Chapter 2
Interpolation

Experiments usually produce a discrete set of data points $(\mathbf{x}_i, f_i)$ which represent the value of a function $f(\mathbf{x})$ for a finite set of arguments $\{\mathbf{x}_0 \cdots \mathbf{x}_n\}$. If additional data points are needed, for instance to draw a continuous curve, interpolation is necessary. Interpolation also can be helpful to represent a complicated function by a simpler one or to develop more sophisticated numerical methods for the calculation of numerical derivatives and integrals. In the following we concentrate on the most important interpolating functions which are polynomials, splines and rational functions. Trigonometric interpolation is discussed in Chap. 7. An interpolating function reproduces the given function values at the interpolation points exactly (Fig. 2.1). The more general procedure of curve fitting, where this requirement is relaxed, is discussed in Chap. 10.

The interpolating polynomial can be explicitly constructed with the Lagrange method. Newton's method is numerically efficient if the polynomial has to be evaluated at many interpolating points and Neville's method has advantages if the polynomial is not needed explicitly and has to be evaluated only at one interpolation point.

Polynomials are not well suited for interpolation over a larger range. Spline functions can be superior which are piecewise defined polynomials. Especially cubic splines are often used to draw smooth curves. Curves with poles can be represented by rational interpolating functions whereas a special class of rational interpolants without poles provides a rather new alternative to spline interpolation.

2.1 Interpolating Functions

Consider the following problem: Given are $n + 1$ sample points (x_i, f_i), $i = 0 \cdots n$ and a function of x which depends on $n + 1$ parameters a_i:

$$\Phi(x; a_0 \cdots a_n). \tag{2.1}$$

Fig. 2.1 (Interpolating
function) The interpolating
function $\Phi(x)$ reproduces a
given data set $\Phi(x_i) = f_i$ and
provides an estimate of the
function $f(x)$ between the
data points
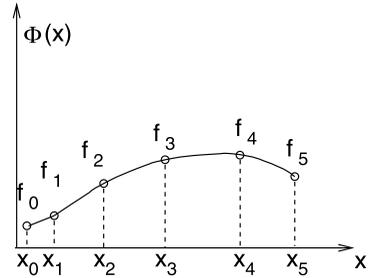
The parameters are to be determined such that the interpolating function has the
proper values at all sample points (Fig. 2.1)

$$\Phi(x_i; a_0 \cdots a_n) = f_i \quad i = 0 \cdots n. \tag{2.2}$$

An interpolation problem is called linear if the interpolating function is a linear
combination of functions

$$\Phi(x; a_0 \cdots a_n) = a_0 \Phi_0(x) + a_1 \Phi_1(x) + \cdots + a_n \Phi_n(x). \tag{2.3}$$

Important examples are

• polynomials

$$a_0 + a_1 x + \cdots + a_n x^n \tag{2.4}$$

• trigonometric functions

$$a_0 + a_1 e^{ix} + a_2 e^{2ix} + \cdots + a_n e^{nix} \tag{2.5}$$

• spline functions which are piecewise polynomials, for instance the cubic spline

$$s(x) = \alpha_i + \beta_i(x - x_i) + \gamma_i(x - x_i)^2 + \delta_i(x - x_i)^3 \quad x_i \le x \le x_{i+1}. \tag{2.6}$$

Important examples for nonlinear interpolating functions are

• rational functions

$$\frac{p_0 + p_1 x + \cdots + p_M x^M}{q_0 + q_1 x + \cdots + q_N x^N} \tag{2.7}$$

• exponential functions

$$a_0 e^{\lambda_0 x} + a_1 e^{\lambda_1 x} + \cdots. \tag{2.8}$$

where amplitudes a_i as well as exponents λ_i have to be optimized.

2.2 Polynomial Interpolation

For $n + 1$ sample points (x_i, f_i), $i = 0 \cdots n$, $x_i \ne x_j$ there exists exactly one inter-
polating polynomial of degree n with

$$p(x_i) = f_i, \quad i = 0 \cdots n. \tag{2.9}$$

2.2.1 Lagrange Polynomials

Lagrange polynomials [137] are defined as

$$L_i(x) = \frac{(x - x_0) \cdots (x - x_{i-1})(x - x_{i+1}) \cdots (x - x_n)}{(x_i - x_0) \cdots (x_i - x_{i-1})(x_i - x_{i+1}) \cdots (x_i - x_n)}. \tag{2.10}$$

They are of degree n and have the property

$$L_i(x_k) = \delta_{i,k}. \tag{2.11}$$

The interpolating polynomial is given in terms of Lagrange polynomials by

$$p(x) = \sum_{i=0}^{n} f_i L_i(x) = \sum_{i=0}^{n} f_i \prod_{k=0, k \neq i}^{n} \frac{x - x_k}{x_i - x_k}. \tag{2.12}$$

2.2.2 Barycentric Lagrange Interpolation

With the polynomial

$$\omega(x) = \prod_{i=0}^{n} (x - x_i) \tag{2.13}$$

the Lagrange polynomial can be written as

$$L_i(x) = \frac{\omega(x)}{x - x_i} \frac{1}{\prod_{k=0, k \neq i}^{n} (x_i - x_k)} \tag{2.14}$$

which, introducing the Barycentric weights [24]

$$u_i = \frac{1}{\prod_{k=0, k \neq i}^{n} (x_i - x_k)} \tag{2.15}$$

becomes the first form of the barycentric interpolation formula

$$L_i(x) = \omega(x) \frac{u_i}{x - x_i}. \tag{2.16}$$

The interpolating polynomial can now be evaluated according to

$$p(x) = \sum_{i=0}^{n} f_i L_i(x) = \omega(x) \sum_{i=0}^{n} f_i \frac{u_i}{x - x_i}. \tag{2.17}$$

Having computed the weights u_i, evaluation of the polynomial only requires $O(n)$ operations whereas calculation of all the Lagrange polynomials requires $O(n^2)$ operations. Calculation of $\omega(x)$ can be avoided considering that

$$p_1(x) = \sum_{i=0}^{n} L_i(x) = \omega(x) \sum_{i=0}^{n} \frac{u_i}{x - x_i} \qquad (2.18)$$

is a polynomial of degree n with

$$p_1(x_i) = 1 \quad i = 0 \cdots n. \qquad (2.19)$$

But this is only possible if

$$p_1(x) = 1. \qquad (2.20)$$

Therefore

$$p(x) = \frac{p(x)}{p_1(x)} = \frac{\sum_{i=0}^{n} f_i \frac{u_i}{x - x_i}}{\sum_{i=0}^{n} \frac{u_i}{x - x_i}} \qquad (2.21)$$

which is known as the second form of the barycentric interpolation formula.

