Mathematical Tripos Part IB: Lent 2020
Numerical Analysis – Lecture 2[1]

In this lecture, we will see another way to compute the interpolant polynomial, known as *Newton's interpolation formula*. Before presenting this formula we first need the definition of *divided differences*.

1.2 Divided differences: a definition

Given pairwise-distinct points $x_0, x_1, \ldots, x_n \in [a,b]$, we let $p \in \mathbb{P}_n[x]$ interpolate $f \in C[a,b]$ there. The coefficient of x^n in p is called the *divided difference* and denoted by $f[x_0, x_1, \ldots, x_n]$. We say that this divided difference is of *degree* n.
We can derive $f[x_0, \ldots, x_n]$ from the Lagrange formula,

$$f[x_0, x_1, \ldots, x_n] = \sum_{k=0}^{n} f(x_k) \prod_{\substack{\ell=0 \\ \ell \neq k}}^{n} \frac{1}{x_k - x_\ell}. \tag{1.2}$$

It is easy to verify that $f[x_0] = f(x_0)$ and $f[x_0, x_1] = \frac{f(x_1)-f(x_0)}{x_1-x_0}$. We can already see why it is called *divided differences*. In fact one has the following general fact:

Theorem Suppose that $x_0, x_1, \ldots, x_{n+1}$ are pairwise distinct, where $n \geq 0$. Then

$$f[x_0, x_1, \ldots, x_{n+1}] = \frac{f[x_1, x_2, \ldots, x_{n+1}] - f[x_0, x_1, \ldots, x_n]}{x_{n+1} - x_0}. \tag{1.3}$$

Proof. Let $p, q \in \mathbb{P}_n[x]$ be the polynomials that interpolate f at

$$\{x_0, x_1, \ldots, x_n\} \qquad \text{and} \qquad \{x_1, x_2, \ldots, x_{n+1}\}$$

respectively and define

$$r(x) := \frac{(x - x_0)q(x) + (x_{n+1} - x)p(x)}{x_{n+1} - x_0} \in \mathbb{P}_{n+1}[x].$$

We readily verify that $r(x_i) = f(x_i)$, $i = 0, 1, \ldots, n+1$. Hence r is the $(n+1)$-degree interpolating polynomial and $f[x_0, \ldots, x_{n+1}]$ is the coefficient of x^{n+1} therein. The recurrence (1.3) follows from the definition of divided differences. □

From this recursive definition it is natural to think that divided differences can approximate the derivatives of f. This is made precise in the following theorem:

Theorem Let $[a,b]$ be the shortest interval that contains $x_0, x_1, \ldots, x_n$ and let $f \in C^n[a,b]$. Then there exists $\xi \in [a,b]$ such that

$$f[x_0, x_1, \ldots, x_n] = \tfrac{1}{n!} f^{(n)}(\xi). \tag{1.4}$$

Proof. Let p be the interpolating polynomial. The error function $f - p$ has at least $n+1$ zeros in $[a,b]$ and, applying Rolle's theorem n times, it follows that $f^{(n)} - p^{(n)}$ vanishes at some $\xi \in [a,b]$. But $p(x) = \frac{1}{n!} p^{(n)}(\zeta) x^n +$ lower order terms (for any $\zeta \in \mathbb{R}$), therefore, letting $\zeta = \xi$,

$$f[x_0, x_1, \ldots, x_n] = \tfrac{1}{n!} p^{(n)}(\xi) = \tfrac{1}{n!} f^{(n)}(\xi)$$

and we deduce (1.4). □

[1] Corrections and suggestions to these notes should be emailed to `h.fawzi@damtp.cam.ac.uk`.

1.3 The Newton interpolation formula

We now present the Newton interpolation formula, which is an alternative to the Lagrange formula we saw in Lecture 1. Again, $f(x_i)$, $i = 0, 1, \ldots, n$, are given and we seek $p \in \mathbb{P}_n[x]$ such that $p(x_i) = f(x_i)$, $i = 0, \ldots, n$.

