Mathematical Tripos Part IB: Easter Term 2002

Numerical Analysis – Lecture 9

9 Polynomial interpolation

Let $f : [a, b] \to \mathbb{R}$ be a real-valued continuous function defined on some interval $[a, b]$ and let $(x_i)_{i=0}^n$ be $n+1$ distinct points in $[a, b]$. We wish to construct a polynomial p of degree n which interpolates f at these points, i.e., satisfies

$$p(x_i) = f(x_i), \quad i = \overline{0, n}, \qquad p \in \mathcal{P}_n.$$

Theorem 9.1 (Existence and uniqueness) *Given $f \in C[a, b]$ and $n+1$ distinct points $(x_i)_{i=0}^n \in [a, b]$, there is exactly one polynomial $p \in \mathcal{P}_n$ such that $p(x_i) = f(x_i)$ all i.*

Proof. 1) There is *at least one* polynomial interpolant $p \in \mathcal{P}_n$, the one in the *Lagrange form*,

$$p(x) = \sum_{i=0}^n f(x_i)\,\ell_i(x) \qquad \text{with} \quad \ell_i(x) := \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j}{x_i - x_j}\,, \quad i = \overline{0, n}\,, \tag{9.1}$$

the latter are the *fundamental* Lagrange polynomials. Each ℓ_i is the product of n linear factors, hence $\ell_i \in \mathcal{P}_n$. It also equals 1 at x_i and vanishes at $x_j \neq x_i$, i.e., $\ell_i(x_j) = \delta_{ij}$. Therefore $p \in \mathcal{P}_n$ and

$$p(x_j) = \sum_{i=0}^n f(x_i)\,\ell_i(x_j) = f(x_j)\,.$$

2) There is *at most one* polynomial interpolant $p \in \mathcal{P}_n$ to f on $(x_i)_{i=0}^n$. For if there are two, $p, q \in \mathcal{P}_n$, then the polynomial $r := p - q$ is of degree n and vanishes at $n+1$ points, whence $r \equiv 0$. ■

Remark 9.2 Let us introduce the so-called *nodal* polynomial

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

Then, in the expression (9.1) for ℓ_i, the numerator is simply $\omega(x)/(x - x_i)$ while the denominator is equal to $\omega'(x_i)$. With that we arrive to a compact Lagrange form

$$\boxed{p(x) = \sum_{i=0}^n f(x_i)\,\ell_i(x) = \sum_{i=0}^n \frac{f(x_i)}{\omega'(x_i)}\,\frac{\omega(x)}{x - x_i}} \tag{9.2}$$

The Lagrange forms (9.1)-(9.2) for the interpolating polynomials are easy to manipulate with, but they are unsuitable for numerical evaluations. An alternative is the *Newton form* which has an *adaptive* nature.

Method 9.3 (The Newton form) For $k = 0, 1, ..., n$, let $p_k \in \mathcal{P}_k$ be the polynomial interpolant to f on $x_0, ..., x_k$. Then two subsequent p_{k-1} and p_k interpolate the same values $f(x_i)$ for $i \leq k-1$, hence their difference is a polynomial of degree k that vanishes at k points $x_0, ..., x_{k-1}$. Thus

$$p_k(x) - p_{k-1}(x) = A_k \prod_{i=0}^{k-1} (x - x_i), \tag{9.3}$$

with some constant A_k which is seen to be the *leading* coefficient of p_k. It follows that $p := p_n$ can be built step by step as one constructs the sequence $(p_0, p_1, ...)$, with p_k obtained from p_{k-1} by addition the term from the right-hand side of (9.3):

$$p(x) := p_n(x) = \sum_{k=0}^n A_k \prod_{i=0}^{k-1} (x - x_i),$$

Definition 9.4 (Divided difference) Given $f \in C[a,b]$ and $k+1$ distinct points $(x_i)_{i=0}^{k} \in [a,b]$, the *divided difference* $f[x_0,...,x_k]$ *of order* k is, by definition, the leading coefficient of the polynomial $p_k \in \mathcal{P}_k$ which interpolates f at these points.

With this definition we arrive at the Newton formula for the interpolating polynomial.