2.2.3 Newton's Divided Differences

Newton's method of divided differences [138] is an alternative for efficient numerical calculations [271]. Rewrite

$$f(x) = f(x_0) + \frac{f(x) - f(x_0)}{x - x_0}(x - x_0). \qquad (2.22)$$

With the first order divided difference

$$f[x, x_0] = \frac{f(x) - f(x_0)}{x - x_0} \qquad (2.23)$$

this becomes

$$f[x, x_0] = f[x_1, x_0] + \frac{f[x, x_0] - f[x_1, x_0]}{x - x_1}(x - x_1) \qquad (2.24)$$

and with the second order divided difference

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

$$= \frac{f(x)}{(x - x_0)(x - x_1)} + \frac{f(x_1)}{(x_1 - x_0)(x_1 - x)} + \frac{f(x_0)}{(x_0 - x_1)(x_0 - x)}$$

$$(2.25)$$

we have

$$f(x) = f(x_0) + (x - x_0)f[x_1, x_0] + (x - x_0)(x - x_1)f[x, x_0, x_1]. \quad (2.26)$$

Higher order divided differences are defined recursively by

$$f[x_1 x_2 \cdots x_{r-1} x_r] = \frac{f[x_1 x_2 \cdots x_{r-1}] - f[x_2 \cdots x_{r-1} x_r]}{x_1 - x_r}. \quad (2.27)$$

They are invariant against permutation of the arguments which can be seen from the explicit formula

$$f[x_1 x_2 \cdots x_r] = \sum_{k=1}^{r} \frac{f(x_k)}{\prod_{i \neq k}(x_k - x_i)}. \quad (2.28)$$

Finally we have

$$f(x) = p(x) + q(x) \quad (2.29)$$

with a polynomial of degree n

$$p(x) = f(x_0) + f[x_1, x_0](x - x_0) + f[x_2 x_1 x_0](x - x_0)(x - x_1) + \cdots$$
$$+ f[x_n x_{n-1} \cdots x_0](x - x_0)(x - x_1) \cdots (x - x_{n-1}) \quad (2.30)$$

and the function

$$q(x) = f[x x_n \cdots x_0](x - x_0) \cdots (x - x_n). \quad (2.31)$$

Obviously $q(x_i) = 0$, $i = 0 \cdots n$, hence $p(x)$ is the interpolating polynomial.

Algorithm The divided differences are arranged in the following way:

$$
\begin{array}{l}
f_0 \\
f_1 \quad f[x_0 x_1] \\
\vdots \quad \vdots \quad\quad \ddots \\
f_{n-1} \; f[x_{n-2} x_{n-1}] \; f[x_{n-3} x_{n-2} x_{n-1}] \cdots f[x_0 \cdots x_{n-1}] \\
f_n \quad f[x_{n-1} x_n] \quad f[x_{n-2} x_{n-1} x_n] \;\cdots\; f[x_1 \cdots x_n] \; f[x_0 \cdots x_n]
\end{array}
\quad (2.32)
$$

Since only the diagonal elements are needed, a one-dimensional data array $t[0] \cdots t[n]$ is sufficient for the calculation of the polynomial coefficients:

```
for i := 0 to n do begin
    t[i] := f[i];
    for k := i − 1 downto 0 do
        t[k] := (t[k + 1] − t[k])/(x[i] − x[k]);
    a[i] := t[0];
end;
```

The value of the polynomial is then evaluated by

$p := a[n]$;
for $i := n-1$ downto 0 do
$\quad p := p*(x-x[i])+a[i]$;

2.2.4 Neville Method

The Neville method [180] is advantageous if the polynomial is not needed explicitly and has to be evaluated only at one point. Consider the interpolating polynomial for the points $x_0 \cdots x_k$, which will be denoted as $P_{0,1\ldots k}(x)$. Obviously

$$P_{0,1\ldots k}(x) = \frac{(x-x_0)P_{1\ldots k}(x)-(x-x_k)P_{0\ldots k-1}(x)}{x_k-x_0} \qquad (2.33)$$

since for $x = x_1 \cdots x_{k-1}$ the right hand side is

$$\frac{(x-x_0)f(x)-(x-x_k)f(x)}{x_k-x_0} = f(x). \qquad (2.34)$$

For $x = x_0$ we have

$$\frac{-(x_0-x_k)f(x)}{x_k-x_0} = f(x) \qquad (2.35)$$

and finally for $x = x_k$

$$\frac{(x_k-x_0)f(x)}{x_k-x_0} = f(x). \qquad (2.36)$$

Algorithm We use the following scheme to calculate $P_{0,1\ldots n}(x)$ recursively:

$$
\begin{array}{lllll}
P_0 & & & & \\
P_1 & P_{01} & & & \\
P_2 & P_{12} & P_{012} & & \\
\vdots & \vdots & \vdots & \ddots & \\
P_n & P_{n-1,n} & P_{n-2,n-1,n} & \cdots & P_{01\ldots n}
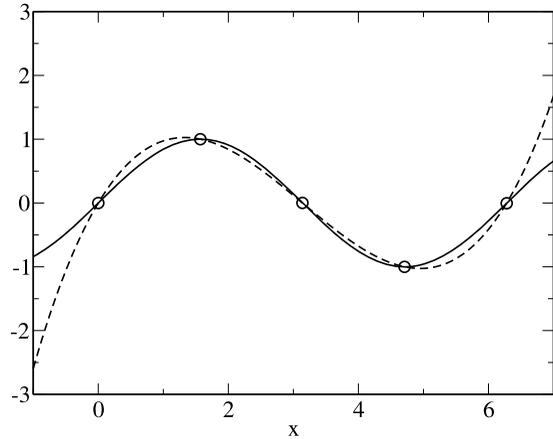\end{array}
\qquad (2.37)
$$

