = 1$ *of multiplicity* $n - i$*;*
4. *The Bernštein polynomials $B_i^n(t)$ have a unique maximum value at $t = i/n$ on* $[0, 1]$*;*
5. *The Bernštein polynomials satisfy the following recursion formula:*

$$B_i^n(t) = (1-t)B_i^{n-1}(t) + tB_{i-1}^{n-1}(t), \quad i = 0 : n; \tag{4.4.3}$$

6. *The Bernštein polynomials of degree n form a basis for the space of polynomials of degree* $\le n$.

***Proof.*** The first four properties follow directly from the definition (4.4.1). The recursion formula is a consequence of the relation

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

between the binomial coefficients.

To show the linear independence we observe that if

$$\sum_{i=0}^{n} a_i B_i^n(t) = 0,$$

then according to property 3,

$$\sum_{i=0}^{n} a_i B_i^n(1) = a_n B_i^n(1) = a_n = 0.$$

By repeatedly dividing by $(1-t)$ and using the same argument we find that $a_0 = \cdots = a_{n-1} = a_n = 0$. □

The unique parametric **Bézier curve** corresponding to a given set of $n+1$ **control points** $p_i$, $i = 0:n$, equals

$$c(t) = \sum_{i=0}^{n} p_i B_i^n(t), \quad t \in [0,1], \tag{4.4.4}$$

where $B_i^n(t)$ are Bernštein polynomials of degree $n$. By property 3 in Theorem 4.4.1 the Bézier curve *interpolates the first and last control points* $p_0$ and $p_n$. Often a curve is constructed by smoothly patching together several Bézier curves of low order.

Starting with $B_0^0(t) = 1$ and setting $B_{-1}^n(t) = B_{n+1}^n(t) = 0$, the recursion in Theorem 4.4.1 can be used to evaluate the Bernštein polynomials at a given point $t$.

It follows directly from the form of (4.4.4) that applying an affine transformation to $c(t)$ can be performed simply by applying the same transformation to the control points. Hence the Bézier curve has the desirable property that it is invariant under translations and rotations.

**Example 4.4.1.**

A quadratic Bézier curve is given by

$$c(t) = (1-t)^2 p_0 + 2t(1-t)p_1 + t^2 p_2, \quad t \in [0,1].$$

Clearly $c(0) = p_0$ and $c(1) = p_2$. For $t = 1/2$ we get

$$c\left(\frac{1}{2}\right) = \frac{1}{2}\left(\frac{p_0 + p_2}{2} + p_1\right).$$

Hence we can construct the point $c(1/2)$ geometrically as the intersection between the midpoint of the line between $p_0$ and $p_2$ and the point $p_1$; see Figure 4.4.2.

The **Bézier polygon** is the closed piecewise linear curve connecting the control points $p_i$ and $p_{i+1}$, $i = 0:n-1$, and finally $p_n$ and back to $p_0$. In Figures 4.4.2 and 4.4.3 this is the polygon formed by the dashed lines. This polygon provides a rough idea of the shape of the Bézier curve.

From the definition (4.4.4) of the Bézier curve it follows that for all $t \in [0,1]$, the curve $c(t)$ is a convex combination of the control points. Therefore, $c(t)$ lies within the convex hull (see Definition 4.3.2) of the control points.

The variation of a function in an interval $[a,b]$ is the least upper bound on the sum of the oscillations in the closed subintervals $[a,x_1]$, $[x_1,x_2]$, $\ldots$, $[x_n,b]$ for all possible such

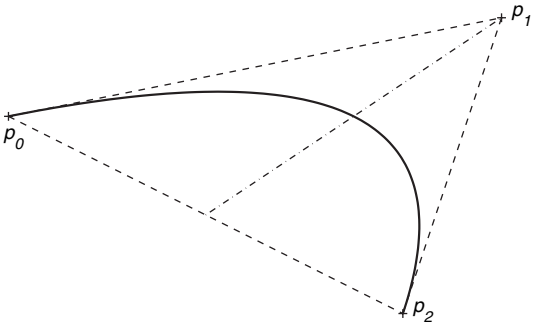



**Figure 4.4.2.** *Quadratic Bézier curve with control points.*

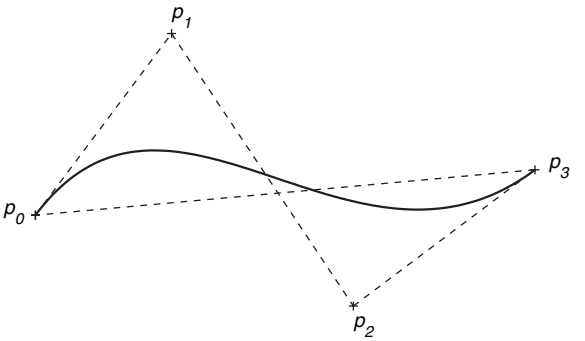



**Figure 4.4.3.** *Cubic Bézier curve with control points* $p_0, \ldots, p_3$.

subdivisions. The Bézier curve is variation diminishing. In particular, if the control points $p_i$ are monotonic, so is $c(t)$.

Usually, not all control points are known in advance. The shape of the curve is controlled by moving the control points until the curve has the desired shape. For example, in the quadratic case moving $p_1$ has a direct and intuitive effect on the curve $c(t)$. An advantage of the Bernštein basis for representing polynomials is that the coefficients (control points) are closely related to the shape of the curve. This is not the case when using a monomial or Chebyshev basis.

**Theorem 4.4.2.**

*The Bézier curve $c(t)$ is tangent to $p_1 - p_0$ and $p_n - p_{n-1}$ for $t = 0$ and $t = 1$, respectively.*

***Proof.*** To show this we compute the derivative of the Bernštein polynomial (4.4.1):

$$\frac{d}{dt}B_i^n(t) = \begin{cases} -nB_0^{n-1}(t) & \text{if } i = 0, \\ n\left(B_{i-1}^{n-1}(t) - B_i^{n-1}(t)\right) & \text{if } 0 < i < n, \\ nB_{n-1}^{n-1}(t) & \text{if } i = n. \end{cases}$$

This follows from

$$\frac{d}{dt}B_i^n(t) = \binom{n}{i}\left(it^{i-1}(1-t)^{n-i} - (n-i)t^i(1-t)^{n-i-1}\right),$$

and using the definition of the Bernštein polynomials. Setting $t = 0$ we find that $\frac{d}{dt}B_i^n(0) = 0$, $i > 1$, and therefore from (4.4.4)

$$\frac{d}{dt}c(t) = n(p_1 - p_0).$$

The result for $t = 1$ follows from symmetry. □

More generally, at a boundary point the $k$th derivative of the Bézier curve depends only on the $k$ closest control points. This fact is useful for smoothly joining together several pieces of Bézier curves.

### De Casteljau's Algorithm

To evaluate the Bézier curve at $t \in [0, 1]$ we use the recursion formula (4.4.3) to obtain

$$\begin{aligned} c(t) &= \sum_{i=0}^{n} p_i B_i^n(t) = (1-t)\sum_{i=0}^{n-1} p_i B_i^{n-1}(t) + t\sum_{i=1}^{n} p_i B_{i-1}^{n-1}(t) \\ &= \sum_{i=0}^{n-1}\big((1-t)p_i + tp_{i+1}\big)B_i^{n-1}(t) = \sum_{i=0}^{n-1} p_i^{(1)}(t)B_i^{n-1}(t). \end{aligned}$$

Here we have introduced the new auxiliary control points

$$p_i^{(1)}(t) = (1-t)p_i + tp_{i+1}, \quad i = 0 : n-1,$$

as convex combinations (depending on $t$) of the original control points. Using this result we can successively lower the grade of the Bernštein polynomial until we arrive at $B_0^0 = 1$. This gives **de Casteljau's algorithm**, a recursion scheme for the auxiliary control points

$$\begin{aligned} p_i^{(0)}(t) &= p_i, \quad i = 0 : n, \\ p_i^{(r)}(t) &= (1-t)p_i^{(r-1)}(t) + tp_{i+1}^{(r-1)}(t), \quad i = 0 : n-r. \end{aligned} \tag{4.4.5}$$

It follows that

$$c(t) = \sum_{i=0}^{n-r} p_i^{(r)}(t)B_i^{n-r}(t), \quad r = 0 : n, \tag{4.4.6}$$

and in particular $c(t) = p_0^{(n)}$.

De Casteljau's algorithm works by building convex combinations (4.4.5) and is therefore numerically very stable. It can conveniently be arranged in a triangular array.

$$\begin{array}{lllll} p_0 = p_0^{(0)} & & & & \\ & p_0^{(1)} & & & \\ p_1 = p_1^{(0)} & & p_0^{(2)} & & \\ & p_1^{(1)} & & & \\ p_2 = p_2^{(0)} & \vdots & p_1^{(2)} & \ddots & \\ \vdots & \vdots & \vdots & \vdots & p_0^{(n)} \\ \vdots & p_{n-2}^{(1)} & \vdots & & \\ p_{n-1} = p_{n-1}^{(0)} & & p_{n-2}^{(2)} & & \\ & p_{n-1}^{(1)} & & & \\ p_n = p_n^{(0)} & & & & \end{array} \tag{4.4.7}$$

The algorithm uses about $n^2$ operations and so is less efficient than Horner's algorithm for evaluating a polynomial in the monomial basis.

The $k$th derivative of $c(t)$ is also available from the de Casteljau scheme. It holds that

$$c'(t) = n\left(p_1^{(n-1)} - p_0^{(n-1)}\right),$$
$$c''(t) = n(n-1)\left(p_2^{(n-2)} - 2p_1^{(n-2)} + p_0^{(n-2)}\right), \ldots,$$

and in general

$$c^{(k)}(t) = \frac{n!}{(n-k)!}\Delta^k p_0^{(n-k)}, \quad 0 \le k \le n, \tag{4.4.8}$$

where the difference operates on the lower index $i$.

De Casteljau's algorithm is illustrated for the quadratic case in Figure 4.4.4, where the following geometric interpretation can be observed. In the interval $[0, t]$ the Bézier curve is represented by a quadratic spline with control points $p_0$, $p_0^{(1)}$, $p_0^{(2)}$. In the remaining interval $[t, 1]$ it is represented by a quadratic spline with control points $p_0^{(2)}$, $p_1^{(1)}$, $p_2$. Note that these

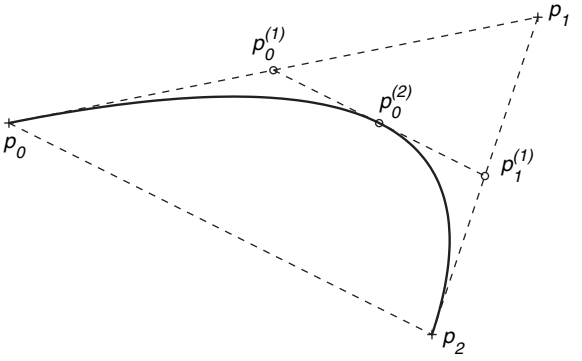



**Figure 4.4.4.** *De Casteljau's algorithm for* $n = 2$, $t = \frac{1}{2}$.

two sets of control points lie closer to the curve $c(t)$. After a few more subdivisions it will be hard to distinguish the polygon joining the control points from the curve.

De Casteljau's algorithm can also be used to subdivide a Bézier curve into two segments. By repeating this partitioning the Bézier polygons converge fast to the curve. This construction is very well suited to control, for example, a milling machine which can only remove materiel.

### 4.4.2 Spline Functions

The name **spline** comes from a very old technique in drawing smooth curves in which a thin strip of wood, called a drafting spline, is bent so that it passes through a given set of points; see Figure 4.4.5. The points of interpolation are called **knots** and the spline is secured at the knots by means of lead weights called **ducks**. Before the computer age splines were used in all kinds of geometric design to produce sufficiently smooth profiles. This process was slow and expensive.

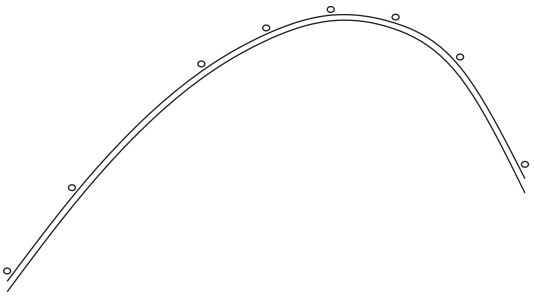


**Figure 4.4.5.** *A drafting spline.*

The mathematical model of a spline is a special case of the elastic line treated in the theory of elasticity. By Hamilton's principle the spline assumes a shape $y(x)$ which minimizes the strain energy. This is proportional to the line integral of the square of the curvature

$$\kappa(x) = \frac{y''(x)}{(1 + (y'(x))^2)^{3/2}}. \tag{4.4.9}$$

Since $ds = \sqrt{1 + (y'(x))^2}dx$, the energy becomes

$$E(y) = \int_a^b \frac{y''(x)^2}{(1 + (y'(x))^2)^{5/2}}\, dx. \tag{4.4.10}$$

A mathematical model of the **nonlinear spline** was given by Daniel Bernoulli (1742) and Euler (1744).[137] A modern description of the development of Bernoulli and Euler is given by Malcolm in [256], where algorithms for computing discrete approximations of nonlinear splines are also discussed and compared. In general these are very costly.

[137] Euler derived the differential equation satisfied by an elastic line using techniques now known as calculus of variation and Lagrange multipliers. When Euler did this work Lagrange was still a small child!

For slowly varying deflections, i.e., when $(y'(x))^2$ is approximately constant, we have the approximation

$$E(y) \approx \text{const} \cdot \int_a^b y''(x)^2\, dx.$$

Under this assumption, $y(x)$ is built up of *piecewise cubic polynomials* in such a way that $y(x)$ and its two first derivatives are everywhere continuous. Let $x_i$, $i = 0 : m$, be the points the spline is forced to interpolate. Then the third derivative can have discontinuities at the points $x_i$.

The mathematical concept of spline functions was introduced in 1946 by Schoenberg in the seminal paper [314]. The importance of the B-spline basis for spline approximation (see Sec. 4.4.3) was also first appreciated by Schoenberg. These were not used in practical calculations for general knot sequences until the early seventies, when a stable recurrence relation was established independently by de Boor [36] and Cox [84].

In the following we restrict ourself to consider curves in the plane. Today B-splines enable the mathematical representation of surfaces far beyond hand techniques. In aircraft design computations they may involve more than 50,000 data points.

### Linear and Cubic Splines

We start by formally defining a spline function of order $k \geq 1$.

**Definition 4.4.3.**
*A spline function $s(x)$ of order $k \geq 1$ (degree $k-1 \geq 0$), on a grid*

$$\Delta = \{a = x_0 < x_1 < \cdots < x_m = b\}$$

*of distinct knots is a real function $s$ with the following properties:*

(a) *For $x \in [x_i, x_{i+1}]$, $i = 0 : m-1$, $s(x)$ is a polynomial of degree $< k$.*

(b) *For $k = 1$, $s(x)$ is a piecewise constant function. For $k \geq 2$, $s(x)$ and its first $k-2$ derivatives are continuous on $[a, b]$, i.e., $s(x) \in C^{k-2}[a, b]$.*

The space of all spline functions of order $k$ on $\Delta$ is denoted by $S_{\Delta,k}$. From the definition it follows that if $s_1(x)$ and $s_2(x)$ are spline functions of the same degree, so is $c_1 s_1(x) + c_2 s_2(x)$. Clearly $S_{\Delta,k}$ is a *linear vector space*. The space $\mathcal{P}_k$ of polynomials of degree less than $k$ is a linear subspace of $S_{\Delta,k}$. The **truncated power** functions

$$(x - x_j)_+^{k-1} = \max\{(x - x_j)^{k-1}, 0\}, \quad j = 1 : m-1,$$

introduced in Sec. 3.3.3 in connection with the Peano kernel are also elements of $S_{\Delta,k}$.

**Theorem 4.4.4.**
*The monomials and truncated power functions*

$$\{1, x, \ldots, x^{k-1}, (x - x_1)_+^{k-1}, \ldots, (x - x_{m-1})_+^{k-1}\} \tag{4.4.11}$$

*form a basis for the spline space $S_{\Delta,k}$. In particular, the dimension of this space is $k+m-1$.*

***Proof.*** All we need for the first subinterval is a basis of $\mathcal{P}_k$, for example, the power basis $\{1, x, \ldots, x^{k-1}\}$. For each additional subinterval $[x_j, x_{j+1})$, $j = 1 : m-1$, we need only add the new basis function $(x - x_j)_+^{k-1}$. It is easy to show that these $k + m - 1$ functions are linearly independent. □

The truncated power basis (4.4.35) has several disadvantages and is not well suited for numerical computations. The basis functions are not local; i.e., they are nonzero on the whole interval $[a, b]$. Further, they (4.4.35) are almost linearly dependent when the knots are close. Therefore, this basis yields dense ill-conditioned linear systems for various tasks. A more suitable basis will be introduced in Sec. 4.4.3.

The simplest case $k = 1$ is the linear spline interpolating given values $y_i = f(x_i)$, $x_i \in [a, b]$, $i = 0 : m$. This is the broken line

$$s(x) = q_i(x) = y_{i-1} + d_i(x - x_{i-1}), \quad x \in [x_{i-1}, x_i), \quad i = 1 : m. \tag{4.4.12}$$

Here

$$d_i = [x_{i-1}, x_i]f(x) = (y_i - y_{i-1})/h_i, \quad h_i = x_i - x_{i-1}, \tag{4.4.13}$$

are the divided differences of $f$ at $[x_{i-1}, x_i]$. By (4.2.16) the error satisfies

$$|f(x) - s(x)| \le \frac{1}{8} \max_i \left( h_i^2 \max_{x \in [x_{i-1}, x_i]} |f''(x)| \right), \tag{4.4.14}$$

if $f \in C^2[a, b]$. Hence, we can make the error arbitrarily small by decreasing $\max_i h_i$. An important property of interpolation with a linear spline function is that it preserves monotonicity and convexity of the interpolated function.

The broken line interpolating function has a discontinuous first derivative at the knots which makes it unsuitable for many applications. To get better smoothness piecewise polynomials of higher degree need to be used. Cubic spline functions, which *interpolate* a given function $f(x)$ on the grid $\Delta$ and have continuous first and second derivatives, are by far the most important; see Figure 4.4.6.

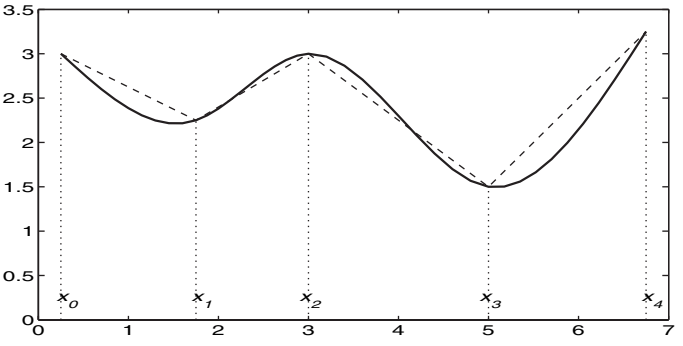



**Figure 4.4.6.** *Broken line and cubic spline interpolation.*

With piecewise Lagrange–Hermite interpolation we can interpolate given function values and first derivatives of a function on the grid $\Delta$. First recall that in cubic Lagrange–Hermite interpolation (see Theorem 4.3.1 and Problem 4.3.1) a cubic polynomial $q_i(x)$ is fitted to values of a function and its first derivative at the endpoints of the interval $[x_{i-1}, x_i)$. Let

$$\theta = \frac{x - x_{i-1}}{h_i} \in [0, 1), \quad x \in [x_{i-1}, x_i), \tag{4.4.15}$$

be a local variable. Then, by (4.3.11) translated to the notation in (4.4.13), the cubic $q_i(x)$ can be written in the form

$$q_i(x) = \theta y_i + (1-\theta)y_{i-1} + h_i\theta(1-\theta)\left[(k_{i-1} - d_i)(1-\theta) - (k_i - d_i)\theta\right], \tag{4.4.16}$$

where $h_i$, $d_i$, $i = 1 : m$, are as defined in (4.4.13), and

$$k_i = q_i'(x_i) \tag{4.4.17}$$

is the derivative at $x_i$.

The second derivative (and the curvature) of this piecewise polynomial will in general be discontinuous at the grid points. We now show how to choose the parameters $k_i$, $i = 0 : m$, to also get a continuous *second* derivative. This will yield an interpolating cubic spline. In contrast to piecewise Lagrange–Hermite interpolation, *each piece of the cubic spline will depend on all data points*.

**Theorem 4.4.5.**

*Every cubic spline function, with knots $a = x_0 < x_1 < \cdots < x_m = b$, which interpolates the function $y = f(x)$,*

$$s(x_i) = f(x_i) = y_i, \quad i = 0 : m,$$

*equals for $x \in [x_{i-1}, x_i)$, $i = 1 : m$, a third degree polynomial of the form* (4.4.16). *The $m + 1$ parameters $k_i$, $i = 0 : m$, satisfy $m - 1$ linear equations*

$$h_{i+1}k_{i-1} + 2(h_i + h_{i+1})k_i + h_i k_{i+1} = 3(h_i d_{i+1} + h_{i+1}d_i), \quad i = 1 : m-1, \tag{4.4.18}$$

*where $h_i = x_i - x_{i-1}$, $d_i = (y_i - y_{i-1})/h_i$.*

***Proof.*** We require the second derivative of the spline $s(x)$ to be continuous at $x_i$, $i = 1 : m - 1$. We have

$$s(x) = \begin{cases} q_i(x), & x \in [x_{i-1}, x_i), \\ q_{i+1}(x), & x \in [x_i, x_{i+1}), \end{cases}$$

where $q_i(x)$ is given by (4.4.22)–(4.4.23). Differentiating $q_i(x)$ twice we get $\frac{1}{2}q_i''(x) = a_{2,i} + 3a_{3,i}(x - x_{i-1})$, and putting $x = x_i$ we obtain

$$\frac{1}{2}q_i''(x_i) = a_{2,i} + 3a_{3,i}h_i = \frac{(k_{i-1} + 2k_i - 3d_i)}{h_i}. \tag{4.4.19}$$

Replacing $i$ by $i + 1$ we get $\frac{1}{2}q_{i+1}''(x) = a_{2,i+1} + 3a_{3,i+1}(x - x_i)$, and hence

$$\frac{1}{2}q_{i+1}''(x_i) = a_{2,i+1} = \frac{(3d_{i+1} - 2k_i - k_{i+1})}{h_{i+1}}. \tag{4.4.20}$$

These last two expressions must be equal, which gives the conditions

$$\frac{1}{h_i}(k_{i-1} + 2k_i - 3d_i) = \frac{1}{h_{i+1}}(3d_{i+1} - 2k_i - k_{i+1}), \quad i = 1 : m-1. \tag{4.4.21}$$

Multiplying both sides by $h_i h_{i+1}$ we get (4.4.18). □

If the cubic spline $s(x)$ is to be evaluated at many points, then it is more efficient to first convert it from the form (4.4.16) to **piecewise polynomial form** (pp-form):

$$q_i(x) = y_{i-1} + a_{1i}(x - x_{i-1}) + a_{2i}(x - x_{i-1})^2 + a_{3i}(x - x_{i-1})^3, \quad i = 1 : m. \tag{4.4.22}$$

From (4.4.16) we obtain after some calculation

$$\begin{aligned} a_{1i} &= q_i'(x_{i-1}) = k_{i-1}, \\ a_{2i} &= \frac{1}{2}q_i''(x_{i-1}) = \frac{(3d_i - 2k_{i-1} - k_i)}{h_i}, \\ a_{3i} &= \frac{1}{6}q_i'''(x_{i-1}) = \frac{(k_{i-1} + k_i - 2d_i)}{h_i^2}. \end{aligned} \tag{4.4.23}$$

Using Horner's scheme $q_i(x)$ can be evaluated from (4.4.22) using only four multiplications. Algorithms for performing conversion to pp-form are given in [37, Chapter X].

The conditions (4.4.18) are $(m-1)$ linearly independent equations for the $(m+1)$ unknowns $k_i$, $i = 0 : m$. Two additional conditions are therefore needed to uniquely determine the interpolating spline. The most important choices are discussed below.

- If the derivatives at the endpoints are known we can take

$$k_0 = f'(a), \qquad k_m = f'(b). \tag{4.4.24}$$

  The corresponding spline function $s(x)$ is called the **complete cubic spline interpolant**.

- If $k_0$ and $k_m$ are determined by numerical differentiation with a truncation error $O(h^4)$, we call the spline interpolant **almost complete**. For example, $k_0$ and $k_m$ may be the sum of (at least) four terms of the expansions

$$Df(x_0) = \frac{1}{h}\ln(1+\Delta)y_0, \qquad Df(x_m) = -\frac{1}{h}\ln(1-\nabla)y_m,$$

  into powers of the operators $\Delta$ and $\nabla$, respectively.[138]

- A physical spline is straight outside the interval $[a, b]$. The corresponding boundary conditions are $q_1''(x_0) = q_m''(x_m) = 0$. From (4.4.20) and (4.4.19) it follows that

$$\frac{1}{2}q_i''(x_{i-1}) = \frac{(3d_i - 2k_{i-1} - k_i)}{h_i}, \qquad \frac{1}{2}q_i''(x_i) = \frac{-(3d_i - k_{i-1} - 2k_i)}{h_i}.$$

[138] Two terms of the central difference expansion in (3.3.46).

Setting $i = 1$ in the first equation and $i = m$ in the second gives the two conditions

$$\begin{aligned} 2k_0 + k_1 &= 3d_1, \\ k_{m-1} + 2k_m &= 3d_m. \end{aligned} \tag{4.4.25}$$

The spline function corresponding to these boundary conditions is called the **natural spline interpolant**. It should be stressed that when a cubic spline is used for the approximation of a smooth function, these boundary conditions are *not natural*! Although the natural spline interpolant in general converges only with order $h^2$, it has been shown that on any compact subinterval of the open interval $(a, b)$ the order of convergence is $O(h^4)$.

- If the endpoint derivatives are not known, a convenient boundary condition is to require that $s'''(x)$ be continuous across the first and last interior knots $x_1$ and $x_{m-1}$. Hence

$$q_1'''(x) = q_2'''(x), \qquad q_{m-1}'''(x) = q_m'''(x).$$

Then $x_1$ and $x_{m-1}$ are no longer knots, and the corresponding spline interpolant is called the **not-a-knot** spline interpolant.

From (4.4.23) we obtain,

$$\frac{1}{6}q_i'''(x) = a_{3i} = \frac{(k_{i-1} + k_i - 2d_i)}{h_i^2}, \quad x \in [x_{i-1}, x_i), \quad i = 1 : m.$$

Hence the condition $q_1''' = q_2'''$ gives $(k_0 + k_1 - 2d_1)/h_1^2 = (k_1 + k_2 - 2d_2)/h_2^2$, or

$$h_2^2 k_0 + (h_2^2 - h_1^2)k_1 - h_1^2 k_2 = 2(h_2^2 d_1 - h_1^2 d_2).$$

Since this equation would destroy the tridiagonal form of the system, we use (4.4.18), with $i = 1$ to eliminate $k_2$. This gives the equation

$$h_2 k_0 + (h_2 + h_1)k_1 = 2h_2 d_1 + \frac{h_1(h_2 d_1 + h_1 d_2)}{h_2 + h_1}. \tag{4.4.26}$$

If the right boundary condition is treated similarly we get

$$(h_{m-1} + h_m)k_{m-1} + h_{m-1}k_m = 2h_{m-1}d_m + \frac{h_m(h_{m-1}d_m + h_m d_{m-1})}{h_{m-1} + h_m}. \tag{4.4.27}$$

The error of the not-a-knot spline interpolant is of the same order as that of the complete spline. For functions $f \in C^4[a, b]$ one has an error estimate of the same form as () for $r = 0 : 2$; see Beatson [22]. Indeed, the error analysis below shows that the Lagrange–Hermite interpolation error is, in the whole interval $[a, b]$, asymptotically, the dominant source of error for both complete and not-a-knot spline approximation.

- If the spline is used to represent *a periodic function*, then $y_0 = y_m$ and the boundary conditions

$$s'(a) = s'(b), \qquad s''(a) = s''(b) \tag{4.4.28}$$

suffice to determine the spline uniquely. From the first condition it follows that $k_0 = k_m$, which can be used to eliminate $k_0$ in (4.4.18) for $k = 1$. Using (4.4.21) the

second condition in (4.4.28) gives

$$(k_0 + 2k_1 - 3d_1)/h_1 = -(2k_{m-1} + k_m - 3d_m)/h_m$$

or, after eliminating $k_0$,

$$2h_m k_1 + 2h_1 k_{m-1} + (h_1 + h_m)k_m = 3(h_m d_1 + h_1 d_m).$$

- The natural spline interpolant has the following best approximation property.

**Theorem 4.4.6.**

*Among all functions $g$ that are twice continuously differentiable on $[a, b]$ and which interpolate $f$ at the points $a = x_0 < x_1 < \cdots < x_m = b$, the natural spline function minimizes*

$$\int_a^b \big(s''(t)\big)^2 dt.$$

*The same minimum property holds for the complete spline interpolant if the functions $g$ satisfy $g'(a) = f'(a)$ and $g'(b) = f'(b)$.*

***Proof.*** See de Boor [37, Chapter 5]. □

Due to this property spline functions yield smooth interpolation curves, except for rather thin oscillatory layers near the boundaries if the "natural" boundary conditions $s''(a) = s''(b) = 0$ are far from being satisfied. For the complete or almost complete cubic spline and for cubic splines determined by the not-a-knot conditions, these oscillations are much smaller as we shall see below. Hence, when a spline is to be used for the approximate representation of a smooth function, the natural spline is *not* a natural choice!

Equations (4.4.18) together with any of these boundary conditions gives a linear system for determining the derivatives $k_i$. It can be shown that this system is well-conditioned using the following result, which will be proved in Volume II.

**Lemma 4.4.7.**

*Assume that the matrix $A \in \mathbf{R}^{n\times n}$ is strictly row* **diagonally dominant**, *i.e.,*

$$\alpha_i := |a_{ii}| - \sum_{j\neq i} |a_{ij}| > 0, \quad i = 1 : n. \tag{4.4.29}$$

*Then $A$ is nonsingular, and for row diagonally dominant linear systems Gaussian elimination* without pivoting *is stable.*

For the first three boundary conditions the system is **tridiagonal** and, which is easily verified, also strictly row diagonally dominant. In Example 1.3.2, and algorithm was given for solving a tridiagonal linear systems of order $m$ by Gaussian elimination in $\mathcal{O}(m)$ flops.

The linear system resulting from the not-a-knot boundary condition is not diagonally dominant in the first and last row. However, it can be shown that also in this case the system is well-conditioned and can be solved stably by Gaussian elimination without pivoting.

**Example 4.4.2.**

In the case of spline interpolation with constant step size $h_i = h$, (4.4.18) becomes

$$k_{i-1} + 4k_i + k_{i+1} = 3(d_i + d_{i+1}), \quad i = 1 : m-1. \tag{4.4.30}$$

The not-a-knot boundary conditions (4.4.26)–(4.4.27) become

$$k_0 + 2k_1 = \frac{1}{2}(5d_1 + d_2), \qquad 2k_{m-1} + k_m = \frac{1}{2}(d_{m-1} + 5d_m). \tag{4.4.31}$$

We obtain for $(k_0, k_1, \ldots, k_m)$ a tridiagonal system $Tk = g$, where

$$T = \begin{pmatrix} 1 & 2 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 2 & 1 \end{pmatrix}, \qquad g = 3 \begin{pmatrix} (5d_1 + d_2)/6 \\ d_1 + d_2 \\ \vdots \\ d_{m-1} + d_m \\ (d_{m-1} + 5d_m)/6 \end{pmatrix}.$$

Except for the first and last row, the elements of $T$ are constant along the diagonals. The condition number of $T$ increases very slowly with $m$; for example, $\kappa(T) < 16$ for $m = 100$.

Consider now the periodic boundary conditions in (4.4.28). Setting $k_m = k_0$ in the last equation we obtain a linear system of equations $Tk = g$ for $k_1, \ldots, k_{m-1}$ where

$$T = \left(\begin{array}{ccccc|c} b_1 & c_1 & & & & a_m \\ a_1 & b_2 & c_2 & & & 0 \\ & \ddots & \ddots & \ddots & & \vdots \\ & & a_{m-3} & b_{m-2} & c_{m-2} & 0 \\ & & & a_{m-2} & b_{m-1} & c_{m-1} \\ \hline c_m & 0 & \cdots & 0 & a_{m-1} & b_m \end{array}\right). \tag{4.4.32}$$

Here $T$ is tridiagonal except for its last row and last column, where an extra nonzero element occurs. Such systems, called **arrowhead systems**, can be solved with about twice the work of a tridiagonal system; see [30].

In some applications one wants to smoothly interpolate given points $(x_j, y_j)$, $j = 0 : m$, where a representation of the form $y = f(x)$ is not possible. Then we can use a **parametric spline** representation $x = x(t)$, $y = y(t)$, where the parameter values $0 = t_0 \le t_1 \le \cdots \le t_m$ correspond to the given points. Using the approach described previously, two spline functions $s_x(t)$ and $s_y(t)$ can then be determined that interpolate the points $(t_i, x_i)$ and $(t_i, y_i)$, $i = 0 : m$, respectively. The parametrization is usually chosen as

$$t_i = d_i/d, \quad i = 1 : m,$$

where $d_0 = 0$,

$$d_i = d_{i-1} + \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}, \quad i = 1 : m,$$

are the cumulative distance, and $d = d_m$.

For boundary conditions we have the same choices as mentioned previously. In particular, using periodic boundary conditions for $s_x(t)$ and $s_y(t)$ allows the representation of *closed curves* (see Problem 4.4.8 (b)). Parametric splines can also be used to approximate curves in higher dimensions.

### Error in Cubic Spline Interpolation

The following standard error estimate for the complete cubic spline interpolant is due to Beatson [22].

**Theorem 4.4.8.**

*Let the function $f$ be four times continuously differentiable in $[a, b]$ and let $s$ be the complete cubic spline interpolant on the grid $a = x_0 < x_1 < \cdots < x_m = b$. Then*

$$\max_{x\in[a,b]} |f^{(r)}(x) - s^{(r)}(x)| \le c_r(\beta) h^{4-r} \max_{x\in[a,b]} |f^{(iv)}(x)|, \quad r = 0:3, \tag{4.4.33}$$

*where $h = \max_i h_i$, $\beta = h/\min_i h_i$, and*

$$c_0 = \frac{5}{384}, \quad c_1 = \frac{1}{216}(9+\sqrt{3}), \quad c_2(\beta) = \frac{1}{12}(1+3\beta), \quad c_3(\beta) = \frac{1}{2}(1+\beta^2).$$

We comment below on the above result for $r = 0$ and how it is influenced by the choice of other boundary conditions. Let $x \in I_i = [x_{i-1}, x_i]$, $i = 1:m$, and set

$$t = (x - x_{i-1})/h_i, \qquad y_i = f(x_i), \qquad y_i' = f'(x_i).$$

The error in cubic spline interpolation can be expressed as the sum of two components:

i. The error $E_H(x)$ due to Lagrange–Hermite interpolation with correct values of $f'(x_{i-1})$, $f'(x_i)$. From (4.3.12) we obtain

$$\max_{x\in I_i} |E_H(x)| \le \frac{1}{384} \max_{x\in I_i} |h_i^4 f^{(iv)}(x)|.$$

ii. The error $E_S(x)$ due to the errors of the slopes $e_i = k_i - y_i'$, $i = 0:m$. In particular, $e_0$ and $e_m$ are the errors in the boundary derivatives.

The bound in Theorem 4.4.8 for $r = 0$ is only a factor of five larger than that for piecewise Lagrange–Hermite interpolation with correct derivatives of $f$ at all points $x_i$, $i = 0:m$, not just at $x_0 = a$ and $x_m = b$. Indeed, typically the error $E_H(x)$ is the dominant part. By (4.4.16) the second part of the error is

$$E_S(x) = h_i t(1-t)\big[e_{i-1}(1-t) - e_i t\big], \quad x = x_{i-1} + t h_i, \quad t \in [0, 1).$$

Since $|1-t| + |t| = 1$, and the maxima of $t(1-t)$ on $[0, 1]$ equals $1/4$, it follows that

$$|E_S(x)| \le \frac{1}{4} \max_{1\le i\le m} |h_i e_j|, \quad j = i-1, i, \tag{4.4.34}$$

where $h_i = x_i - x_{i-1}$.

We shall consider the case of constant step size $h_i = h$. For complete splines $e_0 = e_m = 0$ and for almost complete splines $e_0 = O(h^4)$, $e_m = O(h^4)$. It can be shown that then $e_i = O(h^4)$, and its contribution to $E_S$ is $O(h^5)$. Thus if $h$ is sufficiently small, *the Lagrange–Hermite interpolation error is asymptotically the dominant source of error in the whole interval* $[a, b]$.

Finally, we discuss the *effect of the boundary slope errors* for other boundary conditions. Figure 4.4.7 shows (for $m = 20$, $e_0 = e_m = -1$) how rapidly the error component from the boundary dies out. At the midpoint $x = 0.5$ the error is $0.3816 \cdot 10^{-5}$. If $m \gg 1$, $e_0 \neq 0$, and $e_m \neq 0$, the error is negligible outside thin *oscillatory boundary layers* near $x = x_0$ and $x = x_m$. The height and thickness of the layers depend on $e_0$ and $e_m$.

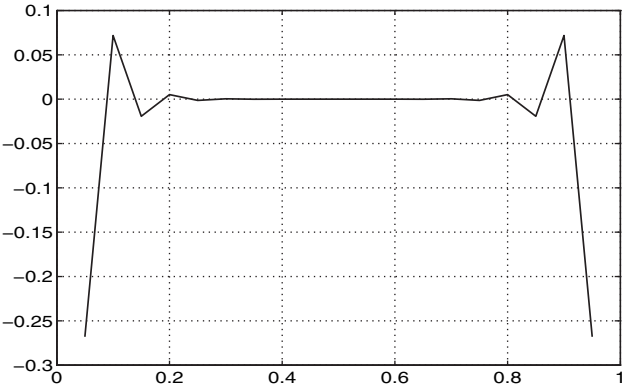



**Figure 4.4.7.** *Boundary slope errors $e_{B,i}$ for a cubic spline, $e_0 = e_m = -1$; $m = 20$.*

Consider the left boundary; the right one is analogous. For the *natural splines*, there is a peak error near $x = x_0$ of approximately $0.049h^2|y''|$, i.e., 40% of the *linear* interpolation error (instead of cubic). This is often clearly visible in a graph of $s(x)$.

