Numerical Calculus

Pătrulescu Flavius

Technical University of Cluj-Napoca

2024

References

1. K.E. Atkinson, *An Introduction to Numerical Analysis*, 2nd Ed., John Wiley & Sons, New-York, 1989
2. J.P. Berrut, L.N. Trefethen, Barycentric Lagrange Interpolation, Siam Review, **46**, no. 2 (2004), pp. 501–517.
3. R.L. Burden, D.J. Faires, A.M. Burden, *Numerical Analysis*, 10th Ed., Cengage Learning, Boston, 2022.
4. G. Dahlquist, A. Björk, *Numerical Methods in Scientific Computing*, Vol. I, SIAM, Philadelphia, 2008.
5. A. Quarteroni, R. Sacco, F. Saleri, *Numerical Mathematics*, Springer-Verlag, New-York, 2000.
6. E. Süli, D. Mayers, *An Introduction to Numerical Analysis*, Cambridge University Press, Cambridge, 2003.

Lagrange Polynomial Interpolation (Burden *et al.*, 2022)

If $x_0, x_1, \ldots, x_n$ are $n+1$ distinct points and f is a function whose values are given at these numbers, then a unique polynomial $L_n(x)$ of degree at most n exists with

$$f(x_k) = L_n(x_k), \quad k = 0, 1, \ldots, n.$$

This polynomial is given by

$$L_n(x) = f(x_0)l_0(x) + f(x_1)l_1(x) + \ldots + f(x_n)l_n(x_n)$$

where

$$\begin{aligned} l_k(x) &= \frac{(x - x_0)\ldots(x - x_{k-1})(x - x_{k+1})\ldots(x - x_n)}{(x_k - x_0)\ldots(x_k - x_{k-1})(x_k - x_{k+1})\ldots(x_k - x_n)} \\ &= \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}, \; k = \overline{0, n}. \end{aligned}$$

Lagrange Polynomial Interpolation (Burden *et al.*, 2022)

$$l_k(x_j) = \delta_{kj} = \begin{cases} 1, k = j \\ 0, k \neq j \end{cases}, \quad \sum_{k=0}^{n} l_k(x) = 1$$

Suppose $f \in C^{n+1}[a, b]$. Then, for each $x \in [a, b]$, a number $\xi(x)$ (generally unknown) between $\min\{x_0, x_1, \ldots, x_n\}$ and $\max\{x_0, x_1, \ldots, x_n\}$ (and hence in (a, b)) exists with

$$f(x) = L_n(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x - x_0)(x - x_1)\ldots(x - x_n)$$

Note that the error form for the Lagrange polynomial is quite similar to that for the Taylor polynomial. The nth Taylor polynomial about x_0 concentrates all the known information at x_0. The Lagrange polynomial of degree n uses information at the distinct numbers $x_0, x_1, \ldots, x_n$ and in place of $(x - x_0)^n$, its error formula uses a product $(x - x_0)(x - x_1)\ldots(x - x_n)$.

Lagrange Polynomial Interpolation (Dahlquist and Björk, 2008; Berrut and Trefethen, 2004)

Quite often it is asserted that the Lagrange form is a bad choice for practical computations, since

- for each new value of x the polynomials $l_k(x)$ have to be recomputed at a cost $O(4n^2)$;
- adding a new data point $(x_{n+1}, f(x_{n+1}))$ will require a new computation from scratch.

The Lagrange representation can easily be rewritten in two more attractive forms (*Newton form* and *Barycentric form*) which are both eminently suitable for computation.

Divided Differences (Burden *et al.*, 2022)

Although L_n is unique, there are alternate algebraic representations that are useful in certain situations. The divided differences of f with respect to $x_0, x_1, \ldots, x_n$ are used to express L_n in the form

$$L_n(x) = a_0 + a_1(x - x_0) + \ldots + a_n(x - x_0)(x - x_1)\ldots(x - x_{n-1})$$

for appropriate constants $a_0, a_1, \ldots, a_n$.
The *zeroth divided difference of the function* f with respect to x_i, denoted with $[x_i; f]$, is simply the value of f at x_i

$$[x_i; f] = f(x_i)$$

Divided Differences (Burden *et al.*, 2022)

The *first divided difference of* f with respect to x_i and x_{i+1} is denoted $[x_i, x_{i+1}; f]$ and defined as

$$[x_i, x_{i+1}; f] = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

The *kth divided difference* relative to $x_i, x_{i+1}, \ldots, x_{i+k}$ is

$$[x_i, \ldots, x_{i+k}; f] = \frac{[x_{i+1}, \ldots, x_{i+k}; f] - [x_i, \ldots, x_{i+k-1}; f]}{x_{i+k} - x_i}$$

where $[x_{i+1}, \ldots, x_{i+k}; f]$ and $[x_i, \ldots, x_{i+k-1}; f]$ are the $(k-1)$th divided differences.
The process ends with the single *nth divided difference*

$$[x_0, \ldots, x_n; f] = \frac{[x_1, \ldots, x_n; f] - [x_0, \ldots, x_{n-1}; f]}{x_n - x_0}$$