The first column contains the function values $P_i(x) = f_i$. The value $P_{01\ldots n}$ can be calculated using a 1-dimensional data array $p[0] \cdots p[n]$:

for $i := 0$ to n do begin
$\quad p[i] := f[i]$;
$\quad$ for $k := i-1$ downto 0 do
$\qquad p[k] := (p[k+1]*(x-x[k])-p[k]*(x-x[i]))/(x[k]-x[i])$;
end;
$f := p[0]$;

Fig. 2.2 (Interpolating polynomial) The interpolated function (*solid curve*) and the interpolating polynomial (*broken curve*) for the example (2.40) are compared

2.2.5 Error of Polynomial Interpolation

The error of polynomial interpolation [12] can be estimated with the help of the following theorem:

If $f(x)$ is $n + 1$ times differentiable then for each $\overline{x}$ there exists ξ within the smallest interval containing $\overline{x}$ as well as all the x_i with

$$q(\overline{x}) = \prod_{i=0}^{n}(\overline{x} - x_i)\frac{f^{(n+1)}(\xi)}{(n + 1)!}. \tag{2.38}$$

From a discussion of the function

$$\omega(x) = \prod_{i=0}^{n}(x - x_i) \tag{2.39}$$

it can be seen that the error increases rapidly outside the region of the sample points (extrapolation is dangerous!). As an example consider the sample points (Fig. 2.2)

$$f(x) = \sin(x) \quad x_i = 0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi. \tag{2.40}$$

The maximum interpolation error is estimated by ($|f^{(n+1)}| \leq 1$)

$$\left|f(x) - p(x)\right| \leq \left|\omega(x)\right|\frac{1}{120} \leq \frac{35}{120} \approx 0.3 \tag{2.41}$$

whereas the error increases rapidly outside the interval $0 < x < 2\pi$ (Fig. 2.3).

Fig. 2.3 (Interpolation error)
The polynomial $\omega(x)$ is
shown for the example (2.40).
Its roots x_i are given by the x
values of the sample points
(*circles*). Inside the interval
$x_0 \cdots x_4$ the absolute value of
ω is bounded by $|\omega(x)| \leq 35$
whereas outside the interval it
increases very rapidly
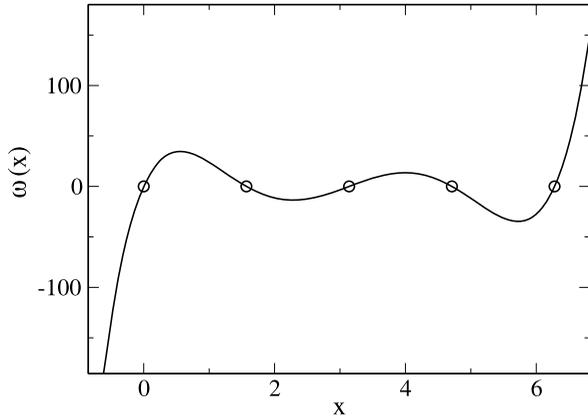
2.3 Spline Interpolation

Polynomials are not well suited for interpolation over a larger range. Often spline
functions are superior which are piecewise defined polynomials [186, 228]. The
simplest case is a linear spline which just connects the sampling points by straight
lines:

$$p_i(x) = y_i + \frac{y_{i+1} - y_i}{x_{i+1} - x_i}(x - x_i) \tag{2.42}$$

$$s(x) = p_i(x) \quad \text{where } x_i \leq x < x_{i+1}. \tag{2.43}$$

The most important case is the cubic spline which is given in the interval $x_i \leq x < x_{i+1}$ by

$$p_i(x) = \alpha_i + \beta_i(x - x_i) + \gamma_i(x - x_i)^2 + \delta_i(x - x_i)^3. \tag{2.44}$$

We want to have a smooth interpolation and assume that the interpolating function
and their first two derivatives are continuous. Hence we have for the inner bound-
aries:

$$i = 0 \cdots n - 1$$

$$p_i(x_{i+1}) = p_{i+1}(x_{i+1}) \tag{2.45}$$

$$p_i'(x_{i+1}) = p_{i+1}'(x_{i+1}) \tag{2.46}$$

$$p_i''(x_{i+1}) = p_{i+1}''(x_{i+1}). \tag{2.47}$$

We have to specify boundary conditions at x_0 and x_n. The most common choice are
natural boundary conditions $s''(x_0) = s''(x_n) = 0$, but also periodic boundary con-
ditions $s''(x_0) = s''(x_n)$, $s'(x_0) = s'(x_n)$, $s(x_0) = s(x_n)$ or given derivative values
$s'(x_0)$ and $s'(x_n)$ are often used. The second derivative is a linear function [244]

$$p_i''(x) = 2\gamma_i + 6\delta_i(x - x_i) \tag{2.48}$$

which can be written using $h_{i+1} = x_{i+1} - x_i$ and $M_i = s''(x_i)$ as

$$p_i''(x) = M_{i+1}\frac{(x-x_i)}{h_{i+1}} + M_i\frac{(x_{i+1}-x)}{h_{i+1}} \quad i = 0 \cdots n-1 \tag{2.49}$$

since

$$p_i''(x_i) = M_i\frac{x_{i+1}-x_i}{h_{i+1}} = s''(x_i) \tag{2.50}$$

$$p_i''(x_{i+1}) = M_{i+1}\frac{(x_{i+1}-x_i)}{h_{i+1}} = s''(x_{i+1}). \tag{2.51}$$