For the not-a-knot splines we obtain approximately $e_0 \sim 0.180h^3 y^{(4)}$, and the peak near $x_0$ becomes $0.031h^4 y^{(4)}$, typically very much smaller than we found for natural splines. Still it is about 11.5 times as large as the Lagrange–Hermite interpolation error, but since the oscillations quickly die out, we conclude that *the Lagrange–Hermite interpolation is the dominant error source in cubic not-a-knot spline interpolation in (say) the interval* $[a + 3h, b - 3h]$ .

Similar conclusions seem to hold in the case of variable step size also, under the reasonable assumption that $h_{n+1} - h_n = O(h_n^2)$. (The use of variable step size in the context of ordinary differential equations will be treated in a later volume.)

### 4.4.3 The B-Spline Basis

In the following we will introduce a more convenient **B-spline** basis for the linear space $S_{\Delta,k}$ of spline functions of degree $k$. The term B-spline was coined by I. J. Schoenberg [314] and is short for basis spline.

It was shown in Sec. 4.4.2 that the set of spline functions of order $k$, $S_{\Delta,k}$, on the grid

$$\Delta = \{a = x_0 < x_1 < \cdots < x_m = b\},$$

is a linear space of dimension $k + m - 1$. One basis was shown to be the **truncated power basis**:

$$\left\{1, x, \ldots, x^{k-1}\right\} \cup \left\{(x - x_1)_+^{k-1}, (x - x_2)_+^{k-1}, \ldots, (x - x_{m-1})_+^{k-1}\right\}. \tag{4.4.35}$$

In particular, a basis for $S_{\Delta,2}$ consists of continuous piecewise linear functions

$$\{1, x\} \cup \{l_1(x), \ldots, l_{m-1}(x)\}, \quad l_i(x) = (x - x_i)_+.$$

Another basis for $S_{\Delta,2}$ is obtained by introducing an artificial exterior knot $x_{-1} \le x_0$. Then it is easy to see that using the functions $l_i(x)$, $i = -1 : m-1$, every linear spline on $[x_0, x_m]$ can also be written as

$$s(x) = \sum_{i=-1}^{m-1} c_i l_i(x).$$

In anticipation of the fact that it may be desirable to interpolate at other points than the knots, we consider from now on the sequence of knots

$$\Delta = \{\tau_0 \le \tau_1 \le \cdots \le \tau_m\}, \tag{4.4.36}$$

where $\tau_i < \tau_{i+k}$, $i = 0 : m - k$, i.e., at most $k$ successive knots are allowed to coincide. We start by considering the space $S_{\Delta,1}$. This consists of piecewise constant functions and a basis is

$$N_{i,1}(x) = \begin{cases} 1 & x \in [\tau_i, \tau_{i+1}), \\ 0 & \text{otherwise}, \end{cases} \qquad i = 0 : m - 1. \tag{4.4.37}$$

The basis functions are arbitrarily chosen to be continuous from the right, i.e., $N_{i,1}(\tau_i) = N_{i,1}(\tau_i + 0)$.

For $k = 2$ we define the **hat functions**[139] by

$$N_{i,2}(x) = \begin{cases} (x - \tau_i)/(\tau_{i+1} - \tau_i), & x \in [\tau_i, \tau_{i+1}], \\ (\tau_{i+2} - x)/(\tau_{i+2} - \tau_{i+1}), & x \in [\tau_{i+1}, \tau_{i+2}), \\ 0, & x \notin (\tau_i, \tau_{i+2}), \end{cases} \quad i = -1 : m - 1. \tag{4.4.38}$$

We have here introduced two **exterior knots** $\tau_{-1} \le \tau_0$ and $\tau_{m+1} \ge \tau_m$ at the boundaries. (In the following we refer to the knots $\tau_0, \ldots, \tau_m$ as **interior knots**.)

At a distinct knot $\tau_i$ just one hat function is nonzero, $N_{i+1,2}(\tau_i) = 1$. If all knots are distinct it follows that the spline function of order $k = 2$ interpolating the points $(\tau_i, y_i)$, $i = 0 : m$, can be written uniquely as

$$s(x) = \sum_{i=-1}^{m-1} c_i N_{i,2}(x), \tag{4.4.39}$$

[139] The generalization of hat functions to several dimensions plays a very important role in finite element methods; see Sec. 5.4.4.

where $c_i = y_{i+1}$. This shows that the restriction of the functions $N_{i,2}(x)$, $i = -1 : m-1$, to the interval $[\tau_0, \tau_m]$ are $(m+1)$ linearly independent functions in $S_{\Delta,2}$ and form a basis for $S_{\Delta,2}$.

If we allow two interior knots to coalesce, $\tau_i = \tau_{i+1}$, $0 < i < m-1$, then $N_{i-1,2}(x)$ and $N_{i,2}(x)$ will have a discontinuity at $\tau_i$. This generalizes the concept of a B-spline of order 2 given in Definition 4.4.3 and allows us to model functions with discontinuities at certain knots. Figure 4.4.8 illustrates the formation of a double knot for a linear spline.

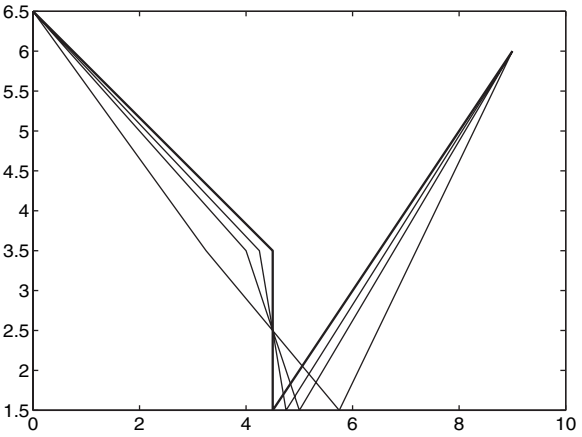



**Figure 4.4.8.** *Formation of a double knot for a linear spline.*

By definition the basis function $N_{i,2}(x)$ is nonzero only on the interval $(\tau_i, \tau_{i+2})$. It follows that for $x \in (\tau_i, \tau_{i+1})$ we have $N_{j,2}(x) = 0$, $j \neq i-1, i$. *Hence, for any given value of x at most two hat functions will be nonzero* and

$$s(x) = c_{i-1}N_{i-1,2}(x) + c_i N_{i,2}(x), \quad x \in (\tau_i, \tau_{i+1}).$$

The exterior knots are usually taken to coincide with the boundary so that $\tau_{-1} = \tau_0$ and $\tau_{m+1} = \tau_m$. In this case $N_{-1,1}$ and $N_{m-1,1}$ become "half-hats" with a discontinuity at $\tau_0$ and $\tau_m$, respectively.

It is easily verified that the functions $N_{i,2}(x)$ can be written as a linear combination of the basis function $l_i(x) = (x-\tau_i)_+$, $i = 1 : m+1$. We have

$$\begin{aligned} N_{i,2}(x) &= \big((x-\tau_{i+2})_+ - (x-\tau_{i+1})_+\big)/(\tau_{i+2}-\tau_{i+1}) \\ &\quad -\big((x-\tau_{i+1})_+ - (x-\tau_i)_+\big)/(\tau_{i+1}-\tau_i) \\ &= [\tau_{i+1}, \tau_{i+2}]_t (t-x)_+ - [\tau_i, \tau_{i+1}]_t (t-x)_+ \\ &= (\tau_{i+2}-\tau_i)[\tau_i, \tau_{i+1}, \tau_{i+2}]_t (t-x)_+, \quad i = 1 : m. \end{aligned} \tag{4.4.40}$$

Here $[\tau_i, \tau_{i+1}, \tau_{i+2}]_t$ means the second order divided-difference functional[140] operating on a function of $t$; i.e., the values $\tau_i$ are to be substituted for $t$, not for $x$. Recall that divided differences are defined also for *coincident values* of the argument; see Sec. 4.3.1.

The Peano kernel and its basic properties were given in Sec. 3.3.3. The last expression in (4.4.40) tells us that $N_{i,2}$ is the Peano kernel of a second order divided-difference functional

[140]The notation is defined in Sec. 4.2.1.

multiplied by the constant $\tau_{i+2} - \tau_i$. This observation suggests a definition of B-splines of arbitrary order $k$ and a B-spline basis for the space $S_{\Delta,k}$.

**Definition 4.4.9.**

*Let $\Delta = \{\tau_0 \le \tau_1 \le \cdots \le \tau_m\}$ be an arbitrary sequence of knots such that $\tau_i < \tau_{i+k}$, $i = 0 : m - k$. Then a B-spline of order $k$ (apart from a stepsize factor) equals the Peano kernel of a $k$th order divided-difference functional; more precisely, we define (with the notations used in this chapter)*

$$N_{i,k}(x) = (\tau_{i+k} - \tau_i)[\tau_i, \tau_{i+1}, \ldots, \tau_{i+k}]_t (t - x)_+^{k-1}, \tag{4.4.41}$$

*where $[\tau_i, \tau_{i+1}, \ldots, \tau_{i+k}]l_x^{k-1}$ denotes the $k$th divided difference of the function $l_x^{k-1}(\cdot)$ with respect to the set of points $\tau_i, \tau_{i+1}, \ldots, \tau_{i+k}$. This definition remains valid for knots that are not distinct.*

It can be shown that $N_{i,k}(x)$ is defined for all $x$. If the knots are distinct, then by Problem 4.2.7

$$N_{i,k}(x) = (\tau_{i+k} - \tau_i) \sum_{j=i}^{i+k} \frac{(\tau_j - x)_+^{k-1}}{\Phi'_{i,k}(\tau_j)}, \qquad \Phi_{i,k}(x) = \prod_{j=i}^{i+k} (x - \tau_j), \tag{4.4.42}$$

which shows that $N_{i,k}$ is a linear combination of functions $(\tau_j - x)_+^{k-1}$, $j = i : i + k$. Hence it is a spline of order $k$ (as anticipated in the terminology).

For equidistant knots the B-spline is related to the probability density of the sum of $k$ uniformly distributed random variables on $[-\frac{1}{2}, \frac{1}{2}]$. This was known already to Laplace. B-splines of order $k = 1, 2$, and 3 are shown in Figure 4.4.9.

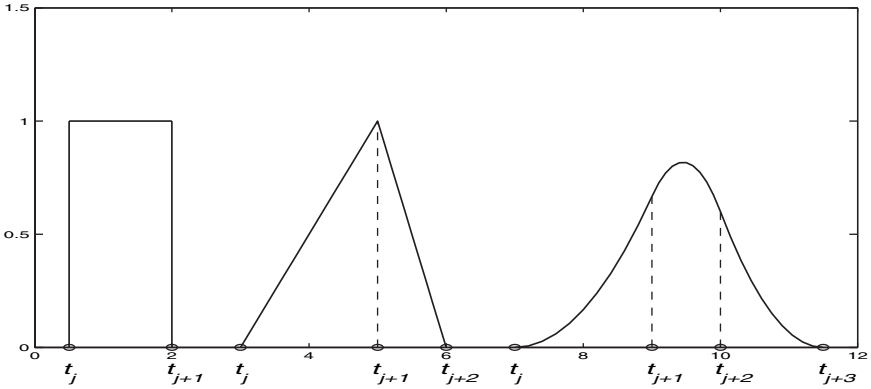



**Figure 4.4.9.** *B-splines of order $k = 1, 2, 3$.*

**Theorem 4.4.10.** *The B-splines of order $k$ have the following properties:*

(i) *Positivity:* $N_{i,k}(x) > 0, \quad x \in (\tau_i, \tau_{i+k})$.

(ii) *Compact support:* $N_{i,k}(x) = 0, \quad x \notin [\tau_i, \tau_{i+k}]$.

(iii) *Summation property:* $\sum_i N_{i,k}(x) = 1$ *for all* $x \in [\tau_0, \tau_m]$.

***Proof.*** A proof can be based on the general facts concerning Peano kernels found in Sec. 3.3.3, where also an expression for the B-spline ($k = 3$) is calculated for the equidistant case. (Unfortunately the symbol $x$ means different things here and in Sec. 3.3.3.)

(i) By (4.2.11), $Rf = [\tau_i, \tau_{i+1}, \ldots, \tau_{i+k}]f = f^{(k)}(\xi)/k!$, $\xi \in (\tau_i, \tau_{i+k})$, and $Rp = 0$, for $p \in \mathcal{P}_k$. It then follows from the corollary of Peano's remainder theorem that the Peano kernel does not change sign in $[\tau_i, \tau_{i+k}]$. It must then have the same sign as $\int K(u)\,du = R(x-a)^k/k! = 1$. This proves a somewhat weaker statement than (i) ($N_{i,k}(x) \geq 0$ instead of $N_{i,k}(x) > 0$).

(ii) This property follows since a Peano kernel always vanishes outside its interval of support of the functional, in this case $[\tau_i, \tau_{i+k}]$. (A more general result concerning the number of zeros is found, e.g., in Powell [292, Theorem 19.1]. Among other things this theorem implies that the $j$th derivative of a B-spline, $j \leq k-2$, changes sign exactly $j$ times. This explains the "bell shape" of B-splines.)

(iii) For a sketch of a proof of the summation property,[141] see Problem 4.4.11. □

To get a basis of B-splines for the space $S_{\Delta,k}$, $\Delta = \{\tau_0 \leq \tau_1 \leq \cdots \leq \tau_m\}$, $(m+k-1)$ B-splines of order $k$ are needed. We therefore choose $2(k-1)$ additional knots $\tau_{-k+1} \leq \cdots \leq \tau_{-1} \leq \tau_0$, $\tau_{m+k-1} \geq \cdots \geq \tau_{m+1} \geq \tau_m$, and B-splines $N_{i,k}(x)$, $i = -k+1 : m-1$.

As for $k = 2$ it is convenient to let the exterior knots coincide with the endpoints,

$$\tau_{-k+1} = \cdots = \tau_{-1} = \tau_0, \qquad \tau_m = \tau_{m+1} = \cdots = \tau_{m+k-1}.$$

It can be shown that this choice tends to optimize the conditioning of the B-spline basis. Figure 4.4.10 shows the first four cubic B-splines for $k = 4$ (the four last B-splines are

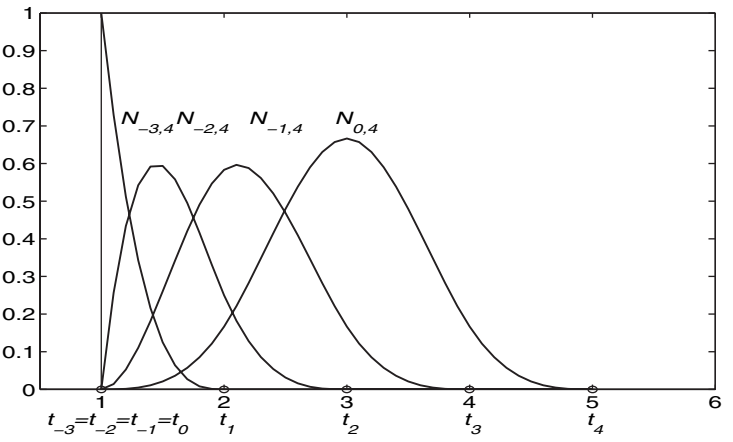



**Figure 4.4.10.** *The four cubic B-splines nonzero for $x \in (t_0, t_1)$ with coalescing exterior knots $t_{-3} = t_{-2} = t_{-1} = t_0$.*

[141] The B-splines $M_{i,k}$ originally introduced by Schoenberg in 1946 were normalized so that $\int_{-\infty}^{\infty} M_{i,k}\,dx = 1$.

a mirror image of these). We note that $N_{-3,4}$ is discontinuous, $N_{-2,4}$ has a nonzero first derivative, and $N_{-2,4}$ has a nonzero second derivative at the left boundary.

Interior knots of multiplicity $r > 1$ are useful when we want to model a function which has less than $k-2$ continuous derivatives at a particular knot. If $r \le k$ interior knots coalesce, then the spline will only have $k-1-r$ continuous derivatives at this knot.

**Lemma 4.4.11.**
*Let $\tau_i$ be a knot of multiplicity $r \le k$, i.e.,*

$$\tau_{i-1} < \tau_i = \cdots = \tau_{i+r-1} < \tau_{i+r}.$$

*Then $N_{i,k}$ is at least $(k-r-1)$ times continuously differentiable at $\tau_i$. For $r = k$, the B-spline becomes discontinuous.*

***Proof.*** The truncated power $(t-\tau_i)_+^{k-1}$ is $(k-2)$ times continuously differentiable and $[\tau_i, \ldots, \tau_{i+k}]g$ contains at most the $(r-1)$st derivative of $g$. Hence the lemma follows. ∎

Consider the spline function

$$s(x) = \sum_{i=-k+1}^{m-1} c_i N_{i,k}(x). \tag{4.4.43}$$

If $s(x) = 0$, $x \in [\tau_0, \tau_m]$, then $s(\tau_0) = s'(\tau_0) = \cdots = s^{(k-1)}(\tau_0) = 0$, and $s(\tau_i) = 0$, $i = 1 : m-1$. From this it can be determined by induction that in (4.4.43) $c_i = 0$, $i = -k+1 : m-1$. This shows that the $(m+k-1)$ B-splines $N_{i,k}(x)$, $i = -k+1 : m-1$, are linearly independent and form a basis for the space $S_{\Delta,k}$. (A more general result is given in de Boor [37, Theorem IX.1].) Thus any spline function $s(x)$ of order $k$ (degree $k-1$) on $\Delta$ can be uniquely written in the form (4.4.43). Note that from the compact support property it follows that for any fixed value of $x \in [\tau_0, \tau_m]$ at most $k$ terms will be nonzero in the sum in (4.4.43), and thus we have

$$s(x) = \sum_{i=j-k+1}^{j} c_i N_{i,k}(x), \quad x \in [\tau_j, \tau_{j+1}). \tag{4.4.44}$$

We will now develop a very important stable recurrence relation for computing B-splines. For this we need the following difference analogue of **Leibniz' formula**.[142]

**Theorem 4.4.12** (*Leibniz' Formula*).
*Let $f(x) = g(x)h(x)$, and $x_i \le x_{i+1} \le \cdots \le x_{i+k}$. Then*

$$[x_i, \ldots, x_{i+k}]f = \sum_{r=i}^{i+k} [x_i, \ldots, x_r]g \cdot [x_r, \ldots, x_{i+k}]h, \tag{4.4.45}$$

[142] Gottfried Wilhelm von Leibniz (1646–1716), a German mathematician who developed his version of calculus at the same time as Newton. Many of the notations he introduced are still used today.

*provided that $g(x)$ and $f(x)$ are sufficiently many times differentiable so that the divided differences on the right-hand side are defined for any coinciding points $x_j$.*

***Proof.*** Note that the product polynomial

$$P(x) = \sum_{r=i}^{i+k} (x - x_i) \cdots (x - x_{r-1})[x_i, \ldots, x_r]g \cdot \sum_{s=i}^{i+k} (x - x_{s+1}) \cdots (x - x_{i+k})[x_s, \ldots, x_{i+k}]h$$

agrees with $f(x)$ at $x_i, \ldots, x_{i+k}$ since by Newton's interpolation formula the first factor agrees with $g(x)$ and the second with $h(x)$ there. If we multiply out we can write $P(x)$ as a sum of two polynomials:

$$P(x) = \sum_{r,s=i}^{i+k} \ldots = \sum_{r \le s} \ldots + \sum_{r > s} \ldots = P_1(x) + P_2(x).$$

Since in $P_2(x)$ each term in the sum has $\prod_{j=i}^{i+k}(x - x_j)$ as a factor, it follows that $P_1(x)$ will also interpolate $f(x)$ at $x_i, \ldots, x_{i+k}$. The theorem now follows since the leading coefficient of $P_1(x)$, which equals $\sum_{r=i}^{i+k}[x_i, \ldots, x_r]g \cdots [x_r, \ldots, x_{i+k}]h$, must equal the leading coefficient $[x_i, \ldots, x_{i+k}]f$ of the unique interpolation polynomial of degree $k$. □

**Theorem 4.4.13.**

*The B-splines satisfy the recurrence relation*

$$N_{i,k}(x) = \frac{x - \tau_i}{\tau_{i+k-1} - \tau_i} N_{i,k-1}(x) + \frac{\tau_{i+k} - x}{\tau_{i+k} - \tau_{i+1}} N_{i+1,k-1}(x). \tag{4.4.46}$$

***Proof.*** (de Boor [37, pp. 130–131]) The recurrence is derived by applying Leibniz' formula for the $k$th divided difference to the product

$$(t - x)_+^{k-1} = (t - x)(t - x)_+^{k-2}.$$

This gives

$$[\tau_i, \ldots, \tau_{i+k}]_t (t - x)_+^{k-1} = (\tau_i - x)[\tau_i, \ldots, \tau_{i+k}]_t (t - x)_+^{k-2} + 1 \cdot [\tau_{i+1}, \ldots, \tau_{i+k}]_t (t - x)_+^{k-2}, \tag{4.4.47}$$

since $[\tau_i]_t(t - x) = (\tau_i - x)$, $[\tau_i, \tau_{i+1}]_t(t - x) = 1$, and $[\tau_i, \ldots, \tau_j]_t(t - x) = 0$ for $j > i + 1$. By the definition of a divided difference

$$(\tau_i - x)[\tau_i, \ldots, \tau_{i+k}]_t = \frac{\tau_i - x}{\tau_{i+k} - \tau_i} \left( [\tau_{i+1}, \ldots, \tau_{i+k}]_t - [\tau_i, \ldots, \tau_{i+k-1}]_t \right).$$

Substitute this in (4.4.47), simplify, and apply the definition of B-splines. This yields (4.4.46). □

Note that with $k$ multiple knots at the boundaries the denominators in (4.4.46) can become zero. In this case the corresponding numerator is also zero and the term should be set equal to zero.

From Property (ii) in Theorem 4.4.10 we conclude that only $k$ B-splines of order $k$ may be nonzero on a particular interval $[\tau_j, \tau_{j+1}]$. Starting from $N_{i,1}(x) = 1, x \in [\tau_i, \tau_{i+1})$ and 0 otherwise (cf. (4.4.37)), these B-splines of order $k$ can be *simultaneously* evaluated using this recurrence by forming successively their values for order $1 : k$ in only about $\frac{3}{2}k^2$ flops. This recurrence is very stable, since it consists of taking a convex combination of two lower-order splines to get the next one.

Suppose that $x \in [\tau_i, \tau_{i+1}]$, and $\tau_i \neq \tau_{i+1}$. Then the B-splines of order $k = 1, 2, 3, \ldots$ nonzero at $x$ can be simultaneously evaluated by computing the following triangular array.

$$\begin{array}{ccccc}
 & & 0 & & \\
 & 0 & & \cdots & \\
0 & & N_{i-3,4} & & \\
 & N_{i-2,3} & & \cdots & \\
N_{i-1,2} & & N_{i-2,4} & & \\
 & N_{i-1,3} & & \cdots & \\
N_{i,2} & & N_{i-1,4} & & \\
 & N_{i,3} & & \cdots & \\
0 & & N_{i,4} & & \\
 & 0 & & \cdots & \\
 & & 0 & &
\end{array} \qquad N_{i,1} \qquad (4.4.48)$$

The boundary of zeros in the array is due to the fact that all other B-splines not mentioned explicitly vanish at $x$. This array can be generated column by column. The first column is known from (4.4.37), and each entry in a subsequent column can be computed as a linear combination with nonnegative coefficients of its two neighbors using (4.4.46). Note that if this is arranged in a suitable order the elements in the new column can overwrite the elements in the old column.

To evaluate $s(x)$, we first determine the index $i$ such that $x \in [\tau_i, \tau_{i+1})$ using, for example, a linear search or bisection (see Sec. 6.1.2). The recurrence above is then used to generate the triangular array (4.4.48) which provides $N_{j,k}(x)$, $j = i - k + 1 : i$, in the sum (4.4.44).

Assume now that $\tau_0 < \tau_1 < \cdots < \tau_m$ are distinct knots. Using the B-spline basis we can formulate a more general interpolation problem, where the $n = m + k - 1$ interpolation points (knots) $x_j$ do not necessarily coincide with the knots $\tau_i$. We consider determining a spline function $s(x) \in S_{\Delta,k}$ such that

$$s(x_j) = f_j, \quad j = 1 : m + k - 1.$$

Since any spline $s(x) \in S_{\Delta,k}$ can be written as a linear combination of B-splines, the interpolation problem can equivalently be written

$$\sum_{i=-k+1}^{m-1} c_i N_{i,k}(x_j) = f_j, \quad j = 1 : m + k - 1. \qquad (4.4.49)$$

These equations form a linear system $Ac = f$ for the coefficients, where

$$a_{ij} = N_{i-k,k}(x_j), \quad i, j = 1 : m + k - 1, \tag{4.4.50}$$

and

$$c = (c_{-k+1}, \ldots, c_{m-1})^T, \qquad f = (f_1, \ldots, f_{m+k-1})^T.$$

The elements $a_{ij} = N_{i-k,k}(x_j)$ of the matrix $A$ can be evaluated by the recurrence (4.4.46). The matrix $A$ will have a banded structure since $a_{ij} = 0$ unless $x_j \in [\tau_i, \tau_{i+k}]$. Hence at most $k$ elements are nonzero in each row of $A$. (Note that if $x_j = \tau_i$ for some $i$ only $k - 1$ elements will be nonzero. This explains why tridiagonal systems were encountered in cubic spline interpolation in earlier sections.)

Schoenberg and Whitney [315] showed that *the matrix A is nonsingular if and only if its diagonal elements are nonzero*,

$$a_{jj} = N_{j-k,k}(x_j) \neq 0, \quad j = 1 : n,$$

or, equivalently, if the knots $x_j$ satisfy

$$\tau_{j-k} < x_j < \tau_j, \quad j = 1 : n. \tag{4.4.51}$$

Further, the matrix can be shown to be **totally nonnegative**, i.e., the determinant of every submatrix is nonnegative. For such systems, if Gaussian elimination is carried out *without pivoting*, the error bound is particularly favorable; see [40]. This will also preserve the banded structure of $A$ during the elimination.

When the B-spline representation (4.4.43) of the interpolant has been determined it can be evaluated at a given point using the recursion formula (4.4.46). If it has to be evaluated more than two or three times per polynomial piece it is more efficient to first convert the B-spline to pp-form (4.4.22). For hints on how to do that, see Problem 4.4.12 (b) and (c). A detailed discussion is found in de Boor [37, Chapter X].

Unless the Schoenberg–Whitney condition (4.4.51) is well-satisfied the system may become ill-conditioned. For splines of even order $k$ the interior knots

$$\tau_0 = x_0, \quad \tau_{j+1} = x_{j+k/2}, \quad j = 0 : n - k - 1, \quad \tau_m = x_n,$$

is a good choice in this respect. In the important case of cubic splines this means that knots are positioned at each data point except the second- and next-last (cf. the not-a-knot condition in Sec. 4.4.2).

### 4.4.4 Least Squares Splines Approximation

In some application we are given function values $f_j = f(x_j)$, $j = 1 : n$, that we want to approximate with a spline functions with *much fewer knots* so that $m + k - 1 \leq n$. Then (4.4.49) is an *overdetermined linear system* and the interpolation conditions cannot be satisfied exactly. We therefore consider the linear **least squares spline approximation** problem:

$$\min \sum_{j=1}^{n} \left( \sum_{i=-k+1}^{m-1} c_i N_{i,k}(x_j) - f_j \right)^2. \tag{4.4.52}$$

Using the same notation as above this can be written in matrix form as

$$\min_c \|Ac - f\|_2^2. \tag{4.4.53}$$

The matrix $A$ will have full column rank equal to $m + k - 1$ if and only if there is a subset of points $\tau_j$ satisfying the Schoenberg–Whitney conditions (4.4.51).

If $A$ has full column rank, then the least squares solution $c$ is uniquely determined by the normal equations $A^T Ac = A^T f$. Since $A$ has at most $k$ nonzero elements in each row the matrix $A^T A$ will have symmetric banded form with at most $2k + 1$ nonzero elements in each row; see Figure 4.4.11.

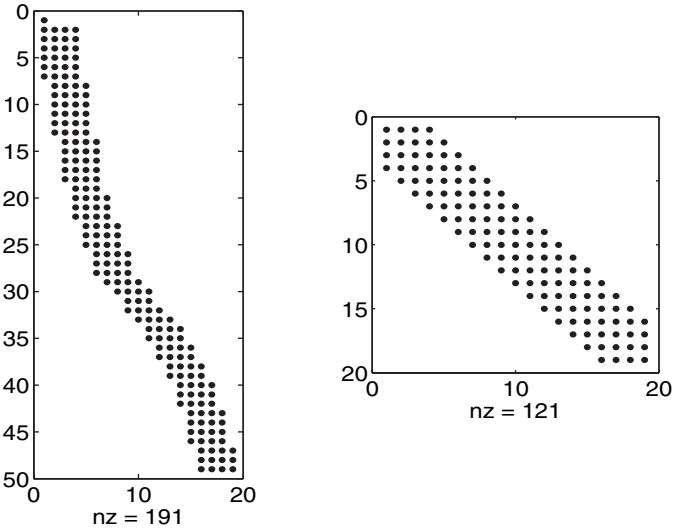



**Figure 4.4.11.** *Banded structure of the matrices A and $A^T A$ arising in cubic spline approximation with B-splines (nonzero elements shown).*

**Example 4.4.3** (*de Boor* [37]).

Consider experimental data describing a property of titanium as a function of temperature. Experimental values for $t_i = 585 + 10i$, $i = 1 : 49$, are given. We want to fit these data using a least squares cubic spline. Figure 4.4.12 shows results from using a least squares fitted cubic spline with 17 knots. The spline with 9 knots shows oscillations near the points where the curve flattens out and the top of the peak is not well matched. Increasing the number of knots to 17 we get a very good fit.

We have in the treatment above assumed that the set of (interior) knots $\{\tau_0 \le \tau_1 \le \cdots \le \tau_m\}$ is given. In many spline approximation problems it is more realistic to consider the location of knots to be free and try to determine a small set of knots such that the given data can be approximated to a some preassigned accuracy. Several schemes have been developed to treat this problem.

One class of algorithms start with only a few knots and iteratively add more knots guided by some measure of the error; see de Boor [36, Chapter XII]. The placement of the

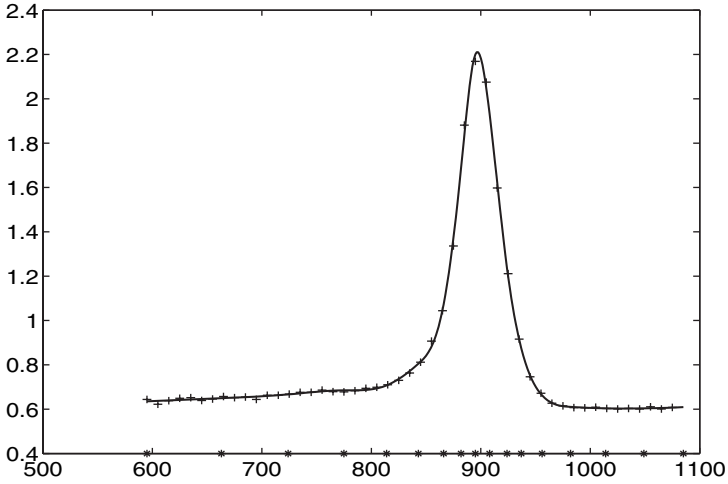



**Figure 4.4.12.** *Least squares cubic spline approximation of titanium data using* 17 *knots marked on the axes by an "o."*

knots is chosen so that the Schoenberg–Whitney conditions are always satisfied. The iterations are stopped when the approximation is deemed satisfactory. If a knot $\bar{\tau} \in [\tau_j, \tau_{j+1})$ is inserted, then the B-spline series with respect to the enlarged set of knots can cheaply and stably be computed from the old one (see Dierckx [99]).

Other algorithms start with many knots and successively *remove* knots which are not contributing much to the quality of the approximation. In these two classes of algorithms one does not seek an optimal knot placement at each step. This is done in more recent algorithms; see Schwetlick and Schütze [319].

## Review Questions

**4.4.1** What is meant by a cubic spline function? Give an example where such a function is better suited than a polynomial for approximation over the whole interval.

**4.4.2** (a) What is the dimension of the space $S_{\Delta,k}$ of spline functions of order $k$ on a grid $\Delta = \{x_0, x_1, \ldots, x_m\}$? Give a basis for this space.

(b) Set up the linear system for cubic spline interpolation in the equidistant case for some common boundary conditions. What do the unknown quantities mean, and what conditions are expressed by the equations? About how many operations are required to interpolate a cubic spline function to $m + 1$, $m \gg 1$, given values of a function?

**4.4.3** What error sources have influence on the results of cubic spline interpolation? How fast do the boundary errors die out? How do the results in the interior of the interval depend on the step size (asymptotically)? One of the common types of boundary conditions yields much larger error than the others. Which one? Compare it quantitatively with one of the others.

**4.4.4** Approximately how many arithmetic operations are required to evaluate the function values of all cubic B-splines that are nonzero at a given point?

**4.4.5** Express the restrictions of $f(x) = 1$ and $f(x) = x$ to the interval $[x_0, x_m]$ as linear combinations of the hat functions defined by (4.4.38).

**4.4.6** The Schoenberg–Whitney conditions give necessary and sufficient conditions for a certain interpolation problem with B-splines of order $k$. What is the interpolation problem and what are the conditions?

## Problems and Computer Exercises

**4.4.1** Consider a cubic Bézier curve defined by the four control points $p_0$, $p_1$, $p_2$, and $p_3$. Show that at $t = 1/2$

$$c\left(\frac{1}{2}\right) = \frac{1}{4}\frac{p_0 + p_3}{2} + \frac{3}{4}\frac{p_1 + p_2}{2}$$

and interpret this formula geometrically.

**4.4.2** (G. Eriksson) Approximate the function $y = \cos x$ on $[-\pi/2, \pi/2]$ by a cubic Bézier curve. Determine the four control points in such a way that it interpolates $\cos x$ and its derivative at $-\pi/2$, 0, and $\pi/2$.

*Hint:* Use symmetry and the result of Problem 4.4.1 to find the $y$-coordinate of $p_1$ and $p_2$.

**4.4.3** Suppose that $f(x)$ and the grid $\Delta$ are symmetric around the midpoint of the interval $[a, b]$. You can then considerably reduce the amount of computation needed for the construction of the cubic spline interpolant by replacing the boundary condition at $x = b$ by an adequate condition at the midpoint. Which?

(a) Set up the matrix and right-hand side for this in the case of constant step size $h$.

(b) Do the same for a general case of variable step size.

**4.4.4** (a) Write a program for solving a tridiagonal linear system by Gaussian elimination without pivoting. Assume that the nonzero diagonals are stored in three vectors. Adapt it to cubic spline interpolation with equidistant knots with several types of boundary conditions.

(b) Consider the tridiagonal system resulting from the not-a-knot boundary conditions. Show that after eliminating $k_0$ between the first two equations and $k_m$ between the last two equations the remaining tridiagonal system for $k_1, \ldots, k_{m-1}$ is diagonally dominant.

(c) Interpolate a cubic spline $s(x)$ through the points $(x_i, f(x_i))$, where

$$f(x) = (1 + 25x^2)^{-1}, \quad x_i = -1 + \frac{2}{10}(i - 1), \quad i = 1 : 11.$$

Compute a natural spline, a complete spline (here $f'(x_1)$ and $f'(x_{11})$ are needed) and a not-a-knot spline. Compute and compare error curves (natural and logarithmic).

(d) Conduct similar runs as in (b), though for $f(x) = 1/x$, $1 \le x \le 2$, with $h = 0.1$

and $h = 0.05$. Compare the "almost complete," as described in the text, with the complete and the natural boundary condition.

**4.4.5** If $f''$ is known at the boundary points, then the boundary conditions can be chosen so that $f'' = s''$ at the boundary points. Show that this leads to the conditions

$$2k_0 + k_1 = 3d_1 - h_1 f''(x_0),$$
$$k_{m-1} + 2k_m = 3d_m + h_m f''(x_m).$$

**4.4.6** Show that the formula

$$\int_{x_0}^{x_m} s(x)dx = \sum_{i=1}^{m} \Big( \frac{1}{2} h_i (y_{i-1} + y_i) + \frac{1}{12}(k_{i-1} - k_i) h_i^2 \Big)$$

is exact for all cubic spline functions $s(x)$. How does the formula simplify if all $h_i = h$?

*Hint:* Integrate (4.4.16) from $x_{i-1}$ to $x_i$.

**4.4.7** In (4.4.16) the cubic spline $q_i(x)$ on the interval $[x_{i-1}, x_i)$ is expressed in terms of function values $y_{i-1}$, $y_i$, and the first derivatives $k_{i-1}$, $k_i$.

(a) Show that if $M_i = s''(x_i)$, $i = 0 : m$, are the *second derivatives* (also called **moments**) of the spline function, then

$$k_i - d_i = \frac{h_i}{6}(2M_i + M_{i-1}), \qquad k_{i-1} - d_i = -\frac{h_i}{6}(M_i + 2M_{i-1}).$$

Hence $q_i(x)$ can also be uniquely expressed in terms of $y_{i-1}$, $y_i$ and $M_{i-1}$, $M_i$.

(b) Show that, using the parametrization in (a), the continuity of the *first* derivative of the spline function at an interior point $x_i$ gives the equation

$$h_i M_{i-1} + 2(h_i + h_{i+1})M_i + h_{i+1} M_{i+1} = 6(d_{i+1} - d_i).$$

**4.4.8** (a) Develop an algorithm for solving the arrowhead linear system $Tk = g$ (4.4.32), using Gaussian elimination without pivoting. Show that about twice the number of arithmetic operations are needed compared to a tridiagonal system.

(b) At the end of Sec. 4.4.2 parametric spline interpolation to given points $(x_i, y_i)$, $i = 0 : m$, is briefly mentioned. Work out the details on how to use this to represent a closed curve. Try it out on a boomerang, an elephant, or what have you.