Theorem 9.5 (Newton formula) *Given $n+1$ distinct points $(x_i)_{i=0}^{n}$, let $p_n \in \mathcal{P}_n$ be the polynomial that interpolates f at these points. Then it may be written in the Newton form*

$$\begin{aligned} p_n(x) \quad &= \quad f[x_0] + f[x_0,x_1]\,(x-x_0) + f[x_0,x_1,x_2]\,(x-x_0)(x-x_1) \;+\; \cdots \\ &\quad\quad \cdots + f[x_0,x_1,\ldots,x_n]\,(x-x_0)(x-x_1)\cdots(x-x_{n-1})\,, \end{aligned}$$

or, more compactly,

$$\boxed{p_n(x) = \sum_{k=0}^{n} f[x_0,...,x_k] \prod_{i=0}^{k-1}(x-x_i)} \tag{9.4}$$

To make this formula of any use, we need expression for $f[x_0,...,x_k]$. One such can be derived from the Lagrange formula (9.2) by identifying the leading coefficient of p. This turns to be

$$f[x_0,...,x_n] = \sum_{i=0}^{n} \frac{f(x_i)}{\omega'(x_i)}, \qquad \omega(x) := \prod_{i=0}^{n}(x-x_i).$$

However, it has the same computational disadvantages as the Lagrange form itself. A useful way to calculate divided difference is again an adaptive (or recurrence) approach.

Theorem 9.6 (Recurrence relation) *For distinct $x_0, x_1, \ldots, x_k$, we have*

$$\boxed{f[x_0,...,x_k] = \frac{f[x_1,...,x_k] - f[x_0,...,x_{k-1}]}{x_k - x_0}} \tag{9.5}$$

Proof. Let $q_0, q_1 \in \mathcal{P}_{k-1}$ be the polynomials such that q_0 interpolates f on $(x_0,x_1,\ldots,x_{k-1})$
q_1 interpolates f on $(x_1,\ldots,x_{k-1},x_k)$
and define the polynomial

$$p(x) := \frac{x-x_0}{x_k-x_0}\,q_1(x) + \frac{x_k-x}{x_k-x_0}\,q_0(x)\,, \qquad p \in \mathcal{P}_k\,.$$

One readily sees that $p(x_i) = f(x_i)$ for all i, hence, p is the k-th degree interpolating polynomial for f. Moreover, the leading coefficient of p is equal to the difference of those of q_1 and q_0 divided by $x_k - x_0$, and that is exactly what the recurrence (9.5) says. ∎

Method 9.7 The recursive formula (9.5) allows for fast evaluation of the *divided difference table*

$$\begin{array}{cllll}
x_i & f[*]=f(*) & f[*,*] & f[*,*,*] & f[*,*,*,*] \\
\hline
x_0 & \to f[x_0] & & & \\
 & & \searrow\!\!\nearrow f[x_0,x_1] & & \\
x_1 & \to f[x_1] & & \searrow\!\!\nearrow f[x_0,x_1,x_2] & \\
 & & \searrow\!\!\nearrow f[x_1,x_2] & & \searrow\!\!\nearrow f[x_0,x_1,x_2,x_3] \\
x_2 & \to f[x_2] & & \searrow\!\!\nearrow f[x_1,x_2,x_3] & \\
 & & \searrow\!\!\nearrow f[x_2,x_3] & & \\
x_3 & \to f[x_3] & & &
\end{array}$$

This can be done in $\mathcal{O}(n^2)$ operations and the outcome are the numbers $\{f[x_0,\ldots,x_k]\}_{k=0}^{n}$ at the head of the columns which can be used in the Newton form (9.4).

Method 9.8 (Horner scheme) Finally, evaluation of p at a given point x using Newton formula (provided that divided differences $A_k := f[x_0, ...x_k]$ are known) requires just $\mathcal{O}(n)$ operations, as long as we do it by the *Horner scheme*

$$p_n(x) = \Big\{\cdots\Big\{\{A_n \times (x-x_{n-1}) + A_{n-1}\} \times (x-x_{n-2}) + A_{n-2}\Big\} \times (x-x_{n-3}) + \cdots + A_1\Big\} \times (x-x_0) + A_0.$$