Integration gives with the two constants A_i and B_i

$$p_i'(x) = M_{i+1}\frac{(x-x_i)^2}{2h_{i+1}} - M_i\frac{(x_{i+1}-x)^2}{2h_{i+1}} + A_i \tag{2.52}$$

$$p_i(x) = M_{i+1}\frac{(x-x_i)^3}{6h_{i+1}} + M_i\frac{(x_{i+1}-x)^3}{6h_{i+1}} + A_i(x-x_i) + B_i. \tag{2.53}$$

From $s(x_i) = y_i$ and $s(x_{i+1}) = y_{i+1}$ we have

$$M_i\frac{h_{i+1}^2}{6} + B_i = y_i \tag{2.54}$$

$$M_{i+1}\frac{h_{i+1}^2}{6} + A_i h_{i+1} + B_i = y_{i+1} \tag{2.55}$$

and hence

$$B_i = y_i - M_i\frac{h_{i+1}^2}{6} \tag{2.56}$$

$$A_i = \frac{y_{i+1}-y_i}{h_{i+1}} - \frac{h_{i+1}}{6}(M_{i+1} - M_i). \tag{2.57}$$

Now the polynomial is

$$\begin{aligned} p_i(x) &= \frac{M_{i+1}}{6h_{i+1}}(x-x_i)^3 - \frac{M_i}{6h_{i+1}}(x-x_i-h_{i+1})^3 + A_i(x-x_i) + B_i \\ &= (x-x_i)^3\left(\frac{M_{i+1}}{6h_{i+1}} - \frac{M_i}{6h_{i+1}}\right) + \frac{M_i}{6h_{i+1}}3h_{i+1}(x-x_i)^2 \\ &\quad + (x-x_i)\left(A_i - \frac{M_i}{6h_{i+1}}3h_{i+1}^2\right) + B_i + \frac{M_i}{6h_{i+1}}h_{i+1}^3. \end{aligned} \tag{2.58}$$

Comparison with

$$p_i(x) = \alpha_i + \beta_i(x-x_i) + \gamma_i(x-x_i)^2 + \delta_i(x-x_i)^3 \tag{2.59}$$

gives

$$\alpha_i = B_i + \frac{M_i}{6} h_{i+1}^2 = y_i \tag{2.60}$$

$$\beta_i = A_i - \frac{h_{i+1} M_i}{2} = \frac{y_{i+1} - y_i}{h_{i+1}} - h_{i+1} \frac{M_{i+1} + 2M_i}{6} \tag{2.61}$$

$$\gamma_i = \frac{M_i}{2} \tag{2.62}$$

$$\delta_i = \frac{M_{i+1} - M_i}{6h_{i+1}}. \tag{2.63}$$

Finally we calculate M_i from the continuity of $s'(x)$. Substituting for A_i in $p_i'(x)$ we have

$$p_i'(x) = M_{i+1} \frac{(x - x_i)^2}{2h_{i+1}} - M_i \frac{(x_{i+1} - x)^2}{2h_{i+1}} + \frac{y_{i+1} - y_i}{h_{i+1}} - \frac{h_{i+1}}{6} (M_{i+1} - M_i) \tag{2.64}$$

and from $p_{i-1}'(x_i) = p_i'(x_i)$ it follows

$$M_i \frac{h_i}{2} + \frac{y_i - y_{i-1}}{h_i} - \frac{h_i}{6} (M_i - M_{i-1})$$

$$= -M_i \frac{h_{i+1}}{2} + \frac{y_{i+1} - y_i}{h_{i+1}} - \frac{h_{i+1}}{6} (M_{i+1} - M_i) \tag{2.65}$$

$$M_i \frac{h_i}{3} + M_{i-1} \frac{h_i}{6} + M_i \frac{h_{i+1}}{3} + M_{i+1} \frac{h_{i+1}}{6} = \frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i} \tag{2.66}$$

which is a system of linear equations for the M_i. Using the abbreviations

$$\lambda_i = \frac{h_{i+1}}{h_i + h_{i+1}} \tag{2.67}$$

$$\mu_i = 1 - \lambda_i = \frac{h_i}{h_i + h_{i+1}} \tag{2.68}$$

$$d_i = \frac{6}{h_i + h_{i+1}} \left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i} \right) \tag{2.69}$$

we have

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i \quad i = 1 \cdots n - 1. \tag{2.70}$$

We define for natural boundary conditions

$$\lambda_0 = 0 \quad \mu_n = 0 \quad d_0 = 0 \quad d_n = 0 \tag{2.71}$$

and in case of given derivative values

$$\lambda_0 = 1 \quad \mu_n = 1 \quad d_0 = \frac{6}{h_1} \left(\frac{y_1 - y_0}{h_1} - y_0' \right) \quad d_n = \frac{6}{h_n} \left(y_n' - \frac{y_n - y_{n-1}}{h_n} \right). \tag{2.72}$$

The system of equation has the form

$$
\begin{bmatrix}
2 & \lambda_0 & & & & \\
\mu_1 & 2 & \lambda_1 & & & \\
& \mu_2 & 2 & \lambda_2 & & \\
& & \ddots & \ddots & \ddots & \\
& & & \mu_{n-1} & 2 & \lambda_{n-1} \\
& & & & \mu_n & 2
\end{bmatrix}
\begin{bmatrix}
M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n
\end{bmatrix}
=
\begin{bmatrix}
d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n
\end{bmatrix}.
\tag{2.73}
$$