**4.4.9** (a) Compute and plot a B-spline basis of order $k = 3$ (locally quadratic) and $m = 6$ subintervals of equal length.

*Hint:* In the equidistant case there is some translation invariance and symmetry, so you do not really need more than essentially three different B-splines. You need one spline with triple knot at $x_0$ and a single knot at $x_1$ (very easy to construct), and two more splines.

(b) Set up a scheme to determine a locally quadratic B-spline which interpolates given values at the *midpoints*

$$x_i = (\tau_{i+1} + \tau_i)/2, \quad (\tau_{i+1} \neq \tau_i), \quad i = 0 : m - 1,$$

and the boundary points $\tau_0$, $\tau_m$. Show that the spline is uniquely determined by these interpolation conditions.

**4.4.10** Derive the usual formula of Leibniz for the $k$th derivative from (4.4.45) by a passage to the limit.

**4.4.11** Use the recurrence (4.4.46)

$$N_{i,k}(x) = \frac{x - \tau_i}{\tau_{i+k-1} - \tau_i} N_{i,k-1}(x) + \frac{\tau_{i+k} - x}{\tau_{i+k} - \tau_{i+1}} N_{i+1,k-1}(x)$$

to show that

$$\sum_i N_{i,k}(x) = \sum_i N_{i,k-1}(x), \quad \tau_0 \le x \le \tau_m,$$

where the sum is taken over all nonzero values. Use this to give an induction proof of the summation property in Theorem 4.4.10.

**4.4.12** (a) Using the result $\frac{d}{dx}(t-x)_+^{k-1} = -(k-1)(t-k)_+^{k-2}$, $k \ge 1$, show the formula for differentiating a B-spline,

$$\frac{d}{dx} N_{i,k}(x) = (k-1)\left( \frac{N_{i,k-1}(x)}{\tau_{i+k-1} - \tau_i} - \frac{N_{i+1,k-1}(x)}{\tau_{i+k} - \tau_{i+1}} \right).$$

Then use the relation (4.4.46) to show

$$\frac{d}{dx} \sum_{i=r}^{s} c_i N_{i,k}(x) = (k-1) \sum_{i=r}^{s+1} \frac{c_i - c_{i-1}}{\tau_{i+k-1} - \tau_i} N_{i,k-1}(x),$$

where $c_{r-1} := c_{s+1} := 0$.

(b) Given the B-spline representation of a cubic spline function $s(x)$. Show how to find its polynomial representation (4.4.22) by computing the function values and first derivatives $s(\tau_i)$, $s'(\tau_i)$, $i = 0 : m$.

(c) Apply the idea in (a) recursively to show how to compute all derivatives of $s(x)$ up to order $k-1$. Use this to develop a method for computing the polynomial representation of a spline of arbitrary order $k$ from its B-spline representation.

**4.4.13** Three different bases for the space of cubic polynomials of degree $\le 3$ on the interval $[0, 1]$ are the monomial basis $\{1, t, t^2, t^3\}$, the Bernštein basis $\{B_0^3(t), B_1^3(t), B_2^3(t), B_3^3(t)\}$, and the Hermite basis. Determine the matrices for these basis changes.

## 4.5 Approximation and Function Spaces

Function space concepts have been introduced successively in this book. Recall, for example, the discussion of operators and functionals in Sec. 3.3.2, where the linear space $\mathcal{P}_n$, the $n$-dimensional space of polynomials of degree less than $n$, was also introduced. This terminology was used and extended in Sec. 4.1, in the discussion of various bases and coordinate transformations in $\mathcal{P}_n$.

For forthcoming applications of functional analysis to interpolation and approximation it is now time for a digression about

- distances and norms in function spaces;
- a general error bound that we call *the norm and distance formula*;
- inner-product spaces and orthogonal systems.

### 4.5.1 Distance and Norm

For the study of accuracy and convergence of methods of interpolation and approximation we now introduce the concept of a **metric space**. By this we understand a set of points $\mathcal{S}$ and a real-valued function $d$, a **distance** defined for pairs of points in $\mathcal{S}$ in such a way that the following axioms are satisfied for all $x, y, z$ in $\mathcal{S}$. (Draw a triangle with vertices at the points $x, y, z$.)

1. $d(x, x) = 0$ (reflexivity),
2. $d(x, y) > 0$ if $x \neq y$ (positivity),
3. $d(x, y) = d(y, x)$ (symmetry),
4. $d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality).

The axioms reflect familiar features of distance concepts used in mathematics and everyday life, such as the absolute value of complex numbers, the shortest distance along a geodesic on the surface of the Earth, or the shortest distance along a given road system.[143]

Many other natural and useful relations can be derived from these axioms, such as

$$d(x, y) \geq |d(x, z) - d(y, z)|, \qquad d(x_1, x_n) \leq \sum_{i=1}^{n-1} d(x_i, x_{i+1}), \tag{4.5.1}$$

where $x_1, x_2, \ldots, x_n$ is a sequence of points in $\mathcal{S}$.

**Definition 4.5.1.**

*A sequence of points $\{x_n\}$ in a metric space $\mathcal{S}$ is said to* **converge** *to a limit $x^* \in \mathcal{S}$ if $d(x_n, x^*) \to 0$. As $n \to \infty$, we write $x_n \to x^*$ or $\lim_{n\to\infty} x_n = x^*$. A sequence $\{x_n\}$ in $\mathcal{S}$ is called a* **Cauchy sequence** *if for every $\epsilon > 0$ there is an integer $N(\epsilon)$ such that $d(x_m, x_n) < \epsilon$ for all $m, n \geq N(\epsilon)$. Every convergent sequence is a Cauchy sequence, but the converse is not necessarily true. $\mathcal{S}$ is called a* **complete space** *if every Cauchy sequence in $\mathcal{S}$ converges to a limit in $\mathcal{S}$.*

It is well known that $\mathbf{R}$ satisfies the characterization of a complete space, but the set of *rational* numbers is not complete. For example, the Newton iteration $x_1 = 1$, $x_{n+1} = \frac{1}{2}(x_n + 2/x_n)$, defines a sequence of rational numbers that converges to $\sqrt{2}$, which is not a rational number.

Many important problems in pure and applied mathematics can be formulated as minimization problems. The function space terminology often makes proofs and algorithms less abstract.

[143] If $\mathcal{S}$ is a functions space, the points of $\mathcal{S}$ are functions with operands in some other space, for example, in $\mathbf{R}$ or $\mathbf{R}^n$.

Most spaces that we shall encounter in this book are **linear spaces**. Their elements are called vectors, which is why these spaces are called **vector spaces**. Two operations are defined in these spaces, namely the addition of vectors and the multiplication of a vector by a scalar. They obey the usual rules of algebra.[144] The set of scalars can be either **R** or **C**; the vector space is then called *real* or *complex*, respectively.

**Definition 4.5.2.**

*Let $f$ be in a metric space $\mathcal{B}$ with a distance function $d(x, y)$, and let $\mathcal{S}$ be a subset or linear subspace of $\mathcal{B}$. We define the distance of $f$ to $\mathcal{S}$ by*

$$\operatorname{dist}(f, \mathcal{S}) = \inf_{g \in \mathcal{S}} d(f, g). \tag{4.5.2}$$

Much of our discussion also applies to linear spaces of **infinite dimension**, i.e., **function spaces**. The elements (vectors) then are functions of one or several real variables on a compact set, i.e., a closed bounded region. The idea of such a functions space is now illustrated in an example.

**Example 4.5.1.**

Consider the set of functions representable by a convergent power series on the interval $[-1, 1]$,

$$f(t) = c_0 + c_1 t + c_2 t^2 + \cdots.$$

This is an infinite-dimensional linear space. The functions $1, t, t^2, \ldots$ can be considered as a standard basis of this space. The coordinates of $f(t)$ then are the vector $c_0, c_1, c_2, \ldots$.

We shall be concerned with the problem of **linear approximation**, i.e., a function $f$ is to be approximated using a function $f^*$ that can be expressed as a linear combination of $n$ given linearly independent functions $\phi_1(x), \phi_2(x), \ldots, \phi_n(x)$,

$$f^* = c_1\phi_1(x) + c_2\phi_2(x) + \cdots + c_n\phi_n(x), \tag{4.5.3}$$

where $c_1, c_2, \ldots, c_n$ are parameters to be determined.[145] They may be considered as coordinates of $f^*$ in the functions space spanned by $\phi_1(x), \phi_2(x), \ldots, \phi_n(x)$.

In a vector space the distance of the point $f$ from the origin is called the **norm** of $f$ and denoted by $\|f\|$, typically with some subscript that more or less cryptically indicates the relevant space. The definition of the norm depends on the space. The following axioms must be satisfied.

**Definition 4.5.3.**

*A real-valued function $\|f\|$ is called a norm on a vector space $\mathcal{S}$ if it satisfies conditions* 1–3 *below for all $f, g \in \mathcal{S}$, and for all scalars $\lambda$.*

[144] See Sec. A.1 in Online Appendix A for a summary about vector spaces. In more detailed texts on linear algebra or functional analysis you find a collection of eight axioms (commutativity, associativity, etc.) required by a linear vector space.

[145] The functions $\phi_j$, however, are typically *not* linear. The term "linear interpolation" is, from our present point of view, rather misleading.

1. $\|f\| > 0$, *unless* $f = 0$ *(positivity),*
2. $\|\lambda f\| = |\lambda|\|f\|$ *(homogeneity),*
3. $\|f + g\| \le \|f\| + \|g\|$ *(triangle inequality).*

A normed vector space is a metric space with the distance

$$d(x, y) = \|x - y\|.$$

If it is also a complete space, it is called a **Banach space**.[146]

The most common norms in spaces of (real and complex) infinite sequences $x = (\xi_1, \xi_2, \xi_3, \ldots)^T$ or spaces of functions on a bounded and closed interval $[a, b]$ are

$$\|x\|_\infty = \max_j |\xi_j|, \qquad \|f\|_\infty = \max_{x\in[a,b]} |f(x)|,$$

$$\|x\|_2 = \left(\sum_{j=1}^{\infty} |\xi_j|^2\right)^{1/2}, \qquad \|f\|_2 = \|f\|_{2,[a,b]} = \left(\int_a^b |f(x)|^2\,dx\right)^{1/2},$$

$$\|x\|_1 = \sum_{j=1}^{\infty} |\xi_j|, \qquad \|f\|_1 = \|f\|_{1,[a,b]} = \int_a^b |f(x)|\,dx.$$

These norms are called

- the $\ell_\infty$ or **max(imum) norm** (or uniform norm);
- the **Euclidean norm** (or the $l_2$-norm for coordinate sequences and $L_2$ norm for integrals);
- the $\ell_1$-norm.

It is possible to introduce weight function $w(x)$, which is continuous and strictly positive on the open interval $(a, b)$, into these norms. For example,

$$\|x\|_{2,w} = \left(\sum_{j=1}^{\infty} w_j|\xi_j|^2\right)^{1/2}, \qquad \|f\|_{2,w} = \left(\int_a^b |f(x)|^2 w(x)\,dx\right)^{1/2},$$

is the **weighted Euclidean norm**. We assume that the integrals

$$\int_a^b |x|^k w(x)\,dx$$

exist for all $k$. Integrable singularities at the endpoints are permitted; an important example is $w(x) = (1 - x^2)^{-1/2}$ on the interval $[-1, 1]$.

[146]Stefan Banach (1892–1945), a Polish mathematician at the University in Lvov. Banach founded modern functional analysis and made major contributions to the theory of topological vector spaces, measure theory, and related topics. In 1939 he was elected President of the Polish Mathematical Society.

The above norms are special cases or limiting cases ($p \to \infty$ gives the max norm) of the $l_p$- or $L_p$-norms and weighted variants of these. They are defined for $p \geq 1$ as follows:[147]

$$\|x\|_p = \left( \sum_{j=1}^{\infty} |\xi_j|^p \right)^{1/p}, \qquad \|f\|_p = \left( \int_a^b |f(x)|^p \, dx \right)^{1/p}. \tag{4.5.4}$$

(The sum in the $l_p$-norm has a finite number of terms, if the space is finite dimensional.)

Convergence in a space, equipped with the max norm, means **uniform convergence**. Therefore, *the completeness of the space $C[a, b]$ follows from a classical theorem of analysis* that tells us that the limit of a uniformly convergent sequence is a continuous function. The generalization of this theorem to several dimensions implies the completeness of the space of continuous functions, equipped with the max norm on a closed bounded region in $\mathbf{R}^n$.

Other classes of functions can be normed with the max norm $\max_{x\in[a,b]} |f(x)|$, for example, $C^1[a, b]$. This space is not complete, but one can often live well with incompleteness.

The notation $L_2$-norm comes from the function space $L_2[a, b]$, which is the class of functions for which the integral $\int_a^b |f(x)|^2 \, dx$ exists, in the sense of Lebesgue. The Lebesgue integral is needed in order to make the space complete and greatly extends the scope of Fourier analysis. No knowledge of Lebesgue integration is needed for the study of this book, but this particular fact can be interesting as background. One can apply this norm also to the (smaller) class of continuous functions on $[a, b]$. In this case the Riemann integral is equivalent. This also yields a normed linear space but *it is not complete*.[148]

Although $C[0, 1]$ is an infinite-dimensional space, the restriction of the continuous functions $f$ to the equidistant grid defined by $x_i = ih$, $h = 1/n$, $i = 0 : n$, constitutes an $(n + 1)$-dimensional space, with the function values on the grid as coordinates. If we choose the norm

$$\|f\|_{2,G_h} = \left( h \sum_{i=0}^{1/h} |f(x_i)|^2 \right)^{1/2},$$

then

$$\lim_{h\to 0} \|f\|_{2,G_h} = \left( \int_0^1 |f(x)|^2 \, dx \right)^{1/2} = \|f\|_{2,[0,1]}.$$

Limit processes of this type are common in numerical analysis.

Notice that even if $n + 1$ functions $\phi_1(x), \phi_1(x), \ldots, \phi_{n+1}(x)$ are linearly independent on the interval $[0, 1]$ (say), their restrictions to a grid with $n$ points must be linearly dependent; but if a number of functions are linearly independent on a set $\mathcal{M}$ (a discrete set or continuum), any extensions of these functions to a set $\mathcal{M}' \supset \mathcal{M}$ will also be linearly independent.

[147] The triangle inequality for $\|x\|_p$ is derived from two classical inequalities due to Hölder and Minkowski. Elegant proofs of these are found in Hairer and Wanner [178, p. 327].

[148] A modification of the $L_2$-norm that also includes higher derivatives of $f$ is used in the Sobolev spaces, which are used as a theoretical framework for the study of the practically very important finite element methods (FEMs), used in particular for the numerical treatment of partial differential equations.

The class of functions, analytic in a simply connected domain $\mathcal{D} \subset \mathbf{C}$, normed with $\|f\|_{\mathcal{D}} = \max_{z\in\partial\mathcal{D}} |f(z)|$, is a Banach space denoted by $\mathbf{Hol}(\mathcal{D})$. (The explanation of this term is that analytic functions are also called **holomorphic**.) By the maximum principle for analytic functions $|f(z)| \le \|f\|_{\mathcal{D}}$ for $z \in \mathcal{D}$.

### 4.5.2 Operator Norms and the Distance Formula

The concepts of **linear operator** and **linear functional** were introduced in Sec. 3.3.2. Here we extend to a general vector space $\mathcal{B}$ some definitions for a finite-dimensional vector space given in Online Appendix A.

Next we shall generalize the concept **operator norm** that is used for matrices; see Sec. A.4.3 in Online Appendix A. Consider an arbitrary bounded linear operator $A : S_1 \mapsto S_2$ in a normed vector space $S$:

$$\|A\| = \sup_{\|f\|_{S_1}} \|Af\|_{S_2}. \tag{4.5.5}$$

Note that $\|A\|$ depends on the vector norm in both $S_1$ and $S_2$. It follows that $\|Af\| \le \|A\|\|f\|$. Moreover, whenever the ranges of the operators $A_1, A_2$ are such that $A_1 + A_2$ and $A_1A_2$ are defined,

$$\|\lambda A\| \le |\lambda|\|A\|, \qquad \|A_1 + A_2\| \le \|A_1\| + \|A_2\|, \qquad \|A_1 \cdot A_2\| \le \|A_1\| \cdot \|A_2\|. \tag{4.5.6}$$

Sums with an arbitrary number of terms and integrals are defined similarly. It follows that $\|A^n\| \le \|A\|^n$, $n = 2, 3, \ldots$.

**Example 4.5.2.**

Let $f \in C[0, 1]$, $\|f\| = \|f\|_\infty$,

$$Af = \sum_{i=1}^{m} a_i f(x_i) \;\Rightarrow\; \|A\| = \sum_{i=1}^{m} |a_i|,$$

$$Af = \int_0^1 e^{-x} f(x)\,dx \;\Rightarrow\; \|A\| = \int_0^1 e^{-x}\,dx = 1 - e^{-1},$$

$$B = \sum_{i=1}^{m} a_i A^i \;\Rightarrow\; \|B\| \le \sum_{i=1}^{m} |a_i|\|A\|^i, \quad (a_i \in \mathbf{C}),$$

$$Kf = \int_0^1 k(x,t) f(t)\,dt \;\Rightarrow\; \|K\|_\infty \le \sup_{x\in[0,1]} \int_0^1 |k(x,t)|\,dt.$$

The proofs of these results are left as a problem. In the last example, approximate the unit square by a uniform grid $(x_i, t_j)_{i,j=1}^m$, $h = 1/m$, and approximate the integrals by Riemann sums. Then approximate $\|K\|_\infty$ by the max norm for the matrix with the elements $k_{i,j} = hk(x_i, t_j)$; see Sec. A.4.3 in Online Appendix A.

**Example 4.5.3.**

For the forward difference operator $\Delta$ we obtain $\|\Delta\|_\infty = 2$, hence $\|\Delta^k\|_\infty \le 2^k$. In fact $\|\Delta^k\|_\infty = 2^k$, because the upper bound $2^k$ is attained (for every integer $k$) by the sequence $\{(-1)^n\}_0^\infty$.

**Example 4.5.4.**

Let $\mathcal{D}$ be a domain in $\mathbf{C}$, the interior of which contains the closed interval $[a, b]$. Define the mapping $D^k$: $\mathbf{Hol}\,\mathcal{D} \Rightarrow C[a, b]$ (with max norm), by

$$D^k f(x) = \frac{\partial^k}{\partial x^k} \frac{1}{2\pi i} \int_{\partial \mathcal{D}} f(z) \frac{1}{(z-x)}\, dz = \frac{1}{2\pi i} \int_{\partial \mathcal{D}} \frac{k! f(z)}{(z-x)^{k+1}}\, dz.$$

Then

$$sup_{x\in[a,b]} \|D^k f(x)\| \le \max_{z\in\partial\mathcal{D}} |f(z)| \cdot \sup_{x\in[a,b]} \frac{k!}{2\pi} \int_{\partial\mathcal{D}} \frac{|dz|}{|z-x|^{k+1}} < \infty.$$

Note that $D^k$ is in this setting a *bounded* operator, while if we had considered $D^k$ to be a mapping from $C^k[a, b]$ to $C[a, b]$, where both spaces are normed with the max norm in $[a, b]$, $D^k$ would have been an *unbounded* operator.

Many of the procedures for the approximate computation of linear functionals that we encounter in this book may be characterized as follows. Let $A$ be a linear functional such that $Af$ cannot be easily computed for an arbitrary function $f$, but it can be approximated by another linear functional $\tilde{A}_k$, more easily computable, such that $\tilde{A}_k f = Af$ for all $f \in \mathcal{P}_k$. A general error bound to such procedures was given in Sec. 3.3.3 by the Peano remainder theorem, in terms of an integral

$$\int f^{(k)}(u) K(u)\, du,$$

where the Peano kernel $K(u)$ is determined by the functional $R = \tilde{A} - A$.

Now we shall give a different type of error bound for more general approximation problems, where other classes of functions and operators may be involved. Furthermore, no estimate of $f^{(k)}(u)$ is required. It is based on the following almost trivial theorem. It yields, however, often less sharp bounds than the Peano formula, in situations when the latter can be applied.

**Theorem 4.5.4** (*The Norm and Distance Formula*).

*Let $A$, $\tilde{A}$ be two linear operators bounded in a Banach space $\mathcal{B}$ such that for any vector $s$ in a certain linear subspace $S$, $\tilde{A}s = As$. Then*

$$\|\tilde{A}f - Af\| \le \|\tilde{A} - A\|\, \mathrm{dist}\,(f, \mathcal{S}) \quad \forall f \in \mathcal{B}.$$

***Proof.*** Set $R = \tilde{A} - A$. For any positive $\epsilon$, there exists a vector $s_\epsilon \in \mathcal{S}$ such that

$$\|f - s_\epsilon\| = \mathrm{dist}\,(f, s_\epsilon) < \inf_{s\in\mathcal{S}} \mathrm{dist}\,(f, s) + \epsilon = \mathrm{dist}\,(f, \mathcal{S}) + \epsilon.$$

Then $\|Rf\| = \|Rf - Rs_\epsilon\| = \|R(f - s_\epsilon)\| \le \|R\|\|f - s_\epsilon\| < (\mathrm{dist}\,(f, \mathcal{S}) + \epsilon)\|R\|$. The theorem follows, since this holds for every $\epsilon > 0$. □

The following is a common particular case of the theorem. *If $A$, $A_k$ are* **linear functionals** *such that $A_k p = Ap$ for all $p \in \mathcal{P}_k$, then*

$$|A_k f - Af| \le (\|A_k\| + \|A\|)\mathrm{dist}\,(f, \mathcal{P}_k). \tag{4.5.7}$$

Another important particular case of the theorem concerns projections $P$ from a function space to a finite-dimensional subspace $\mathcal{S}_k$, for example, *interpolation and series truncation operators*, $A = I$, $\tilde{A} = P$; see the beginning of Sec. 4.5.2. Then

$$\|Pf - f\| \le \|(P - I)f\| \le (\|P\| + 1)\text{dist}\,(f, \mathcal{S}_k). \tag{4.5.8}$$

The norm and distance formula requires bounds for $\|\tilde{A}\|$, $\|A\|$ and dist $(f, \mathcal{S})$. We have seen examples above of how to obtain bounds for operator norms. Now we shall exemplify how to obtain bounds for the distance of $f$ from some relevant subspace $\mathcal{S}$, in particular spaces of polynomials or trigonometric polynomials restricted to some real interval $[a, b]$. For the efficient estimation of dist $(f, \mathcal{S})$ it may be important to take into account that $f$ is analytic in a larger domain than $[a, b]$.

**Theorem 4.5.5** (*Estimation of* dist $_\infty(f, \mathcal{P}_k)$ *in Terms of* $\|f^{(k)}\|_\infty$).
*Let $f \in C^k[a, b] \subset \mathbf{R}$, and define the norm*

$$\|g\|_{\infty,[a,b]} = \max_{t\in[a,b]} |g(t)| \quad \forall\ g \in C[a, b].$$

*Then*

$$\text{dist}\,_{\infty,[a,b]}(f, \mathcal{P}_k) \le \frac{2}{k!}\Big(\frac{b-a}{4}\Big)^k \|f^{(k)}\|_{\infty,[a,b]}.$$

***Proof.*** Let $p(t)$ be the polynomial which interpolates $f(t)$ at the points $t_j \in [a, b]$, $j = 1 : k$. By the remainder term (4.2.10) in interpolation,

$$|f(t) - p(t)| \le \max_{\xi\in[a,b]} \frac{|f^{(k)}(\xi)|}{k!} \prod_{j=1}^{k} |t - t_j|.$$

Set $t = \frac{1}{2}(b+a)+\frac{1}{2}(b-a)x$, and choose $t_j = \frac{1}{2}(b+a)+\frac{1}{2}(b-a)x_j$, where $x_j$ are the zeros of the Chebyshev polynomial $T_k(x)$. Then $p$ is the Chebyshev interpolation polynomial for $f$ on the interval $[a, b]$, and with $M = \max_{t\in[a,b]} |f^{(k)}|/k!$,

$$|f(t) - p(t)| \le M \prod_{j=1}^{k} \frac{(b-a)|x - x_j|}{2}, \quad x \in [-1, 1].$$

Since the leading coefficient of $T_k(x)$ is $2^{k-1}$ and $|T_k(x)| \le 1$, we have, for $t \in [a, b]$,

$$|f(t) - p(t)| \le M \left(\frac{b-a}{2}\right)^k \frac{1}{2^{k-1}} |T_k(x)| \le M \left(\frac{b-a}{2}\right)^k \frac{1}{2^{k-1}}.$$

The bound stated for dist $_\infty(f, \mathcal{P}_k)$ is thus satisfied. □

**Example 4.5.5.**

By the above theorem, the function $e^t$ can be approximated on the interval $[0, 1]$ by a polynomial in $\mathcal{P}_6$ with the error bound $2e \cdot 4^{-6}/6! \approx 2 \cdot 10^{-6}$. According to the proof this accuracy is attained by Chebyshev interpolation on $[0, 1]$.

If one instead uses the Maclaurin series, truncated to $\mathcal{P}_6$, then the remainder is $e^\theta/(6!) \ge 1.3 \cdot 10^{-3}$. Similarly, with the truncated Taylor series about $t = \frac{1}{2}$ the remainder

is $e^{\theta}/(2^6 6!) \geq 2 \cdot 10^{-5}$, still significantly less accurate than the Chebyshev interpolation. Economization of power series (see Problem 3.2.5), yields approximately the same accuracy as Chebyshev interpolation.

If we do these things on an interval of length $h$ (instead of the interval [0, 1]) all the bounds are to multiplied by $h^6$.

**Definition 4.5.6.**

*Let $f \in C[a, b]$ and $\mathcal{P}_n$ be the space of polynomials of degree less than $n$. Then we set*

$$E_n(f) = \operatorname{dist}_{\infty}(f, \mathcal{P}_n). \tag{4.5.9}$$

**Example 4.5.6.**

The use of analyticity in estimates for $\operatorname{dist}_{\infty}(f, \mathcal{P}_n)$: Denote by $\mathcal{E}_R$ an ellipse in $\mathbf{C}$ with foci at $-1$ and 1; $R$ is equal to the sum of the semiaxes. Bernštein's approximation theorem, Theorem 3.2.5, gives the following truncation error bound for the Chebyshev expansion for a function $f \in \mathbf{Hol}(\mathcal{E}_R)$, real-valued on $[-1, 1]$ and with $\|f\|_{\mathcal{E}_R} \leq M$:

$$\left| f(x) - \sum_{j=0}^{n-1} c_j T_j(x) \right| \leq \frac{2MR^{-n}}{1 - R^{-1}}, \quad x \in [-1, 1].$$

This implies that, on the same assumptions concerning $f$,

$$\operatorname{dist}_{\infty,[-1,1]}(f, \mathcal{P}_n) \leq \frac{2MR^{-n}}{1 - R^{-1}}.$$

By the application of a theorem of Landau concerning bounded power series one can obtain the following bound that can be sharper when $R \approx 1$:

$$\operatorname{dist}_{\infty,[-1,1]}(f, \mathcal{P}_n) \leq 2MR^{-(n+1)} G_n, \tag{4.5.10}$$

where

$$G_n = 2 + \left(\frac{1}{2}\right)^2 + \left(\frac{1 \cdot 3}{2 \cdot 4}\right)^2 + \cdots + \left(\frac{1 \cdot 3 \cdots (2n-1)}{2 \cdot 4 \cdots 2n}\right)^2 \sim \frac{\log n}{\pi}.$$

Suppose that $f \in \mathbf{Hol}(\mathcal{D})$, where $\mathcal{D} \supseteq \frac{1}{2}(b + a) + \frac{1}{2}(b - a)\mathcal{E}_R$. Then transforming from $[-1, 1]$ to a general interval $[a, b] \subset \mathbf{R}$, we have

$$\operatorname{dist}_{\infty,[a,b]}(f, \mathcal{P}_n) \leq 2\|f\|_{\infty,\mathcal{D}} \left(\frac{b-a}{2R}\right)^n \frac{2R}{2R - (b-a)}.$$

**Example 4.5.7.**

As a first simple example we shall derive an error bound for one step with the trapezoidal rule. Set

$$Af = \int_0^h f(x)\, dx, \qquad A_2 f = \frac{h}{2}\left(f(0) + f(h)\right).$$

We know that $A_2 p = Ap$ if $p \in \mathcal{P}_2$. By Theorem 4.5.5, $\operatorname{dist}_{\infty}(f, \mathcal{P}_2) \leq \|f''\|_{\infty} h^2/16$.

Furthermore, $\|A\|_\infty = \int_0^h dx = h$, $\|A_2\|_\infty = h$, hence by (4.5.7) the requested error bound becomes

$$\|A_2 f - Af\|_\infty \le 2h \cdot \|f''\|_\infty h^2/16 = \|f''\|_\infty h^3/8.$$

This general method does not always give the best possible bounds but, typically, it gives no gross overestimate. For the trapezoidal rule we know by Peano's method (Example 3.3.7) that $\|f''\|h^3/12$ is the best possible estimate, and thus we now obtained a 50% overestimate of the error.

The norm and distance formula can also be written in the form

$$\text{dist}\,(f, \mathcal{S}) \ge |Af - \tilde{A}f|/\|A - \tilde{A}\|. \tag{4.5.11}$$

This can be used for finding a simple lower bound for dist $(f, \mathcal{P}_k)$ in terms of an easily computed functional that vanishes on $\mathcal{P}_k$.

**Example 4.5.8.**

Let $\tilde{A} = 0$. The functional $Af = f(1) - 2f(0) + f(-1)$ vanishes for $f \in \mathcal{P}_2$. If the maximum norm is used on [-1,1], then $\|A\| = 1 + 2 + 1 = 4$. Thus

$$\text{dist}\,(f, \mathcal{P}_2)_{\infty,[-1,1]} \ge \frac{|Af|}{\|A\|} = \frac{1}{4}|f(1) - 2f(0) + f(-1)|.$$

It follows, for example, that the curve $y = e^x$ cannot be approximated by a straight line in $[-1, 1]$ with an error less than $(e - 2 + e^{-1})/4 \approx 0.271$. (This can also be seen without the use of the norm and distance formula.)

It is harder to derive the following generalization without the norm and distance formula. By Example 4.5.3, $\|\Delta^k\| = 2^k$, $\Delta^k p = 0$, if $p \in \mathcal{P}_k$; hence

$$\text{dist}\,(f, \mathcal{P}_k)_{\infty,[x_0,x_k]} \ge 2^{-k}|\Delta^k f(x_0)|. \tag{4.5.12}$$

There is another inequality that is usually sharper but less convenient to use (it follows from the discrete orthogonality property of the Chebyshev polynomials; see Sec. 4.6):

$$\text{dist}\,(f, \mathcal{P}_k)_{\infty,[x_0,x_k]} \ge \frac{1}{k}\left|\sum_{j=0}^{k}(-1)^j a_j f\left(\cos\frac{j\pi}{k}\right)\right|, \tag{4.5.13}$$

where $a_j = \frac{1}{2}$, $j = 0, k$, and $a_j = 1$ otherwise. Inequalities of this type can reveal when one had better use *piecewise polynomial approximation* of a function on an interval instead of using a single polynomial over the whole interval. See also Sec. 4.4.

Denote by $C[a, b]$ the space of continuous functions on a closed bounded interval $[a, b]$. We shall study the approximation of a continuous function $f \in C[a, b]$ by polynomials. We introduce a metric by

$$\|f\| = \max_{x\in[a,b]} |f(x)|.$$

We define the **modulus of continuity** of $f$ by

$$\omega(\delta; f) = \sup_{|x-y|\le\delta} |f(x) - f(y)|. \tag{4.5.14}$$

Then $\omega(\delta; f) \downarrow 0$ as $\delta \downarrow 0$, because a continuous function is uniformly continuous on a closed bounded interval.

One of the fundamental theorems in approximation theory is Weierstrass'[149] approximation theorem from 1885. It is concerned with the uniform approximation of a continuous function $f$ on a closed bounded interval by polynomials. It is of basic importance for many convergence proofs, which we shall see examples of in this book.

**Theorem 4.5.7** (*Weierstrass' Approximation Theorem*).

*Suppose that $f$ is continuous on a closed, bounded interval $[a, b]$. Let $\mathcal{P}_n$ denote the space of polynomials of degree less than n. Then we can, for any $\epsilon > 0$, find a polynomial $p_n \in \mathcal{P}_n$ such that*

$$\max_{x\in[a,b]} |f(x) - p_n(x)| < \epsilon.$$

*Two equivalent formulations are*

- dist $(f, \mathcal{P}_n) \downarrow 0$ *as* $n \to \infty$;
- *the set of polynomials are dense in* $C[a, b]$.

***Proof.*** Our proof is based on ideas of Lebesgue [241]. For simplicity we assume that the interval is $[0, 1]$. First we interpolate $f(x)$ by a piecewise affine functions on the grid $x_k = k/r$, $k = 0 : r$, i.e., by a linear spline; see Sec. 4.4.2. A basis for the space of linear splines is given by

$$\{1, (x - x_0)_+, (x - x_1)_+, \ldots, (x - x_{r-1})_+\};$$

see (4.4.11), where we recall the notation $(x - a)_+ = \max(x - a, 0)$. Hence we can write the interpolating linear spline in the form

$$g(x) = f(0) + \sum_{k=0}^{r-1} c_k (x - x_k)_+. \tag{4.5.15}$$

Set $M_k = \max(f(x_k), f(x_{k+1}))$ and $m_k = \min(f(x_k), f(x_{k+1}))$. For $x \in [x_k, x_{k+1}]$ we have

$$M_k - \omega(1/r) \le f(x) \le m_k + \omega(1/r), \qquad -M_k \le -g(x) \le -m_k,$$

where $\omega$ is the modulus of continuity of $f$. Adding the last two inequalities we obtain

$$-\omega(1/r) \le f(x) - g(x) \le \omega(1/r).$$

[149] Kurt Theodor Wilhelm Weierstrass (1815–1897), German mathematician, whose lectures at Berlin University attracted students from all over the world. He set high standards of rigor in his work and is considered the father of modern analysis. For a more detailed biography and a survey of Weierstrass' work in approximation theory, see Pinkus [291].

This holds for $x \in [x_k, x_{k+1}]$, $k = 0 : r-1$, hence $\|f - g\| \le \omega(1/r)$. We now *choose* $r$ so that $\omega(1/r) < \epsilon/2$ so that $\|f - l\| < \epsilon/2$. Now that $r$ is fixed we see that the coefficients are bounded, (say) $|c_k| \le L$. (In fact we may choose any $L \ge r\epsilon$.)

If we can find any polynomial $p$ such that $\|p - g\| < \epsilon/2$, we are through. In order to achieve this we look at each term in the sum (4.5.15), and set $t = x - x_k$. We shall next find a polynomial $P$ such that

$$|P(t) - t_+| < \frac{\epsilon}{2rL}, \quad t \in [-1, 1]. \tag{4.5.16}$$

Note that

$$t_+ = \frac{1}{2}(t + |t|) = \frac{1}{2}\big(t + \sqrt{1 - (1 - t^2)}\big), \quad t \in [-1, 1].$$

Set $z = 1 - t^2$. We know that the binomial expansion of $\sqrt{1 - z}$ is valid for $|z| < 1$. By a classical theorem of Abel, see, e.g., Titchmarsh [351, Sec. 1.22], the sequence of partial sums also converges uniformly to $\sqrt{1 - z}$ for $z \in [0, 1]$. After return to the variable $t$, we thus have a sequence of polynomials that converge uniformly to $|t|$, $t+ \in [-1, 1]$, and this trivially yields a polynomial $p(t)$ that satisfies (4.5.16).

If we now set

$$P(x) = f(0) + \sum_{k=0}^{r-1} c_k p(x - x_k),$$

then

$$|P(x) - g(x)| = \Big| \sum_{k=0}^{r-1} l_k (p(x - x_k) - (x - x_k)_+) \Big| < 2\frac{\epsilon}{2} = \epsilon.$$

This finishes our proof. □

Several other proofs have been given, e.g., one by S. Bernštein, using Bernštein polynomials; see Davis [92, Sec. 6.2].

The smoother $f$ is, the quicker dist$(f, P_n)$ decreases, and the narrower the interval is, the less dist$(f, P_n)$ becomes. In many cases dist$(f, P_n)$ decreases so slowly toward zero (as $n$ grows) that it is impractical to attempt to approximate $f$ with only one polynomial in the entire interval $[a, b]$.

In infinite-dimensional spaces, certain operators may not be defined everywhere, but only in a set that is everywhere dense in the space. For example, in the space $C[a, b]$ of continuous functions on a bounded interval (with the maximum norm), the operator $A = d/dx$ is not defined everywhere, since there are continuous functions which are not differentiable. By Weierstrass' approximation theorem the set of polynomials is everywhere dense in $C$, and hence the set of differentiable functions is so too. Moreover, even if $Au$ exists, $A^2u$ may not exist. That $A^{-1}$ may not exist is no novel feature of infinite-dimensional spaces. In $C[a, b]$ the norm of $A = d/dx$ is infinite. This operator is said to be unbounded.

### 4.5.3 Inner Product Spaces and Orthogonal Systems

An abstract foundation for least squares approximation is furnished by the theory of **inner product spaces** which we now introduce.

**Definition 4.5.8.**

*A normed linear space $\mathcal{S}$ will be called an inner product space if for each two elements $f, g$ in $\mathcal{S}$ there is a scalar designated by $(f, g)$ with the following properties:*

| | | |
|---|---|---|
| 1. | $(f+g, h) = (f, h) + (g, h)$ | *(linearity),* |
| 2. | $(f, g) = \overline{(g, f)}$ | *(symmetry),* |
| 3. | $(f, \alpha g) = \alpha(f, g), \quad \alpha$ *scalar* | *(homogeneity),* |
| 4. | $(f, f) \geq 0, \qquad (f, f) = 0$ | *(positivity).* |

*The* **inner product** $(f, g)$ *is scalar, i.e., real in a real space and complex in a complex space. We set* $\|f\| = (f, f)^{1/2}$. *If* $\|f\| = 0$ *implies that* $f = 0$ *this is a* **norm on** $\mathcal{S}$*; otherwise it is a* **seminorm** *on* $\mathcal{S}$.