Nodal Polynomial (Burden *et al.*,2022)

$$l(x) = (x - x_0)(x - x_1)\dots(x - x_n)$$

$$l \in \Pi_{n+1}\,(\deg(l) = n + 1)$$

$$l(x_i) = 0, \quad \forall\, i = \overline{0, n}$$

$$\begin{aligned} l'(x) &= (x - x_1)(x - x_2)\dots(x - x_n) + \\ &+ (x - x_0)(x - x_2)\dots(x - x_n) + \\ &+ \dots + (x - x_0)(x - x_1)\dots(x - x_{n-1}) = \\ &= l(x)\Big(\frac{1}{x - x_0} + \frac{1}{x - x_1} + \dots + \frac{1}{x - x_n}\Big) \end{aligned}$$

$$l'(x_i) = (x_i - x_0)\dots(x_i - x_{i-1})(x_i - x_{i+1})\dots(x_i - x_n),\ \forall\, i = \overline{0, n}$$

Divided Differences (Burden *et al.*, 2022)

$$[x_0, \ldots, x_n; \alpha f + \beta g] = \alpha[x_0, \ldots, x_n; f] + \beta[x_0, \ldots, x_n; g]$$

$$[x_0, \ldots, x_n; f] = \sum_{i=0}^{n} \frac{f(x_i)}{\prod_{j=0, j\neq i}^{n} (x_i - x_j)} = \sum_{i=0}^{n} \frac{f(x_i)}{l'(x_i)}$$

For any permutation $(i_0, \ldots, i_n)$ of integers $(0, \ldots, n)$

$$[x_{i_0}, \ldots, x_{i_n}; f] = [x_0, \ldots, x_n; f]$$

Suppose that $f \in C^n[a, b]$ and $x_0, x_1, \ldots, x_n$ are distinct number in $[a, b]$. Then $\exists!\, \xi \in (a, b)$ with

$$[x_0, \ldots, x_n; f] = \frac{f^{(n)}(\xi)}{n!}$$

Divided Differences (Burden *et al.*, 2022)

$$[x_0, \ldots, x_n; x^p] = \begin{cases} 0, \ p < n \\ 1, \ p = n \\ \sum_{k=0}^{n} x_k, \ p = n+1 \\ \left(\sum_{k=0}^{n} x_k\right)^2 - \sum_{0 \le i < k \le n} x_i x_k, \ p = n+2 \end{cases}$$

$$[a, a+h, \ldots, a+nh; f] = \frac{1}{n! h^n} \sum_{k=0}^{n} (-1)^{n-k} \binom{n}{k} f(a+kh)$$

$$[x_0, \ldots, x_k, \ldots, x_n; f(x)] = [x_0, \ldots, x_k; \frac{f(x)}{(x - x_{k+1}) \ldots (x - x_n)}] +$$

$$+ [x_{k+1}, \ldots, x_n; \frac{f(x)}{(x - x_0) \ldots (x - x_k)}]$$

Newton table of divided differences (Burden *et al.*,2022)

The *Newton table* requires $n^2 + n$ subtractions and $\frac{n^2+n}{2}$ divisions.

$$
\begin{array}{lllll}
f(x_0)| & & & & \\
f(x_1)| & [x_0, x_1; f] = \frac{f(x_1)-f(x_0)}{x_1-x_0} & & & \\
f(x_2)| & [x_1, x_2; f] = \frac{f(x_2)-f(x_1)}{x_2-x_1} & [x_0, x_1, x_2; f] = \frac{[x_1,x_2;f]-[x_0,x_1;f]}{x_2-x_0} & & \dots \\
 & [x_2, x_3; f] = \frac{f(x_3)-f(x_2)}{x_3-x_2} & [x_1, x_2, x_3; f] = \frac{[x_2,x_3;f]-[x_1,x_2;f]}{x_3-x_1} & & \dots \\
 & & [x_2, x_3, x_4; f] = \frac{[x_3,x_4;f]-[x_2,x_3;f]}{x_4-x_2} & & \dots \\
\vdots & \vdots & \vdots & & \vdots \\
f(x_{n-2})| & [x_{n-2}, x_{n-1}; f] = \frac{f(x_{n-1})-f(x_{n-2})}{x_{n-1}-x_{n-2}} & [x_{n-2}, x_{n-1}, x_n; f] = \frac{[x_{n-1},x_n;f]-[x_{n-2},x_{n-1};f]}{x_n-x_{n-2}} & & \\
f(x_{n-1})| & [x_{n-1}, x_n; f] = \frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}} & * & & \\
f(x_n)| & * & * & &
\end{array}
$$

Complexity: $O(\frac{3}{2}n^2)$

Newton form of Lagrange interpolation(Burden *et al.*, 2022)

$$
\begin{aligned}
N_n(x) &= \\
&= \underbrace{[x_0; f]}_{=f(x_0)} + [x_0, x_1; f](x - x_0) + \\
&+ [x_0, x_1, x_2; f](x - x_0)(x - x_1) + \\
&+ \ldots + [x_0, x_1, \ldots, x_n; f](x - x_0)\ldots(x - x_{n-1}) = \\
&= f(x_0) + \sum_{k=1}^{n} [x_0, x_1, \ldots, x_k; f](x - x_0)\ldots(x - x_{k-1})
\end{aligned}
$$

Newton form of Lagrange interpolation (Dahlquist and Björk, 2008)

For given interpolation points, the divided differences in Newton's interpolation formula depend on the order in which the points x_i are introduced. Mathematically, all orderings give the same unique interpolation polynomial. However, the condition number for the coefficients in the Newton polynomial can depend strongly on the ordering of the interpolation points. If the point x where the polynomial is to be evaluated is known, then an ordering such that

$$|x - x_1| \leq |x - x_2| \leq \ldots \leq |x - x_n|$$

can be recommended.