For periodic boundary conditions we define

$$
\lambda_n = \frac{h_1}{h_1 + h_n} \quad \mu_n = 1 - \lambda_n \quad d_n = \frac{6}{h_1 + h_n} \left(\frac{y_1 - y_n}{h_1} - \frac{y_n - y_{n-1}}{h_n} \right) \tag{2.74}
$$

and the system of equations is (with $M_n = M_0$)

$$
\begin{bmatrix}
2 & \lambda_1 & & & & \mu_1 \\
\mu_2 & 2 & \lambda_2 & & & \\
& \mu_3 & 2 & \lambda_3 & & \\
& & \ddots & \ddots & \ddots & \\
& & & \mu_{n-1} & 2 & \lambda_{n-1} \\
\lambda_n & & & & \mu_n & 2
\end{bmatrix}
\begin{bmatrix}
M_1 \\ M_2 \\ M_3 \\ \vdots \\ M_{n-1} \\ M_n
\end{bmatrix}
=
\begin{bmatrix}
d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{n-1} \\ d_n
\end{bmatrix}.
\tag{2.75}
$$

All this tridiagonal systems can be easily solved with a special Gaussian elimination method (Sects. 5.3, 5.4).

2.4 Rational Interpolation

The use of rational approximants allows to interpolate functions with poles, where polynomial interpolation can give poor results [244]. Rational approximants without poles [90] are also well suited for the case of equidistant x_i, where higher order polynomials tend to become unstable. The main disadvantages are additional poles which are difficult to control and the appearance of unattainable points. Recent developments using the barycentric form of the interpolating function [25, 90, 227] helped to overcome these difficulties.

2.4.1 Padé Approximant

The Padé approximant [13] of order $[M/N]$ to a function $f(x)$ is the rational function

$$
R_{M/N}(x) = \frac{P_M(x)}{Q_N(x)} = \frac{p_0 + p_1 x + \cdots + p_M x^M}{q_0 + q_1 x + \cdots + q_N x^N}
\tag{2.76}
$$

which reproduces the McLaurin series (the Taylor series at $x = 0$) of

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots \tag{2.77}$$

up to order $M + N$, i.e.

$$
\begin{aligned}
f(0) &= R(0) \\
\frac{\mathrm{d}}{\mathrm{d}x} f(0) &= \frac{\mathrm{d}}{\mathrm{d}x} R(0) \\
&\vdots \\
\frac{\mathrm{d}^{(M+N)}}{\mathrm{d}x^{(M+N)}} f(0) &= \frac{\mathrm{d}^{(M+N)}}{\mathrm{d}x^{(M+N)}} R(0).
\end{aligned}
\tag{2.78}
$$

Multiplication gives

$$p_0 + p_1 x + \cdots + p_M x^M = \left(q_0 + q_1 x + \cdots + q_N x^N\right)(a_0 + a_1 x + \cdots) \tag{2.79}$$

and collecting powers of x we find the system of equations

$$
\begin{aligned}
p_0 &= q_0 a_0 \\
p_1 &= q_0 a_1 + q_1 a_0 \\
p_2 &= q_0 a_2 + a_1 q_1 + a_0 q_2 \\
&\vdots \\
p_M &= q_0 a_M + a_{M-1} q_1 + \cdots + a_0 q_M \\
0 &= q_0 a_{M+1} + q_1 a_M + \cdots + q_N a_{M-N+1} \\
&\vdots \\
0 &= q_0 a_{M+N} + q_1 a_{M+N-1} + \cdots + q_N a_M
\end{aligned}
\tag{2.80}
$$

where

$$a_n = 0 \quad \text{for } n < 0 \tag{2.81}$$

$$q_j = 0 \quad \text{for } j > N. \tag{2.82}$$

Example (Calculate the $[3, 3]$ approximant to $\tan(x)$) The Laurent series of the tangent is

$$\tan(x) = x + \frac{1}{3} x^3 + \frac{2}{15} x^5 + \cdots. \tag{2.83}$$

We set $q_0 = 1$. Comparison of the coefficients of the polynomial

$$p_0 + p_1 x + p_2 x^2 + p_3 x^3 = \left(1 + q_1 x + q_2 x^2 + q_3 x^3\right)\left(x + \frac{1}{3}x^3 + \frac{2}{15}x^5\right) \quad (2.84)$$

gives the equations

$$
\begin{aligned}
x^0: \quad & p_0 = 0 \\
x^1: \quad & p_1 = 1 \\
x^2: \quad & p_2 = q_1 \\
x^3: \quad & p_3 = q_2 + \frac{1}{3} \\
x^4: \quad & 0 = q_3 + \frac{1}{3}q_1 \\
x^5: \quad & 0 = \frac{2}{15} + \frac{1}{3}q_2 \\
x^6: \quad & 0 = \frac{2}{15}q_1 + \frac{1}{3}q_3.
\end{aligned}
\quad (2.85)
$$

We easily find

$$p_2 = q_1 = q_3 = 0 \quad q_2 = -\frac{2}{5} \quad p_3 = -\frac{1}{15} \quad (2.86)$$

and the approximant of order [3, 3] is

$$R_{3,3} = \frac{x - \frac{1}{15}x^3}{1 - \frac{2}{5}x^2}. \quad (2.87)$$

This expression reproduces the tangent quite well (Fig. 2.4). Its pole at $\sqrt{10}/2 \approx$ 1.581 is close to the pole of the tangent function at $\pi/2 \approx 1.571$.