We shall show below that the triangle inequality is satisfied. (The other axioms for a norm are obvious.) The standard vector inner products introduced in Sec. A.1.2 in Online Appendix A are particular cases, if we set $(x, y) = y^T x$ in $\mathbf{R}^n$ and $(x, y) = y^H x$ in $\mathbf{C}^n$. A **complete** inner-product space is called a **Hilbert space** and is often denoted $\mathcal{H}$ in this book.

One can make computations using the more general definition of $(f, g)$ given above in the same way that one does with scalar products in linear algebra. Note, however, the *conjugations* necessary in a complex space,

$$(\alpha f, g) = \bar{\alpha}(f, g), \tag{4.5.17}$$

because, by the axioms,

$$(\alpha f, g) = \overline{(g, \alpha f)} = \overline{\alpha(g, f)} = \bar{\alpha}\overline{(g, f)} = \bar{\alpha}(f, g).$$

By the axioms it follows by induction that

$$\left(\phi_k, \sum_{j=0}^{n} c_j \phi_j\right) = \sum_{j=0}^{n} (\phi_k, c_j \phi_j) = \sum_{j=0}^{n} c_j (\phi_k, \phi_j). \tag{4.5.18}$$

**Theorem 4.5.9** (*Cauchy–Schwarz inequality*).

*The Cauchy–Schwarz inequality in a complex space is*

$$|(f, g)| \leq \|f\| \, \|g\|.$$

***Proof.*** Let $f$, $g$ be two arbitrary elements in an inner-product space. Then,[150] for every *real* number $\lambda$,

$$0 \leq (f + \lambda(f, g)g, \, f + \lambda(f, g)g) = (f, f) + 2\lambda|(f, g)|^2 + \lambda^2 |(f, g)|^2 (g, g).$$

This polynomial in $\lambda$ with real coefficients cannot have two distinct zeros, hence the discriminant cannot be positive, i.e.,

$$|(f, g)|^4 - (f, f)|(f, g)|^2 (g, g) \leq 0.$$

So, even if $(f, g) = 0$, $|(f, g)|^2 \leq (f, f)(g, g)$. □

[150] We found this proof in [304, sec. 83]. The application of the same idea in a *real* space can be made simpler.

By Definition 4.5.8 and the Cauchy–Schwarz inequality,

$$\begin{aligned}\|f+g\|^2 &= (f+g,\, f+g) = (f,\, f) + (f,\, g) + (g,\, f) + (g,\, g)\\ &\le \|f\|^2 + \|f\|\,\|g\| + \|g\|\,\|f\| + \|g\|^2 = (\|f\| + \|g\|)^2.\end{aligned}$$

This shows that the **triangle inequality** is satisfied by the norm defined above.

**Example 4.5.9.**

The set of all complex infinite sequences $\{x_i\}$ for which $\sum_{i=1}^{\infty} |x_i|^2 < \infty$ and which is equipped with the inner product

$$(x,\, y) = \sum_{i=1}^{\infty} x_i \overline{y}_i$$

constitutes a Hilbert space.

**Definition 4.5.10.**

*Two functions $f$ and $g$ are said to be* **orthogonal** *if $(f, g) = 0$. A finite or infinite sequence of functions $\phi_0, \phi_1, \ldots, \phi_n$ constitutes an* **orthogonal system** *if*

$$(\phi_i, \phi_j) = 0, \quad i \ne j, \text{ and } \|\phi_i\| \ne 0 \quad \forall i. \tag{4.5.19}$$

*If, in addition, $\|\phi_i\| = 1$, for all $i \ge 0$, then the sequence is called an* **orthonormal system**.

**Theorem 4.5.11** (*Pythagoras' Theorem*).

*Let $\{\phi_1, \phi_2, \ldots, \phi_n\}$ be an orthogonal system in an inner-product space. Then*

$$\Big\| \sum_{j=0}^{n} c_j \phi_j \Big\|^2 = \sum_{j=0}^{n} |c_j|^2 \|\phi_j\|^2.$$

*The elements of an orthogonal system are linearly independent.*

***Proof.*** We start as in the proof of the triangle inequality:

$$\|f+g\|^2 = (f,\, f) + (f,\, g) + (g,\, f) + (g,\, g) = (f,\, f) + (g,\, g) = \|f\|^2 + \|g\|^2.$$

Using this result and induction the first statement follows. The second statement then follows because $\sum c_j \phi_j = 0 \Rightarrow |c_j| = 0$ for all $j$. □

**Theorem 4.5.12.**

*A linear operator $P$ is called* **idempotent** *if $P = P^2$. Let $\mathcal{V}$ be the range of $P$. Then $P$ is a* **projection** *(or projector) onto $\mathcal{V}$ if and only if $P$ is idempotent and $Pv = v$ for each $v \in \mathcal{V}$.*

***Proof.*** If $P$ is a projection, then $v = Px$ for some $x \in \mathcal{B}$, hence $Pv = P^2x = Px = v$. Conversely, if $Q$ is a linear operator such that $Qx \in \mathcal{V}$ for all $x \in \mathcal{B}$, and $v = Qv$ for all $v \in V$, then $Q$ is a projection; in fact $Q = P$. □

Note that $I - P$ is also a projection, because

$$(I - P)(I - P) = I - 2P + P^2 = I - P.$$

Any vector $x \in \mathcal{B}$ can be written *uniquely* in the form

$$x = u + w, \quad u = Px, \quad w = (I - P)x. \tag{4.5.20}$$

Important examples of projections in function spaces are interpolation operators, for example, the mapping of $C[a, b]$ into $\mathcal{P}_k$ by Newton or Lagrange interpolation, because each polynomial is mapped to itself. The two types of interpolation are the same projection, although they use different bases in $\mathcal{P}_k$. Another example is the mapping of a linear space of functions, *analytic* on the unit circle, into $\mathcal{P}_k$ so that each function is mapped to its *Maclaurin expansion truncated to* $\mathcal{P}_k$. There are analogous projections where periodic functions and trigonometric polynomials are involved

In an inner-product space, the **adjoint operator** $A^*$ of a linear operator $A$ is defined by the requirement that

$$(A^*u, v) = (u, Av) \quad \forall\, u, v. \tag{4.5.21}$$

An operator $A$ is called **self-adjoint** if $A = A^*$. In $\mathbf{R}^n$, we define $(u, v) = u^T v$, i.e., the standard scalar product. Then $(A^*u)^T v = u^T Av$, i.e., $u^T((A^*)^T v = u^T Av$, hence $A^* = A^T$. It follows that symmetric matrices are self-adjoint in $\mathbf{R}^n$.

In $\mathbf{C}^n$, with the inner product $(u, v) = u^H v$, it follows that $A^* = A^H$, i.e., Hermitian matrices are self-adjoint. An operator $B$ is **positive definite** if $(u, Bu) > 0$ for all $u \neq 0$.

**Example 4.5.10.**

An important example of an orthogonal system is the sequence of trigonometric functions $\phi_j(x) = \cos jx$, $j = 0 : N - 1$. These form an orthogonal system, with either of the two inner products

$$(f, g) = \begin{cases} \displaystyle\int_0^\pi f(x)g(x)\,dx & \text{(continuous case)}, \\ \displaystyle\sum_{i=0}^{N-1} f(x_i)g(x_i), \quad x_i = \frac{2i+1}{N}\frac{\pi}{2} & \text{(discrete case)}. \end{cases} \tag{4.5.22}$$

Moreover, it holds that

$$\|\phi_j\|^2 = \begin{cases} \dfrac{1}{2}\pi, \quad j \geq 1, \quad \|\phi_0\|^2 = \pi & \text{(continuous case)}, \\ \dfrac{1}{2}N, \quad j = 1 : N - 1, \quad \|\phi_0\|^2 = N & \text{(discrete case)}. \end{cases}$$

These results are closely related to the orthogonality of the Chebyshev polynomials; see Theorem 4.5.20. Trigonometric interpolation and Fourier analysis will be treated in Sec. 4.6.

There are many other examples of orthogonal systems. Orthogonal systems of polynomials play an important role in approximation and numerical integration. Orthogonal systems also occur in a natural way in connection with eigenvalue problems for differential equations, which are quite common in mathematical physics.

### 4.5.4 Solution of the Approximation Problem

Orthogonal systems give rise to extraordinary formal simplifications in many situations. We now consider the least squares approximation problem.

**Theorem 4.5.13.**

*If $\phi_0, \phi_1, \ldots, \phi_n$ are linearly independent, then the least squares approximation problem of minimizing the norm of the error function $\|f^* - f\|$ over all functions $f^* = \sum_{j=0}^{n} c_j \phi_j$ has the unique solution*

$$f^* = \sum_{j=0}^{n} c_j^* \phi_j. \tag{4.5.23}$$

*The solution is characterized by the orthogonality property that $f^* - f$ is orthogonal to all $\phi_j$, $j = 0 : n$. The coefficients $c_j^*$ are called* **orthogonal coefficients** *(or* **Fourier coefficients***), and satisfy the linear system of equations*

$$\sum_{j=0}^{n} (\phi_j, \phi_k) c_j^* = (f, \phi_k), \tag{4.5.24}$$

*called normal equations. In the important special case when $\phi_0, \phi_1, \ldots, \phi_n$ form an orthogonal system, the coefficients are computed more simply by*

$$c_j^* = (f, \phi_j)/(\phi_j, \phi_j), \quad j = 0 : n. \tag{4.5.25}$$

***Proof.*** Let $(c_0, \ldots, c_n)$ be a sequence of coefficients with $c_j \neq c_j^*$ for at least one $j$. Then

$$\sum_{j=0}^{n} c_j \phi_j - f = \sum_{j=0}^{n} (c_j - c_j^*)\phi_j + (f^* - f).$$

If $f^* - f$ is orthogonal to all the $\phi_j$, then it is also orthogonal to the linear combination $\sum_{j=0}^{n} (c_j - c_j^*)\phi_j$ and, according to the Pythagorean theorem,

$$\left\| \sum_{j=0}^{n} c_j \phi_j - f \right\|^2 = \left\| \sum_{j=0}^{n} (c_j - c_j^*)\phi_j \right\|^2 + \|(f^* - f)\|^2 > \|(f^* - f)\|^2.$$

Thus if $f^* - f$ is orthogonal to all $\phi_k$, then $f^*$ is a solution to the approximation problem. It remains to show that the orthogonality conditions

$$\left( \sum_{j=0}^{n} c_j^* \phi_j - f, \phi_k \right) = 0, \quad k = 0 : n,$$

can be fulfilled. The above conditions are equivalent to the normal equations in (4.5.24). If $\{\phi_j\}_{j=0}^{n}$ constitutes an orthogonal system, then the system can be solved immediately, since in each equation all the terms with $j \neq k$ are zero. The formula in (4.5.25) then follows immediately.

Suppose now that we know only that $\{\phi_j\}_{j=0}^n$ are linearly independent. The solution to the normal equations exists and is unique, unless the homogeneous system,

$$\sum_{j=0}^{n}(\phi_j, \phi_k)c_j^* = 0, \quad k = 0 : n,$$

has a solution $c_0, c_1, \ldots, c_n$ with at least one $c_i \neq 0$. But this would imply

$$\Big\| \sum_{j=0}^{n} c_j\phi_j \Big\|^2 = \Big( \sum_{j=0}^{n} c_j\phi_j, \sum_{k=0}^{n} c_k\phi_k \Big) = \sum_{k=0}^{n}\sum_{j=0}^{n}(\phi_j, \phi_k)c_jc_k = \sum_{k=0}^{n} 0 \cdot c_k = 0,$$

which contradicts that the $\phi_j$ were linearly independent. □

In the case where $\{\phi_j\}_{j=0}^n$ form an orthogonal system, *the Fourier coefficients $c_j^*$ are independent of $n$* (see formula (4.5.25)). This has the important advantage that one can increase the total number of parameters without recalculating any previous ones. Orthogonal systems are advantageous not only because they greatly simplify calculations; using them, one can often avoid numerical difficulties with roundoff error which may occur when one solves the normal equations for a nonorthogonal set of basis functions.

With every continuous function $f$ one can associate an infinite series,

$$f \sim \sum_{j=0}^{\infty} c_j^*\phi_j, \quad c_j^* = \frac{(f, \phi_j)}{(\phi_j, \phi_j)}.$$

Such a series is called an **orthogonal expansion**. For certain orthogonal systems this series converges with very mild restrictions on the function $f$.

**Theorem 4.5.14.**

*If $f^*$ is defined by formulas* (4.5.23) *and* (4.5.25)*, then*

$$\|f^* - f\|^2 = \|f\|^2 - \|f^*\|^2 = \|f\|^2 - \sum_{j=0}^{n}(c_j^*)^2\|\phi_j\|^2.$$

***Proof.*** Since $f^* - f$ is, according to Theorem 4.5.13, orthogonal to all $\phi_j$, $0 \leq j \leq n$, then $f^* - f$ is orthogonal to $f^*$. The theorem then follows directly from Pythagoras' theorem. □

We obtain as corollary **Bessel's inequality**:

$$\sum_{j=0}^{n}(c_j^*)^2\|\phi_j\|^2 \leq \|f\|^2. \tag{4.5.26}$$

The series $\sum_{j=0}^{\infty}(c_j^*)^2\|\phi_j\|^2$ is convergent. If $\|f^* - f\| \to 0$ as $n \to \infty$, then the sum of the latter series is equal to $\|f\|^2$, which is **Parseval's identity**.

**Theorem 4.5.15.**

*If $\{\phi_j\}_{j=0}^m$ are linearly independent on the grid $G = \{x_i\}_{i=0}^m$, then the interpolation problem of determining the coefficients $\{c_j\}_{j=0}^m$ such that*

$$\sum_{j=0}^{m} c_j \phi_j(x_i) = f(x_i), \quad i = 0 : m, \tag{4.5.27}$$

*has exactly one solution. Interpolation is a special case ($n = m$) of the method of least squares. If $\{\phi_j\}_{j=0}^m$ is an orthogonal system, then the coefficients $c_j$ are equal to the orthogonal coefficients in* (4.5.25).

***Proof.*** The system of equations (4.5.27) has a unique solution, since its column vectors are the vectors $\phi_j(G)$, $j = 0 : n$, which are linearly independent. For the solution of the interpolation problem it holds that $\|c_j\phi_j - f\| = 0$; i.e., the error function has the least possible seminorm. The remainder of the theorem follows from Theorem 4.5.13. □

The following collection of important and equivalent properties is named the **fundamental theorem of orthonormal expansions** by Davis [92, Theorem 8.9.1], whom we follow closely at this point.

**Theorem 4.5.16.**

*Let $\phi_0, \phi_1, \phi_2, \cdots$ be a sequence of orthonormal elements in a complete inner product space $\mathcal{H}$. The following six statements are equivalent:*[151]

(A) *The $\phi_j$ is a complete orthonormal system in $\mathcal{H}$.*

(B) *The orthonormal expansion of any element $y \in \mathcal{H}$ converges in norm to $y$; i.e.,*

$$\lim_{n\to\infty} \Big\| y - \sum_{j=0}^{n} (y, \phi_j)\phi_j \Big\|. \tag{4.5.28}$$

(C) *Parseval's identity holds for every $y \in \mathcal{H}$, i.e.,*

$$\|y\|^2 = \sum_{j=0}^{\infty} |(y, \phi_j)|^2. \tag{4.5.29}$$

(D) *There is no strictly larger orthonormal system containing $\phi_1, \phi_2, \ldots$.*

(E) *If $y \in \mathcal{H}$ and $(y, \phi_j) = 0$, $j = 0, 2, \ldots$, then $y = 0$.*

(F) *An element of $\mathcal{H}$ is determined uniquely by its Fourier coefficients, i.e., if $(w, \phi_j) = (y, \phi_j)$, $j = 0, 2, \ldots$, then $w = y$.*

[151] We assume that $\mathcal{H}$ is not finite-dimensional, in order to simplify the formulations. Only minor changes are needed in order to cover the finite-dimensional case.

***Proof.*** The proof that $A \Leftrightarrow B$ is formulated with respect to the previous statements. By the conjugations necessary in the handling of complex scalars in inner products (see (4.5.17) and (4.5.18)),

$$\left(x - \sum_{j=0}^{\infty}(x, \phi_j)\phi_j,\ y - \sum_{j=0}^{n}(y, \phi_j)\phi_j\right) = (x, y) - \sum_{j=0}^{n}(x, \phi_j)(\phi_j, y).$$

By the Schwarz inequality,

$$\left|(x, y) - \sum_{j=0}^{n}(x, \phi_j)(\phi_j, y)\right| \le \left\|x - \sum_{j=0}^{\infty}(x, \phi_j)\phi_j\right\| \cdot \left\|y - \sum_{j=0}^{n}(y, \phi_j)\phi_j\right\|.$$

For the rest of the proof, see Davis [92, pp. 192 ff.]. □

**Theorem 4.5.17.**

*The converse of statement* (F) *holds, i.e., let $\mathcal{H}$ be a complete inner product space, and let $a_j$ be constants such that $\sum_{j=0}^{\infty}|a_j|^2 < \infty$. Then there exists an $y \in \mathcal{H}$ such that $y = \sum_{j=0}^{\infty} a_j\phi_j$ and $(y, \phi_j) = a_j$ for all $j$.*

***Proof.*** See Davis [92, Sec. 8.9]. □

## 4.5.5 Mathematical Properties of Orthogonal Polynomials

By a family of **orthogonal polynomials** we mean a triangle family of polynomials (see (4.1.8)), which (in the continuous case) is an orthogonal system with respect to a given inner product. The theory of orthogonal polynomials is also of fundamental importance for many problems which at first sight seem to have little connection with approximation (e.g., numerical integration, continued fractions, and the algebraic eigenvalue problem).

We assume in the following that in the continuous case the inner product is

$$(f, g) = \int_a^b f(x)g(x)w(x)\,dx, \quad w(x) \ge 0, \tag{4.5.30}$$

where $-\infty \le a < b \le \infty$. We assume that the weight function $w(x) \ge 0$ is such that the **moments**

$$\mu_k = (x^k, 1) = \int_a^b x^k w(x)\,dx \tag{4.5.31}$$

are defined for all $k \ge 0$, and $\mu_0 > 0$. In the discrete case, we define the weighted discrete inner product of two real-valued functions $f$ and $g$ on the grid $\{x_i\}_{j=0}^{m}$ of distinct points by

$$(f, g) = \sum_{i=0}^{m} w_i f(x_i)g(x_i), \quad w_i > 0. \tag{4.5.32}$$

Note that both these inner products have the property that

$$(xf, g) = (f, xg). \tag{4.5.33}$$

The continuous and discrete case are both special cases of the more general inner product

$$(f, g) = \int_a^b f(x)g(x)\, d\alpha(x), \tag{4.5.34}$$

where the integral is a Stieltjes integral (see Definition 3.4.4) and $\alpha(x)$ is allowed to be discontinuous. However, in the interest of clarity, we will in the following treat the two cases separately.

The weight function $w(x)$ determines the orthogonal polynomials $\phi_n(x)$ up to a constant factor in each polynomial. The specification of those factors is referred to as **standardization**. These polynomials satisfy a number of relationships of the same general form. In the case of a continuously differentiable weight function $w(x)$ we have an explicit expression

$$\phi_n(x) = \frac{1}{a_n w(x)} \frac{d^n}{dx^n}\{w(x)(g(x))^n\}, \quad n = 0, 1, 2, \ldots, \tag{4.5.35}$$

where $g(x)$ is a polynomial in $x$ independent of $n$. This is **Rodrigues' formula**. The orthogonal polynomials also satisfy a second order differential equation,

$$g_2(x)\phi_n'' + g_1(x)\phi_n' + a_n\phi_n = 0, \tag{4.5.36}$$

where $g_2(x)$ and $g_1(x)$ are independent of $n$ and $a_n$ is a constant only dependent on $n$.

Let $p_n(x) = k_n x^n + \cdots, n = 0, 1, 2, \ldots$, be a family of real orthogonal polynomials. The symmetric function

$$K_n(x, y) = \sum_{k=0}^{n} p_k(x)\, p_k(y) \tag{4.5.37}$$

is called the **kernel polynomial** of order $n$ for the orthogonal system. It can be shown that the kernel polynomial has the reproducing property that for every polynomial $p$ of degree at most $n$

$$(p(x), K_n(x, y)_x) = p(y). \tag{4.5.38}$$

Here the subscript $x$ indicates that the inner product is taken with respect to $x$. Conversely, if $K(x, y)$ is a polynomial of degree at most $n$ in $x$ and $y$ and if $(p(x), K(x, y)_x) = p(y)$, for all polynomials $p$ of degree at most $n$, then $K(x, y) = K_n(x, y)$.

An alternative expression, the **Christoffel–Darboux formula**, can be given for the kernel polynomial.

**Theorem 4.5.18.**

*Let $p_n(x) = k_n x^n + \cdots, n = 0, 1, 2, \ldots$, be real orthonormal polynomials. Then*

$$K_n(x, y) = \frac{k_n}{k_{n+1}} \frac{p_{n+1}(x)p_n(y) - p_n(x)p_{n+1}(y)}{x - y}. \tag{4.5.39}$$

***Proof.*** See Davis [92, Theorem 10.1.6]. □

Given a linearly independent sequence of vectors, an orthogonal system can be derived by a process analogous to Gram–Schmidt orthogonalization.

**Theorem 4.5.19.**

*For every weight function in an inner product space there is a triangle family of orthogonal polynomials $\phi_k(x)$, $k = 0, 1, 2, \ldots$, such that $\phi_k(x)$ has exact degree $k$, and is orthogonal to all polynomials of degree less than $k$. The family is uniquely determined apart from the fact that the leading coefficients can be given arbitrary positive values.*

*The monic orthogonal polynomials satisfy the three-term recurrence formula,*

$$\phi_{k+1}(x) = (x - \beta_k)\phi_k(x) - \gamma_{k-1}^2\phi_{k-1}(x), \quad k \geq 1, \tag{4.5.40}$$

*with initial values $\phi_{-1}(x) = 0$, $\phi_0(x) = 1$. The recurrence coefficients are given by Darboux's formulas*

$$\beta_k = \frac{(x\phi_k, \phi_k)}{\|\phi_k\|^2}, \qquad \gamma_{k-1}^2 = \frac{\|\phi_k\|^2}{\|\phi_{k-1}\|^2}. \tag{4.5.41}$$

***Proof.*** The proof is by induction. We have $\phi_{-1} = 0$, $\phi_0 = 1$. Suppose that $\phi_j \neq 0$ have been constructed for $0 \leq j \leq k$, $k \geq 0$. We now seek a polynomial $\phi_{k+1}$ of degree $k + 1$ with leading coefficient equal to 1 which is orthogonal to all polynomials of degree $\leq k$. Since $\{\phi_j\}_{j=0}^k$ is a triangle family, every polynomial of degree $k$ can be expressed as a linear combination of these polynomials. Therefore, we can write

$$\phi_{k+1} = x\phi_k - \sum_{i=0}^{k} c_{k,i}\phi_i, \tag{4.5.42}$$

where $\phi_{k+1}$ has leading coefficient one. The orthogonality condition is fulfilled if and only if

$$(x\phi_k, \phi_j) - \sum_{i=0}^{k} c_{k,i}(\phi_i, \phi_j) = 0, \quad j = 0 : k.$$

But $(\phi_i, \phi_j) = 0$ for $i \neq j$, and thus $c_{k,j}\|\phi_j\|^2 = (x\phi_k, \phi_j)$. This determines the coefficients uniquely. From the definition of inner product (4.5.30), it follows that

$$(x\phi_k, \phi_j) = (\phi_k, x\phi_j).$$

But $x\phi_j$ is a polynomial of degree $j + 1$. Thus if $j < k$, then it is orthogonal to $\phi_k$. So $c_{kj} = 0$ for $j < k - 1$. From (4.5.42) it then follows that

$$\phi_{k+1} = x\phi_k - c_{k,k}\phi_k - c_{k,k-1}\phi_{k-1}, \tag{4.5.43}$$

with $c_{k,k-1} = 0$ if $k = 0$. This has the same form as the original assertion of the theorem if we set

$$\begin{aligned} \beta_k &= c_{k,k} = \frac{(x\phi_k, \phi_k)}{\|\phi_k\|^2}, \\ \gamma_{k-1}^2 &= c_{k,k-1} = \frac{(\phi_k, x\phi_{k-1})}{\|\phi_{k-1}\|^2}, \quad k \geq 1. \end{aligned} \tag{4.5.44}$$

In the discrete case the division in (4.5.44) can always be performed, as long as $k \leq m$. In the continuous case, no reservation need be made.

The expression for $\gamma_{k-1}^2$ can be written in another way. If we take the inner product of (4.5.42) and $\phi_{k+1}$ we get

$$(\phi_{k+1}, \phi_{k+1}) = (\phi_{k+1}, x\phi_k) - \sum_{i=0}^{k} c_{k,i}(\phi_{k+1}, \phi_i) = (\phi_{k+1}, x\phi_k).$$

Thus $(\phi_{k+1}, x\phi_k) = \|\phi_{k+1}\|^2$, or if we decrease all indices by one, $(\phi_k, x\phi_{k-1}) = \|\phi_k\|^2$. Substituting this in the expression for $\gamma_{k-1}^2$ gives the second equation of (4.5.41). □

If the weight distribution $w(x)$ is symmetric about $\beta$, i.e., (in the continuous case) $w(\beta - x) = w(x + \beta)$, then $\beta_k = \beta$ for all $k \geq 0$. Further,

$$\phi_k(\beta - x) = (-1)^k \phi_k(x + \beta), \quad k \geq 0; \tag{4.5.45}$$

that is, $\phi_k$ is symmetric about $\beta$ for $k$ even and antisymmetric for $k$ odd. The proof is by induction. We have $\phi_0 = 1$ and $\phi_1(x) = x - \beta_0$. Clearly $(\phi_1, \phi_0) = 0$ implies that $\phi_1$ is antisymmetric about $\beta$ and therefore $\beta_0 = \beta$. Thus the hypothesis is true for $k \leq 1$. Now assume that (4.5.45) holds for $k \leq n$. Then

$$\beta_n = \frac{(x\phi_n, \phi_n)}{\|\phi_n\|^2} = \frac{((x - \beta)\phi_n, \phi_n)}{\|\phi_n\|^2} + \beta.$$

Here the first term is zero since it is an integral of an antisymmetric function. It follows that

$$\phi_{n+1}(x) = (x - \beta)\phi_n(x) - \gamma_{n-1}^2 \phi_{n-1}(x),$$

which shows that (4.5.45) holds for $k = n + 1$. An analog result holds for a symmetric discrete inner product.

Often it is more convenient to consider corresponding **orthonormal polynomials** $\hat{\phi}_k(x)$, which satisfy $\|\hat{\phi}_k\| = 1$. We set $\hat{\phi}_0 = 1/\sqrt{\mu_0}$, $\mu_0 = \|\phi_0\|_2^2$, and scale the monic orthogonal polynomials according to

$$\phi_k = (\gamma_1 \cdots \gamma_{k-1})\hat{\phi}_k, \quad k \geq 1; \tag{4.5.46}$$

then we find using (4.5.41) that

$$\frac{\|\hat{\phi}_k\|}{\|\hat{\phi}_{k-1}\|} = \frac{\gamma_1 \cdots \gamma_{k-2}}{\gamma_1 \cdots \gamma_{k-1}} \frac{\|\phi_k\|}{\|\phi_{k-1}\|} = 1.$$

Substituting (4.5.46) in (4.5.40) we obtain the recurrence relation for the orthonormal polynomials

$$\gamma_k \hat{\phi}_{k+1}(x) = (x - \beta_k)\hat{\phi}_k(x) - \gamma_{k-1}\hat{\phi}_{k-1}(x), \quad k \geq 1, \tag{4.5.47}$$

where $\gamma_k$ is determined by the condition $\|\hat{\phi}_{k+1}\| = 1$.

Perhaps the most important example of a family of orthogonal polynomials is the Chebyshev polynomials $T_n(x) = \cos(n \arccos(x))$ introduced in Sec. 3.2.3. These are orthogonal on $[-1, 1]$ with respect to the weight function $(1 - x^2)^{-1/2}$ and also with respect to a discrete inner product. Their properties can be derived by rather simple methods.

**Theorem 4.5.20.**

*The Chebyshev polynomials have the following two orthogonality properties. Set*

$$(f, g) = \int_{-1}^{1} f(x)g(x)(1-x^2)^{-1/2}\,dx \tag{4.5.48}$$

*(the continuous case). Then* $(T_0, T_0) = \pi$, *and*

$$(T_j, T_k) = \begin{cases} 0 & \text{if } j \neq k, \\ \pi/2 & \text{if } j = k \neq 0. \end{cases} \tag{4.5.49}$$

*Let* $x_k$ *be the zeros of* $T_{m+1}(x)$ *and set*

$$(f, g) = \sum_{k=0}^{m} f(x_k)g(x_k), \quad x_k = \cos\Big(\frac{2k+1}{m+1}\frac{\pi}{2}\Big) \tag{4.5.50}$$

*(the discrete case). Then* $(T_0, T_0) = m + 1$, *and*

$$(T_j, T_k) = \begin{cases} 0 & \text{if } j \neq k, \\ (m+1)/2 & \text{if } j = k \neq 0. \end{cases} \tag{4.5.51}$$

***Proof.*** In the continuous case, let $j \neq k$, $j \geq 0$, $k \geq 0$. From $x = cos\phi$ it follows that $dx = \sin\phi d\phi = (1-x^2)^{1/2}d\phi$. Hence

$$\begin{aligned}(T_j, T_k) &= \int_0^{\pi} \cos jx \cos kx\,dx = \int_0^{\pi} \frac{1}{2}\big(\cos(j-k)x + \cos(j+k)x\big)\,dx \\ &= \frac{1}{2}\left(\frac{\sin(j-k)\pi}{j-k} + \frac{\sin(j+k)\pi}{j+k}\right) = 0,\end{aligned}$$

whereby orthogonality is proved.

In the discrete case, set $h = \pi/(m+1)$, $x_\mu = h/2 + \mu h$,

$$(T_j, T_k) = \sum_{\mu=0}^{m} \cos jx_\mu \cos kx_\mu = \frac{1}{2}\sum_{\mu=0}^{m}\big(\cos(j-k)x_\mu + \cos(j+k)x_\mu\big).$$

Using notation from complex numbers ($i = \sqrt{-1}$) we have

$$(T_j, T_k) = \frac{1}{2}\mathrm{Re}\Big(\sum_{\mu=0}^{m} e^{i(j-k)h(1/2+\mu)} + \sum_{\mu=0}^{m} e^{i(j+k)h(1/2+\mu)}\Big). \tag{4.5.52}$$

The sums in (4.5.52) are geometric series with ratios $e^{i(j-k)h}$ and $e^{i(j+k)h}$, respectively. If $j \neq k$, $0 \leq j \leq m$, $0 \leq k \leq m$, then the ratios are never equal to one, since

$$0 < |\,(j \pm k)h| \leq \frac{2m}{m+1}\pi < \pi.$$

Using the formula for the sum of a geometric series the first sum in (4.5.52) is

$$e^{i(j-k)(h/2)}\frac{e^{i(j-k)(m+1)h}-1}{e^{i(j-k)h}-1} = \frac{e^{i(j-k)\pi}-1}{e^{i(j-k)(h/2)}-e^{-i(j-k)(h/2)}-1} = \frac{(-1)^{j-k}-1}{2i\sin(j-k)h/2}.$$

The real part of the last expression is clearly zero. An analogous computation shows that the real part of the other sum in (4.5.52) is also zero. Thus the orthogonality property holds in the discrete case also. It is left to the reader to show that the expressions when $j = k$ given in the theorem are correct. □

For the uniform weight distribution $w(x) = 1$ on $[-1, 1]$ the relevant orthogonal polynomials are the **Legendre polynomials**[152] $P_n(x)$. The Legendre polynomials are defined by Rodrigues' formula

$$P_n(x) = (-1)^n \frac{1}{2^n n!}\frac{d^n}{dx^n}\{(1-x^2)^n\}, \quad n = 0, 1, 2, \ldots. \tag{4.5.53}$$

Since $(1 - x^2)^n$ is a polynomial of degree $2n$, $P_n(x)$ is a polynomial of degree $n$. The Legendre polynomials $P_n = A_n x^n + \cdots$ have leading coefficient and norm

$$A_n = \frac{(2n)!}{2^n (n!)^2}, \qquad \|P_n\| = \frac{2}{2n+1}. \tag{4.5.54}$$

This standardization corresponds to setting $P_n(1) = 1$ for all $n \geq 0$. The extreme values are

$$|P_n(x)| \leq 1, \quad x \in [-1, 1].$$

There seems to be no easy proof for the last result; see [193, p. 219].

Since the weight distribution is symmetric about the origin, these polynomials have the symmetry property

$$P_n(-x) = (-1)^n P_n(x).$$

The Legendre polynomials satisfy the three-term recurrence formula $P_0(x) = 1$, $P_1(x) = x$,

$$P_{n+1}(x) = \frac{2n+1}{n+1} x P_n(x) - \frac{n}{n+1} P_{n-1}(x), \quad n \geq 1. \tag{4.5.55}$$

The first few Legendre polynomials are

$$P_2(x) = \frac{1}{2}(3x^2 - 1), \qquad P_3(x) = \frac{1}{2}(5x^3 - 3x),$$
$$P_4(x) = \frac{1}{8}(35x^4 - 30x^2 + 3), \qquad P_5(x) = \frac{1}{8}(63x^5 - 70x^3 + 15), \ldots.$$

A graph of the Legendre polynomial $P_{21}(x)$ is shown in Figure 4.5.1.

[152] Legendre had obtained these polynomials in 1784–1789 in connection with his investigation concerning the attraction of spheroids and the shape of planets.

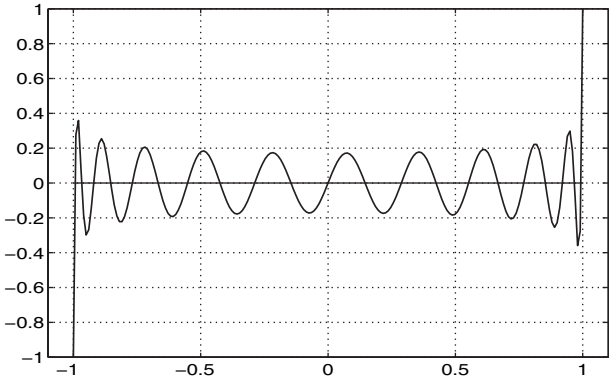



**Figure 4.5.1.** *The Legendre polynomial* $P_{21}$.

Often a more convenient standardization is to consider *monic* Legendre polynomials, with leading coefficient equal to one. These satisfy $P_0(x) = 1$, $P_1(x) = x$, and the recurrence formula

$$P_{n+1}(x) = xP_n(x) - \frac{n^2}{4n^2 - 1}P_{n-1}(x), \quad n \geq 1; \tag{4.5.56}$$

note that we have kept the same notation for the polynomials. It can be shown that

$$P_n = x^n + c_n x^{n-2} + \cdots, \quad c_n = -\frac{n(n-1)}{2(2n-1)}.$$

The **Jacobi polynomials**[153] $J_n(x; \alpha, \beta)$ arise from the weight function

$$w(x) = (1 - x)^\alpha (1 + x)^\beta, \quad x \in [-1, 1], \quad \alpha, \beta > -1.$$

They are special cases of Gauss' hypergeometric function $F(a, b, c : x)$,

$$F(-n, \alpha + 1 + \beta + n, \alpha + 1; x),$$

(see (3.1.16)). The Jacobi polynomials are usually standardized so that the coefficient $A_n$ of $x^n$ in $J_n(x; \alpha, \beta)$ is given by

$$A_n = \frac{1}{2^n n!}\frac{\Gamma(2n + \alpha + \beta + 1)}{\Gamma(n + \alpha + \beta + 1)}.$$

The Legendre polynomials are obtained as the special case when $\alpha = \beta = 0$. The case $\alpha = \beta = -1/2$, which corresponds to the weight function $w(x) = 1/\sqrt{1 - x^2}$, gives the Chebyshev polynomials.

[153] Carl Gustav Jacob Jacobi (1805–1851) was a German mathematician. Jacobi joined the faculty of Berlin University in 1825. Like Euler, he was a proficient calculator who drew a great deal of insight from immense algorithmic work.

The generalized **Laguerre polynomials** $L_n^{(\alpha)}(x)$ are orthogonal with respect to the weight function

$$w(x) = x^{\alpha}e^{-x}, \quad x \in [0, \infty], \quad \alpha > -1.$$

Setting $\alpha = 0$, we get the Laguerre polynomials $L_n^{(0)}(x) = L_n(x)$. Standardizing these so that $L_n(0) = 1$, they satisfy the three-term recurrence relation

$$(n+1)L_{n+1}(x) = (2n+1-x)L_n(x) - nL_{n-1}(x), \tag{4.5.57}$$

Rodrigues' formula becomes

$$L_n^{\alpha}(x) = \frac{e^x}{n!x^{(\alpha)}}\frac{d^n}{dx^n}(x^{n-\alpha}e^{-x}).$$

The **Hermite polynomials** are orthogonal with respect to the weight function

$$w(x) = e^{-x^2}, \quad -\infty < x < \infty.$$

With the classic standardization they satisfy the recurrence relation $H_0(x) = 1$, $H_1(x) = 2x$,

$$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x),$$

and

$$H_n(0) = \begin{cases} (-1)^m(2m)!/m! & \text{if } n = 2m, \\ 0 & \text{if } n = 2m+1. \end{cases}$$

The Hermite polynomials can also be defined by Rodrigues' formula:

$$H_n(x) = (-1)^n e^{x^2}\frac{d^n}{dx^n}\{e^{-x^2}\}.$$

It can be verified that these polynomials are identical to those defined by the recurrence relation.

The properties of some important families of orthogonal polynomials are summarized in Table 4.5.1. Note that here the coefficients in the three-term recurrence relation are given for the *monic orthogonal polynomials*; cf. (4.5.40).

