Approximation Theory.

The poroblem: How to approximate derivatives and integrals.
The goal: Solution of ordinary diff. equations and partial diff. equations.

Preliminaries from high-school calculus

Let $f: A \to \mathbb{R}$ be a function. The set of all such f that are continuous at A is written as $C(A)$.
We also define

$$C^1(A) = \{f \in C(A) \mid f' \in C(A)\}$$
$$C^2(A) = \{f \in C(A) \mid f'' \in C(A)\}$$

and in general:

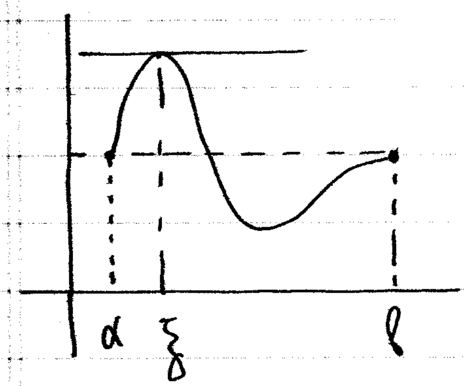
$$C^n(A) = \{f \in C(A) \mid f^{(n)} \in C(A)\}.$$

We also introduce the space of analytic functions:

$$C^\infty(A) = \bigcap_{k=1}^{\infty} C^k(A).$$

① Rolle theorem

$$\left.\begin{array}{l} f \text{ continuous at } [a,b] \\ f \text{ differentiable at } (a,b) \\ f(a)=f(b) \end{array}\right\} \Rightarrow \exists \xi \in (a,b) : f'(\xi)=0$$

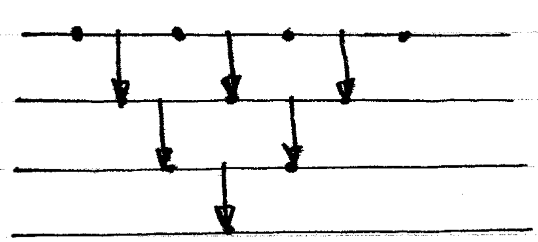


If f is reasonably "smooth" and $f(a)=f(b)$, then there is a point $\xi \in (a,b)$ where the tangent to the graph of f is horizontal.

An immediate corollary of Rolle theorem is:

$$\left.\begin{array}{l} x_1 < \dots < x_n \\ f(x_k)=0,\ \forall k \in [n] \\ f \in C^{n-1}([x_1,x_n]) \end{array}\right\} \Rightarrow \exists \xi \in (x_1,x_n) : f^{(n-1)}(\xi)=0$$

example : For $n=4$



$f(x_k)=0$

$f'(y_k)=0$

$f''(z_k)=0$

$f'''(\xi)=0$

② Mean Value Theorem

$$\left.\begin{array}{l} f \text{ continuous at } [a,b] \\ f \text{ differentiable at } (a,b) \end{array}\right\} \Rightarrow \exists \xi \in (a,b) : f'(\xi) = \frac{f(b)-f(a)}{b-a}$$

③ Taylor Expansion theorem

Assume that
$f \in C^n[a,b]$
$f^{(n)}$ differentiable at (a,b)
Then

$\forall x < c \in [a,b] : \exists \xi \in (x,c) :$

$$: f(x) = \sum_{k=0}^{n} \frac{1}{k!} f^{(k)}(c)(x-c)^k + E_n(x)$$

with $E_n(x) = \dfrac{1}{(n+1)!} f^{(n+1)}(\xi)(x-c)^{n+1}$

A similar statement is applicable when $x > c$.

- The error term can be written in closed form (without ξ dependence) as follows:

$$E_n(x) = \frac{1}{n!} \int_c^x f^{(n+1)}(t)(x-t)^n \, dt$$

4 Integral Mean Value Theorem

$$\left.\begin{array}{l} f,g \in C[a,b] \\ f(x) \geq 0, \forall x \in [a,b] \end{array}\right\} \Rightarrow \exists \xi \in [a,b] : \int_a^b f(x) g(x) dx = f(\xi) \int_a^b g(x) dx$$

Polynomial Interpolation

Let $x_1 < x_2 < \dots < x_n$. Want a polynomial $p \in \mathbb{R}[x]$ with $\deg p = n-1$ such that

$$\forall k \in [n] : p(x_k) = y_k .$$

The Lagrange construction of $p(x)$ is as follows:

- 1 Define $l_k(x)$ such that $l_k(x_k) = 1$ and $l_k(x_j) = 0, \forall j \in [n] - \{k\}$:

$$l_k(x) = \prod_{a \in [n]-\{k\}} \frac{x - x_a}{x_k - x_a}$$

- 2 Define $p(x)$ as:

$$p(x) = \sum_{a=1}^{n} y_a l_a(x) = y_1 l_1(x) + \dots + y_n l_n(x).$$

examples

1) Two points (x_1, y_1), (x_2, y_2)

$$p(x) = y_1 \frac{x - x_2}{x_1 - x_2} + y_2 \frac{x - x_1}{x_2 - x_1} \qquad \text{(line)}$$