## *2.4.2 Barycentric Rational Interpolation*

If the weights of the barycentric form of the interpolating polynomial (2.21) are taken as general parameters $u_i \neq 0$ it becomes a rational function

$$R(x) = \frac{\sum_{i=0}^{n} f_i \frac{u_i}{x - x_i}}{\sum_{i=0}^{n} \frac{u_i}{x - x_i}} \quad (2.88)$$

which obviously interpolates the data points since

$$\lim_{x \to x_i} R(x) = f_i. \quad (2.89)$$

Fig. 2.4 (Padé
approximation to $\tan(x)$) The
Padé approximant ((2.87),
dash-dotted curve)
reproduces the tangent (*full
curve*) quite well
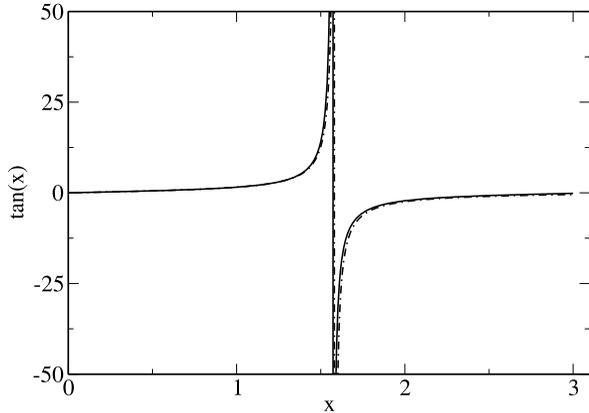
With the polynomials[1]

$$P(x) = \sum_{i=0}^{n} u_i f_i \prod_{j=0; j \neq i}^{n} (x - x_j) = \sum_{i=0}^{n} u_i f_i \frac{\omega(x)}{x - x_i}$$

$$Q(x) = \sum_{i=0}^{n} u_i \prod_{j=0; j \neq i}^{n} (x - x_j) = \sum_{i=0}^{n} u_i \frac{\omega(x)}{x - x_i}$$

a rational interpolating function is given by[2]

$$R(x) = \frac{P(x)}{Q(x)}.$$

Obviously there are infinitely different rational interpolating functions which differ
by the weights $\mathbf{u} = (u_0, u_1 \cdots u_n)$ (an example is shown in Fig. 2.5). To fix the
parameters u_i, additional conditions have to be imposed.

2.4.2.1 Rational Interpolation of Order $[M, N]$

One possibility is to assume that $P(x)$ and $Q(x)$ are of order $\leq M$ and $\leq N$, respec-
tively with $M + N = n$. This gives n additional equations for the $2(n + 1)$ polyno-
mial coefficients. The number of unknown equals $n + 1$ and the rational interpolant
is uniquely determined up to a common factor in numerator and denominator.

---

[1] $\omega(x) = \prod_{i=0}^{n} (x - x_i)$ as in (2.39).

[2] It can be shown that any rational interpolant can be written in this form.
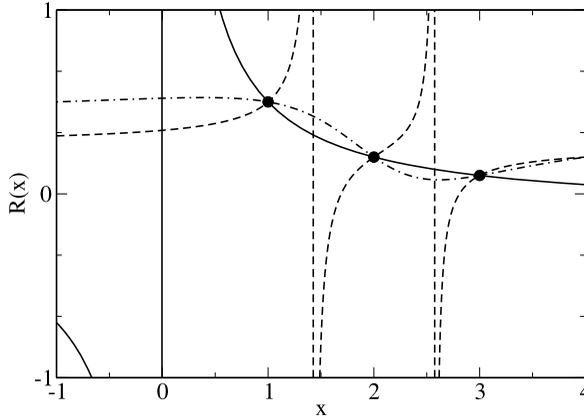
Fig. 2.5 (Rational interpolation) The data points $(1, \frac{1}{2})$, $(2, \frac{1}{5})$, $(3, \frac{1}{10})$ are interpolated by several rational functions. The [1, 1] approximant (2.95) corresponding to $\mathbf{u} = (5, -20, 15)$ is shown by the *solid curve*, the *dashed curve* shows the function $R(x) = \frac{8x^2-36x+38}{10(3x^2-12x+11)}$ which is obtained for $\mathbf{u} = (1, 1, 1)$ and the *dash-dotted curve* shows the function $R(x) = \frac{4x^2-20x+26}{10(5-4x+x^2)}$ which follows for $\mathbf{u} = (1, -1, 1)$ and has no real poles

Example (Consider the data points $f(1) = \frac{1}{2}$, $f(2) = \frac{1}{5}$, $f(3) = \frac{1}{10}$) The polynomials are

$$P(x) = \frac{1}{2}u_0(x-2)(x-3) + \frac{1}{5}u_1(x-1)(x-3) + \frac{1}{10}u_2(x-1)(x-2)$$

$$= 3u_0 + \frac{3}{5}u_1 + \frac{1}{5}u_2 + \left[-\frac{5}{2}u_0 - \frac{4}{5}u_1 - \frac{3}{10}u_2\right]x$$

$$+ \left[\frac{1}{2}u_0 + \frac{1}{5}u_1 + \frac{1}{10}u_2\right]x^2 \tag{2.90}$$

$$Q(x) = u_0(x-2)(x-3) + u_1(x-1)(x-3) + u_2(x-1)(x-2)$$

$$= 6u_0 + 3u_1 + 2u_2 + [-5u_0 - 4u_1 - 3u_2]x + [u_0 + u_1 + u_2]x^2. \tag{2.91}$$

To obtain a [1, 1] approximant we have to solve the equations

$$\frac{1}{2}u_0 + \frac{1}{5}u_1 + \frac{1}{10}u_2 = 0 \tag{2.92}$$

$$u_0 + u_1 + u_2 = 0 \tag{2.93}$$

which gives

$$u_2 = 3u_0 \quad u_1 = -4u_0 \tag{2.94}$$

and thus

$$R(x) = \frac{\frac{6}{5}u_0 - \frac{1}{5}u_0 x}{2u_0 x} = \frac{6-x}{10x}. \tag{2.95}$$

General methods to obtain the coefficients u_i for a given data set are described in [25, 227]. They also allow to determine unattainable points corresponding to $u_i = 0$ and to locate the poles. Without loss of generality it can be assumed [227] that $M \geq N$.[3]