For equidistant data, the **Gram polynomials** $\{P_{n,m}\}_{n=0}^m$ are of interest.[154] These polynomials are orthogonal with respect to the discrete inner product

$$(f, g) = (1/m)\sum_{i=1}^{m} f(x_i)g(x_i), \quad x_i = -1 + (2i-1)/m.$$

The weight distribution is symmetric around the origin $\alpha_k = 0$. For the *monic* Gram polynomials the recursion formula is (see [16])

$$P_{-1,m}(x) = 0, \qquad P_{0,m} = 1,$$
$$P_{n+1,m}(x) = xP_{n,m}(x) - \beta_{n,m}P_{n-1,m}(x), \quad n = 0 : m-1,$$

[154] Jørgen Pedersen Gram (1850–1916), a Danish mathematician, graduated from Copenhagen University and worked as a director for a life insurance company. He introduced the Gram determinant in connection with his study of linear independence, and his name is also associated with Gram–Schmidt orthogonalization.

**Table 4.5.1.** *Weight functions and recurrence coefficients for some classical monic orthogonal polynomials.*

| $[a,b]$ | $w(x)$ | Orthog. pol. | $\mu_0$ | $\beta_k$ | $\gamma_k^2$ |
|---|---|---|---|---|---|
| $[-1,1]$ | $1$ | $P_n(x)$ Legendre | $2$ | $0$ | $\dfrac{k^2}{4k^2-1}$ |
| $[-1,1]$ | $(1-x^2)^{-1/2}$ | $T_n(x)$ Cheb. 1st | $\pi$ | $0$ | $\frac{1}{2}$ $(k=1)$ |
| | | | | | $\frac{1}{4}$ $(k>1)$ |
| $[-1,1]$ | $(1-x^2)^{1/2}$ | $U_n(x)$ Cheb. 2nd | $\pi/2$ | $0$ | $\frac{1}{4}$ |
| $[-1,1]$ | $(1-x)^\alpha(1+x)^\beta$ | $J_n(x;\alpha,\beta)$ Jacobi | | | |
| $[0,\infty]$ | $x^\alpha e^{-x}$, $\alpha>-1$ | $L_n^{(\alpha)}(x)$ Laguerre | $\Gamma(1+\alpha)$ | $2k+\alpha+1$ | $k(k+\alpha)$ |
| $[-\infty,\infty]$ | $e^{-x^2}$ | $H_n(x)$ Hermite | $\sqrt{\pi}$ | $0$ | $\frac{1}{2}k$ |

where $(n<m)$ and

$$\beta_{n,m}=\frac{n^2}{4n^2-1}\left(1-\frac{n^2}{m^2}\right).$$

When $n \ll m^{1/2}$, these polynomials are well behaved. But when $n \geq m^{1/2}$, the Gram polynomials have very large oscillations between the grid points, and a large maximum norm in $[-1,1]$. This fact is related to the recommendation that when fitting a polynomial to *equidistant data*, one should never choose $n$ larger than about $2m^{1/2}$.

### Complex Orthogonal Polynomials

So far we have considered the inner products (4.5.30) defined by an integral over the real interval $[a,b]$. Now let $\Gamma$ be a rectifiable curve (i.e., a curve of finite length) in the complex plane. Consider the linear space of all polynomials with complex coefficients and $z=x+iy$ on $\Gamma$. Let $\alpha(s)$ be a function on $\Gamma$ with infinitely many points of increase and define an inner product by the line integral

$$(p,q)=\int_\Gamma p(z)\overline{q(z)}w(s)\,d(s). \tag{4.5.58}$$

The complex monomials $1, z, z^2, \ldots$ are independent functions, since if $a_0+a_1z+a_2z^2+\cdots+a_nz^n \equiv 0$ on $\Gamma$ it would follow from the fundamental theorem of algebra that $a_0=a_1=a_2=\cdots=a_n=0$. There is a unique infinite sequence of polynomials $\phi_j=z^j+c_1z^{j-1}+\cdots+c_j$, $j=0,1,2,\ldots$, which are orthonormal with respect to (4.5.58). They can be constructed by Gram–Schmidt orthogonalization as in the real case.

An important case is when $\Gamma$ is the unit circle in the complex plane. We then write

$$(p, q) = \frac{1}{2\pi} \int_{-\pi}^{\pi} p(z)\overline{q(z)}\, d\alpha(t), \quad z = e^{it}, \tag{4.5.59}$$

where the integral is to be interpreted as a Stieltjes integral. The corresponding orthogonal polynomials are known as **Szegő polynomials** and have applications, e.g., in signal processing.

Properties of Szegő polynomials are discussed in [174]. Together with the reverse polynomials $\tilde{\phi}_j(z) = z^j \phi_j(1/z)$ they satisfy special recurrence relations. A linear combination $\sum_{j=0}^{n} c_j \phi_j(z)$ can be evaluated by an analogue to the Clenshaw algorithm; see [5].

### 4.5.6 Expansions in Orthogonal Polynomials

Expansions of functions in terms of orthogonal polynomials are very useful. They are easy to manipulate, have good convergence properties in the continuous case, and usually give a well-conditioned representation.

Let $\hat{p}_n$ denote the polynomial of degree less than $n$ for which

$$\|f - \hat{p}_n\|_\infty = E_n(f) = \min_{p \in \mathcal{P}_n} \|f - p\|_\infty,$$

and set

$$p_n = \sum_{j=0}^{n-1} c_j \phi_j,$$

where $c_j$ is the $j$th Fourier coefficient of $f$ and $\{\phi_j\}$ are the orthogonal polynomials with respect to the inner product (4.5.30). If we use the weighted Euclidean norm, $\hat{p}_n$ is of course not a better approximation than $p_n$. In fact,

$$\begin{aligned} \|f - p_n\|_w^2 &= \int_a^b |f(x) - p_n(x)|^2 w(x)\, dx \\ &\le \int_a^b |f(x) - \hat{p}_n(x)|^2 w(x)\, dx \le E_n(f)^2 \int_a^b w(x)\, dx. \end{aligned} \tag{4.5.60}$$

This can be interpreted as saying that a kind of weighted mean of $|f(x) - p_n(x)|$ is less than or equal to $E_n(f)$, which is about as good a result as one could demand. The error curve has an oscillatory behavior. In small subintervals, $|f(x) - p_n(x)|$ can be significantly greater than $E_n(f)$. This is usually near the ends of the intervals or in subintervals where $w(x)$ is relatively small. Note that from (4.5.60) and Weierstrass' approximation theorem it follows that

$$\lim_{n \to \infty} \|f - p\|_{2,w}^2 = 0$$

for every continuous function $f$. From (4.5.60) one gets after some calculations

$$\sum_{j=n}^{\infty} c_j^2 \|\phi_j\|^2 = \|f - p_n\|_{2,w}^2 \le E_n(f)^2 \int_a^b w(x)\, dx,$$

which gives one an idea of how quickly the terms in the orthogonal expansion decrease.

**Example 4.5.11** (*Chebyshev Interpolation*).

Let $p(x)$ denote the interpolation polynomial in the Chebyshev points $x_k = \cos(\frac{2k+1}{m+1}\frac{\pi}{2})k = 0 : m$. For many reasons it is practical to write this interpolation polynomial in the form

$$p(x) = \sum_{i=0}^{m} c_i T_i(x). \tag{4.5.61}$$

Then using the discrete orthogonality property (4.5.51)

$$c_i = \frac{(f, T_i)}{\|T_i\|^2} = \frac{1}{\|T_i\|^2} \sum_{k=0}^{m} f(x_k) T_i(x_k) \tag{4.5.62}$$

where

$$\|T_0\|^2 = m + 1, \quad \|T_i\|^2 = \frac{1}{2}(m+1), \quad i > 0.$$

The recursion formula (3.2.20) can be used for calculating the orthogonal coefficients according to (4.5.62).

For computing $p(x)$ with (4.5.61), Clenshaw's algorithm [71] can be used. Clenshaw's algorithm holds for any sum of the form $S = \sum_{k=1}^{n} c_k \phi_k$, where $\{\phi_k\}$ satisfies *a three-term recurrence relation*. It can also be applied to series of Legendre functions, Bessel functions, Coulomb wave functions, etc., because they satisfy recurrence relations of this type, where the $\alpha_k$, $\gamma_k$ depend on $x$; see the Handbook [1] or any text on special functions.

**Theorem 4.5.21** (*Clenshaw's Algorithm*).

*Suppose that a sequence $\{p_k\}$ satisfies the three-term recurrence relation*

$$p_{k+1} = \gamma_k p_k - \beta_k p_{k-1}, \quad k = 0 : n-1, \tag{4.5.63}$$

*where $p_{-1} = 0$. Then*

$$S = \sum_{k=0}^{n} c_k p_k = y_0 p_0,$$

*where $y_{n+1} = 0$, $y_n = c_n$, and $y_0$ is obtained by the recursion*

$$y_k = c_k + \gamma_{k-1} y_{k+1} - \beta_k y_{k+2}, \quad k = n-1 : -1 : 0. \tag{4.5.64}$$

***Proof.*** Write the recursion (4.5.63) in matrix form as $L_n p = p_0 e_1$, where

$$L_n = \begin{pmatrix} 1 & & & & \\ -\gamma_0 & 1 & & & \\ \beta_1 & -\gamma_1 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{n-1} & -\gamma_{n-1} & 1 \end{pmatrix}$$

is unit lower triangular and $e_1$ is the first column of the unit matrix. Then

$$S = c^T p = c^T L_n^{-1} g = g^T (L_n^T)^{-1} c = g^T y,$$

where $y$ is the solution to the upper triangular system $L_n^T y = c$. Solving this by back-substitution we get the recursion (4.5.64). □

It can be proved that Clenshaw's algorithm is componentwise backward stable with respect to the data $\gamma_k$ and $\beta_k$; see Smoktunowicz [329].

Occasionally one is interested in the partial sums of (4.5.61). For example, if the values of $f(x)$ are afflicted with statistically independent errors with standard deviation $\sigma$, then the series can be broken off when for the first time

$$\left\| f - \sum_{i=0}^{p} c_i T_i(x) \right\| < \sigma m^{1/2}.$$

The proof of Theorem 4.5.19 is constructive and leads to a unique construction of the sequence of orthogonal polynomials $\phi_k$, $n \geq 1$, with leading coefficients equal to one. The coefficients $\beta_k$ and $\gamma_k$ and the polynomials $\phi_k$ can be computed in the order

$$\beta_0, \phi_1(x), \beta_1, \gamma_0, \phi_2(x), \ldots.$$

(Recall that $\phi_{-1} = 0$ and $\phi_0 = 1$.) This procedure is called the **Stieltjes procedure**. This assumes that the inner product $(x\phi_k, \phi_k)$ and norm $\|\phi_k\|_2$ can be computed. This clearly can be done for any discrete $n$-point weight.

For a continuous weight function $w(x)$ this is more difficult. One possible way to proceed is then to approximate the integrals using some appropriate quadrature rule. For a discussion of such methods and examples we refer to Gautschi [148].

There are many computational advantages of using the Stieltjes procedure when computing the discrete least squares approximation

$$f(x) \approx p_m(x) = \sum_{k=0}^{m} c_k \phi_k(x), \tag{4.5.65}$$

where $\phi_0(x), \ldots, \phi_m(x)$ are the orthogonal polynomials with respect to the discrete weight function (4.5.32). Recall the family ends with $\phi_m(x)$, since

$$\phi_{m+1}(x) = (x - x_0)(x - x_1) \cdots (x - x_m)$$

is zero at each grid point and $\|\phi_{m+1}\| = 0$. The construction cannot proceed, which is natural since there cannot be more than $m + 1$ orthogonal (or even linearly independent) functions on a grid with $m + 1$ points.

By Theorem 4.5.13 the coefficients are given by

$$c_k = (f, \phi_k)/\|\phi_k\|^2, \quad k = 0 : m. \tag{4.5.66}$$

Approximations of increasing degree can be recursively generated as follows. Suppose that $\phi_j$, $j = 0 : k$, and the least squares approximation $p_k$ of degree $k$, have been computed. In

the next step the coefficients $\beta_k$, $\gamma_{k-1}$, and $\phi_{k+1}$ are computed and the next approximation is obtained by

$$p_{k+1} = p_k + c_{k+1}\phi_{k+1}, \qquad c_{k+1} = (f, \phi_{k+1})/\|\phi_{k+1}\|^2. \tag{4.5.67}$$

To compute numerical values of $p(x)$ **Clenshaw's algorithm** is used; see Theorem 4.5.21.

For unit weights and a symmetric grid the Stieltjes procedure requires about $4mn$ flops to compute the orthogonal functions $\phi_k$ at the grid points. If there are differing weights, then about $mn$ additional operations are needed. Similarly, $mn$ additional operations are required if the grid is not symmetric. If the orthogonal coefficients are determined simultaneously for several functions on the same grid, then only about $mn$ additional operations per function are required. (In the above, we assume $m \gg 1, n \gg 1$.) Hence the procedure is much more economical than the general methods based on normal equations, which require $O(mn^2)$ flops.

Since $p_k$ is a linear combination of $\phi_j$, $j = 0 : k$, $\phi_{k+1}$ is orthogonal to $p_k$. Therefore, an alternative expression for the new coefficient is

$$c_{k+1} = (r_k, \phi_{k+1})/\|\phi_{k+1}\|^2, \quad r_k = f - p_k, \tag{4.5.68}$$

where $r_k$ is the residual. Mathematically the two formulas (4.5.67) and (4.5.68) for $c_{k+1}$ are equivalent. In finite precision, as higher-degree polynomials $p_{k+1}$ are computed, they will gradually lose orthogonality to previously computed $p_j$, $j \le k$. In practice there is an advantage in using (4.5.68) since cancellation will then take place mostly in computing the residual $r_k = f - p_k$, and the inner product $(r_k, p_{k+1})$ is computed more accurately. Theoretically the error $\|p_k - f\|$ must be a nonincreasing function of $k$. Often the error decreases rapidly with $k$ and then $p_k$ provides a good representation of $f$ already for small values of $k$. Note that for $n = m$ we obtain the (unique) interpolation polynomial for the given points.

When using the first formula one sometimes finds that *the residual norm increases when the degree of the approximation is increased*! With the modified formula (4.5.68) this is very unlikely to happen; see Problem 4.5.17.[155]

One of the motivations for the method of least squares is that it effectively reduces the influence of random errors in measurements. We now consider some statistical aspects of the method of least squares. First, we need a slightly more general form of Gauss–Markov's theorem.

**Corollary 4.5.22.**

*Assume that in the linear model* (1.4.2)*, $\epsilon$ has zero mean and positive definite covariance matrix $V$. Then the normal equations for estimating $c$ are*

$$(A^T V^{-1} A)\hat{c} = A^T V^{-1} y. \tag{4.5.69}$$

*In particular, if $V = \operatorname{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2)$ this corresponds to a row scaling of the linear model $y = Ac + \epsilon$.*

[155]The difference between the two variants discussed here is similar to the difference between the so-called classical and modified Gram–Schmidt orthogonalization methods.

Suppose that the values of a function have been measured in the points $x_1, x_2, \ldots, x_m$. Let $f(x_p)$ be the measured value, and let $\bar{f}(x_p)$ be the "true" (unknown) function value which is assumed to be the same as the *expected value* of the measured value. Thus *no systematic errors* are assumed to be present. Suppose further that *the errors in measurement at the various points are statistically independent*. Then we have a linear model $f(x_p) = \bar{f}(x_p) + \epsilon$, where

$$E(\epsilon) = 0, \qquad V(\epsilon) = \text{diag}\,(\sigma_1^2, \ldots, \sigma_n^2). \tag{4.5.70}$$

The problem is to use the measured data to estimate the coefficients in the series

$$f(x) = \sum_{j=1}^{n} c_j \phi(x), \quad n \le m,$$

where $\phi_1, \phi_2, \ldots, \phi_n$ are known functions. According to the Gauss–Markov theorem and its corollary the estimates $c_j^*$, which one gets by minimizing the sum

$$\sum_{p=1}^{m} w_p \left( f(x_p) - \sum_{j=1}^{n} c_j \phi_j(x_p) \right)^2, \quad w_p = \sigma_p^{-2},$$

have a smaller variance than the values one gets by any other linear unbiased estimation method. This minimum property holds not only for the estimates of the coefficients $c_j$, but also for every linear functional of the coefficients, for example, the estimate

$$f_n^*(\alpha) = \sum_{j=1}^{n} c_j^* \phi_j(\alpha)$$

of the value $f(\alpha)$ at an arbitrary point $\alpha$.

Suppose now that $\sigma_p = \sigma$ for all $p$ and that the functions $\{\phi_j\}_{j=1}^n$ form an *orthonormal system* with respect to the discrete inner product

$$(f, g) = \sum_{p=1}^{m} f(x_p) g(x_p).$$

Then the least squares estimates are $c_j^* = (f, \phi_j)$, $j = 1 : n$. According to (1.4.4) the estimates $c_j^*$ and $c_k^*$ are *uncorrelated if* $j \neq k$ and

$$E\{(c_j^* - \bar{c}_j)(c_k^* - \bar{c}_k)\} = \begin{cases} 0 & \text{if } j \neq k, \\ \sigma^2 & \text{if } j = k. \end{cases}$$

From this it follows that

$$\text{var}\{f_n^*(\alpha)\} = \text{var}\left\{ \sum_{j=1}^{n} c_j^* \phi_j(\alpha) \right\} = \sum_{j=1}^{n} \text{var}\{c_j^*\} |\phi_j(\alpha)|^2 = \sigma^2 \sum_{j=1}^{n} |\phi_j(\alpha)|^2.$$

As an average, *taken over the grid of measurement points*, the variance of the smoothed function values is

$$\frac{1}{m} \sum_{j=1}^{n} \text{var}\{f_n^*(x_i)\} = \frac{\sigma^2}{m} \sum_{j=1}^{n} \sum_{i=1}^{m} |\phi_j(x_i)|^2 = \sigma^2 \frac{n}{m}.$$

Between the grid points, however, the variance can in many cases be significantly larger. For example, when fitting a polynomial to measurements in *equidistant points*, the Gram polynomial $P_{n,m}$ can be much larger between the grid points when $n > 2m^{1/2}$. Set

$$\sigma_I^2 = \sigma^2 \sum_{j=1}^{n} \frac{1}{2} \int_{-1}^{1} |\phi(x)|^2 \, dx.$$

Thus $\sigma_I^2$ is an average variance for $f_n^*(x)$ *taken over the entire interval* $[-1, 1]$. The following values for the ratio $\rho$ between $\sigma_I^2$ and $\sigma^2(n+1)/(m+1)$ when $m = 42$ were obtained by H. Björk [32]. Related results are found in Reichel [298].

| $n+1$ | 5 | 10 | 15 | 20 | 25 | 30 | 35 |
|---|---|---|---|---|---|---|---|
| $\rho$ | 1.0 | 1.1 | 1.7 | 26 | $7 \cdot 10^3$ | $1.7 \cdot 10^7$ | $8 \cdot 10^{11}$ |

These results are related to the recommendation that one should choose $n < 2m^{1/2}$ when fitting a polynomial to equidistant data. This recommendation seems to contradict the Gauss–Markov theorem, but in fact it just means that one gives up the requirement that the estimates are unbiased. Still, it is remarkable that this can lead to such a drastic reduction of the variance of the estimates $f_n^*$.

If the measurement points are the Chebyshev abscissae, then no difficulties arise in fitting polynomials to data. The Chebyshev polynomials have a magnitude between grid points not much larger than their magnitude at the grid points. The average variance for $f_n^*$ becomes the same on the interval $[-1, 1]$ as on the net of measurements, $\sigma_2(n+1)/(m+1)$.

The choice of $n$ when $m$ is given is a question of compromising between taking into account the truncation error (which decreases as $n$ increases) and the random errors (which grow when $n$ increases). If $f$ is a sufficiently smooth function, then in the Chebyshev case $|c_j|$ decreases quickly with $j$. In contrast, the part of $c_j$ which comes from errors in measurements varies randomly with a magnitude of about $\sigma(2/(m+1)^{1/2})$, using (4.5.50) and $\|T_j\|^2 = (m+1)/2$. The expansion should be broken off when the coefficients begin to "behave randomly." An expansion in terms of Chebyshev polynomials can hence be used for *filtering away the "noise" from the signal, even when $\sigma$ is initially unknown.*

**Example 4.5.12.**

Fifty-one equidistant values of a certain analytic function were rounded to four decimals. In Figure 4.5.2, a semilog diagram is given which shows how $|c_i|$ varies in an expansion in terms of the Chebyshev polynomials for these data. For $i > 20$ (approximately) the contribution due to noise dominates the contribution due to signal. Thus it is sufficient to break off the series at $n = 20$.

## 4.5.7 Approximation in the Maximum Norm

In some applications it is more natural to seek an approximation that minimizes the *maximum deviation* instead of the Euclidean norm. Let $U$ be an $(n+1)$-dimensional vector space of continuous (real or complex) functions $u_k(x)$, $k = 0 : n$, defined on an interval $I = [a, b]$.

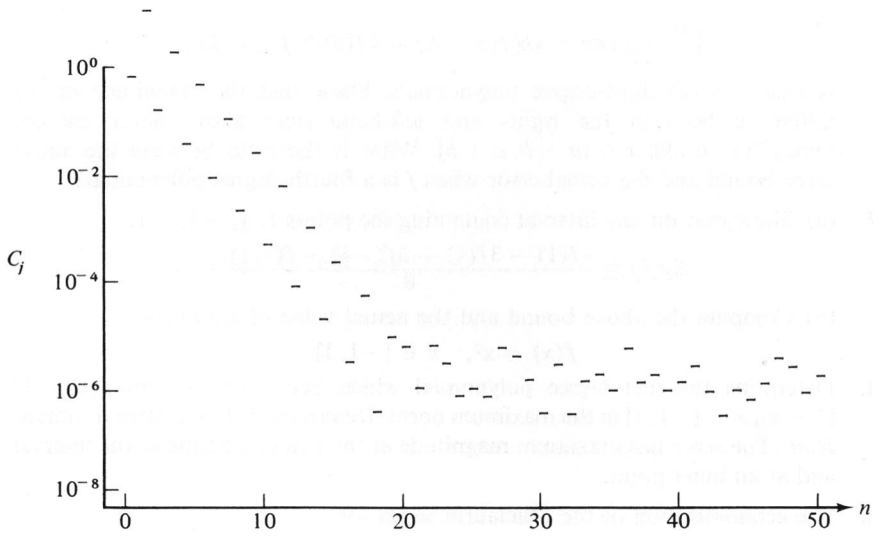



**Figure 4.5.2.** *Magnitude of coefficients $c_i$ in a Chebyshev expansion of an analytic function contaminated with roundoff noise.*

Given a function $f \in C[a, b]$, we consider the approximation problem of finding a function

$$\tilde{u} = \sum_{k=0}^{n} c_k u_k \in U, \tag{4.5.71}$$

which minimizes the maximum norm of the error,

$$\eta(f, I) := \inf_{u \in U} \|f - u\|_{\infty, I} = \inf_{u \in U} \max_{x \in I} |f(x) - u(x)|.$$

Approximation in the maximum norm is often called Chebyshev approximation. We avoid this terminology, since Chebyshev's name is also naturally associated with two entirely different approximation methods: interpolation in the Chebyshev abscissae, and expansion in a series of Chebyshev polynomials.

The related *discrete approximating problem* is that of minimizing

$$\inf_{u \in U} \|f - u\|_{\infty, X} = \inf_{u \in U} \max_{0 \le i \le m} |f(x_i) - u(x_i)|, \tag{4.5.72}$$

where $X = \{x_0, \ldots, x_m\}$ is a finite set of points. We note that the discrete approximation can be posed as a linear programming problem with $2(m + 1)$ linear inequalities:

$$\begin{aligned} &\text{Minimize } \eta \\ &\text{subject to } -\eta \le f(x_i) - \sum_{k=0}^{n} c_k u_k(x_i) \le \eta, \quad i = 0 : m. \end{aligned}$$

In principle this problem can be solved by the simplex method. Algorithms for solving this problem are given in [17].

The discrete approximation problem in $l_1$-*norm* corresponds to the linear programming problem

$$\text{Minimize} \quad \sum_{i=0}^{m} \eta_i$$

$$\text{subject to} \quad -\eta_i \le f(x_i) - \sum_{k=0}^{n} c_k u_k(x_i) \le \eta_i, \quad i = 0:m.$$

Algorithms for solving this problem are discussed in [18].

As a simple example, consider the problem of best approximation in the maximum norm to the function $f(x) = x^n$ on $[-1, 1]$ by a polynomial of lower degree. From the minimax property of the Chebyshev polynomials it follows that the solution is given by the polynomial

$$f_n^*(x) = x^n - 2^{1-n} T_n(x),$$

which has in fact degree $n-2$. The error function $-2^{1-n}T_n(x)$ assumes its extrema $2^{1-n}$ on a sequence of $n+1$ points, $x_k = \cos(k\pi/n)$. The sign of the error alternates at these points, and thus for $f(x) = x^n$

$$E_{n-1}(f) = E_{n-2}(f) = 2^{1-n}.$$

The above property can be generalized. We shall now give a theorem which is the basis of many algorithms.

**Theorem 4.5.23** (*Chebyshev's Equioscillation Theorem*).

*Let $f$ be a continuous function on $[a, b]$ and let $\tilde{p}$ be an nth degree polynomial which best approximates $f$ in the maximum norm. Then $\tilde{p}$ is characterized by the fact that there exists at least $n+2$ points,*

$$a \le x_0 < x_1 < \cdots < x_{n+1} \le b,$$

*where the error $(f - \tilde{p})$ takes on its maximal magnitude $\eta = \|\tilde{f} - p\|_\infty$ with alternating signs, that is,*

$$f(x_k) - \tilde{p}(x_k) = -(f(x_{k+1}) - \tilde{p}(x_{k+1})), \quad k = 0:n. \tag{4.5.73}$$

*This characterization constitutes both a necessary and a sufficient condition. If $f^{(n+1)}$ has constant sign in [a,b], then $x_0 = a$ and $x_{n+1} = b$.*

***Proof.*** We prove here the sufficiency of the condition (4.5.73). For the rest of the proof, see, e.g., Cheney [66]. We first derive another characterization of a best approximation. Given a polynomial $\tilde{p}$ of degree $n$, consider the set of points

$$M = \{x \in [a, b] \mid |f(x) - \tilde{p}(x)| = \|f - \tilde{p}\|_\infty\},$$

where the difference $\tilde{d} = f - \tilde{p}$ takes on the extreme values $\pm\|f - \tilde{p}\|_\infty$. If $\tilde{p}$ is *not* a best approximation, then there is a best approximation that can be written $p^* = \tilde{p} + p$, where $p \ne 0$ is a polynomial of degree at most $n$. Then for all $x \in M$, we have

$$|f(x) - (\tilde{p}(x) + p(x))| < |f(x) - \tilde{p}(x)|.$$

This can only happen if the sign of $p(x)$ and the sign of $(f(x) - \tilde{p}(x))$ are the same, that is,

$$(f(x) - \tilde{p}(x))p(x) > 0, \quad x \in M.$$

Reversing this implication, it follows that $\tilde{p}$ is a best approximation whenever there is no polynomial satisfying this condition.

Suppose now that $\tilde{p}$ satisfies the conditions in the theorem. Then we claim that the conditions $(f(x_k) - \tilde{p}(x_k))p(x_k) > 0$, $k = 0 : n + 2$, cannot hold for any polynomial $p \neq 0$ of degree $n$. For if they did hold, then $p$ would have at least $n + 1$ sign changes in $[a, b]$ and hence also $n + 1$ zeros there. But by the fundamental theorem of algebra this is impossible. □

Notice that Theorem 4.5.23 states that the best solution in maximum norm has *at least* $n + 2$ points where the error alternates. In general, there can be more points where the maximal deviation is achieved.

Suppose that the function $f \in C[a, b]$ is known at $m > n + 1$ distinct points $X = \{\xi_i\}_{i=0}^{m+1}$ in $[a, b]$, and let $\tilde{p}$ be a polynomial of degree $n$. Then $\tilde{p}$ is a best approximation of $f$ in the discrete maximum norm if there are $n+2$ points $x_k \in X$ such that $(f - \tilde{p})$ assumes its maximal absolute value $d = \|\tilde{f} - p\|_\infty$ at these points with alternating sign. The proof of this is identical to the proof above.

**Example 4.5.13.**

In certain simple cases the alternating property can be used to construct the best uniform approximation. Suppose we want to find the best linear approximation to a convex function $f$ on $[a, b]$. In this case the maximum error occurs with alternating signs at the three points $a = x_0, x_1, x_2 = b$, where $x_1$ is chosen so that

$$f'(x_1) = [a, b]f = (f(b) - f(a))/(b - a)$$

gives the solution. The linear polynomial equals the mean value of the tangent of $f$ at $x_1$ and the linear interpolant through the endpoints of $[a, b]$, i.e.,

$$\tilde{p}(x) = (f(a) + f(b))/2 + (x - (a + x_1)/2)[a, b]f,$$

is illustrated in Figure 4.5.3. Notice that linear interpolation (dotted line) using $f(a)$ and $f(b)$ gives a maximum error which is exactly twice as large.

The only property that was used in the proof of Chebyshev's equioscillation theorem was the fundamental theorem of algebra. This property also holds for other subspaces of $C[a, b]$, which we now define.

**Definition 4.5.24.**

*Let $U$ be an $(n+1)$-dimensional vector space of continuous (real or complex) functions $u_i$, $i = 0 : n$, defined on a domain $\mathcal{D}$ containing at least $n + 1$ distinct points. The set of functions $\{u_0, \ldots, u_n\}$ are said to satisfy the* **Haar condition**[156] *(or form a* **Chebyshev system***) if one of the three following equivalent conditions is satisfied:*

[156]This concept is named after the Austrian-Hungarian mathematician Alfred Haar (1885–1933) who, together with Friedrich Riesz established a mathematical center in Szeged, Hungary. Many important contributions to modern functional analysis originated in this center.

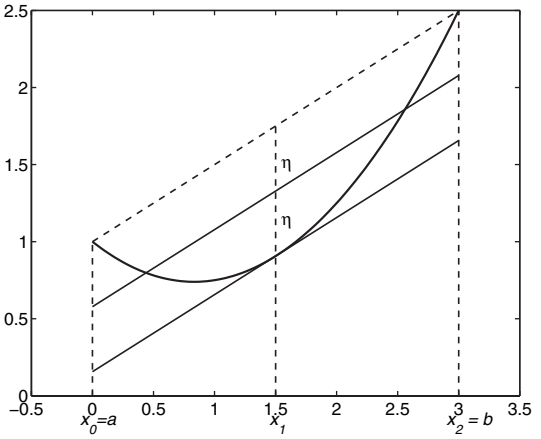



**Figure 4.5.3.** *Linear uniform approximation.*

1. *For any $n+1$ distinct points $x_i$, $i = 0:n$, it holds that*

$$\det \begin{pmatrix} x_0, x_1, \ldots, x_n \\ u_0, u_1, \ldots, u_n \end{pmatrix} := \det \begin{pmatrix} u_0(x_0) & u_1(x_0) & \cdots & u_n(x_0) \\ u_0(x_1) & u_1(x_1) & \cdots & u_n(x_1) \\ \vdots & \vdots & \vdots & \vdots \\ u_0(x_n) & u_1(x_n) & \cdots & u_n(x_n) \end{pmatrix} \neq 0.$$

2. *The interpolation problem to determine $u \in U$ such that $u(x_i) = f_i$, $i = 0:n$, is uniquely solvable.*

3. *Each $u \in U$, not identically zero, has at most $n$ distinct zeros on $\mathcal{D}$.*

We remark that continuous functions of more than one variable cannot form a Chebyshev system. A large part of the theory of polynomial interpolation can be generalized to Chebyshev systems. Theorem 4.5.23 holds for any Chebyshev system, since only the Haar condition is needed in the proof.

**Example 4.5.14.**

The classical example of a Chebyshev system is the set of monomials $u_k(x) = x^k$, $k = 0:n$. The Vandermonde determinant

$$\det(V) = \prod_{0 \le i,\, j \le n} (x_i - x_j)$$

is nonzero if the points $x_i$ are pairwise distinct. Some other important examples are as follows:

1. If $\{u_0, u_1, \ldots, u_n\}$ is a Chebyshev system on $[a, b]$ and $w(x) \neq 0$, then $w(x)u_0, w(x)u_1, \ldots, w(x)u_n$ is a Chebyshev system on $[a, b]$.

2. If $\alpha_0 < \alpha_1 < \cdots \alpha_n$ is a strictly increasing sequence of real numbers, then the power functions $u_k(x) = x^{\alpha_k}$, $k = 0 : n$, form a Chebyshev system on any closed subinterval of $[0, \infty]$.

3. $1, e^x, e^{2x}, \ldots, e^{nx}$, $x \in [a, b]$.

4. $1, \sin x, \sin 2x, \ldots, \sin mx, \cos x, \cos x, \ldots, \cos\, mx$, $x \in [0, 2\pi]$.

5. Let $s_0, s_1, \ldots, s_n$ be distinct values not in $[a, b]$. Then the functions

$$u_i(x) = \frac{1}{x - s_i}, \quad i = 0 : n,$$

form a Chebyshev system on $[a, b]$.

The last example follows from Cauchy's formula for the determinant

$$\det \begin{pmatrix} x_0, x_1, \ldots, x_n \\ u_0, u_1, \ldots, u_n \end{pmatrix} = \prod_{i>j}^{n} (x_i - x_j)(s_i - s_j) \Big/ \prod_{i,j=0}^{n} (x_i + s_j);$$

see Davis [92, Lemma 11.3.1].

As shown by Haar, the Haar condition is necessary and sufficient for the best approximation problem in maximum norm to have a unique solution. (Note that because the $\|\cdot\|_\infty$ is not a strict norm for $C[a, b]$, this does not directly follow from general principles.)

**Theorem 4.5.25.**

*Let $U = \operatorname{span}(u_0, u_1, \ldots, u_n)$ be a Haar subspace of $C[a, b]$. Then for any $f \in C[a, b]$ there is a unique best uniform approximation $u \in U$.*

Even when the alternating property is only approximately satisfied, the following theorem by de la Vallée-Poussin still gives a practical estimate for how good an approximation it is.

**Theorem 4.5.26.**

*Let $f(x)$ and $u_k(x)$, $k = 0 : n$, be real-valued continuous functions defined on the interval $I = [a, b]$ which satisfy the Haar condition on $I$. Assume that an approximation $\tilde{u}(x) \in U$ has the alternating property*

$$(f(x_j) - \tilde{u}(x_j))(-1)^j \sigma = \mu_j > 0 \tag{4.5.74}$$

*at $(n + 2)$ distinct points $x_0, \ldots, x_{n+1} \in I$, where $\sigma = \pm 1$. Let $\eta(f, D) = \inf_{u \in U} \|f - u\|_{\infty, D}$ denote the error of the best approximation on the domain $D$. Then it holds that*

$$\mu := \min_{0 \le j \le n+1} \mu_j \le \eta(f, X) \le \eta(f, I) \le \|f - \tilde{u}\|_{\infty, I}, \tag{4.5.75}$$

$$\min_{0 \le j \le n+1} \mu_j \le \eta(f, X) \le \max_{0 \le j \le n+1} \mu_j = \|f - \tilde{u}\|_{\infty, X}. \tag{4.5.76}$$

***Proof.*** Since $X \subset I$ we have $\eta(f, X) \le \eta(f, I)$. It then suffices to show that for any $u \in U$, the inequality $\mu \le \|f - u\|_{\infty,X}$ holds. This follows from

$$\begin{aligned}\mu \le \mu_j &= (f(x_j) - \tilde{u}(x_j))(-1)^j\sigma \\ &= (f(x_j) - u(x_j))(-1)^j\sigma + (u(x_j) - \tilde{u}(x_j))(-1)^j\sigma \\ &\le (f(x_j) - u(x_j))(-1)^j\sigma \le \|f - u\|_{\infty,X}\end{aligned}$$

for those $x_j \in X$ for which $(u(x_j) - \tilde{u}(x_j))(-1)^j\sigma \le 0$. Such an $x_j$ must exist since otherwise $u - \tilde{u}$ has at least $n + 1$ zeros and does not vanish. □

It follows that an approximation $\tilde{u}$ with equality in (4.5.76) is the best approximation on $X$. If $X$ is chosen so that equality also holds in (4.5.75), then $\tilde{u}$ is the best approximation on the whole interval $I = [a, b]$.

The alternating property of the best approximation plays a key role in the solution of the best approximation problem. The idea is to solve a sequence of discrete uniform approximation problems each using $(n + 2)$ reference points:

$$X = \{x_0, \ldots, x_{n+1}\}, \quad a \le x_0 \le x_1 \cdots \le x_{n+1} \le b.$$

We therefore now consider how we can determine $\tilde{u} = u^*$ so that equality in (4.5.76) is attained. For a given set of points $X = x_0, \ldots, x_{n+1}$ assume that $u^*$ has the alternating property

$$(f(x_j) - u^*(x_j))(-1)^{n+1-j}\eta, \quad j = 0 : n + 1, \tag{4.5.77}$$

where $\eta$ is to be determined. We extend the function $u_0, \ldots, u_n$ with the function $u_{n+1}(x_j) = (-1)^j$ defined on $X$. Then we obtain a linear system $\tilde{U}_{n+1}c = f$, or

$$\begin{pmatrix} u_0(x_0) & \cdots & u_n(x_0) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ u_0(x_n) & \cdots & u_n(x_n) & (-1)^n \\ u_0(x_{n+1}) & \cdots & u_n(x_{n+1}) & (-1)^{n+1} \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_n \\ \eta \end{pmatrix} = \begin{pmatrix} f_0 \\ \vdots \\ f_n \\ f_{n+1} \end{pmatrix}. \tag{4.5.78}$$

The first leading submatrix of order $n + 1$ of $\tilde{U}_{n+1}$ is nonsingular since $u_0, \ldots, u_n$ form a Chebyshev system. Hence its first $n + 1$ columns are linearly independent. Further, there cannot exist a linear combination of the first $n + 1$ columns that is parallel to the last column since this would imply that some function $u \in U$ has $n+1$ zeros in $[a, b]$. It follows that the system (4.5.78) is nonsingular and has a unique solution. Note that once $d$ has been determined, then the remaining coefficients $c_i$, $i = 0 : n$, can be computed by solving an interpolating problem.