Modified Lagrange Interpolation (Berrut and Trefethen, 2004)

We consider the *nodal polynomial*

$$l(x) = (x - x_0)(x - x_1)\dots(x - x_n)$$

and *barycentric weights (support coefficients)*:

$$w_k = \frac{1}{\prod_{j=0, j\neq k}^{n} (x_k - x_j)} = \frac{1}{l'(x_k)}, \quad k = \overline{0, n}$$

The polynomials l_k can then be written as

$$l_k(x) = l(x)\frac{w_k}{x - x_k}, k = \overline{0, n}$$

Modified Lagrange Interpolation (Berrut and Trefethen, 2004)

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

$$L_n(x) = l(x) \sum_{k=0}^{n} \frac{w_k}{x - x_k} f(x_k)$$

Now Lagrange interpolation is a formula requiring $O(2n^2)$ operations for calculating some quantities independent of x, the numbers w_k, followed by $O(n)$ operations for evaluating L_n once these numbers are known.

Barycentric Lagrange Interpolation (Berrut and Trefethen, 2004)

Since $L_n \equiv f$, $\forall\, f \in \Pi_n$, we have

$$f \equiv 1 \Rightarrow 1 = l(x) \sum_{k=0}^{n} \frac{w_k}{x - x_k} \cdot 1 \Rightarrow l(x) = \frac{1}{\sum_{k=0}^{n} \frac{w_k}{x - x_k}}$$

Barycentric form for Lagrange interpolation

$$L_n(x) = \frac{\sum_{k=0}^{n} \frac{w_k}{x - x_k} f(x_k)}{\sum_{k=0}^{n} \frac{w_k}{x - x_k}}$$

Barycentric Lagrange Interpolation (Berrut and Trefethen, 2004)

The barycentric formula is a Lagrange formula, but one with a special and beautiful symmetry. The weights w_k appear in the denominator exactly as in the numerator, except for the data factors $f(x_k)$. This means that any common factor in all the weights w_k can be canceled without affecting the value of L_n. For certain special sets of nodes x_k, we can give explicit formulas for the barycentric weights w_k.

Barycentric Lagrange Interpolation (Berrut and Trefethen, 2004)

For equidistant nodes on $[a, b]$,

$$x_k = a + kh,\ h = \frac{b-a}{n},\ k = \overline{0, n}$$

we obtain

$$w_k = \frac{(-1)^{n-k}}{h^n n!}\binom{n}{k} = \frac{(-1)^n}{h^n n!}(-1)^k\binom{n}{k}$$

$$L_n(x) = \frac{\sum_{k=0}^{n} \frac{\frac{(-1)^n}{h^n n!}(-1)^k\binom{n}{k}}{x - x_k} f(x_k)}{\sum_{k=0}^{n} \frac{\frac{(-1)^n}{h^n n!}(-1)^k\binom{n}{k}}{x - x_k}} = \frac{\sum_{k=0}^{n} \frac{(-1)^k\binom{n}{k}}{x - x_k} f(x_k)}{\sum_{k=0}^{n} \frac{(-1)^k\binom{n}{k}}{x - x_k}}$$

Barycentric versus Newton (Berrut and Trefethen, 2004)

A notable advantage of Barycentric over Newton interpolation is that the weights w_k do not depend on the data $f(x_k)$. This feature permits the interpolation of as many functions as desired once the weights are known, whereas Newton interpolation requires the recomputation of the divided difference table for each new function.

This is not to say that Newton interpolation has no advantages. One is that it leads to elegant methods to incorporate information on derivatives $f^{(p)}(x_k)$ when solving the so-called Hermite interpolation problems.

Interpolation matrix (Quarteroni *et al.*, 2000)

An infinite triangular array (matrix) of nodes

$$
X: \quad
\begin{array}{ccccccc}
x_{00} & & & & & & \\
x_{10} & x_{11} & & & & & \\
x_{20} & x_{21} & x_{22} & & & & \\
\cdots & \cdots & \cdots & & & & \\
x_{n-10} & x_{n-11} & \cdots & x_{n-1n-2} & x_{n-1n-1} & & \\
x_{n0} & x_{n1} & \cdots & x_{nn-2} & x_{nn-1} & x_{nn} & \\
x_{n+10} & x_{n+11} & \cdots & x_{n+1n-2} & x_{n+1n-1} & x_{n+1n} & x_{n+1n+1} \\
\cdots & \cdots & \cdots & \cdots & & &
\end{array}
$$

Drawbacks of polynomial interpolation (Quarteroni *et al.*, *2000*)