Let $P(x)$ be the unique polynomial which interpolates the product $f(x)Q(x)$

$$P(x_i) = f(x_i)Q(x_i) \quad i = 0 \cdots M. \tag{2.96}$$

Then from (2.31) we have

$$f(x)Q(x) - P(x) = (fQ)[x_0 \cdots x_M, x](x - x_0) \cdots (x - x_M). \tag{2.97}$$

Setting

$$x = x_i \quad i = M + 1 \cdots n \tag{2.98}$$

we have

$$f(x_i)Q(x_i) - P(x_i) = (fQ)[x_0 \cdots x_M, x_i](x_i - x_0) \cdots (x - x_M) \tag{2.99}$$

which is zero if $P(x_i)/Q(x_i) = f_i$ for $i = 0 \cdots n$. But then

$$(fQ)[x_0 \cdots x_M, x_i] = 0 \quad i = M + 1 \cdots n. \tag{2.100}$$

The polynomial $Q(x)$ can be written in Newtonian form (2.30)

$$Q(x) = \sum_{i=0}^{N} v_i \prod_{j=0}^{i-1}(x - x_j) = v_0 + v_1(x - x_0) + \cdots + v_N(x - x_0) \cdots (x - x_{N-1}). \tag{2.101}$$

With the abbreviation

$$g_j(x) = x - x_j \quad j = 0 \cdots N \tag{2.102}$$

we find

$$(fg_j)[x_0 \cdots x_M, x_i] = \sum_{k=0\cdots M,i} \frac{f(x_k)g(x_k)}{\prod_{r\neq k}(x_k - x_r)} = \sum_{k=0\cdots M,i,k\neq j} \frac{f(x_k)}{\prod_{r\neq k,r\neq j}(x_k - x_r)}$$

$$= f[x_0 \cdots x_{j-1}, x_{j+1} \cdots x_M, x_i] \tag{2.103}$$

---

[3]The opposite case can be treated by considering the reciprocal function values $1/f(x_i)$.

Fig. 2.6 (Interpolation of a step function) A step function with uniform x-values (*circles*) is interpolated by a polynomial (*full curve*), a cubic spline (*dashed curve*) and with the rational Floater-Hormann $d = 1$ function ((2.105), *dash-dotted curve*). The rational function behaves similar to the spline function but provides in addition an analytical function with continuous derivatives
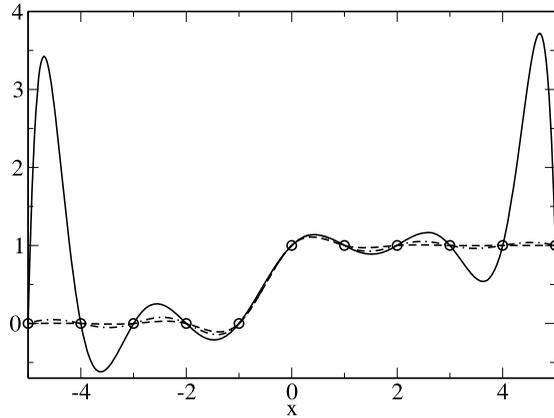
which we apply repeatedly to (2.100) to get the system of $n - M = N$ equations for $N + 1$ unknowns

$$\sum_{j=0}^{N} v_j f[x_j, x_{j+1} \cdots x_M, x_i] = 0 \quad i = M + 1 \cdots n \qquad (2.104)$$

from which the coefficients v_j can be found by Gaussian elimination up to a scaling factor. The Newtonian form of $Q(x)$ can then be converted to the barycentric form as described in [271].

2.4.2.2 Rational Interpolation Without Poles

Polynomial interpolation of larger data sets can be ill behaved, especially for the case of equidistant x-values. Rational interpolation without poles can be a much better choice here (Fig. 2.6).

Berrut [23] suggested to choose the following weights

$$u_k = (-1)^k.$$

With this choice $Q(x)$ has no real roots. Floater and Hormann [90] used the different choice

$$u_k = (-1)^{k-1} \left(\frac{1}{x_{k+1} - x_k} + \frac{1}{x_k - x_{k-1}} \right) \quad k = 1 \cdots n - 1$$

$$u_0 = -\frac{1}{x_1 - x_0} \quad u_n = (-1)^{n-1} \frac{1}{x_n - x_{n-1}} \qquad (2.105)$$

which becomes very similar for equidistant x-values.

Table 2.1 Floater-Hormann weights for uniform data

$|u_k|$	d
$1, 1, 1, \ldots, 1, 1, 1$	0
$1, 2, 2, 2, \ldots, 2, 2, 2, 1$	1
$1, 3, 4, 4, 4, \ldots, 4, 4, 4, 3, 1$	2
$1, 4, 7, 8, 8, 8, \ldots, 8, 8, 8, 7, 4, 1$	3
$1, 5, 11, 15, 16, 16, 16, \ldots, 16, 16, 16, 15, 11, 5, 1$	4

Floater and Hormann generalized this expression and found a class of rational interpolants without poles given by the weights

$$u_k = (-1)^{k-d} \sum_{i=\max(k-d,0)}^{\min(k,n-d)} \prod_{j=i, j \neq k}^{i+d} \frac{1}{|x_k - x_j|} \tag{2.106}$$

where $0 \leq d \leq n$ and the approximation order increases with d. In the uniform case this simplifies to (Table 2.1)

$$u_k = (-1)^{k-d} \sum_{i=\min(k-d,0)}^{\max(k,n-d)} \binom{d}{k-i}. \tag{2.107}$$

2.5 Multivariate Interpolation

The simplest 2-dimensional interpolation method is bilinear interpolation.[4] It uses linear interpolation for both coordinates within the rectangle $x_i \leq x \leq x_{i+1}$, $y_i \leq y \leq y_{i+1}$:

$$
\begin{aligned}
&p(x_i + h_x, y_i + h_y) \\
&= p(x_i + h_x, y_i) + h_y \frac{p(x_i + h_x, y_{i+1}) - p(x_i + h_x, y_i)}{y_{i+1} - y_i} \\
&= f(x_i, y_i) + h_x \frac{f(x_{i+1}, y_i) - f(x_i, y_i)}{x_{i+1} - x_i} \\
&\quad + h_y \frac{f(x_i, y_{i+1}) + h_x \frac{f(x_{i+1}, y_{i+1}) - f(x_i, y_{i+1})}{x_{i+1} - x_i} - f(x_i, y_i) - h_x \frac{f(x_{i+1}, y_i) - f(x_i, y_i)}{x_{i+1} - x_i}}{y_{i+1} - y_i}
\end{aligned}
\tag{2.108}
$$

---

[4]Bilinear means linear interpolation in two dimensions. Accordingly linear interpolation in three dimensions is called trilinear.