In the special case of polynomial approximation the system (4.5.78) can be solved in $O(n^2)$ flops. It can be shown that $\eta$ can be determined as the quotient between two divided differences

$$\eta = [x_0, x_1, \ldots, x_{n+1}]f/[x_0, x_1, \ldots, x_{n+1}]s, \tag{4.5.79}$$

where $s(x)$ is a function such that $s(x_i) = (-1)^i$. The Newton form of the solution $p(x)$ is then given by

$$p(x) = \sum_{j=0}^{n} [x_0, \ldots, x_j](f - \eta s) \prod_{k=0}^{j-1} (x - x_k). \tag{4.5.80}$$

The **Remez exchange algorithm**, proposed by Remez [300] in 1934, is an iterative method for computing the best uniform approximation on an interval $[a, b]$. (Sometimes the name is transcribed as Remes.) In this method one proceeds as follows:

1. Choose an initial reference set $X$ consisting of $(n + 2)$ points $a \le x_0 < x_1 < \cdots < x_{n+1} \le b$.
2. Compute the (unique) discrete approximation $\tilde{u}_X$ to $f(x)$ on the reference set.
3. Adjust the points in the reference set to be the extreme points of the error curve $f(x) - \tilde{u}_X(x)$.
4. Repeat steps 2 and 3 until convergence.

Usually the initial reference set is chosen as Chebyshev points transformed to the interval $[a, b]$. The Remez algorithm can also be applied to the discrete case where $f$ is only defined on a grid $\xi_0, \xi_1, \ldots, \xi_m$, $m \gg n$. In this case the Remez algorithm converges in a finite number of steps.

In the continuous case, step 3, one can use Newton's method on the equation $f(x) - \tilde{u}_X(x) = 0$, with starting values equal to $x_i$ for all interior points. If derivatives are not available, then a combination of inverse parabolic interpolation and golden section search given by Brent [44, Chap. 5] is a good alternative; see Sec. 6.4.3.

A discussion of the exchange algorithm and its convergence can be found in Meinardus [264]. A careful implementation of the Remez algorithm, where the delicate points to consider in an implementation are discussed, is given by Golub and Smith [168]. Assuming that $f(x)$ is sufficiently smooth quadratic, convergence is achieved.

## Review Questions

**4.5.1** State the axioms that any norm must satisfy. Define the maximum norm and the Euclidean norm for a continuous function $f$ on a closed interval.

**4.5.2** Define dist$(f, P_n)$, and state Weierstrass' approximation theorem.

**4.5.3** Prove the Pythagorean theorem in an inner product space.

**4.5.4** Define and give examples of orthogonal systems of functions.

**4.5.5** Formulate and prove Bessel's inequality and Parseval's identity, and interpret them geometrically.

**4.5.6** (a) Give some reasons for using orthogonal polynomials in polynomial approximation with the method of least squares.

(b) Give some argument against the assertion that orthogonal polynomials are difficult to work with.

**4.5.7** In Chebyshev interpolation we seek the coefficients $c_j$ such that

$$p(x_k) = \sum_{j=0}^{m} c_j T_j(x_k), \quad k = 0 : m,$$

where $x_k$ are the zeros of $T_{m+1}(x)$. How would you compute $c_j$?

**4.5.8** The Gram polynomials are examples of orthogonal polynomials. With respect to what inner product are they orthogonal?

**4.5.9** Let $\phi_k(x), k = 1, 2, 3, \ldots$, be a triangle family of orthogonal polynomials in an inner product space. What property can be used to generate this sequence of polynomials?

**4.5.10** What is the characteristic property of the $n$th degree polynomial which best approximates a given continuous function in the maximum norm over a closed interval?

**4.5.11** Give at least three examples of a Chebyshev system of functions.

## Problems and Computer Exercises

**4.5.1** Compute $\|f\|_\infty$ and $\|f\|_2$ for the function $f(x) = (1 + x)^{-1}$ on the interval $[0, 1]$.

**4.5.2** Determine straight lines which approximate the curve $y = e^x$ such that

(a) the discrete Euclidean norm of the error function on the grid $(-1, -0.5, 0, 0.5, 1)$ is as small as possible;

(b) the Euclidean norm of the error function on the interval $[-1, 1]$ is as small as possible;

(c) the line is tangent to $y = e^x$ at the point $(0, 1)$, i.e., the Taylor approximation at the midpoint of the interval.

**4.5.3** Determine, for $f(x) = \pi^2 - x^2$, the "cosine polynomial" $f^* = \sum_{j=0}^{n} c_j \cos jx$ which makes $\|f^* - f\|_2$ on the interval $[0, \pi]$ as small as possible.

**4.5.4** (a) Show that on any interval containing the points $-1, -1/3, 1/3, 1$,

$$E_2(f) \geq \frac{1}{8}\left| f(1) - 3f\left(\frac{1}{3}\right) + 3f\left(-\frac{1}{3}\right) - f(-1)\right|.$$

(b) Compute the above bound and the actual value of $E_2(f)$ for $f(x) = x^3$.

**4.5.5** (a) Let a scalar product be defined by $(f, g) = \int_a^b f(x)g(x)\,dx$. Calculate the matrix of normal equations, when $\phi_j(x) = x^j$, $j = 0 : n$, when $a = 0$, $b = 1$.

(b) Do the same when $a = -1$, $b = 1$. Show how in this case the normal equations can be easily decomposed into two systems, with approximately $(n+1)/2$ equations in each.

**4.5.6** Verify the formulas for $\|\phi_j\|^2$ given in Example 4.5.10.

**4.5.7** (a) Show that $\|f - g\| \geq \|f\| - \|g\|$ for all norms. (Use the axioms mentioned in Sec. 4.5.1.)

(b) Show that if $\{c_j\}_0^n$ is a set of real numbers and if $\{f_j\}_0^n$ is a set of vectors, then $\|\sum c_j f_j\| \leq \sum |c_j| \|f_j\|$.

**4.5.8** Let $G \in \mathbf{R}^{n\times n}$ be a symmetric positive definite matrix. Show that an inner product is defined by the formula $(u, v) = u^T G v$. Show that $A^* = G^{-1}A^T G$.

**4.5.9** In a space of complex-valued twice differentiable functions of $t$ which vanish at $t = 0$ and $t = 1$, let the inner product be

$$(u, v) = \int_0^1 u(t)\bar{v}(t)\, dt.$$

What is the adjoint of the operator $A = d/dt$? Is it true that the operator $iA$ is self-adjoint, and that $-A^2$ is self-adjoint and positive definite?

**4.5.10** (a) Show that, in a real inner-product space,

$$4(u, v) = \|u + v\|^2 - \|u - v\|^2.$$

In a complex space this gives only the real part of the inner product. Show that one has to add $\|u - iv\|^2 - \|u + iv\|^2$.

(b) This can be used to reduce many questions concerning inner products to questions concerning norms. For example, in a general inner-product space a **unitary operator** is defined by the requirement that $\|Au\| = \|u\|$ for all $u$. Show that $(Au, Av) = (u, v)$ for all $u, v$.

Note, however, that the relation $(u, Au) = (Au, u)$ for all $u$ which, in a real space, holds for every operator $A$, does *not* imply that $(u, Av) = (Au, v)$ for all $u, v$. The latter holds only if $A$ is self-adjoint.

**4.5.11** Show that $(AB)^* = B^*A^*$. Also show that if $C$ is self-adjoint and positive definite, then $A^*CA$ is so too. ($A$ is not assumed to be self-adjoint.)

**4.5.12** Show that

$$(A^{-1})^* = (A^*)^{-1}, \qquad (A^p)^* = (A^*)^p,$$

for all integers $p$, provided that the operators mentioned exist. Is it true that $C^p$ is self-adjoint and positive definite if $C$ is so?

**4.5.13** Show the following **minimum property** of orthogonal polynomials: Among all $n$th-degree polynomials $p_n$ with leading coefficient 1, the smallest value of

$$\|p_n\|^2 = \int_a^b p_n^2(x)w(x)dx, \quad w(x) \geq 0,$$

is obtained for $p_n = \phi_n/A_n$, where $\phi_n$ is the orthogonal polynomial with leading coefficient $A_n$ associated with the weight distribution $w(x)$.

*Hint:* Determine the best approximation to $x^n$ in the above norm or consider the expansion $p_n = \phi_n/A_n + \sum_{j=0}^{n-1} c_j\phi_j$.

**4.5.14** Verify the formulas for $\|T_j\|^2$ given in Theorem 4.5.20.

**4.5.15** Modify Clenshaw's algorithm to a formula for the derivative of an orthogonal expansion.

**4.5.16** (a) Let $\alpha_j$, $j = 1 : n$, be the zeros of the Chebyshev polynomial $T_n(x)$, $n \geq 1$. (There are, of course, simple trigonometric expressions for them.) Apply Clenshaw's algorithm to compute

$$\sum_{m=0}^{n-1} T_m(\alpha_1)T_m(x), x = \alpha_j, j = 1 : n.$$

It turns out that the results are remarkably simple.

(b) Show that $S = \sum_{k=0}^{n-1} c_k\phi_k$ can be computed by a forward version of Clenshaw's algorithm that reads

$$\begin{aligned}
&y_{-2} = 0; \quad y_{-1} = 0;\\
&\textbf{for } k = 0 : n-1,\\
&\quad y_k = (-y_{k-2} + \alpha_k y_{k-1} + c_k)/\gamma_{k+1};\\
&\textbf{end}\\
&S = c_n\phi_n + \gamma_n y_{n-1}\phi_{n-1} - y_{n-2}\phi_n.
\end{aligned}$$

Add this version as an option to your program, and study [294, Sec. 5.4], from which this formula is quoted (with adaptation to our notation). Make some test example of your own choice.

**4.5.17** (a) Write a MATLAB function `c = stieltjes(f,x,w,m,n)` that computes the orthogonal coefficients in a least squares polynomial fit to the data $(f_i, x_i)$ and weights $w_i$, $i = 0 : m$. Compute the orthogonal polynomials $\phi_k$ using the Stieltjes procedure. For computing $c_k$, $k = 0 : n$, use either (4.5.67) or (4.5.68).

(b) Apply the function in (a) to the case $f_i = x_i^7$, $w_i = 1/(f_i)^2$, and $m = 20$. Compute and print the error $\|p_k - f\|$ for $k = 0 : 10$, using the expression (4.5.67) for $c_{k+1}$. Note that for $k > 7$ the fit should be exact.

(c) Repeat the calculations in (b) using the modified formula (4.5.68). Compare the error with the results in (b).

**4.5.18** (a) What first-degree polynomial does one get with Chebyshev interpolation to $f(x) = 1/(3 + x)$, $[-1, 1]$? What is the maximum norm of the error?

*Answer:* $p(x) = (-2x + 6)/17$; 0.0294 attained at $x = -1$.

(b) Determine the first-degree polynomial which best approximates the function $f(x) = 1/(3 + x)$, $[-1, 1]$, in the maximum norm. Determine $E_1(f)$ to at least three decimals.

*Hint:* The error has maximum magnitude at the two endpoints of the interval and at one inner point.

*Answer:* $p(x) = (-x + 2\sqrt{2})/8$; $E_1(f) = 0.0214$.

**4.5.19** (a) Show that the three conditions in Definition 4.5.24 are equivalent.

(b) Show the formula for *eta* given in (4.5.79).

**4.5.20** Determine, using the Remez algorithm, the second-degree polynomial $p(x) = c_0 + c_1x + c_2x^24$ that best approximates $f(x) = \sin\left(\frac{\pi}{2}x\right)$ on $I = [0, 1]$.

*Hint:* Since the second derivative $f''$ has constant sign in $I$, we can take $x_0 = a$, $x_3 = b$, and only the two interior points $x_1$ and $x_2$ are unknown.

## 4.6 Fourier Methods

Many natural phenomena, for example, acoustical and optical, are of a periodic character. For instance, it was known already by Pythagoras (600 B.C.) that a musical sound is composed of regular oscillations, partly a fundamental tone with a certain frequency $f$, and partly overtones with frequencies $2f$, $3f$, $4f$, .... The ratio of the strength of the fundamental tone to that of the overtones is decisive for our impression of the sound. Sounds which are free from overtones occur, for instance, in electronic music, where they are called pure sine tones.

In an electronic oscillator, a current is generated whose strength at time $t$ varies according to the formula $r\sin(\omega t + v)$, where $r$ is called the amplitude of the oscillation; $\omega$ is called the angular frequency and is equal to $2\pi$ times the frequency; and $v$ is a constant which defines the state at the time $t = 0$. In a loudspeaker, variations of current are converted into variations in air pressure which, under ideal conditions, are described by the same function. In practice, however, there is always a certain distortion; overtones occur. The variations in air pressure which reach the ear can, from this viewpoint, be described as a sum of the form

$$\sum_{k=0}^{\infty} r_k \sin(k\omega t + v_k). \tag{4.6.1}$$

An expansion of this form is called a **Fourier series**.

The separation of a periodic phenomenon into a fundamental tone and overtones permeates not only acoustics, but also many other areas. It is related to an important, purely mathematical theorem first given by Fourier.[157] According to this theorem, every function $f(t)$ with period $2\pi/\omega$ can, under certain very general conditions, be expanded in a Fourier series of the form (4.6.1). (A function has period $p$ if $f(t + p) = f(t)$ for all $t$.) A more precise formulation will be given later in Theorem 4.6.2.

Fourier series are valuable aids in the study of phenomena which are periodic in time (such as vibrations, sound, light, alternating currents) or in space (waves, crystal structure, etc.). One very important area of application is in digital signal and image processing which is used in interpreting radar and sonar signals. Another is statistical time series, which are used in communications theory, control theory, and the study of turbulence. For the numerical analyst, Fourier analysis is partly a very common computational task and partly an important aid in the analysis of properties of numerical methods.

Modifications of pure Fourier methods are used as a means of analyzing nonperiodic phenomena; see, e.g., Sec. 4.6.3 (periodic continuation of functions) and Sec. 4.6.5 (Fourier transforms). The approximation of Fourier expansions using sampled data and discrete Fourier analysis is treated in Sec. 4.6.2. The fast Fourier transform (FFT) for discrete Fourier analysis and synthesis is treated Sec. 4.7.1. It has caused a complete change of attitude toward what can be done using discrete Fourier methods.

[157] Jean Baptist Joseph Fourier (1768–1830), French mathematician. In 1807 Fourier completed and read to the Paris Institute his important memoir *Théorie Analytique de la Chaleur* [Analytical Theory of Heat], in which he used what is now called Fourier series. It won a prize competition set by the Institute in 1811, but was not published until 1822.

### 4.6.1 Basic Formulas and Theorems

The basic formulas and theorems derived in this section rely to a great extent on the theory in Sec. 4.5. An expansion of the form of (4.6.1) can be expressed in many equivalent ways. If we set $a_k = r_k \sin v_k$, $b_k = r_k \cos v_k$, then using the addition theorem for the sine function we can write

$$f(t) = \sum_{k=0}^{\infty} (a_k \cos k\omega t + b_k \sin k\omega t), \tag{4.6.2}$$

where $a_k$, $b_k$ are real constants. Another form, which is often the most convenient, can be found with the help of Euler's formulas,

$$\cos x = \frac{1}{2}(e^{ix} + e^{-ix}), \qquad \sin x = \frac{1}{2i}(e^{ix} - e^{-ix}), \quad (i = \sqrt{-1}).$$

Here and in what follows $i$ denotes the imaginary unit. Then one gets

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{ik\omega t}, \tag{4.6.3}$$

where

$$c_0 = a_0, \quad c_k = \frac{1}{2}(a_k - i\, b_k), \qquad c_{-k} = \frac{1}{2}(a_k + i\, b_k), \quad k \geq 1, \ldots. \tag{4.6.4}$$

In the rest of this chapter we shall use the term **Fourier series** to denote an expansion of the form of (4.6.1) or (4.6.3). We shall call the partial sums of the form of these series **trigonometric polynomials**. Sometimes the term **spectral analysis** is used to describe the above methods.

We shall study *functions with period* $2\pi$. These are fully defined by their values on the **fundamental interval** $[-\pi, \pi]$. If a function of $t$ has period $L$, then the substitution $x = 2\pi t/L$ transforms the function to a function of $x$ with period $2\pi$. We assume that the function can have complex values, since the complex exponential function is convenient for manipulations.

In the **continuous case** the inner product of two complex-valued functions $f$ and $g$ of period $2\pi$ is defined in the following way (the bar over $g$ indicates complex conjugation):

$$(f, g) = \int_{-\pi}^{\pi} f(x)\bar{g}(x)dx. \tag{4.6.5}$$

(It makes no difference what interval one uses, as long as it has length $2\pi$—the value of the inner product is unchanged.) As usual the norm of the function $f$ is defined by $\|f\| = (f, f)^{1/2}$. Notice that $(g, f) = \overline{(f, g)}$.

**Theorem 4.6.1.**

*The following orthogonality relations hold for the functions*

$$\phi_j(x) = e^{ijx}, \quad j = 0, \pm 1, \pm 2, \ldots,$$

*where*

$$(\phi_j, \phi_k) = \begin{cases} 2\pi & \text{if } j = k, \\ 0 & \text{if } j \neq k. \end{cases} \tag{4.6.6}$$

***Proof.*** In the continuous case, if $j \neq k$, it holds that

$$(\phi_j, \phi_k) = \int_{-\pi}^{\pi} e^{ijx} e^{-ikx} dx = \Big|_{-\pi}^{\pi} \frac{e^{i(j-k)x}}{i(j-k)} = \frac{(-1)^{j-k} - (-1)^{j-k}}{i(j-k)} = 0,$$

whereby orthogonality is proved. For $j = k$

$$(\phi_k, \phi_k) = \int_{-\pi}^{\pi} e^{ikx} e^{-ikx} dx = \int_{-\pi}^{\pi} 1 \, dx = 2\pi. \qquad \square$$

If one knows that the function $f(x)$ has an expansion of the form

$$f = \sum_{j=-\infty}^{\infty} c_j \phi_j,$$

then from Theorem 4.6.1 it follows formally that

$$(f, \phi_k) = \sum_{j=a}^{b} c_j (\phi_j, \phi_k) = c_k (\phi_k, \phi_k), \quad a \leq k \leq b,$$

since $(\phi_j, \phi_k) = 0$ for $j \neq k$. Thus, changing $k$ to $j$, we have

$$c_j = \frac{(f, \phi_j)}{(\phi_j, \phi_j)} = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) e^{-ijx} dx. \tag{4.6.7}$$

These coefficients are called **Fourier coefficients**; see the more general case in Theorem 4.5.13. In accordance with (4.6.4) set

$$a_j = c_j + c_{-j}, \qquad b_j = i(c_j - c_{-j}).$$

Then

$$\begin{aligned} \sum_{j=-N}^{N} c_j e^{ijx} &= c_0 + \sum_{j=1}^{N} \big( c_j (\cos jx + i \sin jx) + c_{-j} (\cos jx - i \sin jx) \big) \\ &= \frac{1}{2} a_0 + \sum_{j=1}^{N} (a_j \cos jx + b_j \sin jx), \end{aligned}$$

where

$$a_j = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos jx \, dx, \quad j \geq 0, \qquad b_j = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin jx \, dx, \quad j \geq 1. \tag{4.6.8}$$

(Notice that the factors preceding the integral are different in the expressions for $c_j$ and for $a_j, b_j$, respectively.)

From a generalization of Theorem 4.5.13, we also know that the error

$$\Big\| f - \sum_{j=-n}^{n} k_j \phi_j \Big\|, \quad n < \infty,$$

becomes as small as possible if we choose $k_j = c_j$, $-n \le j \le n$. Theorem 4.5.14 and its corollary, Parseval's identity,

$$2\pi \sum_{j=-\infty}^{\infty} |c_j|^2 = \|f\|^2 = \int_{-\pi}^{\pi} |f(x)|^2 \, dx, \tag{4.6.9}$$

are of great importance in many applications of Fourier analysis. The integral in (4.6.9) can be interpreted as the "energy" of the function $f(x)$.

**Theorem 4.6.2** (*Fourier Analysis, Continuous Case*).

*Assume that the function $f$ is defined at every point in the interval $[-\pi, \pi]$ and that $f(x)$ is finite and piecewise continuous. Associate with $f$ a Fourier series in the following two ways:*

$$\frac{1}{2}a_0 + \sum_{j=1}^{\infty}(a_j \cos jx + b_j \sin jx) \quad \textit{and} \quad \sum_{j=-\infty}^{\infty} c_j e^{ijx},$$

*where the coefficients $a_j, b_j$, and $c_j$ are defined by* (4.6.8) *in the first case and* (4.6.7) *in the second case. Then the partial sums of the above expansions give the best possible approximations to $f(x)$ by trigonometric polynomials, in the least squares sense.*

*If $f$ is of bounded variation and has at most a finite number of discontinuities, then the series is everywhere convergent to $f(x)$. At a point $x = a$ of discontinuity $f(a)$ equals the mean $f(a) = \frac{1}{2}(f(a+) + f(a-))$.*

***Proof.*** The proof of the convergence results is outside the scope of this book (see, e.g., Courant and Hilbert [83]). The rest of the assertions follow from previously made calculations in Theorem 4.6.1 and the comments following; see also the proof of Theorem 4.5.13. □

The more regular a function is, the faster its Fourier series converges. The following useful result is relatively easy to prove using (4.6.7) and integrating by parts $k + 1$ times (cf. (3.2.8)).

**Theorem 4.6.3.**

*If $f$ and its derivatives up to and including order $k$ are periodic and* everywhere *continuous, and if $f^{(k+1)}$ is piecewise continuous, then*

$$|c_j| \le \frac{1}{j^{(k+1)}} \|f^{(k+1)}\|_\infty. \tag{4.6.10}$$

Sometimes it is convenient to separate a function $f$ defined on $[-\pi, \pi]$ into an even and an odd part. We set $f(x) = g(x) + h(x)$, where

$$g(x) = \frac{1}{2}(f(x) + f(-x)), \qquad h(x) = \frac{1}{2}(f(x) - f(-x)) \quad \forall x. \tag{4.6.11}$$

Then $g(x) = g(-x)$ and $h(x) = -h(-x)$. For both $g(x)$ and $h(x)$ it suffices to give the function only on $[0, \pi]$. For the even function $g(x)$ the sine part of the Fourier series drops

and we have

$$g(x) = \frac{1}{2}a_0 + \sum_{j=1}^{\infty} a_j \cos jx, \quad a_j = \frac{2}{\pi}\int_0^{\pi} g(x)\cos jx\,dx. \tag{4.6.12}$$

For $h(x)$ the cosine part drops out and the Fourier series becomes a sine series:

$$h(x) = \sum_{j=1}^{\infty} b_j \sin jx, \quad b_j = \frac{2}{\pi}\int_0^{\pi} h(x)\sin jx\,dx. \tag{4.6.13}$$

The proof is left as an exercise to the reader (use the formulas for the coefficients given in (4.6.8)).

**Example 4.6.1.**

Consider the rectangular wave function obtained by periodic continuation outside the interval $(-\pi, \pi)$ of

$$f(x) = \begin{cases} -1/2, & -\pi < x < 0, \\ 1/2, & 0 < x < \pi; \end{cases}$$

see Figure 4.6.1. The function is odd, so $a_j = 0$ for all $j$, and

$$b_j = \frac{1}{\pi}\int_0^{\pi} \sin jx\,dx = \frac{1}{j\pi}(1 - \cos j\pi) = \begin{cases} 0 & \text{if } j \text{ even}, \\ 2/(j\pi) & \text{if } j \text{ odd}. \end{cases}$$

Hence

$$f(x) = \frac{2}{\pi}\Big(\sin x + \frac{\sin 3x}{3} + \frac{\sin 5x}{5} + \cdots\Big). \tag{4.6.14}$$

Notice that the coefficients $c_j$ decay as $j^{-1}$ in agreement with Theorem 4.6.3.

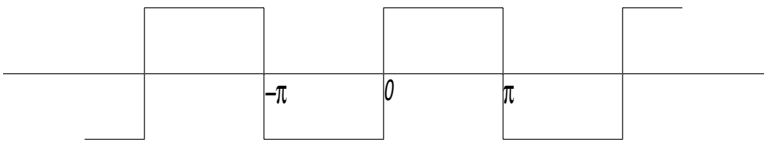



**Figure 4.6.1.** *A rectangular wave.*

The sum of the series is zero at the points where $f$ has a jump discontinuity; this agrees with the fact that the sum should equal the average of the limiting values to the left and to the right of the discontinuity.

Figure 4.6.2 shows the approximations to the square wave using one, two, five, and ten terms of the series (4.6.14). As can be seen, there is a ringing effect near the discontinuities. The width and energy of this error are reduced when the number of terms in the approximation is increased. However, the height of the overshoot and undershoot near the

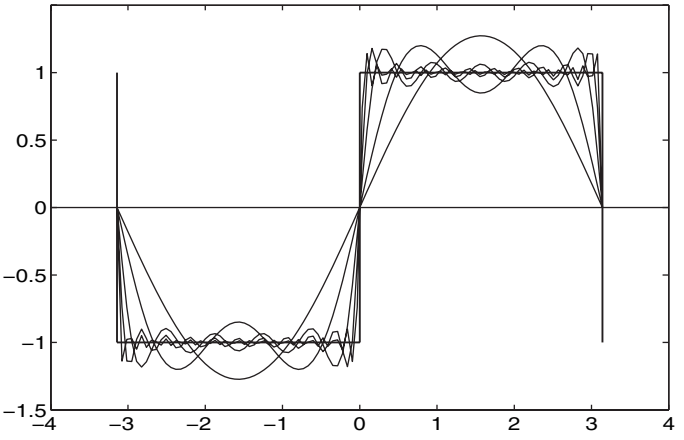



**Figure 4.6.2.** *Illustration of Gibbs' phenomenon.*

discontinuity converges to a fixed height, which is equal to about 0.179 times the jump in the function value. This artifact is known as **Gibbs' phenomenon**.[158]

The Gibbs' oscillations can be smoothed by multiplying the terms by factors which depend on $m$, the order of the partial sum. Let us consider the finite expansion

$$f_m(x) = \sum_{k=-(m-1)}^{m-1} c_k e^{ikx}.$$

Then in the smoothed expansion each term in the sum is multiplied by the Lanczos $\sigma$-factors,

$$f_m(x) = \sum_{k=-(m-1)}^{m-1} \sigma_k c_k e^{ikx}, \quad \sigma_k = \frac{\sin \pi k/m}{\pi/m} \tag{4.6.15}$$

(see Lanczos [235, Chap. IV, Secs. 6 and 9]). Since the coefficients in the real form of the Fourier series are

$$a_k = c_k + c_{-k}, \qquad b_k = i(c_k - c_{-k}),$$

the same $\sigma$-factor applies to them.

The Gibbs' oscillations can also be suppressed by using the epsilon algorithm. For this purpose one adds the conjugate Fourier series, applies the epsilon algorithm, and keeps only the real part of the result.

## 4.6.2 Discrete Fourier Analysis

Although the data to be treated in Fourier analysis are often continuous in the time or space domain, for computational purposes these data must usually be represented in terms of a

[158] Named after the American physicist J. William Gibbs, who in 1899 proved that this ringing effect will always occur when approximating a discontinuous function with a Fourier series.

finite discrete sequence. For example, a function $f(t)$ of time is recorded at evenly spaced intervals $\Delta t$ in time. Assume that the function $f$ is known at equidistant arguments in the interval $[0, 2\pi]$,

$$x_k = 2\pi k/N, \quad k = 0 : N-1.$$

Such data can be analyzed by discrete Fourier analysis. Define the inner product

$$(f, g) = \sum_{k=0}^{N-1} f(x_k)\bar{g}(x_k), \quad x_k = 2\pi k/N. \tag{4.6.16}$$

Then, with $\phi_j(x) = e^{ijx}$ we have

$$(\phi_j, \phi_k) = \sum_{k=0}^{N-1} e^{ijx_k} e^{-ikx_k} = \sum_{k=0}^{N-1} e^{i(j-k)hk}.$$

From Lemma 3.2.2 it now follows that

$$(\phi_j, \phi_k) = \begin{cases} N & \text{if } (j-k)/N \text{ is an integer,} \\ 0 & \text{otherwise.} \end{cases} \tag{4.6.17}$$

**Theorem 4.6.4** (*Trigonometric Interpolation*).

*Every function, defined on the equidistant grid $x_k = 2\pi k/N$, $k = 0 : N-1$, can be interpolated by the trigonometric polynomial*

$$f(x) = \begin{cases} \displaystyle\sum_{j=-k}^{k+\theta} c_j e^{ijx}, \\ \displaystyle\frac{1}{2}a_0 + \sum_{j=1}^{k}(a_j \cos jx + b_j \sin jx) + \frac{1}{2}\theta a_{k+1}\cos(k+1)x. \end{cases} \tag{4.6.18}$$

*Here*

$$\theta = \begin{cases} 1 & \text{if } N \text{ even,} \\ 0 & \text{if } N \text{ odd,} \end{cases} \qquad k = \begin{cases} N/2-1 & \text{if } N \text{ even,} \\ (N-1)/2 & \text{if } N \text{ odd,} \end{cases} \tag{4.6.19}$$

*and*

$$c_j = \frac{1}{N}\sum_{k=0}^{N-1} f(x_k)e^{-ijx_k}, \tag{4.6.20}$$

$$a_j = \frac{2}{N}\sum_{k=0}^{N-1} f(x_k)\cos jx_k, \qquad b_j = \frac{2}{N}\sum_{k=0}^{N-1} f(x_k)\sin jx_k. \tag{4.6.21}$$

*If the sums in* (4.6.18) *are terminated when $|j| < k + \theta$, then one obtains the trigonometric polynomial which is the best least squares approximation, among all trigonometric polynomials with the same number of terms, to $f$ on the grid.*

***Proof.*** The expression for $c_j$ is justified by (4.6.17). Further, by (4.6.20)–(4.6.21) it follows that

$$a_j = c_j + c_{-j}, \qquad b_j = i(c_j - c_{-j}), \qquad c_{k+1} = \frac{1}{2}a_{k+1}.$$

The two expressions for $f(x)$ are equivalent, because

$$\begin{aligned}\sum_{j=-k}^{k+\theta} c_j e^{ijx} &= c_0 + \sum_{j=1}^{k}\big(c_j(\cos jx + i\sin jx) + c_{-j}(\cos jx - i\sin jx)\big)\\ &\quad + \theta c_{k+1}\cos(k+1)x\\ &= c_0 + \sum_{j=1}^{k}(a_j\cos jx + b_j\sin jx) + \frac{1}{2}\theta a_{k+1}\cos(k+1)x. \qquad \square\end{aligned}$$

The function $f(x)$ coincides *on the grid* $x_0, x_1, \ldots, x_{N-1}$ with the function

$$f^*(x) = \sum_{j=0}^{N-1} c_j e^{ijx} \tag{4.6.22}$$

because

$$e^{-i(N-j)x_k} = e^{ijx_k}, \qquad c_{-j} = c_{N-j}.$$

However, the functions $f$ and $f^*$ are *not* identical between the grid points. If we set $\omega = e^{ix}$ and $\omega_k = e^{ix_k}$, then

$$f^*(x) = P(\omega) = \sum_{j=0}^{N-1} c_j \omega^j,$$

where $P(\omega)$ is a polynomial of degree less than $N$. It becomes clear that trigonometric interpolation is equivalent to polynomial interpolation at the grid points $\omega_k$. The mapping $\mathbf{C}^N \to \mathbf{C}^N$

$$(f_0, f_1, \ldots, f_{N-1}) \mapsto (c_0, c_1, \ldots, c_{N-1})$$

is called the **discrete Fourier transform** (DFT).

The calculations required to compute the coefficients $c_j$ according to (4.6.20), **Fourier analysis**, are of essentially the same type as the calculations needed to compute $f^*(x)$ at the grid points

$$x_k = 2\pi k/N, \quad k = 0 : N-1,$$

when the expansion in (4.6.22) is known, so-called **Fourier synthesis**. Both calculations can be performed very efficiently using FFT algorithms; see Sec. 4.7.1.

Functions of several variables are treated analogously. Quite simply, one takes one variable at a time. In the discrete case with two variables we set

$$x_k = 2\pi k/N, \qquad y_\ell = 2\pi \ell/N,$$

and assume that $f(x_k, y_\ell)$ is known for $k = 0 : N-1$, $\ell = 0 : N-1$. Set

$$c_j(y_\ell) = \frac{1}{N}\sum_{k=0}^{N-1} f(x_k, y_\ell)e^{-ijx_k},$$

$$c_{j,k} = \frac{1}{N}\sum_{\ell=0}^{N-1} c_j(y_\ell)e^{-iky_\ell}.$$

From Theorem 4.6.4, then (with obvious changes in notations)

$$c_j(y_\ell) = \sum_{k=0}^{N-1} c_{j,k} e^{iky_\ell},$$

$$f(x_k, y_\ell) = \sum_{j=0}^{N-1} c_j(y_\ell) e^{ijx_k} = \sum_{j=0}^{N-1}\sum_{k=0}^{N-1} c_{j,k} e^{(ijx_k + iky_\ell)}.$$

The above expansion is of considerable importance in, e.g., crystallography.

The Fourier coefficients

$$c_j(f) = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)e^{ijx}\,dx, \quad j = 0, \pm 1, \pm 2, \ldots, \tag{4.6.23}$$

of a function $f$ with period $2\pi$ are often difficult to compute. On the other hand the coefficients of the DFT

$$\hat{c}_j(f) = \frac{1}{N}\sum_{k=0}^{N-1} f(x_k)e^{-ijx_k}, \quad x_k = \frac{2\pi k}{N}, \quad j = 0 : N-1, \tag{4.6.24}$$

can be computed very efficiently by the FFT. Now, since $f(x_0) = f(x_N)$, the sum in (4.6.24) can be thought of as a trapezoidal sum

$$\hat{c}_j(f) = \frac{1}{N}\left[\frac{1}{2}f(x_0) + f(x_1)e^{-ijx_1} + \cdots + f(x_{N-1})e^{-ijx_{N-1}} + \frac{1}{2}f(x_N)\right]$$

approximating the integral (4.6.23). Therefore, one might think of using $\hat{c}_j$ as an approximation to $c_j$ for all $j = 0, \pm 1, \pm 2, \ldots$. But $\hat{c}_j(f)$ are periodic in $j$ with period $N$, whereas by Theorem 4.6.3 the true Fourier coefficients $c_j(f)$ decay as some power $j^{-(k+1)}$ as $j \to \infty$.

We now show a way to remove this deficiency. Let $f_k$, $k = 0, \pm 1, \pm 2, \ldots$, be an infinite $N$-periodic sequence with $f_k = f(x_k)$. Let $\varphi = Pf$ be a continuous function such that $\varphi(x_k) = f_k$, $k = 0, \pm 1, \pm 2, \ldots$, and approximate $c_j(f)$ by $c_j(\varphi)$. Then $c_j(\varphi)$ will decay to zero as $j \to \infty$. It is a remarkable fact that if the approximation scheme $P$ is linear and translation invariant, we have

$$c_j(\varphi) = \tau_j \hat{c}_j(f), \tag{4.6.25}$$

where the **attenuation factors** $\tau_j$ *depend only on the approximation scheme* $P$ *and not the function* $f$. This implies that $\tau_j$ can be determined from (4.6.25) by evaluating $c_j(\varphi)$ and $\hat{c}_j(f)$ for some suitable sequence $f_k$, $k = 0, \pm 1, \pm 2, \ldots$. This allows the FFT to be used. For a proof of the above result and a more detailed exposition of attenuation factors in Fourier analysis, we refer to Gautschi [141].

**Example 4.6.2.**

For a given $N$-periodic sequence $f_k$, $k = 0, \pm 1, \pm 2, \ldots$, take $\varphi(x) = Pf$ to be defined as the piecewise linear function such that $\varphi(x_k) = f_k$, $k = 0, \pm 1, \pm 2, \ldots$. Clearly this approximation scheme is linear and translation invariant. Further, the function $\varphi$ is continuous and has period $2\pi$.

Consider in particular the sequence $f_k = 1, k = 0 \bmod N$, and $f_k = 0$ otherwise. We have

$$\hat{c}_j(f) = \frac{1}{N}\sum_{k=0}^{N-1} f(x_k)e^{-ijx_k} = \frac{1}{N}.$$

Further, setting $h = 2\pi/N$ and using symmetry, we get

$$\begin{aligned} c_j(Pf) &= \frac{1}{2\pi}\int_{-h}^{h}\left(1 - \frac{|x|}{h}\right)e^{ijx}\,dx \\ &= \frac{1}{\pi}\int_0^h \left(1 - \frac{x}{h}\right)\cos jx\,dx = \frac{2}{j^2\pi h}\sin^2\left(\frac{jh}{2}\right). \end{aligned}$$

This gives the attenuation factors

$$\tau_j = \left(\frac{\sin(j\pi/N)}{j\pi/N}\right)^2, \quad j = 0, \pm 1, \pm 2, \ldots.$$

Note that the coefficients $c_j(Pf)$ decay as $j^{-2}$, reflecting the fact that the first derivative of $Pf$ is discontinuous. If instead we use a cubic spline approximation, the first and second derivatives are continuous and the attenuation factors decay as $j^{-4}$.

### 4.6.3 Periodic Continuation of a Function

Assume now that the function $f$ is from the beginning defined only on the interval $[0, \pi]$. In that case we can extend the function to $[-\pi, 0]$ as either an even or an odd function. Hence the same function can be expanded as either a sine series or a cosine series. Both expansions will converge to the given $f(x)$ in the interval $[0, \pi]$ provided $f$ satisfies the conditions in Theorem 4.6.2. However, the rate of convergence of the two Fourier expansions may be vastly different depending on continuity properties at the points $x = 0$ and $x = \pi$ of the original interval.

If the function $f$ defined in $[0, \pi]$ satisfies $f(0) = f(\pi) = 0$, it can be continued as an odd function (see Figure 4.6.3). Its Fourier expansion then becomes a sine series. In the continuous case,

$$\sum_{j=1}^{\infty} b_j \sin jx, \qquad b_j = \frac{2}{\pi}\int_0^{\pi} f(x)\sin jx\,dx. \tag{4.6.26}$$

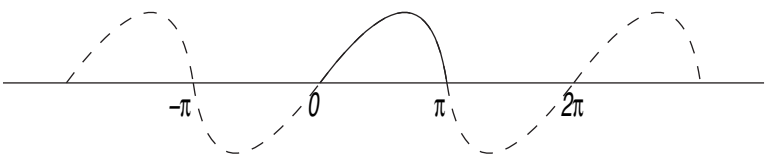



**Figure 4.6.3.** *Periodic continuation of a function $f$ outside $[0, \pi]$ as an odd function.*

In the discrete case ($x_k = \pi k/N$),

$$\sum_{j=1}^{N-1} b_j \sin jx, \qquad b_j = \frac{2}{N} \sum_{k=1}^{N-1} f(x_k) \sin jx_k. \tag{4.6.27}$$

The reflection as an odd function preserves the continuity in function values and the first derivative in the extended function. According to Theorem 4.6.3 the Fourier coefficients $b_j$ will therefore decrease as $j^{-3}$.

