Key Theorem

Given $f(x)$ r-times differentiable on $[a, b]$ then

$$f[x_0 x_1 \cdots x_r] = \frac{f^r(\eta)}{r!}$$

for some $\eta \in [a, b]$.

Proof:

For $r = 1$ this is the Mean Value Theorem for derivatives. Now, for the general case, let $P_r(x)$ be the unique polynomial of at most degree r that interpolates $f(x)$ at $(x_0, x_1, \cdots x_r)$. Then $E_r(x) = f(x) - P_r(x)$ has $r + 1$ zeros in $[a, b]$. By repeatedly applying Rolle's Theorem we see that

Proof (cont.)

$$
\begin{aligned}
& E_r^{'}(x) \text{ has at least } r \text{ distinct zeros in } [a,b], \\
\Rightarrow \quad & E_r^{''}(x) \text{ has at least } r-1 \text{ distinct zeros in } [a,b], \\
\vdots \\
\Rightarrow \quad & E_r^{r}(x) \text{ has at least 1 zero in } [a,b].
\end{aligned}
$$

Let η be one zero of $E_r^r(x)$ in (a,b).

$$0 = E_r^r(\eta) \Rightarrow f^r(\eta) = P_r^r(\eta).$$

But since $P_r(x)$ is a polynomial of at most degree r, $P_r^r(x)$ is a constant. More precisely

$$P_r^r(\eta) = f[x_0 x_1 \cdots x_r] r!.$$

and we have the desired result:

$$\frac{f^r(\eta)}{r!} = f[x_0 x_1 \cdots x_r].$$

Corollary

If $f(x)$ is $n+1$ times differentiable and $P_n(x)$ interpolates $f(x)$ at the $n+1$ distinct points $(x_0, x_1, \cdots x_n) \in [a,b]$ then $\forall x \in (a,b)\ \exists \eta \in (a,b)$ such that,

$$E_n(x) = \frac{f^{n+1}(\eta)}{(n+1)!} \prod_{i=0}^{n} (x - x_i).$$

- This is the exact error in interpolation. We can use it to derive an overall <u>error bound</u> and to provide guidance in the choice of interpolation points.
- We may not have $|E_n(x)| \to 0$ since $|f^{n+1}|$ may grow faster than $(n+1)!$.

Minimising the Error

What can be done to minimise $|E_n(x)|$ for $(x_0, x_1, \cdots x_n) \in [a, b]$?

$$E_n(x) = \frac{f^{n+1}(\eta)}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$$

One approach would be to choose the x_i's to make

$$\max_{a \le x \le b} \prod_{i=0}^{n} |x - x_i|$$

a minimum. This leads to the choice of Chebyshev points where we have

$$\max_{a \le x \le b} \prod_{i=0}^{n} |x - x_i| = 2\left(\frac{b-a}{4}\right)^{n+1}.$$

Computing Divided Differences

We can exploit the properties of divided differences to derive an efficient scheme for computing and estimating the error in polynomial interpolation. If one introduces a two dimensional tableau of divided differences, $d_{ij},\ i = 0, 1, \cdots n;\ \ j = 0, 1, \cdots i$ where

$$
\begin{aligned}
d_{ij} &= f[x_{i-j} x_{i-j+1} \cdots x_i] \\
&= \frac{f[x_{i-j} x_{i-j+1} \cdots x_{i-1}] - f[x_{i-j+1} x_{i-j+2} \cdots x_i]}{x_{i-j} - x_i} \\
&= \frac{d_{i-1,j-1} - d_{i,j-1}}{x_{i-j} - x_i}.
\end{aligned}
$$

then computing the entries in the tableau by rows is easy and effective (by hand or in MATLAB).

Try some examples!

An Example

Consider determining the cubic polynomial, $P_3(x)$, and estimating the associated error (over the interval $[-2, 2]$) given the data values, $f(-2) = 4, f(-1) = 6, f(0) = 1, f(1) = 0$ and $f(2) = 2$. Note that the nodes or interpolation points defining $P_3(x)$ are $(-2, -1, 0, 1)$ while the node $x_4 = 2$ is used only in the derivation of the error estimate.

x_i	$f[x_i]$	$f[x_{i-1}x_i]$	$f[x_{i-2}x_{i-1}x_i]$	$f[x_{i-3}x_{i-2}x_{i-1}x_i]$	$f[x_{i-4}x_{i-3}x_{i-2}x_{i-1}x_i]$
-2	4				
-1	6	$\frac{4-6}{-2+1} = 2$			
0	1	$\frac{6-1}{-1-0} = -5$	$\frac{2+5}{-2-0} = -7/2$		
1	0	$\frac{1-0}{0-1} = -1$	$\frac{-5+1}{-1-1} = 2$	$\frac{-7/2-2}{-2-1} = 11/6$	
2	2	$\frac{0-2}{1-2} = 2$	$\frac{-1-2}{0-2} = 3/2$	$\frac{2-3/2}{-1-2} = -1/6$	$\frac{11/6+1/6}{-2-2} = -1/2$

Example (cont)

We then have

$$P_3(x) = 4 + 2(x+2) - 7/2(x+2)(x+1) + 11/6(x+2)(x+1)x,$$

and the associated error estimate,

$$\begin{aligned} est_n(x) &= f[x_0 x_1 x_2 x_3 x_4] \prod_{i=0}^{3} (x - x_i) \\ &= -1/2(x+2)(x+1)(x)(x-1). \end{aligned}$$

Hermite Interpolation

In some applications one wants polynomials which interpolate derivative as well as function values. That is we want to determine a polynomial $P_n(x)$, of degree at most n that satisfies $P_n(x_i) = f(x_i)$ for $i = 0, 1, \cdots k$ and $P_n^{'}(x_i) = f^{'}(x_i)$ for $i = 0, 1, \cdots r$ (where $n = k + r + 1$).