The *maximum norm*:

$$\|f\|_\infty = \max_{x\in[a,b]} |f(x)|, \forall\, f \in C^0[a,b]$$

Let X a *interpolation matrix* on $[a,b]$, whose entries x_{ij} represent points of $[a,b]$, with the assumption that on each row the entries are all distinct.
$\forall\, n > 0$ the $n+1$ row of X contains $n+1$ distinct values $\Rightarrow$ we can uniquely define for a given function f an interpolating polynomial L_n of degree n.
The *interpolation error* is defined by

$$E_{n,\infty}(X) = \|f - L_n\|_\infty,\ n = 0, 1, \ldots$$

Drawbacks of polynomial interpolation (Quarteroni *et al.*, *2000*)

Let $P_n^* \in \Pi_n$ be the *best approximation polynomial* for which

$$E_n^* := \|f - P_n^*\|_\infty \leq \|f - Q_n\|_\infty, \quad \forall\, Q_n \in \Pi_n$$

Let $f \in C[a, b]$ and X be an interpolation matrix on $[a, b]$. Then

$$E_{n,\infty}(X) \leq E_n^*(1 + \Lambda_n(X)), n = 0, 1, \ldots$$

where $\Lambda_n(X)$ denotes the *Lebesgue constant* of X defined as

$$\Lambda_n(X) = \left\| \sum_{j=0}^{n} |l_{nj}| \right\|_\infty$$

where $l_{nj} \in \Pi_n$ is the jth characteristic (fundamental) Lagrange polynomial associated with the nth row of X,

$$l_{nj}(x_{nk}) = \delta_{jk},\ k = 0, 1, \ldots$$

Drawbacks of polynomial interpolation (Quarteroni *et al.*, *2000*)

E_n^* does not depend on X, all the information concerning the effects of X on $E_{n,\infty}(X)$ must be looked in $\Lambda_n(X)$.
For any possible choice of X there exists a constant $C > 0$ s.t.

$$\Lambda_n(X) > \frac{2}{\pi} ln(n+1) - C, \; n = 0, 1, \ldots$$

This property shows that

$$\Lambda_n(X) \to \infty \text{ as } n \to \infty.$$

This fact has important consequences: in particular, it can be proved that, given an interpolation matrix X on an interval $[a, b]$, there always exists $f \in C[a, b]$, s.t. L_n does not converge uniformly (that is, in the maximum norm) to f.

Drawbacks of polynomial interpolation (Quarteroni *et al.*, *2000*)

Thus, Lagrange polynomial interpolation does not allow approximating any continuous function (see the Runge example). Moreover,

$$\Lambda_n(X_c) \leq \frac{2}{\pi} \ln n + 1$$

where X_c is the array matrix whose nth row contains the zeros of Chebyshev polynomial of the fist kind T_n.
Runge example:

$$f(x) = \frac{1}{1+x^2}, \quad x \in [-5, 5]$$

Lagrange interpolation on equally spaced nodes diverges for $|x| > 3.63...$

Stability of polynomial interpolation (Quarteroni *et al.*, 2000)

Let a set of function values $\{\hat{f}(x_i)\}$ that is a perturbation of the data $\{f(x_i)\}$ relative to the nodes x_i, $i = \overline{0,n}$. We denote by $\hat{L}_n$ the interpolation polynomial on the set values $\hat{f}(x_i)$.

$$\begin{aligned} \|L_n - \hat{L}_n\|_\infty &= \max_{a \le x \le b} \left| \sum_{i=0}^{n} \left(f(x_i) - \hat{f}(x_i)\right) l_i(x) \right| \le \\ &\le \Lambda_n(X) \max_{i=\overline{0,n}} |f(x_i) - \hat{f}(x_i)| \end{aligned}$$

As a consequence, small changes on the data give rise to small changes on the interpolating polynomial only if the Lebesgue constant is small. This constant plays the role of the *condition number for the interpolation problem.*

Stability of polynomial interpolation (Quarteroni *et al.*, 2000)

On equally spaced nodes:

$$\Lambda_n(X) \approx \frac{1}{e}\frac{2^{n+1}}{n \ln n}$$

This shows that for n large, this form of interpolation can become unstable.
Example: in the interval $[-1, 1]$ let us interpolate the function $f(x) = \sin(2\pi x)$ at 22 equally spaced nodes x_i. For a perturbed set of values $\hat{f}(x_i)$ with $\max_{i=\overline{0,21}} \left| f(x_i) - \hat{f}(x_i) \right| \approx 9.5 \cdot 10^{-4}$ we obtain

$$\| L_{21} - \hat{L}_{21} \| \approx 2.1635, \text{ and } \Lambda_{21} = 24000$$

Osculating polynomials (Burden *et al.*, 2022)

Osculating polynomials generalize both the Taylor polynomials and the Lagrange polynomials. Suppose that we are given $n+1$ distinct numbers $x_0, x_1, \ldots, x_n \in [a, b]$ and non-negative integers $m_0, m_1, \ldots, m_n \in \mathbb{N}$ and

$$m = \max\{m_0, m_1, \ldots, m_n\}.$$

The *osculating polynomial* P approximating a function $f \in C^m[a, b]$ at x_i, $i = \overline{0, n}$ is the polynomial of least degree that verifies

$$P^{(k)}(x_i) = f^{(k)}(x_i),\ k = \overline{0, m_i},\ i = \overline{0, n}$$

Osculating polynomials (Burden *et al.*, 2022)

The degree of P is at most

$$deg(P) = \sum_{i=0}^{n} m_i + n$$

because the number of conditions to be satisfied is

$$\sum_{i=0}^{n} m_i + (n+1).$$

When $n = 0$, the osculating polynomial is the m_0th Taylor polynomial for f at x_0. When $m_i = 0$, $\forall\, i = \overline{0, n}$, the osculating polynomial is the nth Lagrange polynomial that interpolates f on $x_0, x_1, \ldots, x_n$.