If $f(0) \neq 0$, one can still use such an expansion but the function will cause a discontinuity at $x = 0$ and $x = \pi$. The discontinuity at $x = 0$ follows from the reflection; the discontinuity at $x = \pi$ follows from the periodicity condition $f(x + 2\pi) = f(x)$ which gives $f(\pi) = f(-\pi)$. This will cause slow convergence (as $j^{-1}$) of the Fourier coefficients and cause undesirable oscillations due to Gibbs' phenomenon. However, by subtracting a linear function $a + bx$ from $f$ we can always ensure that the condition $f(0) = f(\pi) = 0$ is satisfied.

If $f(0) \neq 0$ and one makes a continuation of $f$ into an *even* function on $[-\pi, \pi]$, the extended function will be continuous at $x = 0$ and $x = \pi$. The Fourier series then becomes a pure cosine series In the continuous case,

$$\frac{1}{2}a_0 + \sum_{j=1}^{\infty} a_j \cos jx, \qquad a_j = \frac{2}{\pi} \int_0^{\pi} f(x) \cos jx \, dx. \tag{4.6.28}$$

In the discrete case ($x_k = \pi k/N$),

$$\frac{1}{2}a_0 + \sum_{j=1}^{N-1} a_j \cos jx, \qquad a_j = \frac{2}{N} \sum_{k=0}^{N-1} f(x_k) \cos jx_k. \tag{4.6.29}$$

The coefficients $a_j$ will decrease as $j^{-2}$. If $f'(0) = f'(\pi) = 0$, then the first derivative will also be continuous. One can still use such an expansion even if $f'(0) \neq 0$ or $f'(\pi) \neq 0$, but then a discontinuity appears in the first derivative.

### 4.6.4 Convergence Acceleration of Fourier Series

The generalized Euler transformation described in Sec. 3.4.3 can be used for accelerating the convergence of Fourier series, except in the immediate vicinity of singular points. Consider a complex power series

$$S(z) = \sum_{n=1}^{\infty} u_n z^{n-1}, \quad z = e^{i\phi}. \tag{4.6.30}$$

A Fourier series that is originally of the form $\sum_{n=-\infty}^{\infty} c_n e^{in\phi}$, or in trigonometric form, can easily be brought to this form; see Problem 4.6.7.

We consider the case

$$S(z) = -\frac{1}{z} \log(1 - z) = \sum_{n=1}^{\infty} \frac{1}{n} z^{n-1}, \tag{4.6.31}$$

which is typical for a power series with completely monotonic terms. (The rates of convergence are the same for almost all series of this class.) Numerical computation, essentially by the above algorithm, gave the following results. The coefficients $u_j$ are computed in IEEE double precision arithmetic. We make the rounding errors during the computations less important by subtracting the first row of partial sums by its last element; it is, of course, added again to the final result.[159] The first table shows, for various $\phi$, the most accurate result that can be obtained without thinning. These limits are due to the rounding errors; we can make the pure truncation error arbitrarily small by choosing $N$ large enough.

| $\phi$ | $\pi$ | $2\pi/3$ | $\pi/2$ | $\pi/3$ | $\pi/4$ | $\pi/8$ | $\pi/12$ | $\pi/180$ |
|---|---|---|---|---|---|---|---|---|
| \|error\| | $2\cdot10^{-16}$ | $8\cdot10^{-16}$ | $10^{-14}$ | $6\cdot10^{-12}$ | $10^{-9}$ | $5\cdot10^{-7}$ | $3\cdot10^{-5}$ | $2\cdot10^{-1}$ |
| $N$ | 30 | 33 | 36 | 36 | 36 | 40 | 40 | 100 |
| $kk$ | 21 | 22 | 20 | 21 | 20 | 13 | 10 | (3) |

| $\tau$ | 80 | 120 | 90 | 15 |
|---|---|---|---|---|
| $\tau\cdot\phi$ | $\pi$ | $2\pi/3$ | $\pi/2$ | $\pi/12$ |
| \|error\| | $2\cdot10^{-14}$ | $10^{-14}$ | $3\cdot10^{-13}$ | $3\cdot10^{-5}$ |
| $N$ | 28 | 31 | 33 | 41 |
| $kk$ | 20 | 22 | 18 | 10 |
| no. terms | 5040 | 3720 | 2970 | 615 |

Note that a rather good accuracy is also obtained for $\phi = \pi/8$ and $\phi = \pi/12$, where the algorithm is "unstable," since $|\frac{z}{1-z}| > 1$. In this kind of computation "instability" does not mean that the algorithm is hopeless, but it shows the importance of a good termination criterion. The question is to navigate safely between Scylla and Charybdis. For a small value such as $\phi = \pi/180$, the sum is approximately $4.1 + 1.5i$. The smallest error with 100 terms (or less) is 0.02; it is obtained for $k = 3$. Also note that $kk/N$ increases with $\phi$.

By the application of thinning the results can often be improved considerably for $\phi \ll \pi$, in particular for $\phi = \pi/180$. Let $\tau$ be a positive integer. The thinned form of $S(z)$ reads

$$S(z) = \sum_{p=1}^{\infty} u_p^* z^{\tau\cdot(p-1)}, \quad u_p^* = \sum_{j=1}^{\tau} u_{j+\tau\cdot(p-1)}\, z^{j-1}.$$

The series (4.6.31) has "essentially positive" terms originally that can become "essentially alternating" by thinning. For example, if $z = e^{i\pi/3}$ and $\tau = 3$, the series becomes an alternating series, perhaps with complex coefficients. It does not matter in the numerical work that $u_p^*$ depends on $z$.

We present the errors obtained for four values of the parameter $\tau$, with different amounts of work. Compare $|error|$, $kk$, etc. with appropriate values in the table above. We see that, by thinning, it is possible to calculate the Fourier series very accurately for small values of $\phi$ also.

[159] Tricks like this can often be applied in linear computations with a slowly varying sequence of numbers. See, for example, the discussion of rounding errors in Richardson extrapolation in Sec. 3.4.6.

Roughly speaking, the optimal rate of convergence of the Euler transformation depends on $z$ in the same way for all power series with completely monotonic coefficients, independently of the rate of convergence of the original series. The above tables from a particular example can therefore—with some safety margin—be used as a guide for the application of the Euler transformation with thinning to any series of this class.

Say that you want the sum of a series $\sum u_n z^n$ for $z = e^{i\phi}$, $\phi = \pi/12$, with relative $|error| < 10^{-10}$. You see in the first table that $|error| = 6 \cdot 10^{-12}$ for $\phi = \pi/3 = 4\pi/12$ without thinning. The safety margin is, we hope, large enough. Therefore, try $\tau = 4$. We make two tests with completely monotonic terms: $u_n = n^{-1}$ and $u_n = \exp(-\sqrt{n})$. We hope that tol $= 10^{-10}$ is large enough to make the irregular errors relatively negligible. In both tests the actual magnitude of the error turns out to be $4 \cdot 10^{-11}$, and the total number of terms is $4 \cdot 32 = 128$. The values of *errest* are $6 \cdot 10^{-11}$ and $7 \cdot 10^{-11}$; both slightly overestimate the actual errors and are still smaller than tol.

### 4.6.5 The Fourier Integral Theorem

We have seen how Fourier methods can be used on functions defined on a finite interval usually taken to be $[-\pi, \pi]$. Fourier found that expansion of an arbitrary function in a Fourier series remains possible even if the function is defined on an interval that extends on both sides to infinity. In this case the fundamental frequency converges to zero and the summation process changes into one of integration.

Suppose that the function $f(x)$ is defined on the entire real axis, and that it satisfies the regularity properties which we required in Theorem 4.6.2. Set

$$\varphi(\xi) = f(x), \quad \xi = 2\pi x/L \in [-\pi, \pi],$$

and continue $\varphi(\xi)$ outside $[-\pi, \pi]$ so that it has period $2\pi$. By Theorem 4.6.2, if

$$c_j = \frac{1}{2\pi} \int_{-\pi}^{\pi} \varphi(\xi) e^{-ij\xi}\, d\xi = \frac{1}{L} \int_{-L/2}^{L/2} f(x) e^{-2\pi i x j/L}\, dx, \tag{4.6.32}$$

then $\varphi(\xi) = \sum_{j=-\infty}^{\infty} c_j e^{ij\xi}$, $\xi \in (-\pi, \pi)$, and hence

$$f(x) = \sum_{j=-\infty}^{\infty} c_j e^{2\pi i x j/L}, \quad x \in \left(-\frac{L}{2}, \frac{L}{2}\right).$$

If we set

$$g_L(\omega) = \int_{-L/2}^{L/2} f(x) e^{-2\pi i x \omega}\, dx, \quad \omega = \frac{j}{L}, \tag{4.6.33}$$

then by (4.6.32) we have $c_j = (1/L) g_L(\omega)$, and hence

$$f(x) = \frac{1}{L} \sum_{j=-\infty}^{\infty} g_L(\omega) e^{2\pi i x \omega}, \quad x \in \left(\frac{-L}{2}, \frac{L}{2}\right). \tag{4.6.34}$$

Now by passing to the limit $L \to \infty$, one avoids making an artificial periodic continuation outside a finite interval. The sum in (4.6.34) is a "sum of rectangles" similar to the sum which appears in the definition of a definite integral. But here the argument varies from

$-\infty$ to $+\infty$, and the function $g_L(t)$ depends on $L$. By a somewhat dubious passage to the limit, then, the pair of formulas (4.6.33) and (4.6.34) becomes the pair

$$g(\omega) = \int_{-\infty}^{\infty} f(x)e^{-2\pi i x\omega}\,dx \iff f(x) = \int_{-\infty}^{\infty} g(\omega)e^{2\pi i x\omega}\,d\omega. \tag{4.6.35}$$

One can, in fact, after a rather complicated analysis, show that the above result is correct; see, e.g., Courant–Hilbert [83]. The proof requires, besides the previously mentioned "local" regularity conditions on $f$, the "global" assumption that

$$\int_{-\infty}^{\infty} |f(x)|\,dx$$

is convergent. The beautiful, almost symmetric relation of (4.6.35) is called the **Fourier integral theorem**. This theorem, and other versions of it, with varying assumptions under which they are valid, is one of the most important aids in both pure and applied mathematics. The function $g$ is called the **Fourier transform**[160] of $f$. The Fourier transform is one of the most important tools of applied analysis. It plays a fundamental role in problems relating to input–output relations, e.g., in electrical networks.

Clearly the Fourier transform is a *linear operator*. Another elementary property that can easily be verified is

$$f(ax) \iff \frac{1}{|a|}g\left(\frac{\omega}{a}\right).$$

Whether the function $f(x)$ has even or odd symmetry and is real or purely imaginary leads to relations between $g(\omega)$ and $g(-\omega)$ that can be used to increase computational efficiency. Some of these properties are summarized in Table 4.6.1.

**Table 4.6.1.** *Useful symmetry properties of the continuous Fourier transform.*

| Function | Fourier transform |
|---|---|
| $f(x)$ real | $g(-\omega) = \overline{g}(\omega)$ |
| $f(x)$ imaginary | $g(-\omega) = -\overline{g}(\omega)$ |
| $f(x)$ even | $g(-\omega) = g(\omega)$ |
| $f(x)$ odd | $g(-\omega) = -g(\omega)$ |
| $f(x)$ real even | $g(\omega)$ real even |
| $f(x)$ imaginary odd | $g(\omega)$ real odd |

**Example 4.6.3.**

The function $f(x) = e^{-|x|}$ has Fourier transform

$$g(\omega) = \int_{-\infty}^{\infty} e^{-|x|}e^{-2\pi i x\omega}\,dx = \int_0^{\infty} \left(e^{-(1+2\pi i\omega)x} + e^{-(1-2\pi i\omega)x}\right) dx$$
$$= \frac{1}{1+2\pi i\omega} + \frac{1}{1-2\pi i\omega} = \frac{2}{1+4\pi^2\omega^2}.$$

[160] The terminology in the literature varies somewhat as to the placement of the factor $2\pi$; it can be taken out of the exponent by a simple change of variable.

Here $f(x)$ is real and an even function. In agreement with Table 4.6.1, the Fourier transform is also real and even.

From (4.6.35) it follows that

$$e^{-|x|} = \int_{-\infty}^{\infty} \frac{2}{1+4\pi^2\omega^2} e^{2\pi i x\omega}\, d\omega = \frac{2}{\pi}\int_0^{\infty} \frac{1}{1+x^2}\cos \pi x\, dx, \quad (2\pi\omega = x).$$

It is not so easy to prove this formula directly.

Many applications of the Fourier transform involve the use of convolutions.

**Definition 4.6.5.**

*The* **convolution** *of $f_1$ and $f_2$ is the function*

$$h(\xi) = \text{conv}\,(f_1, f_2) = \int_{-\infty}^{\infty} f_1(x) f_2(\xi - x)\, dx. \tag{4.6.36}$$

It is not difficult to verify that $\text{conv}\,(f_1, f_2) = \text{conv}\,(f_2, f_1)$. The following theorem states that the convolution of $f_1$ and $f_2$ can be computed as the inverse Fourier transform of the product $g_1(\omega)g_2(\omega)$. This fact is of great importance in the application of Fourier analysis to differential equations and probability theory.

**Theorem 4.6.6.**

*Let $f_1$ and $f_2$ have Fourier transforms $g_1$ and $g_2$, respectively. Then the Fourier transform $g$ of the convolution of $f_1$ and $f_2$ is the product $g(\omega) = g_1(\omega)g_2(\omega)$.*

***Proof.*** By definition the Fourier transform of the convolution is

$$\begin{aligned} g(\omega) &= \int_{-\infty}^{\infty} e^{-2\pi i\xi\omega} \left( \int_{-\infty}^{\infty} f_1(x) f_2(\xi - x)\, dx \right) d\xi \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-2\pi i(x+\xi-x)\omega} f_1(x) f_2(\xi - x)\, dx\, d\xi \\ &= \int_{-\infty}^{\infty} e^{-2\pi i x\omega} f_1(x)\, dx \int_{-\infty}^{\infty} e^{-2\pi i(\xi-x)\omega} f_2(\xi - x)\, d\xi \\ &= \int_{-\infty}^{\infty} e^{-2\pi i x\omega} f_1(\xi)\, dx \int_{-\infty}^{\infty} e^{-2\pi i x\omega} f_2(x)\, dx = g_1(\omega)g_2(\omega). \end{aligned}$$

The legitimacy of changing the order of integration is here taken for granted. □

In many physical applications, the following relation, analogous to Parseval's identity (corollary to Theorem 4.5.14), is of great importance. If $g$ is the Fourier transform of $f$, then

$$\int_{-\infty}^{\infty} |g(\omega)|^2\, d\omega = \int_{-\infty}^{\infty} |f(\xi)|^2\, d\xi. \tag{4.6.37}$$

In signal processing this can be interpreted to mean that the total power in a signal is the same whether computed in the time domain or the frequency domain.

### 4.6.6 Sampled Data and Aliasing

The ideas of Sec. 4.3.5 can be applied to the derivation of the celebrated **sampling theorem**. This is an interpolation formula that expresses a function that is **band-limited** to the frequency interval $[-W,\ W]$. Such a function has a Fourier representation of the form (see also Strang [339, p. 325])

$$f(z) = \frac{1}{2\pi}\int_{-W}^{W} \hat{f}(k)e^{ikz}\,dk, \quad |\hat{f}(k)| \le M, \tag{4.6.38}$$

in terms of its values at all integer points.

**Theorem 4.6.7** (*Shannon's Sampling Theorem*).
*Let the function be band-limited to the frequency interval* $[-W,\ W]$. *Then*

$$f(z) = \sum_{j=-\infty}^{\infty} f\left(\frac{j\pi}{W}\right)\frac{\sin(Wz - j\pi)}{(Wz - j\pi)}. \tag{4.6.39}$$

***Proof.*** We shall sketch a derivation of this for $W = \pi$. (Strang [339] gives an entirely different derivation, based on Fourier analysis.) We first note that (4.6.38) shows that $f(z)$ is analytic for all $z$. Then we consider the same Cauchy integral as many times before,

$$I_n(u) = \frac{1}{2\pi i}\int_{\partial\mathcal{D}_n} \frac{\Phi(u)f(z)}{\Phi(z)(z-u)}\,dz, \quad u \in \mathcal{D}_n.$$

Here $\Phi(z) = \sin \pi z$, which vanishes at all integer points, and $\mathcal{D}_n$ is the *open* rectangle with vertices at $\pm(n + 1/2) \pm bi$. By the residue theorem, we obtain after a short calculation

$$I_n(u) = f(u) + \sum_{j=-n}^{n} \frac{\Phi(u)f(j)}{\Phi'(j)(j-u)} = f(u) - \sum_{j=-n}^{n} \frac{f(j)\sin\pi(j-u)}{\pi(j-u)}.$$

Set $z = x + iy$. Note that

$$|f(z)| \le \frac{1}{2\pi}\int_{-\pi}^{\pi} Me^{-ky}dk \le \frac{M(e^{|\pi y|} - e^{-|\pi y|})}{|2\pi y|}, \quad |\Phi(z)| \ge e^{|\pi y|}.$$

These inequalities, applied for $y = b$, allow us to let $b \to \infty$ ($2b$ is the height of the symmetric rectangular contour). Then it can be shown that $I_n(u) \to 0$ as $n \to \infty$, which establishes the sampling theorem for $W = \pi$. The general result is then obtained by "regula de tri,"[161] but it is sometimes hard to get it right. □

Note the similarity of (4.6.39) to Lagrange's interpolation formula. Like Lagrange's, it is a so-called *cardinal interpolation formula*. As $Wz/\pi$ tends to an integer $m$, all terms except one on the right-hand side become zero; for $j = m$ the term becomes $f(m\pi/W)$.

[161] This rule from Euclid's fifth book of *Elementa* tells us how, given three out of four proportional quantities $a/b = c/d$, one determines the fourth. This name is used in elementary mathematics, e.g., in Germany and Sweden, but does not seem to be known under the same name in English-speaking countries.

Let $f$ be a function which is zero outside the interval $[0, L]$. Its Fourier transform is then given by

$$g(\omega) = \int_0^L f(x)e^{-2\pi i\omega x}\,dx. \tag{4.6.40}$$

We want to approximate $g(\omega)$ using values of $f(x)$ sampled at intervals $\Delta x$,

$$f_j = f(j\Delta x), \quad 0 < j < N-1, \quad L = N\Delta x.$$

The integral (4.6.40) can be approximated by

$$g(\omega) \approx \frac{L}{N}\sum_{j=0}^{N-1} f_j e^{-2\pi i\omega j\Delta x}. \tag{4.6.41}$$

Since only $N$ values of $f_j$ are used as input and we want the computed values to be linearly independent, we cannot approximate $g(\omega)$ at more than $N$ points. The wave of lowest frequency associated with the interval $[0, L]$ is $\omega = 1/L = 1/(N\Delta x)$, since then $[0, L]$ corresponds to one full period of the wave. We therefore choose points $\omega_k = k\Delta\omega$, $k = 0 : N$ in the frequency space such that the following **reciprocity relations** hold:

$$LW = N, \qquad \Delta x\Delta\omega = 1/N. \tag{4.6.42}$$

With this choice it holds that

$$W = N\Delta\omega = 1/\Delta x, \qquad L = N\Delta x = 1/\Delta\omega. \tag{4.6.43}$$

Noting that $(j\Delta x)(k\Delta\omega) = jk/N$ we get from the trapezoidal approximation

$$g(\omega_k) \approx \frac{L}{N}\sum_{j=0}^{N-1} f_j e^{-2\pi ikj/N} = Lc_k, \quad k = 0 : N-1,$$

where $c_k$ is the coefficient of the DFT.

The frequency $\omega_c = 1/(2\Delta x) = W/2$ is the so-called **Nyquist's critical frequency**. Sampling the wave $\sin(2\pi\omega_c x)$ with sampling interval $\Delta x$ will sample exactly *two points per cycle*. It is a remarkable fact that if a function $f(x)$, defined on $[-\infty, \infty]$, is *band-width limited* to frequencies smaller than or equal to $\omega_c$, then $f(x)$ is completely determined by its sample values $j\Delta x$, $\infty \le j \le \infty$; see Shannon's sampling theorem, our Theorem 4.6.7.

If the function is not band-width limited the spectral density outside the critical frequency is moved into that range. This is called **aliasing**. The relationship between the Fourier transform $g(\omega)$ and the DFT of a finite sampled representation can be characterized as follows. Assuming that the reciprocity relations (4.6.42) are satisfied, the DFT of $f_j = f(j\Delta x)$, $0 \le j < N$, will approximate the periodic aliased function

$$\tilde{g}_k = \tilde{g}(k\Delta\omega), \quad 0 \le j < N, \tag{4.6.44}$$

where

$$\tilde{g}(\omega) = g(\omega) + \sum_{k=1}^{\infty}\left(g(\omega + kW) + g(\omega - kW)\right), \quad \omega \in [0, W]. \tag{4.6.45}$$

Since by (4.6.43) $W = 1/\Delta x$, we can *increase* the frequency range $[0, W]$ covered by *decreasing* $\Delta x$.

**Example 4.6.4.**

The function $f(x) = e^{-x}$, $x > 0$, $f(x) = 0$, $x < 0$, has Fourier transform

$$g(\omega) = \frac{1}{1 + 2\pi i\omega} = \frac{1 - i2\pi\omega}{1 + 4\pi^2\omega^2}$$

(cf. Example 4.6.3). Set $f(0) = 1/2$, the average of $f(-0)$ and $f(+0)$, which is the value given by the inverse Fourier transform at a discontinuity.

Set $N = 32$, $T = 8$, and sample the $f$ in the interval $[0, T]$ at equidistant points $j\Delta t$, $j = 0 : N - 1$. Note that $T$ is so large that the aliased function (4.6.44) is nearly equal to $f$. This sampling rate corresponds to $\Delta t = 8/32 = 1/4$ and $W = 4$.

The effect of aliasing in the frequency domain is evident. The error is significant for frequencies larger than the critical frequency $W/2$. To increase the accuracy $W$ can be increased by decreasing the sampling interval $\Delta x$; see Figure 4.6.4.

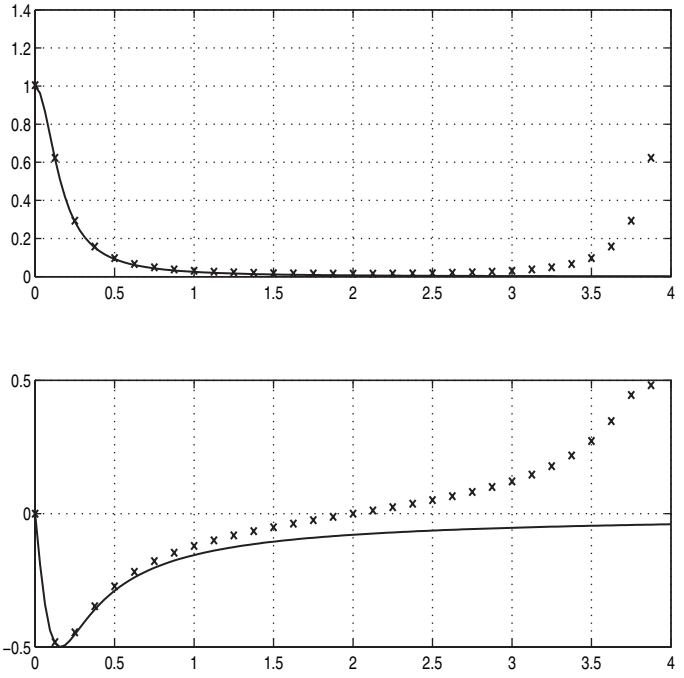



**Figure 4.6.4.** *The real (top) and imaginary (bottom) parts of the Fourier transform (solid line) of $e^{-x}$ and the corresponding DFT (dots) with $N = 32$, $T = 8$.*

## Review Questions

**4.6.1** Derive the orthogonality properties and coefficient formulas which are fundamental to Fourier analysis, for both the continuous and the discrete case.

**4.6.2** Give sufficient conditions for the Fourier series of the function $f$ to be everywhere convergent to $f$. To what value will the Fourier series converge at a point $x = a$ of discontinuity of $f$?

**4.6.3** How can the Fourier expansion be generalized to a function $f(x, y)$ of two variables?

**4.6.4** (a) Give two ways in which a real function $f$ defined on the interval $[0, \pi]$ can be extended to a periodic function.

(b) What disadvantage (for Fourier analysis) is incurred if the periodic continuation has a discontinuity—for example, in its derivative at the end points of 80, $\pi$9?

**4.6.5** Describe how the behavior of the coefficients of the discrete Fourier expansion can be modified to improve the approximation of the corresponding continuous Fourier expansion.

**4.6.6** Formulate the Fourier integral theorem.

**4.6.7** What is meant by aliasing, when approximating the Fourier transform by a discrete transform?

## Problems and Computer Exercises

**4.6.1** Give a simple characterization of the functions which have a sine expansion containing odd terms only.

**4.6.2** Let $f$ be an even function, with period $2\pi$, such that

$$f(x) = \pi - x, \quad 0 \le x \le \pi.$$

(a) Plot the function $y = f(x)$ for $-3\pi \le x \le 3\pi$. Expand $f$ in a Fourier series.

(b) Use this series to show that $1 + 3^{-2} + 5^{-2} + 7^{-2} + \cdots = \pi^2/8$.

(c) Compute the sum $1 + 2^{-2} + 3^{-2} + 4^{-2} + 5^{-2} + \cdots$.

(d) Compute, using (4.6.9), the sum $1 + 3^{-4} + 5^{-4} + 7^{-4} + \cdots$.

(e) Differentiate the Fourier series term by term, and compare with the result for the rectangular wave in Sec. 4.6.1.

**4.6.3** (a) Show that the function $G_1(t) = t - 1/2$, $0 < t < 1$, has the expansion

$$G_1(t) = -\sum_{n=1}^{\infty} \frac{\sin 2n\pi t}{n\pi}.$$

(b) Let $G_1(t)$ be as in (a) and consider a sequence of functions such that $G'_{p+1}(t) = G_p(t)$, $p = 1, 2, \ldots$. Derive by termwise integration the expansion for the functions

$G_p(t)$, and show that $c_p - G_p(t)$ has the same sign as $c_p$. Show also that for $p$ even

$$\sum_{n=1}^{\infty} n^{-p} = \frac{1}{2}|c_p|(2\pi)^p.$$

**4.6.4** (a) Using that $\sin x$ is the imaginary part of $e^{ix}$, prove that

$$\sum_{k=1}^{N-1} \sin \frac{\pi k}{N} = \cot \frac{\pi}{2N}.$$

(b) Determine a sine polynomial $\sum_{j=1}^{n-1} b_j \sin jx$, which takes on the value 1 at the points $x_k = \pi k/n$, $k = 1 : n-1$.

*Hint:* Use (4.6.26) or recall that the sine polynomial is an odd function.

(c) Compare the limiting value for $b_j$ as $n \to \infty$ with the result in Example 4.6.1.

**4.6.5** (a) Prove the inequality in (4.6.10).

(b) Show, under the assumptions on $f$ which hold in (4.6.10), that for $k \geq 1$, $f$ can be approximated by a trigonometric polynomial such that

$$\left\| f - \sum_{j=-n}^{n} c_j e^{ijx} \right\|_\infty < \frac{2}{kn^k} \|f^{(k+1)}\|_\infty.$$

**4.6.6** (a) Fourier approximations to the rectangular wave, $f(x) = 1$, $0 < x < \pi$, $f(x) = -1$, $-\pi < x < 0$, are shown in Figure 4.6.2 for one, two, five, and ten terms. Plot and compare the corresponding smoothed approximations when the $\sigma$-factors in (4.6.15) have been applied.

(b) The delta function $\delta(x)$ is defined as being zero everywhere except between the limits $\pm\epsilon$, where $\epsilon$ tends to zero. At $x = 0$ the function goes to infinity in such a way that the area under the function equals one. The formal Fourier expansion of $\delta(x)$ yields

$$f(x) = \frac{1}{\pi}\left(\frac{1}{2} + \cos x + \cos 2x + \cos 3x + \cdots\right),$$

which does not converge anywhere. If the $\sigma$-factors in (4.6.15) are applied, we obtain the expansion

$$y_m(x) = \frac{1}{\pi}\left(\frac{1}{2} + \sigma_1 \cos x + \sigma_2 \cos 2x + \cdots + \sigma_{m-1} \cos(m-1)x\right), \quad (4.6.46)$$

which can be considered the trigonometric representation of the delta function. Plot $y_m(x)$, $-6 \leq x \leq 6$, for several values of $m$.

**4.6.7** (a) Consider a *real* function with the Fourier expansion

$$F(\phi) = \sum_{n=-\infty}^{\infty} c_n e^{in\phi}.$$

Show that this rewritten for convergence acceleration with the generalized Euler's method is

$$F(\phi) = c_0 + 2\Re \sum_{n=1}^{\infty} c_n z^n, \quad z = e^{i\phi}.$$

*Hint:* Show that $c_{-n} = \bar{c}_n$.

(b) Set $c_n = a_n - ib_n$, where $a_n, b_n$ are real. Show that

$$\sum_{n=0}^{\infty} (a_n \cos n\phi + b_n \sin n\phi) = \Re \sum_{n=0}^{\infty} c_n z^n.$$

(c) How would you rewrite the Chebyshev series $\sum_{n=0}^{\infty} T_n(x)/(1+n^2)$?

(d) Consider also how to handle a complex function $F(\phi)$.

*In the following problems, we do not require any investigation of whether it is permissible to change the order of summations, integrations, differentiations; it is sufficient to treat the problems in a purely formal way.*

**4.6.8** The partial differential equation $\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$ is called the heat equation. Show that the function

$$u(x,t) = \frac{4}{\pi} \sum_{k=0}^{\infty} \frac{\sin(2k+1)x}{2k+1} e^{-(2k+1)^2 t}$$

satisfies the differential equation for $t > 0$, $0 < x < \pi$, with boundary conditions $u(0,t) = u(\pi,t) = 0$ for $t > 0$, and initial condition $u(x,0) = 1$ for $0 < x < \pi$ (see Example 4.6.1).

**4.6.9** Show that if $g(t)$ is the Fourier transform of $f(x)$, then

(a) $e^{2\pi kt} g(t)$ is the Fourier transform of $f(x+k)$.

(b) $(2\pi it)^k g(t)$ is the Fourier transform of $f^{(k)}(x)$, assuming that $f(x)$ and its derivatives up to the $k$th order tend to zero as $x \to \infty$.

**4.6.10** The **correlation** of $f_1(x)$ and $f_2(x)$ is defined by

$$c(\xi) = \int_{-\infty}^{\infty} f_1(x+\xi) f_2(x)\, dx. \tag{4.6.47}$$

Show that if $f_1(x)$ and $f_2(x)$ have Fourier transforms $g_1(t)$ and $g_2(t)$, respectively, then the Fourier transform of $c(\xi)$ is $h(t) = g_1(t) g_2(-t)$.

*Hint:* Compare Theorem 4.6.6.

**4.6.11** (a) Work out the details of the proof of the sampling theorem.

(b) The formulation of the sampling theorem with a general $W$ in Strang [339] does not agree with ours in (4.6.39). Who is right?

**4.6.12** Suppose that $f(z)$ satisfies the assumptions of our treatment of the sampling theorem for $W = \pi/h$. Show by integration term by term that

$$\lim_{R\to\infty} \int_{-R}^{R} f(u)\, du = h \lim_{n\to\infty} \sum_{j=-n}^{n} f(jh).$$

*Hint:* Use the classical formula

$$\int_{-\infty}^{\infty} \frac{\sin x}{x}\, dx = \pi.$$

Full rigor is not necessary.

## 4.7 The Fast Fourier Transform

### 4.7.1 The FFT Algorithm

Consider the discrete Fourier transform (DFT)

$$f(x) = \sum_{j=0}^{N-1} c_j e^{ijx}$$

of a function, whose values $f(x_k)$ are known at the grid points $x_k = 2\pi k/N, k = 0 : N-1$. According to Theorem 4.6.4 the coefficients are given by

$$c_j = \frac{1}{N} \sum_{k=0}^{N-1} f(x_k) e^{-ijx_k}, \quad j = 0 : N-1. \tag{4.7.1}$$

The evaluation of expressions of the form (4.7.1) occur also in discrete approximations to the Fourier transform.

Setting $\omega_N = e^{-2\pi i/N}$ this becomes

$$c_j = \frac{1}{N} \sum_{k=0}^{N-1} \omega_N^{jk} f(x_k), \quad j = 0 : N-1, \tag{4.7.2}$$

where $\omega_N$ is an $N$th root of unity, $(\omega_N)^N = 1$. It seems from (4.7.2) that computing the discrete Fourier coefficients would require $N^2$ complex multiplications and additions. As we shall see, only about $N \log_2 N$ complex multiplications and additions are required using an algorithm called the **fast Fourier transform** (**FFT**).

The modern usage of the FFT started in 1965 with the publication of [78] by James W. Cooley of IBM Research and John W. Tukey, Princeton University.[162] In many areas of application (digital signal processing, image processing, time-series analysis, to name a few), the FFT has caused a complete change of attitude toward what can be done using discrete Fourier methods. Without the FFT many modern devices such as cell phones, digital cameras, CT scanners, and DVDs would not be possible. Some applications considered in astronomy require FFTs of several gigapoints.

[162] Tukey came up with the basic algorithm at a meeting of President Kennedy's Science Advisory Committee. One problem discussed at this meeting was that the ratification of a US–Soviet nuclear test ban depended on a fast method to detect nuclear tests by analyzing seismological time-series data.

In the following we will use the common convention *not* to scale the sum in (4.7.2) by $1/N$.

**Definition 4.7.1.**
*The* DFT *of the vector* $f \in \mathbf{C}^N$ *is*

$$y = F_N f, \tag{4.7.3}$$

*where* $F_n \in \mathbf{C}^{N\times N}$ *is the* **DFT matrix** *with elements*

$$(F_N)_{jk} = \omega_N^{jk}, \quad j,k = 0:N-1, \tag{4.7.4}$$

*where* $\omega_N = e^{-2\pi i/N}$.[163]

From the definition it follows that the DFT matrix $F_N$ is a complex Vandermonde matrix. Since $\omega_N^{jk} = \omega_N^{kj}$, $F_N$ is symmetric. By Theorem 4.6.4

$$\frac{1}{N} F_N^H F_N = I,$$

where $F_N^H$ is the complex conjugate transpose of $F_N$. Hence the matrix $\frac{1}{\sqrt{N}} F_N$ is a unitary matrix and the inverse transform can be written as

$$f = \frac{1}{N} F_N^H y.$$

We now describe the central idea of the FFT algorithm, which is based on the divide and conquer strategy (see Sec. 1.2.3). Assume that $N = 2^p$ and set

$$k = \begin{cases} 2k_1 & \text{if } k \text{ is even,} \\ 2k_1+1 & \text{if } k \text{ is odd,} \end{cases} \quad 0 \le k_1 \le m-1,$$

where $m = N/2 = 2^{p-1}$. Split the DFT sum into an even and an odd part:

$$y_j = \sum_{k_1=0}^{m-1} (\omega_N^2)^{jk_1} f_{2k_1} + \omega_N^j \sum_{k_1=0}^{m-1} (\omega_N^2)^{jk_1} f_{2k_1+1}, \quad j = 0:N-1.$$

Let $\beta$ be the quotient and $j_1$ the remainder when $j$ is divided by $m$, i.e., $j = \beta m + j_1$. Then, since $\omega_N^N = 1$,

$$(\omega_N^2)^{jk_1} = (\omega_N^2)^{\beta m k_1} (\omega_N^2)^{j_1 k_1} = (\omega_N^N)^{\beta k_1} (\omega_N^2)^{j_1 k_1} = \omega_m^{j_1 k_1}.$$

Thus if, for $j_1 = 0 : m-1$, we set

$$\phi_{j_1} = \sum_{k_1=0}^{m-1} f_{2k_1} \omega_m^{j_1 k_1}, \qquad \psi_{j_1} = \sum_{k_1=0}^{m-1} f_{2k_1+1} \omega_m^{j_1 k_1}, \tag{4.7.5}$$

[163] Some authors set $\omega_N = e^{2\pi i/N}$. Which convention is used does not much affect the development.

then $y_j = \phi_{j_1} + \omega_N^j \psi_{j_1}$. The two sums on the right are elements of the DFTs of length $N/2$ applied to the parts of $f$ with odd and even subscripts. *The entire DFT of length N is obtained by combining these two DFTs!* Since $\omega_N^m = -1$, we have

$$y_{j_1} = \phi_{j_1} + \omega_N^{j_1} \psi_{j_1}, \tag{4.7.6}$$

$$y_{j_1+N/2} = \phi_{j_1} - \omega_N^{j_1} \psi_{j_1}, \quad j_1 = 0 : N/2 - 1. \tag{4.7.7}$$

These expressions, noted already by Danielson and Lanczos [90], are often called **butterfly relations** because of the data flow pattern. Note that these can be performed in place, i.e., no extra vector storage is needed.

The computation of $\phi_{j_1}$ and $\psi_{j_1}$ means that one does *two* Fourier transforms with $m = N/2$ terms instead of one with $N$ terms. If $N/2$ is even the same idea can be applied to these two Fourier transforms. One then gets *four* Fourier transforms, each of which has $N/4$ terms. If $N = 2^p$, this reduction can be continued recursively until we get $N$ DFTs with one term. But $F_1 = I$, the identity. A recursive MATLAB implementation of the FFT algorithm is given in Problem 4.7.2.