2) Three points $(x_1, y_1), (x_2, y_2), (x_3, y_3)$

$$p(x) = y_1 \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)} + y_2 \frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)}$$
$$+ y_3 \frac{(x-x_2)(x-x_1)}{(x_3-x_2)(x-x_1)} \qquad \text{(parabola)}$$

<u>Thm</u> : Let $x_1 < x_2 < \cdots < x_n$ and $p \in \mathbb{R}[x]$ the polynomial that interpolates $f \in C^n[a,b]$ at the points $x_k \in [a,b]$. Then

$$\forall x \in [a,b] : \exists\, \xi(x) \in [a,b] : f(x) - p(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi_k)\, w(x)$$

with

$$w(x) = \prod_{a=1}^{n} (x - x_a).$$

Proof

Let $x_0 \in [a,b]$ be given. If $\exists k \in [n] : x_k = x_0$, then $f(x_0) - p(x_0) = 0$ and $w(x_0) = 0$, and the statement holds. So, assume that

$$\forall k \in [n] : x_k \neq x_0 \Rightarrow w(x_0) \neq 0.$$

Define

$$\lambda(x_0) = \frac{f(x_0) - p(x_0)}{w(x_0)}$$

and $\varphi(x, x_0) = f(x) - p(x) - \lambda(x_0) w(x)$

Thus $\varphi(x_0, x_0) = 0$.

Note that:

$$\varphi(x_k, x_0) = f(x_k) - p(x_k) - \lambda(x_0) w(x_k) = 0 \quad , \forall k \in [n]$$

because $f(x_k) = p(x_k)$, $\forall k \in [n]$

$w(x_k) = 0$, $\forall k \in [n]$

Also $\varphi \in C^n[a,b]$.

By Rolle's corollary:

$$\exists \xi(x_0) \in (a,b) : \left.\frac{\partial^n \varphi(x, x_0)}{\partial x^n}\right|_{x = \xi(x_0)} = 0 \qquad (1)$$

Note that

$$\frac{\partial^n \varphi(x, x_0)}{\partial x^n} = f^{(n)}(x) - p^{(n)}(x) - \lambda(x_0)\, w^{(n)}(x)$$

$$= f^{(n)}(x) - p^{(n)}(x) - n!\,\lambda(x_0)$$

$$= f^{(n)}(x) - n!\,\lambda(x_0)$$

because $p^{(n)}(x) = 0$ [since $\deg p = n-1$] and $w^{(n)}(x) = n!$

$(1) \Rightarrow \exists\, \xi(x_0) \in (a,b) : f^{(n)}(\xi(x_0)) - n!\,\dfrac{f(x_0) - p(x_0)}{w(x_0)}$

$\Rightarrow \exists\, \xi(x_0) \in (a,b) : f(x_0) - p(x_0) = \dfrac{1}{n!} f^{(n)}(\xi(x_0))\, w(x_0)$ $\square$

The factor $w(x)$ has two interesting properties:

1) If $x_1, x_2, \ldots, x_n \in [n]$ then
$$\max_{x \in [0,1]} |w(x)| \geq 2^{1-n}$$

~~2) If $x_k = \cos\left(\frac{2k-1}{2n}\right.$~~

2) If $x_k = \cos\left(\dfrac{2k-1}{2n}\pi\right), \forall k \in [n]$ (Chebyshev Nodes)
then
$$\max_{x \in [0,1]} |w(x)| = 2^{1-n}$$

Thus Chebyshev nodes minimize the error in interpolating polynomial.

Chebyshev nodes for interval $[a,b]$

$$\forall k \in [n]: \quad x_k = \frac{b-a}{2} \cos\left(\frac{2k-1}{2n}\right) + \frac{a+b}{2}$$

Numerical Calculation of interpolation polynomial

We use the Newton formulation of the interpolating polynomial:

$$p(x) = \sum_{a=0}^{n-1} c_a q_a(x)$$

$$q_a(x) = \prod_{b=1}^{a} (x - x_b), \quad q_0(x) = 1$$

The condition $p(x_k) = f(x_k), \forall k \in [n]$ gives

$$\sum_{a=0}^{n-1} q_a(x_k)\, c_a = f(x_k)$$

This is a linear system $Ac = f$ with $A_{ab} = q_b(x_a)$.

By definition we see that for $a<b$ we have

$$A_{ab} = q_b(x_a) = 0$$

because $q_b(x)$ has a $(x-x_a)$ factor. Thus A is lower triangular!

<u>example</u> : For $n=3$

$$\begin{bmatrix} q_0(x_1) & 0 & 0 \\ q_0(x_2) & q_1(x_2) & 0 \\ q_0(x_3) & q_1(x_3) & q_2(x_3) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \end{bmatrix}$$

We find c_1, c_2, c_3 using forward substitution. A more efficient approach is the <u>divided differences</u> algorithm.