Hermite interpolation with double nodes (Burden *et al.*, 2022)

When $m_i = 1$, we obtain the *Hermite polynomial with double nodes*, denoted by H_{2n+1}. It verifies

$$H_{2n+1}(x_i) = f(x_i), \quad H'_{2n+1}(x_i) = f'(x_i),\ i = \overline{0, n}.$$

For a given function f, this polynomial agrees with f at x_0, x_1, ..., x_n. In addition, since its first derivative agrees with those of f, it has the same *shape* as the function at $(x_i, f(x_i))$ in the sense that the tangent lines to the polynomial and the function agree.

Hermite interpolation with double nodes (Burden *et al.*, 2022)

If $f \in C^1[a, b]$ and $x_0, x_1, \ldots, x_n \in [a, b]$ are distinct, the unique polynomial of least degree agreeing with f and f' at x_i, $i = \overline{0, n}$ is the Hermite polynomial of degree at most $2n + 1$ given by

$$H_{2n+1}(x) = \sum_{k=0}^{n} f(x_k) h_k(x) + \sum_{k=0}^{n} f'(x_k) \hat{h}_k(x)$$

where

$$h_k(x) = \Big(1 - 2(x - x_k) l_k'(x_k)\Big) l_k^2(x),$$
$$\hat{h}_k(x) = (x - x_k) l_k^2(x)$$

and l_k denotes the kth Lagrange coefficient polynomial.

Hermite interpolation with double nodes (Burden *et al.*, 2022)

If $f \in C^{2n+2}[a, b]$ then

$$f(x) = H_{2n+1}(x) + \frac{(x - x_0)^2 \dots (x - x_n)^2}{(2n + 2)!} f^{(2n+2)}(\xi(x))$$

where $\xi(x) \in (a, b)$.
The need to determine and evaluate the Lagrange polynomials and their derivatives makes the procedure tedious even for small values of n. There is an alternative method for generating Hermite approximations that has as its basis the Newton interpolatory divided-difference formula.

Hermite interpolation with double nodes (Burden *et al.*, 2022)

$$H_{2n+1}(x) = f(z_0) + \sum_{k=1}^{2n+1} [z_0, z_1, \ldots, z_k; f](x - z_0)\ldots(x - z_{k-1}).$$

where $z_0, \ldots, z_{2n+1}$ are given by $z_{2i} = z_{2i+1} = x_i$, $i = \overline{0, n}$ and

$$\begin{array}{cc|cc|c}
z_0 & f(z_0) & f'(z_0) & [z_0, z_1, z_2; f] & \ldots \\
z_1 & f(z_1) & [z_1, z_2; f] = \frac{f(z_2) - f(z_1)}{z_2 - z_1} & [z_1, z_2, z_3; f] & \ldots; \\
z_2 & f(z_2) & f'(z_2) & [z_2, z_3, z_4; f] & \ldots \\
z_3 & f(z_3) & [z_3, z_4; f] = \frac{f(z_4) - f(z_3)}{z_4 - z_3} & [z_3, z_4, z_5; f] & \ldots \\
\vdots & \vdots & \vdots & \vdots & \\
z_{2n-2} & f(z_{2n-2}) & f'(z_{2n-2}) & [z_{2n-2}, z_{2n-1}, z_{2n}; f] & \\
z_{2n-1} & f(z_{2n-1}) & [z_{2n-1}, z_{2n}; f] = \frac{f(z_{2n}) - f(z_{2n-1})}{z_{2n} - z_{2n-1}} & [z_{2n-1}, z_{2n}, z_{2n+1}; f] & \\
z_{2n} & f(z_{2n}) & f'(z_{2n}) & * & \\
z_{2n+1} & f(z_{2n+1}) & * & * &
\end{array}$$

Cubic Hermite Polynomial (Atkinson, 1989)

The *cubic Hermite polynomial* solves

$$p(a) = f(a), \quad p'(a) = f'(a)$$

$$p(b) = f(b), \quad p'(b) = f'(b)$$

It is given by

$$H_2(x) = \left(1 + 2\frac{x-a}{b-a}\right)\left(\frac{b-x}{b-a}\right)^2 f(a) +$$

$$+\left(1 + 2\frac{b-x}{b-a}\right)\left(\frac{x-a}{b-a}\right)^2 f(b)$$

$$+\frac{(x-a)(b-x)^2}{(b-a)^2} f'(a) - \frac{(x-a)^2(b-x)}{(b-a)^2} f'(b)$$

Cubic Hermite Polynomial (Atkinson, 1989)