Note that each of the $k+r+2$ constraints is linear in the unknowns (the coefficients defining the polynomial $P_n(x)$) and, as for standard interpolation, we can solve for these coefficients by solving a linear system of $n+1$ equations in $n + 1$ unknowns. In particular the algorithm based on the divided difference tableau to constructively generate the Newton form of $P_n(x)$ can easily be generalized to handle this class of problems.

Hermite Interp. (cont)

Recall that for standard interpolation we compute

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

using the diagonal entries of the DD tableau. The first three columns of this tableau are :

x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, x_{i-1}, x_i]$
x_0	$f[x_0] = f(x_0)$		
x_1	$f[x_1] = f(x_1)$	$\frac{f(x_0)-f(x_1)}{x_0-x_1} = f[x_0, x_1]$	
x_2	$f[x_2] = f(x_2)$	$\frac{f(x_1)-f(x_2)}{x_1-x_2} = f[x_1, x_2]$	$\frac{f[x_0,x_1]-f[x_1,x_2]}{x_0-x_2}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_n	$f[x_n] = f(x_n)$	$\frac{f(x_{n-1})-f(x_n)}{x_{n-1}-x_n} = f[x_{n-1}, x_n]$	$\frac{f[x_{n-2},x_{n-1}]-f[x_{n-1},x_n]}{x_{n-2}-x_n}$

Hermite Interp. (cont)

In the limit as two interpolation nodes coalesce (ie, $x_m \to x_{m+1}$), the corresponding entries of the DD tableau become:

x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, x_{i-1}, x_i]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_m	$f[x_m] = f(x_m)$		
x_{m+1}	$f[x_{m+1}] = f(x_{m+1})$	$\frac{f(x_m)-f(x_{m+1})}{x_m-x_{m+1}} \to f'(x_m)$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$

This suggests that when the function value and derivative are both prescribed at the node x_m we introduce two rows in this tableau (corresponding to x_m) and the first 2 columns of the tableau are initialised to:

x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, x_{i-1}, x_i]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_m	$f[x_m] = f(x_m)$		
x_m	$f[x_m] = f(x_m)$	$f'(x_m)$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Hermite Interp. (cont)

- With these modifications, the remaining entries in the tableau are computed in the usual way (row by row) with the diagonal entries yielding the Newton form of the Hermite interpolating polynomial, $P_n(x)$.
- Note that the error analysis for Hermite interpolation is analogous to that for standard interpolation and similar error estimates can be justified.

Piecewise Polynomial Approx.

Recall:

$$E_n(x) = \frac{f^{n+1}(\eta)}{(n+1)!}\prod_{i=0}^{n}(x - x_i).$$

The basic idea is to obtain accurate approximations of $f(x)$ on $[a, b]$ by subdividing the interval, $a = x_0 < x_1 < \cdots < x_M = b$ and over each sub-interval, $[x_{i-1}, x_i]$, introduce interpolation points $(\xi_{i0}, \xi_{i1}, \cdots \xi_{in})$, and approximate $f(x)$ by the interpolating polynomial $P_{i,n}(x)$ of degree at most n. The approximating function, $S(x)$ is then defined on $[a, b]$ by,

$$S(x) = P_{i,n}(x) \text{ for } x \in [x_{i-1}, x_i],$$

and is referred to as a piecewise polynomial.

Observations re PP

- The evaluation of $S(x)$ requires an initial search to determine i such that $x \in [x_{i-1}, x_i]$.
- The error satisfies,

$$|f(x) - S(x)| = |\frac{f^{n+1}(\eta)}{(n+1)!} \prod_{j=0}^{n} (x - \xi_{ij})| \leq \frac{L}{(n+1)!} h_i^{n+1},$$

  where L is a bound on $|f^{n+1}(x)|$ and $h_i = (x_i - x_{i-1})$.
- $S(x)$ will be continuous if the endpoints of each subinterval are interpolation points. That is, if the set of points $[\xi_{ij}]_{j=0}^{n}$ includes x_i and x_{i-1}. $S(x)$ will not in general be differentiable.

An Example of PP Approx.

Piecewise Linear Interpolation:

On each subinterval $[x_{i-1}, x_i]$ let $P_{i,1}(x)$ be the linear polynomial interpolating f_{i-1} and f_i,

$$P_{i,1}(x) = a_0^{(i)} + a_1^{(i)}(x - x_{i-1}),$$

where $a_0^{(i)} = f_{i-1}$ and $a_1^{(i)} = f[x_{i-1}, x_i]$. Note that $S(x)$ will then be continuous and satisfy,

$$|f(x) - S(x)| < \frac{L}{2}h^2,$$

where h is the maximum subinterval width. If the x_i's are equally spaced we will have

$h = (b - a)/M$.