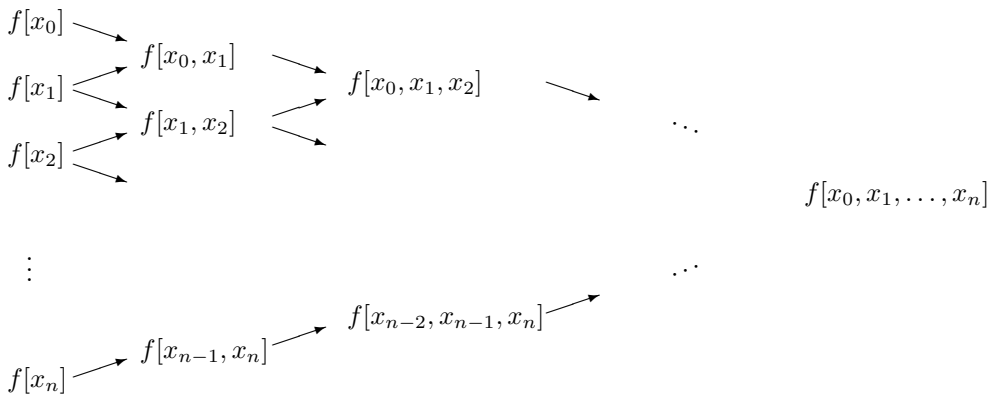
Theorem Suppose that $x_0, x_1, \ldots, x_n$ are pairwise distinct. The polynomial

$$p_n(x) := f[x_0] + f[x_0, x_1](x - x_0) + \cdots + f[x_0, x_1, \ldots, x_n] \prod_{i=0}^{n-1} (x - x_i) \in \mathbb{P}_n[x] \tag{1.5}$$

obeys $p_n(x_i) = f(x_i)$, $i = 0, 1, \ldots, n$.

Proof. By induction on n. The statement is obvious for $n = 0$ and we suppose that it is true for n. We now prove that $p_{n+1}(x) - p_n(x) = f[x_0, x_1, \ldots, x_{n+1}] \prod_{i=0}^{n}(x - x_i)$. Clearly, $p_{n+1} - p_n \in \mathbb{P}_{n+1}[x]$ and the coefficient of x^{n+1} therein is, by definition, $f[x_0, \ldots, x_{n+1}]$. Moreover, $p_{n+1}(x_i) - p_n(x_i) = 0$, $i = 0, 1, \ldots, n$, hence it is a multiple of $\prod_{i=0}^{n}(x - x_i)$, and this proves the asserted form of $p_{n+1} - p_n$. The explicit form of p_{n+1} follows by adding $p_{n+1} - p_n$ to p_n. □

Evaluating the interpolant using Newton's formula We saw last lecture that evaluating the interpolant using Lagrange's formula requires $\mathcal{O}(n^2)$ operations, where n is the number of interpolation points. What about the Newton formula? To evaluate the Newton formula we first need to compute the divided differences. These can be computed using the recursive formula via the *divided difference table,* in the following manner:



The recursive formula allows us to compute all the quantities in this table (i.e., all the $\{f[x_j, x_1, \ldots, x_l]\}_{0 \le j \le l \le n}$), column by column, in $\mathcal{O}(n^2)$ operations.

Having computed the divided differences, the evaluation of the Newton formula (1.5) can be done in just $\mathcal{O}(n)$ time using the Horner scheme, i.e., by writing the Newton formula in the following equivalent way

$$p_n(x) = f[x_0] + (x - x_0)\Big(f[x_0, x_1] + (x - x_1)\Big(f[x_0, x_1, x_2] + \ldots \Big) \Big).$$

On the other hand, the Lagrange formula is often better when we wish to manipulate the interpolation polynomial as part of a larger mathematical expression. We'll see an example in the section on *Gaussian quadrature.*