The divided difference formula becomes

$$H_2(x) = f(a) + (x-a)f'(a) + (x-a)^2[a,a,b;f] +$$
$$+(x-a)^2(x-b)[a,a,b,b;f]$$

in which

$$[a,a,b;f] = \frac{[a,b;f] - f'(a)}{b-a},$$

$$[a,a,b,b;f] = \frac{f'(b) - 2[a,b;f] + f'(a)}{(b-a)^2}$$

Cubic Hermite Polynomial (Atkinson, 1989)

The error formula is

$$\begin{aligned} f(x) - H_2(x) &= (x-a)^2(x-b)^2[a,a,b,b,x;f] = \\ &= \frac{(x-a)^2(x-b)^2}{24} f^{(4)}(\xi), \end{aligned}$$

where $\min\{a,b,x\} \leq \xi \leq \max\{a,b,x\}$

$$\max_{x\in[a,b]} |f(x) - H_2(x)| \leq \frac{(b-a)^4}{384} \max_{x\in[a,b]} |f^{(4)}(x)|.$$

$$\left(\|f - H_2\|_\infty \leq \frac{(b-a)^4}{384} \|f^{(4)}\|_\infty \right)$$

Piecewise Polynomial Approximation (Süli and Mayers, 2003)

For Lagrange interpolation and Hermite interpolation the approximation is defined by the same analytical expression on the whole interval $[a, b]$. An alternative and more flexible way of approximating a function f is to divide the interval $[a, b]$ into a number of subintervals and to look for a piecewise approximation by polynomials of low degree. Such piecewise-polynomial approximations are called *splines*, and the endpoints of the subintervals are known as the *knots*. More specifically, a spline of degree n, $n \geq 1$, is a function which is a polynomial of degree n or less in each subinterval and has a prescribed degree of smoothness.

Linear Interpolating Splines (Süli and Mayers, 2003)

Suppose that f is a real-valued function, defined and continuous on the closed interval $[a, b]$. Further, let $K = \{x_0, \ldots, x_n\}$ be a subset of $[a, b]$, with

$$a = x_0 < x_1 < \ldots < x_{n-1} < x_n = b,\ n \geq 2.$$

The *linear spline* S_L, interpolating f at the points x_i, is defined by

$$S_L(x) = \frac{x_i - x}{x_i - x_{i-1}} f(x_{i-1}) + \frac{x - x_{i-1}}{x_i - x_{i-1}} f(x_i),\ x \in [x_{i-1}, x_i],\ i = \overline{0, n}$$

The set K is referred to as the *set of knots*.

Linear Interpolating Splines (Süli and Mayers, 2003)

The function S_L interpolates the function f at the knots, *i.e.*,

$$S_L(x_i) = f(x_i), i = 0, 1, ..., n,$$

and over each interval $[x_{i-1}, x_i]$ the function S_L is a linear polynomial (and therefore continuous).
Suppose that $f \in C^2[a, b]$ and let S_L be the linear spline that interpolates f at the set of knots K. The following error bound holds:

$$\|f - S_L\|_\infty \leq \frac{1}{8} h \|f^{(2)}\|_\infty$$

where $h = \max_i h_i = \max_i (x_i - x_{i-1})$, and $\|\cdot\|_\infty$ denotes the ∞-norm over $[a, b]$.

Linear Interpolating Splines (Süli and Mayers, 2003)

Suppose that S_L is the linear spline that interpolates $f \in C[a,b]$ at the knots $a = x_0 < x_1 < \ldots < x_{n-1} < x_n = b$. Then, for any function v in $H^1(a,b)$ that also interpolates f at these knots,

$$\|S_L^{(1)}\|_2 \leq \|v^{(1)}\|_2$$

where $\|f\|_2 = \left(\int_a^b |f(x)|^2 \, dx \right)^{\frac{1}{2}}$, $\forall\, f \in H^1(a,b)$ (denotes the 2-norm over $[a,b]$).
A linear spline can be characterised as a minimiser of the functional

$$v \to \|v^{(1)}\|_2$$

over all $v \in H^1(a,b)$ which interpolate a given continuous function at the knots of the spline.

Basis functions for the linear spline (Süli and Mayers, 2003)

Suppose that S_L is a linear spline with knots x_i, $i = 0, 1, \ldots, n$, interpolating the function $f \in C[a, b]$. Instead of specifying the value of S_L on each subinterval $[x_{i-1}, x_i]$, $i = 1, \ldots, n$ we can express S_L as a linear combination of suitable *basis functions* s_k as follows:

$$S_L(x) = \sum_{k=0}^{n} f(x_k) s_k(x), \quad x \in [a, b].$$

Here, we require that each s_k is itself a linear spline which vanishes at every knot except x_k, $s_k(x_i) = \delta_{ki}$. The function s_k is often known as the *linear basis spline* or *hat function*.