**Example 4.7.1.**

For $n = 2^2 = 4$, we have $\omega_4 = e^{-\pi i/2} = -i$, and the DFT matrix is

$$F_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & (-i)^2 & (-i)^3 \\ 1 & (-i)^2 & (-i)^4 & (-i)^6 \\ 1 & (-i)^3 & (-i)^6 & (-i)^9 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix}. \tag{4.7.8}$$

It is symmetric and its inverse is

$$F_4^{-1} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}.$$

The number of complex operations (one multiplication and one addition) required to compute $\{y_j\}$ from the butterfly relations when $\{\phi_{j_1}\}$ and $\{\psi_{j_1}\}$ have been computed is $2^p$, assuming that the powers of $\omega$ are precomputed and stored. Thus, if we denote by $q_p$ the total number of operations needed to compute the DFT when $N = 2^p$, we have

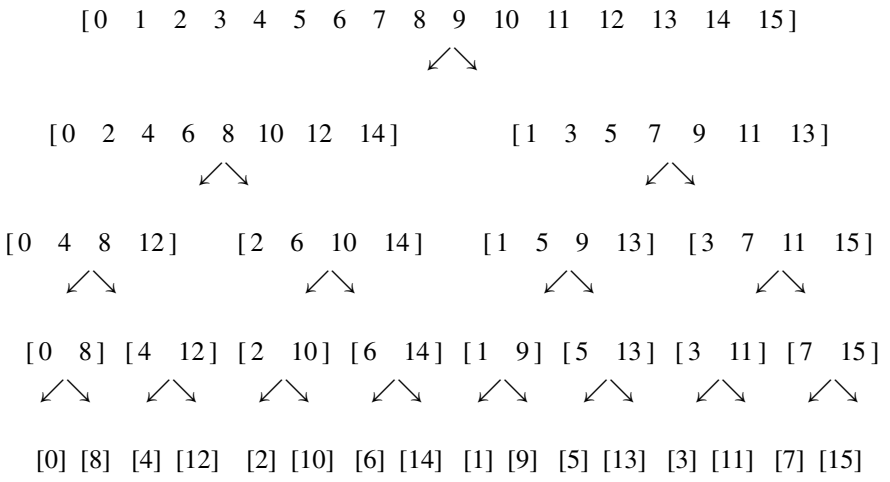
$$q_p \le 2q_{p-1} + 2^p, \quad p \ge 1.$$

Since $q_0 = 0$, it follows by induction that $q_p \le p \cdot 2^p = N \cdot \log_2 N$. *Hence, when N is a power of* 2, *the FFT solves the problem with at most* $N \cdot \log_2 N$ *operations*.

For example, when $N = 2^{20} = 1{,}048{,}576$ the FFT algorithm is theoretically a factor of 84,000 faster than the "conventional" $O(N^2)$ algorithm. On a 3 GHz laptop, a real FFT of this size takes about 0.1 second using MATLAB 6, whereas more than two hours would be required by the conventional algorithm! The FFT not only uses fewer operations to evaluate the DFT, it also is more accurate. Whereas when using the conventional method the roundoff error is proportional to $N$, for the FFT algorithm it is proportional to $\log_2 N$.

**Example 4.7.2.**

Let $N = 2^4 = 16$. Then the 16-point DFT (0:1:15) can be split into two 8-point DFTs (0:2:14) and (1:2:15), which can each be split in two 4-point DFTs. Repeating these splittings we finally get 16 one-point DFTs which are the identity $F_1 = 1$. The structure of this FFT is illustrated below.




In most implementations the explicit recursion is avoided. Instead the FFT algorithm is implemented in two stages:

- a reordering stage in which the data vector $f$ is permuted;
- a second stage in which first $N/2$ FFT transforms of length 2 are computed on adjacent elements, followed by $N/4$ transforms of length 4, etc., until the final result is obtained by merging two FFTs of length $N/2$.

We now consider each stage in turn.

Each step of the recursion involves an even–odd permutation. In the first step the points with last binary digit equal to 0 are ordered first and those with last digit equal to 1 are ordered last. In the next step the two resulting subsequences of length $N/2$ are reordered according to the second binary digit, etc. It is not difficult to see that the combined effect of the reordering in stage 1 is a **bit-reversal permutation** of the data points. For $i = 0 : N-1$, let the index $i$ have the binary expansion

$$i = b_0 + b_1 \cdot 2 + \cdots + b_{t-1} \cdot 2^{t-1}$$

and set

$$r(i) = b_{t-1} + \cdots + b_1 \cdot 2^{t-2} + b_0 \cdot 2^{t-1}.$$

That is, $r(i)$ is the index obtained by reversing the order of the binary digits. If $i < r(i)$, then exchange $f_i$ and $f_{r(i)}$. This reordering is illustrated for $N = 16$ below.

| Decimal | Binary | | Decimal | Binary |
|---|---|---|---|---|
| 0 | 0000 | | 0 | 0000 |
| 1 | 0001 | | 8 | 1000 |
| 2 | 0010 | | 4 | 0100 |
| 3 | 0011 | | 12 | 1100 |
| 4 | 0100 | | 2 | 0010 |
| 5 | 0101 | | 10 | 1010 |
| 6 | 0110 | | 6 | 0110 |
| 7 | 0111 | $\Longrightarrow$ | 14 | 1110 |
| 8 | 1000 | | 1 | 0001 |
| 9 | 1001 | | 9 | 1001 |
| 10 | 1010 | | 5 | 0101 |
| 11 | 1011 | | 13 | 1101 |
| 12 | 1100 | | 3 | 0011 |
| 13 | 1101 | | 11 | 1011 |
| 14 | 1110 | | 7 | 0111 |
| 15 | 1111 | | 15 | 1111 |

We denote the permutation matrix performing the bit-reversal ordering by $P_N$. Note that if an index is reversed twice we end up with the original index. This means that

$$P_N^{-1} = P_N^T = P_N,$$

i.e., $P_N$ is symmetric. The permutation can be carried out "in place" by a sequence of pairwise interchanges or transpositions of the data points. For example, for $N = 16$ the pairs (1,8), (2,4), (3,12), (5,10), (7,14), and (11,13) are interchanged. The bit-reversal permutation can take a substantial fraction of the total time to do the FFT. Which implementation is best depends strongly on the computer architecture.

We now consider the second stage of the FFT. The key observation to develop a matrix-oriented description of this stage is to note that the Fourier matrices $F_N$ after an odd–even permutation of the columns can be expressed as a $2 \times 2$ block matrix, where each block is either $F_{N/2}$ or a diagonal scaling of $F_{N/2}$.

**Theorem 4.7.2** (*Van Loan* [366, *Theorem* 1.2.1]).

*Let $\Pi_N^T$ be the permutation matrix which applied to a vector groups the even-indexed components first and the odd-indexed last.*[164] *If $N = 2m$, then*

$$F_N \Pi_N = \begin{pmatrix} F_m & \Omega_m F_m \\ F_m & -\Omega_m F_m \end{pmatrix} = \begin{pmatrix} I_m & \Omega_m \\ I_m & -\Omega_m \end{pmatrix} \begin{pmatrix} F_m & 0 \\ 0 & F_m \end{pmatrix},$$

$$\Omega_m = \operatorname{diag}(1, \omega_N, \ldots, \omega_N^{m-1}), \quad \omega_N = e^{-2\pi i/N}. \tag{4.7.9}$$

***Proof.*** The proof essentially follows from the derivation of the butterfly relations (4.7.6)–(4.7.7). □

[164] Note that $\Pi_N^T = \Pi_N^{-1}$ is the so-called **perfect shuffle permutation**, which in the permuted vector $\Pi_N^T f$ is obtained by splitting $f$ in half and then "shuffling" the top and bottom halves.

**Example 4.7.3.**

We illustrate Theorem 4.7.2 for $N = 2^2 = 4$. The DFT matrix $F_4$ is given in Example 4.7.1. After a permutation of the columns $F_4$ can be written as a $2 \times 2$ block-matrix

$$F_4\Pi_4^T = \left(\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & -1 & -i & i \\ \hline 1 & 1 & -1 & -1 \\ 1 & -1 & i & -i \end{array}\right) = \begin{pmatrix} F_2 & \Omega_2 F_2 \\ F_2 & -\Omega_2 F_2 \end{pmatrix},$$

where

$$F_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \Omega_2 = \operatorname{diag}(1, -i).$$

When $N = 2^p$ the FFT algorithm can be interpreted as a sparse factorization of the DFT matrix

$$F_N = A_k \cdots A_2 A_1 P_N, \tag{4.7.10}$$

where $P_N$ is the bit-reversal permutation matrix and $A_1, \ldots, A_k$ are block-diagonal matrices,

$$A_q = \operatorname{diag}(\underbrace{B_L, \ldots, B_L}_{r}), \quad L = 2^q, \quad r = N/L. \tag{4.7.11}$$

Here the matrix $B_k \in \mathbf{C}^{L\times L}$ is the radix-2 butterfly matrix defined by

$$B_L = \begin{pmatrix} I_{L/2} & \Omega_{L/2} \\ I_{L/2} & -\Omega_{L/2} \end{pmatrix}, \tag{4.7.12}$$

$$\Omega_{L/2} = \operatorname{diag}(1, \omega_L, \ldots, \omega_L^{L/2-1}), \quad \omega_L = e^{-2\pi i/L}. \tag{4.7.13}$$

The FFT algorithm described above is usually referred to as the Cooley–Tukey FFT algorithm. Using the fact that both the bit-reversal matrix $P_N$ and the DFT matrix $F_n$ are symmetric, we obtain by transposing (4.7.10) the factorization

$$F_N = F_N^T = P_N A_1^T A_2^T \cdots A_k^T. \tag{4.7.14}$$

This gives rise to a "dual" FFT algorithm, referred to as the Gentleman–Sande algorithm [155]. In this the bit-reversal permutation comes *after* the other computations. In many important applications, such as convolution and the solution of discretized Poisson equations (see Sec. 1.1.4), this permits the design of in-place FFT solutions that avoid bit-reversal altogether; see Van Loan [366, Secs. 4.1, 4.5].

In the operation count for the FFT above we assumed that the weights $\omega_L^j$, $j = 1 : L-1$, $\omega_L = e^{-2\pi i/L}$, are precomputed. To do this one could use that

$$\omega_L^j = \cos(j\theta) - i\sin(j\theta), \quad \theta = 2\pi/L,$$

for $L = 2^q, q = 2 : k$. This is accurate, but expensive, since it involves $L-1$ trigonometric functions calls. An alternative is to compute $\omega = \cos(\theta) - i\sin(\theta)$ and use repeated multiplication,

$$\omega^j = \omega\omega^{j-1}, \quad j = 2 : L-1.$$

This replaces one sine/cosine call with a single complex multiplication, but has the drawback that accumulation of roundoff errors will give an error in $\omega_L^j$ of order $ju$.

### 4.7.2 Discrete Convolution by FFT

The most important operation in signal processing is computing the discrete version of the convolution operator. This awkward operation in the time domain becomes very simple in the frequency domain.

**Definition 4.7.3.**

*The convolution of two sequences $f_i$ and $g_i$, $i = 0 : N-1$, is* $\operatorname{conv}(f, g) = (h_0, h_1, \ldots, h_{N-1})^T$, *where*

$$h_k = \sum_{i=0}^{N-1} f_i g_{k-i}, \quad k = 0 : N-1, \tag{4.7.15}$$

*where the sequences are extended to have period N by setting $f_i = f_{i+jN}$, $g_i = g_{i+jN}$ for all integers $i$, $j$.*

The discrete convolution can be used to approximate the convolution defined for continuous functions in Definition 4.6.5 in a similar way as the Fourier transform was approximated using sampled values in Sec. 4.6.6.

We can write the sum in (4.7.15) as a matrix-vector multiplication $h = Gf$, where $G$ is a Toeplitz matrix. Writing out components we have

$$\begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \\ h_{N-1} \end{pmatrix} = \begin{pmatrix} g_0 & g_{N-1} & g_{N-2} & \cdots & g_1 \\ g_1 & g_0 & g_{N-1} & \cdots & g_2 \\ g_2 & g_1 & g_0 & \cdots & g_3 \\ \vdots & \vdots & \vdots & \cdots & \\ g_{N-1} & g_{N-2} & g_{N-3} & \cdots & g_0 \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{pmatrix}.$$

Note that each column in $G$ is a cyclic down-shifted version of the previous column. Such a matrix is called a **circulant matrix**. We have

$$G = [\, g \quad C_N g \quad \cdots \quad C_N^{N-1} g \,] = g_0 I + g_1 C_N + \cdots + g_{N-1} C_N^{N-1}, \tag{4.7.16}$$

and $C_N$ is the circulant permutation matrix

$$C_N = \begin{pmatrix} 0 & & \cdots & & 1 \\ 1 & & & & \\ & 1 & & & \vdots \\ \vdots & & \ddots & & \\ 0 & & \cdots & 1 & 0 \end{pmatrix}. \tag{4.7.17}$$

The following result is easily verified.

**Lemma 4.7.4.**

*The eigenvalues of the circulant matrix $C_N$ in* (4.7.17) *are*

$$\omega_j = e^{-2\pi j/N}, \quad j = 0 : N-1,$$

*where $\omega$ is an $N$th root of unity, i.e., $\omega^N = 1$. The columns of the DFT matrix $F_N$,*

$$x_j = (1, \omega_j, \ldots, \omega_j^{N-1})^T, \quad j = 0 : N-1,$$

*are eigenvectors.*

Since the matrix $G$ in (4.7.16) is a polynomial in $C_N$ it has the same set of eigenvectors, and thus $G$ is diagonalized by the DFT matrix $F_N$,

$$G = F_N \Lambda F_N^{-1}, \quad \Lambda = \text{diag}\,(\lambda_1, \ldots, \lambda_n), \tag{4.7.18}$$

where the eigenvalues of $G$ are

$$\lambda_i = (1, \omega_j, \ldots, \omega_j^{N-1})g, \quad j = 0 : N-1,$$

which is the FFT of the first column in $G$. Hence $\Lambda = \text{diag}\,(F_N g)$, where diag $(x)$ denotes a diagonal matrix with diagonal elements equal to the elements in the vector $x$.

**Theorem 4.7.5.**

*Let $f_i$ and $g_i$, $i = 0 : N-1$, be two sequences with DFTs equal to $F_N f$ and $F_N g$. Then the DFT of the* **convolution** *of $f$ and $g$ is $F_N f .* F_N g$ (.* denotes the elementwise product).*

***Proof.*** From $G = F_N^{-1}\text{diag}\,(F_N g)F_N$ it follows that

$$h = Gf = F_N^{-1}\text{diag}\,(F_N g)F_N f = F_N^{-1}((F_N g) .* (F_N f)). \qquad \square \tag{4.7.19}$$

This shows that using the FFT algorithm the discrete convolution can be computed in $O(N \log_2 N)$ operations as follows: First the two FFTs of $f$ and $g$ are computed and multiplied (pointwise) together. Then the inverse DFT of this product is computed. This is one of the most useful properties of the FFT!

Using the Gentleman–Sande algorithm $F_N = P_N A^T$ for the forward DFT and the Cooley–Tukey algorithm $F_N = A P_N$ for the inverse DFT,

$$F_N^{-1} = \frac{1}{N} F_N^H = \frac{1}{N} \bar{A} P_N,$$

we get from (4.7.19)

$$h = \frac{1}{N} \bar{A} P_N ((P_N A^T f) .* (P_N A^T g)) = \frac{1}{N} \bar{A}((A^T f) .* (A^T g)). \tag{4.7.20}$$

This shows that $h$ can be computed without the bit-reversal permutation $P_N$ which typically can save 10–30 percent of the overall computation time.

### 4.7.3 FFTs of Real Data

Frequently the FFT of a real data vector is required. The complex FFT algorithm can still be used, but is inefficient both in terms of storage and operations. By using symmetries in

the DFT, which correspond to the symmetries noted in the Fourier transform in Table 4.6.1, better alternatives can be found.

Consider the DFT matrix for $N = 4$ in (4.7.8). Note that the fourth row is the conjugate of the second row. This is not a coincidence; the conjugate transpose of the DFT matrix $F_N$ can be obtained by reversing the order of the last $N-1$ rows. Let $T_N$ be the $N \times N$ permutation matrix obtained by reversing the last $N-1$ columns in the unit matrix $I_N$. For example,

$$T_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Then it holds that

$$F_N^H = \overline{F}_N = T_N F_N = F_N T_N. \tag{4.7.21}$$

We first verify that $\overline{F}_N = T_N F_N$, by observing that

$$[T_N F_N]_{jk} = \omega_N^{(N-j)k} = \omega_N^{-jk} = \overline{\omega}_N^{jk} = [\overline{F}_N]_{jk}, \quad 1 \le j \le N-1.$$

Since $F_N$ and $T_N$ are both symmetric, we also have $F_N^H = (T_N F_N)^T = F_N T_N$.

We say that a vector $y \in \mathbf{C}^N$ is **conjugate even** if $\overline{y} = T_N y$, and **conjugate odd** if $\overline{y} = -T_N y$. Suppose now that $f$ is real and $u = F_N f$. Then it follows that

$$\overline{u} = \overline{F}_N f = T_N F_N f = T_N u,$$

i.e., $u$ is conjugate even. If a vector $u$ of even length $N$ is conjugate even, this implies that

$$u_j = \overline{u}_{N-j}, \quad j = 1 : N/2.$$

In particular, $u_j$ is real for $j = 0, N/2$.

For purely imaginary data $g$ and $v = F_N g$, we have

$$\overline{v} = F_N^H \overline{g} = -F_N^H g = -T_N F_N g = -T_N v,$$

i.e., $v$ is conjugate odd. Some other useful symmetry properties are given in Table 4.7.1. We have proved the first two properties; the others are established similarly and we leave the proofs to the reader; see Problem 4.7.4.

**Table 4.7.1.** *Useful symmetry properties of the DFT.*

| Data $f$ | Definition | DFT $F_N f$ |
|---|---|---|
| real | | conjugate even |
| imaginary | | conjugate odd |
| real even | $f = T_N f$ | real |
| real odd | $f = -T_N f$ | imaginary |
| conjugate even | $\overline{f} = T_N f$ | real |
| conjugate odd | $\overline{f} = -T_N f$ | imaginary |

We now outline how symmetries can be used to compute the DFTs $u = F_N f$ and $v = F_N g$ of *two* real functions $f$ and $g$ simultaneously. First form the complex function $f + ig$ and compute its DFT

$$w = F_N(f + ig) = u + iv$$

by any complex FFT algorithm. Multiplying by $T_N$ we have

$$T_N w = T_N F_N(f + ig) = T_N(u + iv) = \overline{u} + i\overline{v},$$

where we have used that $u$ and $v$ are conjugate even. Adding and subtracting these two equations we obtain

$$w + T_N w = (u + \overline{u}) + i(v + \overline{v}),$$
$$w - T_N w = (u - \overline{u}) + i(v - \overline{v}).$$

We can now retrieve the two DFTs from

$$u = F_N f = \frac{1}{2}\big[\mathrm{Re}(w + T_N w) + i\,\mathrm{Im}(w - T_N w)\big], \tag{4.7.22}$$

$$v = F_N g = \frac{1}{2}\big[\mathrm{Im}(w + T_N w) - i\,\mathrm{Re}(w - T_N w)\big]. \tag{4.7.23}$$

Note that because of the conjugate even property of $u$ and $v$ there is no need to save the entire transforms.

The above scheme is convenient when, as for convolutions, two real transforms are involved. It can also be used to efficiently compute the DFT of a single real function of length $N = 2^p$. First express this DFT as a combination of the two real FFTs of length $N/2$ corresponding to even and odd numbered data points (as in (4.7.5)). Then apply the procedure above to simultaneously compute these two real FFTs.

### 4.7.4 Fast Trigonometric Transforms

Two real transforms, the **discrete sine transform (DST)** and **discrete cosine transform (DCT)**, are of interest. There are several variations of these. We define the first two as follows:

- Given real data $f_j$, $j = 1 : N - 1$, compute

$$y_k = \sum_{j=1}^{N-1} \sin\left(kj\frac{\pi}{N}\right) f_j \qquad \text{(DST-1)}. \tag{4.7.24}$$

- Given real data $f_j$, $j = 0 : N$, compute

$$y_k = \sum_{j=0}^{N}{}'' \cos\left(kj\frac{\pi}{N}\right) f_j \qquad \text{(DCT-1)}. \tag{4.7.25}$$

(The double prime on the sum means that the first and last term are to be halved.)

The real and imaginary parts of the DFT matrix $F_N$ consist of cosines and sines, $F_N = C_N - iS_N$, where

$$(C_N)_{kj} = \cos\left(kj\frac{2\pi}{N}\right), \qquad (S_N)_{kj} = \sin\left(kj\frac{2\pi}{N}\right), \quad k, j = 0 : N-1. \tag{4.7.26}$$

The DST and DCT transforms can be expressed in matrix form as

$$y = S_{2N}(1 : N-1, 1 : N-1)f, \qquad y = C_{2N}(0 : N, 0 : N)\tilde{f},$$

respectively, where $\tilde{f} = (\frac{1}{2}f_0, f_1, \ldots, f_{N-1}, \frac{1}{2}f_N)^T$.

These two transforms can be computed by applying the FFT algorithm (for real data) to an auxiliary vector $\tilde{f}$ formed by extending the given data vector $f$ either into an odd or even sequence.

For DST-1 the data $f_j$, $j = 1 : N-1$, are extended to an *odd* sequence of length $2N$ by setting

$$f_0 = f_N = 0, \qquad f_{2N-j} \equiv -f_j, \quad j = 1 : N-1.$$

For example, the data $f = \{f_1, f_2, f_3\}$, $(N = 4)$ are extended to

$$\tilde{f} = \{0, f_1, f_2, f_3, 0, -f_3, -f_2, -f_1\}.$$

The extended vector satisfies $\tilde{f} = -T_{2N}\tilde{f}$, and thus by Table 4.7.1 the DFT of $\tilde{f}$ will be pure imaginary.

For DCT-1 the data $f_j$, $j = 0 : N$, are extended to an *even* sequence of length $2N$ by setting

$$f_{2N-j} \equiv f_j, \quad j = 1 : N.$$

For example, the data $f = \{f_0, f_1, f_2, f_3, f_4\}$, $(N = 4)$ are extended to

$$\tilde{f} = \{f_0, f_1, f_2, f_3, f_4, f_3, f_2, f_1\}$$

so that $\tilde{f} = T_{2N}\tilde{f}$. By Table 4.7.1 the DFT of $\tilde{f}$ will then be real.

**Theorem 4.7.6** (*Van Loan* [366, *Sec.* 4.4]).

*Let $f_j$, $j = 1 : N-1$, form a real data vector $f$ and extend it to a vector $\tilde{f}$ with $\tilde{f}_0 = \tilde{f}_N = 0$ so that $\tilde{f} = -T_{2N}\tilde{f}$. Then $y(1 : N-1)$ is the DST of $\tilde{f}$, where*

$$y = y(0 : 2N-1) = \frac{i}{2}F_{2N}\tilde{f}.$$

*Let $f_j$, $j = 0 : N$, form a real data vector $f$ and extend it to a vector $\tilde{f}$ so that $\tilde{f} = T_{2N}\tilde{f}$. Then $y(0 : N)$ is the DCT of $f$, where*

$$y = y(0 : 2N-1) = \frac{1}{2}F_{2N}\tilde{f}.$$

There is an inefficiency factor of two in the above procedure. This can be eliminated by using a different auxiliary vector. For details we refer to [294, pp. 420–421]; [366, Sec. 4.4].

In some application areas variants of the above transforms called DST-2 and DCT-2 turn out to be more useful. We define them as follows:

- Given real data $f_j$, $j = 1 : N$, compute

$$y_k = \sum_{j=1}^{N} \sin\left(k(2j-1)\frac{\pi}{2N}\right) f_j \qquad \text{(DST-2)}. \tag{4.7.27}$$

- Given real data $f_j$, $j = 0 : N-1$, compute

$$y_k = \sum_{j=0}^{N-1} \cos\left(k(2j+1)\frac{\pi}{2N}\right) f_j \qquad \text{(DCT-2)}. \tag{4.7.28}$$

The DST-2 and DCT-2 transforms can also be obtained by extending the data vector $f$.

For DST-2 the data vector $f_j$, $j = 0 : N-1$, is extended to an *odd* sequence of length $2N$. For example, the data $\{f_1, f_2, f_3, f_4\}$, $(N = 4)$ are extended to

$$\tilde{f} = \{f_1, f_2, f_3, f_4, -f_4, -f_3, -f_2, -f_1\}.$$

The extended vector satisfies $f = -T_{2N} f$, and thus by Table 4.7.1 the DFT of $f$ will be imaginary.

For DCT-2 the data $f_j$, $j = 0 : N$, are extended to an *even* sequence of length $2N$. For example, the data $\{f_0, f_1, f_2, f_3\}$, $(N = 4)$ are extended to

$$\tilde{f} = \{f_0, f_1, f_2, f_3, f_3, f_2, f_1, f_0\}.$$

so that $f = T_{2N} f$. By Table 4.7.1 the DFT of $f$ will then be real.

We give without proof the following result, which allows the computation of DST-2 and DCT-2 from the FFT.

**Theorem 4.7.7.**

*Let $f_j$, $j = 1 : N$, form a real data vector $f$ and extend it to a vector $\tilde{f}$ so that $\tilde{f} = -T_{2N}\tilde{f}$. Then $y(1 : N)$ is the DST-2 of $f$, where*

$$y = y(0 : 2N-1) = \frac{i}{2}\Omega_{2N} F_{2N} \tilde{f},$$

*where*

$$\Omega_{2N} = \operatorname{diag}(1, \omega_{4N}, \ldots, \omega_{4N}^{2N}).$$

*Let $f_j$, $j = 0 : N-1$, form a real data vector $f$ and extend it to a vector $\tilde{f}$ so that $\tilde{f} = T_{2N}\tilde{f}$. Then $y(0 : N-1)$ is the DCT-2 of $f$, where*

$$y = y(0 : 2N-1) = \frac{1}{2}\Omega_{2N} F_{2N} \tilde{f}.$$

The two-dimensional DCT-2 transform has the property that, for a visual image, most of the information is concentrated in the first few coefficients of the DCT. For this reason the DCT-2 transform is often used in image compression algorithms.[165]

[165] This transform is used in the JPEG (Joint Photographic Experts Group) compression algorithm for image processing. Each $8 \times 8$ block in the image is transformed by a two-dimensional DCT-2 transform; see Strang [340].

### 4.7.5 The General Case FFT

It can be argued [294, p. 409] that one should always choose $N = 2^k$ when using the FFT. If necessary the data can be padded with zeros to achieve this. To introduce an odd factor $s$, let $N = sr$, where $r$ is a power of 2. Then one can combine the power of two algorithm for the $r$-point subseries with a special algorithm for the $s$-point subseries. If $s$ is a small number, then one could generate the DFT matrix $F_s$ and use matrix-vector multiplication; (cf. the code given in Problem 4.7.2). But the general case when $N$ is not a power of two is at least of theoretical interest.

Suppose that $N = r_1 r_2 \cdots r_p$. We will describe an FFT algorithm which requires $N(r_1 + r_2 + \cdots + r_p)$ complex operations. Set

$$N_\nu = \prod_{i=\nu+1}^{p} r_i, \quad \nu = 0 : p-1, \quad N_p = 1.$$

Thus

$$N = r_1 r_2 \cdots r_\nu N_\nu, \quad N_0 = N.$$

The algorithm is based on two representations of integers which are generalizations of the position principle (see Sec. 2.2.1).

I. Every integer $j$, $0 \le j \le N-1$, has a unique representation of the form

$$j = \alpha_1 N_1 + \alpha_2 N_2 + \cdots + \alpha_{p-1} N_{p-1} + \alpha_p, \quad 0 \le \alpha_i \le r_i - 1. \tag{4.7.29}$$

II. For every integer $\beta$, $0 \le \beta \le N-1$, $\beta/N$ has a unique representation of the form

$$\frac{\beta}{N} = \frac{k_1}{N_0} + \frac{k_2}{N_1} + \cdots + \frac{k_p}{N_{p-1}}, \quad 0 \le k_i \le r_i - 1. \tag{4.7.30}$$

Set

$$j_\nu = \sum_{i=\nu+1}^{p} \alpha_i N_i, \qquad \frac{\alpha_\nu}{N_\nu} = \sum_{i=\nu}^{p-1} \frac{k_{i+1}}{N_i}, \quad (j_\nu < N_\nu). \tag{4.7.31}$$

As an exercise, the reader can verify that the coefficients in the above representations can be recursively determined from the following algorithms:[166]

$$j_0 = j, \qquad j_{i-1}/N_i = \alpha_i + j_i/N_i, \quad i = 1 : p,$$

$$\beta_0 = \beta, \qquad \beta_{i-1}/r_i = \beta_i + k_i/r_i, \quad i = 1 : p.$$

From (4.7.29)–(4.7.31), it follows that, since $N_i$ is divisible by $N_\nu$ for $i \le \nu$,

$$\frac{j\beta}{N} = \text{integer} + \sum_{\nu=0}^{p-1} \frac{k_{\nu+1}}{N_\nu} \left( \sum_{i=\nu+1}^{p} \alpha_i N_i \right) = \sum_{\nu=0}^{p-1} \frac{k_{\nu+1} j_\nu}{N_\nu} + \text{integer}.$$

[166] These algorithms can, in the special case that $r_i = B$ for all $i$, be used for converting integers or fractions to the number system whose base is $B$; see Algorithm 2.1.

From this, it follows that

$$\omega^{j\beta} = e^{2\pi i j\beta/N} = \prod_{\nu=0}^{p-1} e^{k_{\nu+1} j_\nu 2\pi i/N_\nu} = \prod_{\nu=0}^{p-1} \omega_\nu^{j_\nu k_{\nu+1}}, \tag{4.7.32}$$

where $\omega_\nu = e^{2\pi i/N_\nu}$, $\omega_0 = \omega$.

A split-radix approach for the general case FFT, which has a lower operation count, is described in Van Loan [366, Sec. 2.1.4].

We now give an illustration of how the factorization in (4.7.32) can be utilized in FFT for the case $p = 3$. Set, in accordance with (4.7.30),

$$f_\beta = c^{(0)}(k_1, k_2, k_3).$$

We have then

$$c_j = \sum_{\beta=0}^{N-1} f_\beta \omega^{j\beta} = \sum_{k_1=0}^{r_1-1} \sum_{k_2=0}^{r_2-1} \sum_{k_3=0}^{r_3-1} c^{(0)}(k_1, k_2, k_3) \omega_2^{j_2 k_3} \omega_1^{j_1 k_2} \omega^{j k_1}.$$

One can thus compute successively (see (4.7.31))

$$c^{(1)}(k_1, k_2, \alpha_3) = \sum_{k_3=0}^{r_3-1} c^{(0)}(k_1, k_2, k_3) \omega_2^{j_2 k_3} \quad (j_2 \text{ depends only on } \alpha_3),$$

$$c^{(2)}(k_1, \alpha_2, \alpha_3) = \sum_{k_2=0}^{r_2-1} c^{(1)}(k_1, k_2, \alpha_3) \omega_1^{j_1 k_2} \quad (j_1 \text{ depends only on } \alpha_2, \alpha_3),$$

$$c_j = c^{(3)}(\alpha_1, \alpha_2, \alpha_3) = \sum_{k_1=0}^{r_1-1} c^{(2)}(k_1, \alpha_2, \alpha_3) \omega^{j k_1} \quad (j \text{ depends on } \alpha_1, \alpha_2, \alpha_3).$$

The quantities $c^{(i)}$ are computed for all $r_1 r_2 r_3 = N$ combinations of the values of the arguments. Thus the total number of operations for the entire Fourier analysis becomes at most $N(r_3 + r_2 + r_1)$. The generalization to arbitrary $p$ is obvious.

## Review Questions

**4.7.1** Suppose we want to compute the DFT for $N = 2^{10}$. Roughly how much faster is the FFT algorithm compared to the straightforward $O(N^2)$ algorithm?

**4.7.2** Show that the matrix $U = \frac{1}{\sqrt{N}} F_N$ is unitary, i.e., $U^H U = I$, where $U^H = (\overline{U})^T$.

**4.7.3** Show that the DFT matrix $F_4$ can be written as a $2 \times 2$ block matrix where each block is related to $F_2$. Give a generalization of this for $F_N$, $N = 2^m$ that holds for arbitrary $m$.

**4.7.4** Work out, on your own, the bit-reversal permutation of the vector $[0 : N-1]$ for the case $N = 2^4 = 16$. How many exchanges need to be performed?

## Problems and Computer Exercises

**4.7.1** The following MATLAB script uses an algorithm due to Cooley et al. to permute the vector $x(1:2^m)$, in bit-reversal order:

```
n = 2^m;
nv2 = n/2; nm1 = n - 1;
j = 1;
for i = 1:nm1
  if i < j
    t = x(j); x(j) = x(i); x(i) = t;
  end
  k = nv2;
  while k < j
     j = j - k; k = k/2;
  end
  j = j + k;
end
```

Plot the time taken by this algorithm on your computer for $m = 5:10$. Does the execution time depend linearly on $N = 2^m$?

**4.7.2** The following MATLAB program (Moler and Eddins [269]) demonstrates how the FFT idea can be implemented in a simple but efficient recursive MATLAB program. The program uses the fast recursion as long as $n$ is a power of two. When it reaches an odd length it sets up the Fourier matrix and uses matrix-vector multiplication.

**ALGORITHM 4.3.** *Fast Fourier Transform.*

```
function y = fftx(x);
% FFT computes the Fast Fourier Transform of x(1:n)
x = x(:);
n = length(x);
omega = exp(-2*pi*i/n);
if rem(n,2) == 0
% Recursive divide and conquer
   k = (0:n/2-1)
   w = omega.^k;
   u = fftx(x(1:2:n-1));
   v = w.*fftx(x(2:2:n));
   y = [u+v; u-v];
else
% Generate the Fourier matrix
   j = 0:n-1;
   k = j';
   F = omega.^(k*j);
   y = F*x;
end
```

Apply this program to compute DFT of the rectangular wave in Sec. 4.6.1, sampled at the points $2\pi\alpha/N$, $\alpha = 0:N-1$. Choose, for instance, $N = 32, 64, 128$.

**4.7.3** Write an efficient MATLAB program for computing the DFT of a real data vector of length $N = 2^m$. As outlined in Sec. 4.7.3, first split the data in odd and even data points. Compute the corresponding DFTs using one call of the function `fftx` in Problem 4.7.2 with complex data of length $N/2$.

**4.7.4** Verify the last four symmetry properties of DFTs in Table 4.7.1.

## Notes and References

A modern treatment of divided differences considered as linear functionals on some vector space of functions is given by de Boor [38]. Our notation $[x_1, \ldots, x_k]f$ for the divided difference agrees with that used in de Boor [37]. A general treatment of complex polynomial interpolation on complex nodes is given by Gelfond [151, Sec. 1.4.3]; see also Horn and Johnson [203, Sec. 6.1].

The barycentric form of Lagrange's interpolation formula found in German literature does not seem to have caught on elsewhere, until recently. Berrut and Trefethen [24] argue convincingly that this should be the standard method. The problem of choosing a good ordering of points in Newton and barycentric Lagrange interpolations is discussed in Werner [371]. Newton interpolation using the Leja ordering of points has been analyzed by Reichel [299].

Uniform methods of solving linear problems and computing the inverse of a Vandermonde matrix have been studied by Traub [355]. The algorithm for the inverse Vandermonde matrix given in the text is due to Higham [199, Sec. 22.1]. The $O(n^2)$ Björck–Pereyra algorithm for solving Vandermonde systems appeared in [31].

An introduction to rational interpolation is given in Stoer and Bulirsch [338, Sec. 2.2]. The reciprocal differences of Thiele are also treated by Milne-Thomson [265], with expressions for the truncation error. The $\rho$-algorithm for convergence acceleration is due to Wynn [385].

A good reference for multidimensional polynomial interpolation is Steffensen [330, Sec. 19]; see also Isaacson and Keller [208, Sec. 6.6]. Methods for multidimensional interpolation of scattered data points, including radial basis functions, are surveyed by Powell in [293]. For a more practical approach the survey by Hayes [189] is still very good. The history of multidimensional interpolation is surveyed in [134].

The survey paper [27] gives a good summary of the pioneering work done by Garret Birkhoff on interpolation and approximation of univariate and bivariate data. The book by de Boor [37] has done much to further the use of B-splines in geometric constructions. It contains a clear outline of the theory and also a library of Fortran programs for computations with spline functions. Several packages are available for computing with splines, e.g., the spline toolbox in MATLAB and FITPACK by Dierckx [97, 98]. A more geometric view of splines is taken in Farin [116].

The computational advantage of the Stieltjes approach for discrete least squares fitting was pointed out by Forsythe [119]. Shampine [321] established the advantage of using the alternative formula involving the residual $r_k$.

The theory and application of Chebyshev systems in approximation and statistics is surveyed in Karlin and Studden [222]. An application of uniform approximation with polynomials and the Remez algorithm is the determination of nonrecursive digital filters with minimal ripple; see Parks and McClellan [284]. The discrete $\ell_1$ approximation problem

is related to "sparse approximation" ideas that are currently attracting much attention in the signal processing community.

The series named after Fourier was well established by the work of Bernoulli, Euler, and Lagrange, before the time of Fourier. The Fourier integral is, however, the undisputed discovery of Fourier. A very readable introduction to harmonic analysis is given in Lanczos [235, Chapter IV]. For a more exhaustive treatment of Gibbs' phenomenon, see Tadmor [348].

The rediscovery of the FFT algorithm is surveyed in [76]. Applications are surveyed in [77, 58, 57]. The matrix-oriented framework for the FFT used in this book is due to Van Loan [366]. An early implementation of the FFT is given by Singleton [324, 325]. For a roundoff error analysis of the FFT see [8]. Algorithms for bit-reversal permutations are reviewed in [223].

The FFT algorithm was discovered independently by several mathematicians. Similar ideas were used already by Gauss [137]. The doubling algorithm is contained in the textbook by Runge and König [309] from 1924 and in the paper by Danielson and Lanczos [90].

Some significant developments in approximation theory are not covered in this book, such as wavelets (see Daubechies [91]), radial basis functions (see Buhmann [59]), and box splines (see de Boor, Höllig, and Riemenschneider [39]). The last two constructs are different multidimensional generalizations of univariate spline functions.