Fig. 2.7 Bispline interpolation
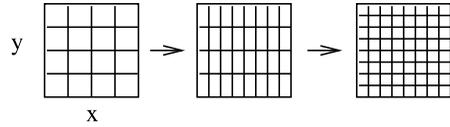
which can be written as a two dimensional polynomial

$$p(x_i + h_x, y_i + h_y) = a_{00} + a_{10}h_x + a_{01}h_y + a_{11}h_xh_y \qquad (2.109)$$

with

$$
\begin{aligned}
a_{00} &= f(x_i, y_i) \\
a_{10} &= \frac{f(x_{i+1}, y_i) - f(x_i, y_i)}{x_{i+1} - x_i} \\
a_{01} &= \frac{f(x_i, y_{i+1}) - f(x_i, y_i)}{y_{i+1} - y_i} \\
a_{11} &= \frac{f(x_{i+1}, y_{i+1}) - f(x_i, y_{i+1}) - f(x_{i+1}, y_i) + f(x_i, y_i)}{(x_{i+1} - x_i)(y_{i+1} - y_i)}.
\end{aligned}
\qquad (2.110)
$$

Application of higher order polynomials is straightforward. For image processing purposes bicubic interpolation is often used.

If high quality is needed more sophisticated interpolation methods can be applied. Consider for instance two-dimensional spline interpolation on a rectangular mesh of data to create a new data set with finer resolution[5]

$$f_{i,j} = f(ih_x, jh_y) \quad \text{with } 0 \le i < N_x \quad 0 \le j < N_y. \qquad (2.111)$$

First perform spline interpolation in x-direction for each data row j to calculate new data sets

$$f_{i',j} = s(x_{i'}, f_{ij}, 0 \le i < N_x) \quad 0 \le j \le N_y \quad 0 \le i' < N'_x \qquad (2.112)$$

and then interpolate in y direction to obtain the final high resolution data (Fig. 2.7)

$$f_{i',j'} = s(y_{j'}, f_{i'j}, 0 \le j < N_y) \quad 0 \le i' < N'_x \quad 0 \le j' < N'_y. \qquad (2.113)$$

2.6 Problems

Problem 2.1 (Polynomial interpolation) This computer experiment interpolates a given set of n data points by

---

[5]A typical task of image processing.

Table 2.2 Zener diode
voltage/current data

Voltage	-1.5	-1.0	-0.5	0.0
Current	-3.375	-1.0	-0.125	0.0

Table 2.3 Additional
voltage/current data

Voltage	1.0	2.0	3.0	4.0	4.1	4.2	4.5
Current	0.0	0.0	0.0	0.0	1.0	3.0	10.0

- a polynomial

$$p(x) = \sum_{i=0}^{n} f_i \prod_{k=0, k \neq i}^{n} \frac{x - x_k}{x_i - x_k}, \tag{2.114}$$

- a linear spline which connects successive points by straight lines

$$s_i(x) = a_i + b_i(x - x_i) \quad \text{for } x_i \leq x \leq x_{i+1} \tag{2.115}$$

- a cubic spline with natural boundary conditions

$$s(x) = p_i(x) = \alpha_i + \beta_i(x - x_i) + \gamma_i(x - x_i)^2 + \delta_i(x - x_i)^3 \quad x_i \leq x \leq x_{i+1} \tag{2.116}$$

$$s''(x_n) = s''(x_0) = 0 \tag{2.117}$$

- a rational function without poles

$$R(x) = \frac{\sum_{i=0}^{n} f_i \frac{u_i}{x - x_i}}{\sum_{i=0}^{n} \frac{u_i}{x - x_i}} \tag{2.118}$$

  with weights according to Berrut

$$u_k = (-1)^k \tag{2.119}$$

  or Floater-Hormann

$$u_k = (-1)^{k-1} \left(\frac{1}{x_{k+1} - x_k} + \frac{1}{x_k - x_{k-1}} \right) \quad k = 1 \cdots n - 1 \tag{2.120}$$

$$u_0 = -\frac{1}{x_1 - x_0} \quad u_n = (-1)^{n-1} \frac{1}{x_n - x_{n-1}}. \tag{2.121}$$

- Interpolate the data (Table 2.2) in the range $-1.5 < x < 0$.
- Now add some more sample points (Table 2.3) for $-1.5 < x < 4.5$.
- Interpolate the function $f(x) = \sin(x)$ at the points $x = 0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi$. Take more sample points and check if the quality of the fit is improved.

Table 2.4 Pulse and step function data

x	−3	−2	−1	0	1	2	3
y_{pulse}	0	0	0	1	0	0	0
y_{step}	0	0	0	1	1	1	1

Table 2.5 Data set for two-dimensional interpolation

x	0	1	2	0	1	2	0	1	2
y	0	0	0	1	1	1	2	2	2
f	1	0	−1	0	0	0	−1	0	1

- Investigate the oscillatory behavior for a discontinuous pulse or step function as given by the data (Table 2.4).

Problem 2.2 (Two-dimensional interpolation) This computer experiment uses bi-linear interpolation or bicubic spline interpolation to interpolate the data (Table 2.5) on a finer grid $\Delta x = \Delta y = 0.1$.