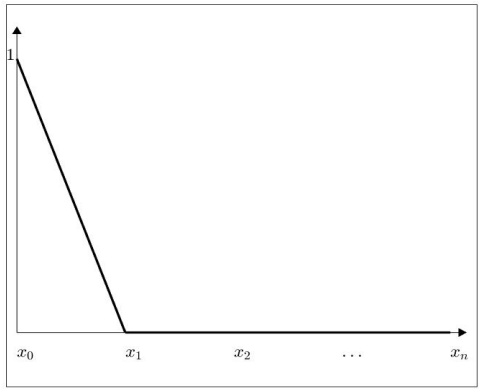
Basis functions for the linear spline (Süli and Mayers, 2003)

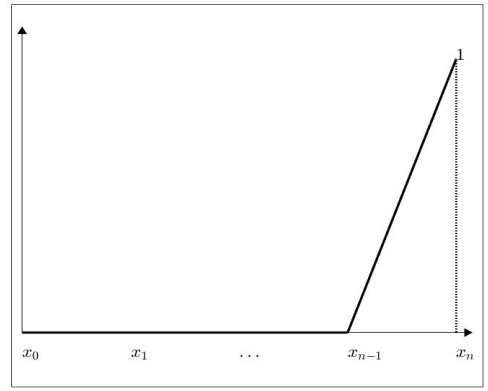
$$s_0(x) = \begin{cases} \frac{x_1 - x}{x_1 - x_0}, \; a = x_0 \leq x \leq x_1 \\ 0, \; x_1 \leq x \end{cases}$$

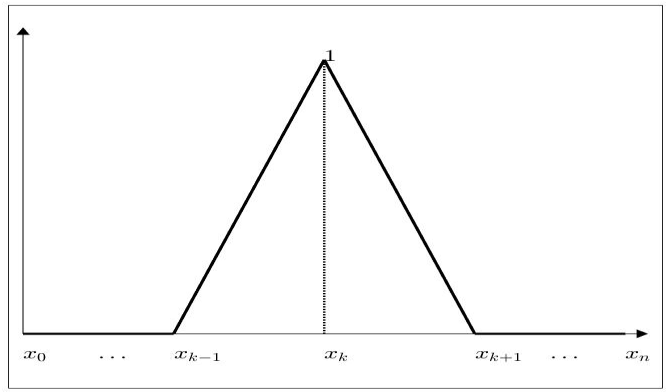
$$s_k(x) = \begin{cases} 0, \; x \leq x_{k-1} \\ \frac{x - x_{k-1}}{x_k - x_{k-1}}, \; x_{k-1} \leq x \leq x_k \\ \frac{x_{k+1} - x}{x_{k+1} - x_k}, \; x_k \leq x \leq x_{k+1} \end{cases} , \; k = \overline{1, n-1}$$

$$s_n(x) = \begin{cases} 0, \; x \leq x_{n-1} \\ \frac{x - x_{n-1}}{x_n - x_{n-1}}, \; x_{n-1} \leq x \leq x_n = b \end{cases}$$

Basis functions for the linear spline (Süli and Mayers, 2003)







Cubic Spline (Süli and Mayers, 2003)

Consider the set $\mathcal{S}$ of all functions $S^c \in C^2[a,b]$ such that

1. $S^c(x_k) = f(x_k), \quad k = \overline{0,n}$
2. S^c is a cubic polynomial on $[x_{k-1}, x_k]$, $k = \overline{1,n}$

$$S^c(x) = \begin{cases} S_0^c(x),\ x \in [x_0, x_1] \\ S_1^c(x),\ x \in [x_1, x_2] \\ \dots \quad \dots \quad \dots \\ S_{n-1}^c(x),\ x \in [x_{n-1}, x_n] \end{cases}$$

Any element of $\mathcal{S}$ is referred to as an *interpolating cubic spline*. We note that, unlike linear splines which are uniquely determined by the interpolating conditions, there is more than one interpolating cubic spline $S^c \in C^2[a,b]$ that satisfies the two conditions stated above.

Cubic Spline (Süli and Mayers, 2003)

There are $4n$ coefficients of cubic polynomials, four on each subinterval $[x_{k-1}, x_k]$, $k = \overline{1, n}$,

$$S_k^c(x) = a_k x^3 + b_k x^2 + c_k x + d_k$$

There are $3(n-1)$ *continuity conditions* (since S^c belongs to $C^2[a, b]$, this means that S^c, $(S^c)'$ and $(S^c)''$ are continuous at the internal knots $x_1, \ldots, x_{n-1}$.

$$S_k^c(x_{k+1}) = S_{k+1}^c(x_{k+1}), \quad k = \overline{1, n-1}$$
$$(S_k^c)'(x_{k+1}) = (S_{k+1}^c)'(x_{k+1}), \quad k = \overline{1, n-1}$$
$$(S_k^c)''(x_{k+1}) = (S_{k+1}^c)''(x_{k+1}), \quad k = \overline{1, n-1}.$$

Cubic Spline (Süli and Mayers, 2003)

There are $n+1$ interpolating conditions

$$S_k^c(x_k) = f(x_k) \text{ and } S_k^c(x_{k+1}) = f(x_{k+1}), \quad k = \overline{0, n-1}.$$

Hence, we have a total of $4n-2$ conditions for the $4n$ unknown coefficients. Depending on the choice of the remaining two conditions we can construct various interpolating cubic splines.

Natural Cubic Spline (Süli and Mayers, 2003)

The *natural cubic spline*, denoted by S^{cn}, is the element of the set $\mathcal{S}$ satisfying the *end conditions*

$$(S^{cn})''(x_0) = (S^{cn})''(x_n) = 0$$

This definition is correct in the sense that the two additional uniquely determine S^{cn}.

Construction of the natural cubic spline: Let us begin by defining

$$\sigma_k = (S^{cn})''(x_k),\ k = 0, \ldots, n.$$

and noting that S^{cn} is a linear function on $[x_{k-1}, x_k]$.

Natural Cubic Spline (Süli and Mayers, 2003)

Therefore, $(S^{cn})''$ can be expressed as

$$(S^{cn})''(x) = \frac{x_k - x}{x_k - x_{k-1}}\sigma_{k-1} + \frac{x - x_{k-1}}{x_k - x_{k-1}}\sigma_k, \quad x \in [x_{k-1}, x_k].$$

Integrating this twice we obtain

$$\begin{aligned} S^{cn}(x) &= \frac{(x_k - x)^3}{6(x_k - x_{k-1})}\sigma_{k-1} + \frac{(x - x_{k-1})^3}{6(x_k - x_{k-1})}\sigma_k + \\ &+ \alpha_k(x - x_{k-1}) + \beta_k(x_k - x) = \\ &= \frac{(x_k - x)^3}{6h_k}\sigma_{k-1} + \frac{(x - x_{k-1})^3}{6h_k}\sigma_k + \\ &+ \alpha_k(x - x_{k-1}) + \beta_k(x_k - x) \end{aligned}$$

where $h_k = x_k - x_{k-1}$ and α_k and β_k are constants of integration.

Natural Cubic Spline (Süli and Mayers, 2003)

Equating S^{cn} with f at the knots x_{k-1}, x_k yields

$$\frac{1}{6}\sigma_{k-1}h_k + h_k\beta_k = f(x_{k-1}),\ \frac{1}{6}\sigma_k h_k + h_k\alpha_k = f(x_k).$$

Expressing α_k and β_k from these and exploiting the continuity of S^{cn} at the internal knots, (*i.e.*, $(S^{cn})'(x_k-) = (S^{cn})'(x_k+)$), gives

$$h_k\sigma_{k-1} + 2(h_{k+1} + h_k)\sigma_k + h_{k+1}\sigma_{k+1} =$$

$$= 6\Big(\frac{f(x_{k+1}) + f(x_k)}{h_{k+1}} - \frac{f(x_k) - f(x_{k-1})}{h_k}\Big)$$

for $k = 1, \ldots, n-1$, together with

$$\sigma_0 = \sigma_n = 0.$$

Natural Cubic Spline (Süli and Mayers, 2003)

The matrix of the system is tridiagonal and nonsingular. By solving this linear system we obtain the σ_k, $k = \overline{0, n}$, and thereby all the α_k, β_k.

Among all functions $v \in H^2(a, b)$ which interpolate a given continuous function f at a fixed set of knots in $[a, b]$, the natural cubic spline S^{cn} is smoothest, in the sense that it minimises $v \to \|v^{(2)}\|_2$, the *average curvature* of v.

More exactly, for any function $v \in H^2(a, b)$ that also interpolates f at the knots,

$$\|(S^{cn})^{(2)}\|_2 \leq \|v^{(2)}\|_2.$$

Hermite Cubic Spline (Süli and Mayers, 2003)

In the previous case we took $f \in C[a,b]$ and demanded that $S^c \in C^2[a,b]$; here we shall strengthen our requirements on the smoothness of the function that we wish to interpolate and assume that $f \in C^1[a,b]$; simultaneously, we shall relax the smoothness requirements on the associated spline approximation S^c by demanding that $S^c \in C^1[a,b]$ only.
We define the *Hermite cubic spline* as a function $S^{ch} \in C^1[a,b]$ such that

$$S^{ch}(x_k) = f(x_k),\ (S^{ch})'(x_k) = f'(x_k),\ k = \overline{0,n}$$

S^{ch} is a cubic polynomial on $[x_{k-1}, x_k]$, $k = \overline{1,n}$.

Hermite Cubic Spline (Süli and Mayers, 2003)

Writing the spline S^{ch} on the interval $[x_{k-1}, x_k]$ as

$$S^{ch}(x) = c_0 + c_1(x - x_{k-1}) + c_2(x - x_{k-1})^2 + c_3(x - x_{k-1})^3$$

where

$$c_0 = f(x_{k-1}),$$

$$c_1 = f'(x_{k-1}),$$

$$c_2 = 3\frac{f(x_k) - f(x_{k-1})}{h_k^2} - \frac{f'(x_k) + 2f'(x_{k-1})}{h_k}$$

$$c_3 = \frac{f'(x_k) + f'(x_{k-1})}{h_k^2} - 2\frac{f(x_k) - f(x_{k-1})}{h_k^3}$$

Hermite Cubic Spline (Süli and Mayers, 2003)

Note that the Hermite cubic spline only has a continuous first derivative at the knots. Unlike natural cubic splines, the coefficients of a Hermite cubic spline on each subinterval can be written down explicitly without the need to solve a tridiagonal system.
Concerning the size of the interpolation error, we have the following result

$$\|f - S^{ch}\|_\infty \leq \frac{1}{384} h^4 \|f^{(4)}\|_\infty.$$