Divided Differences

Let $p(x) = c_0 q_0(x) + \cdots + c_{n-1} q_{n-1}(x)$ be the interpolating polynomial through $x_1 < x_2 < \cdots < x_n$. We define

$$f[x_1, x_2, \ldots, x_n] \equiv c_{n-1}$$

The divided differences algorithm uses the observation that

$$
\begin{aligned}
c_0 &= f(x_1) \\
c_1 &= f[x_1, x_2] \\
c_2 &= f[x_1, x_2, x_3] \\
&\vdots \\
c_{n-1} &= f[x_1, x_2, \dots, x_n]
\end{aligned}
$$

and the recursion

$$f[x_1, x_2, \dots, x_n] = \frac{f[x_2, x_3, \dots, x_n] - f[x_1, x_2, \dots, x_{n-1}]}{x_n - x_1}$$

<u>example</u> : 4 data points

$$
\begin{array}{lllll}
x_1 & f(x_1) & f[x_1, x_2] & f[x_1, x_2, x_3] & f[x_1, x_2, x_3, x_4] \\
x_2 & f(x_2) & f[x_2, x_3] & f[x_2, x_3, x_4] & \\
x_3 & f(x_3) & f[x_3, x_4] & & \\
x_4 & f(x_4) & & &
\end{array}
$$

Given the coefficient c_n, we use a Horner Scheme to calculate $p(x)$:

$$p(x) = \sum_{a=0}^{n-1} c_a q_a(x) = \sum_{a=0}^{n-1} c_a \left[\prod_{b=1}^{a} (x - x_b) \right]$$

$$\begin{aligned} &= c_0 + c_1 (x - x_1) + c_2 (x - x_1)(x - x_2) + \\ &\quad + c_3 (x - x_1)(x - x_2)(x - x_3) + \cdots + \\ &\quad + c_{n-1} (x - x_1)(x - x_2) \cdots (x - x_{n-1}) \end{aligned}$$

$$= c_0 + (x - x_1)\Big[c_1 + (x - x_2)\Big[c_2 + (x - x_3)\Big[c_3 + \cdots (x - x_{n-2}) \times$$

$$\times \big[c_{n-2} + (x - x_{n-1})\, c_{n-1} \big]\Big]\Big] \cdots \Big].$$

which leads to the following algorithm

<u>Algorithm</u> : Takes c_n, evaluates $p = p(x)$

```
p = c_{n-1}
for a = n-2, ..., 0
    p = (x - x_{a+1}) p + c_a
end for
return p
```

Applications of interpolation

(1) Approximating derivatives with finite differences.

The problem : Given $y_k = f(x_k)$ for a set of points $x_1 < x_2 < \cdots < x_n$
to estimate the derivatives $f'(x)$, $f''(x)$, etc.

Method

- 1 We calculate the interpolating polynomial $p(x)$ from the given data.
- 2 Evaluate the derivatices of $p(x)$ at the required point.

Solution

We begin with the Lagrange formulation of $p(x)$:

$$f(x) = \sum_{a=1}^{n} y_a l_a(x) + \frac{1}{n!} f^{(n)}(\xi(x))\, w(x)$$

with

$$w(x) = \prod_{a=1}^{n} (x - x_a)$$

$$l_a(x) = \prod_{b \in [n]-\{a\}} \frac{x - x_b}{x_a - x_b}$$

The derivative of $f(x)$ is

$$f'(x) = \sum_{a=1}^{n} y_a \, l_a'(x) + \frac{1}{n!} \frac{d}{dx} \left[f^{(n)}(\xi(x)) \, w(x) \right]$$

$$= \sum_{a=1}^{n} y_a \, l_a'(x) + \frac{1}{n!} f^{(n)}(\xi(x)) \, w'(x) +$$

$$+ \frac{1}{n!} w(x) \frac{d}{dx} f^{(n)}(\xi(x))$$

For $x = x_k$ we have $w(x_k) = 0$, so third term is zero. Also

$$w'(x) = \frac{d}{dx} \prod_{a=1}^{n} (x - x_a) = \sum_{b=1}^{n} \prod_{a \in [n]-\{b\}} (x - x_a) \Rightarrow$$

$$\Rightarrow w'(x_k) = \prod_{a \in [n]-\{k\}} (x_k - x_a)$$ (only $b = k$ contributes non-zero term).

Consequently

$$\boxed{f'(x_k) = \sum_{a=1}^{n} y_a\, l_a'(x_k) + \frac{1}{n!} f^{(n)}(\xi(x_k)) \prod_{a=1}^{n} (x_k - x_a)}$$

Finite difference scheme for 1st derivative

Consider 3 points: $x_1 = -h_1$, $x_2 = 0$, $x_3 = +h_2$.

$$l_1(x) = \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)} = \frac{x(x-h_2)}{(-h_1)(-h_1-h_2)}$$
$$= \frac{x(x-h_2)}{h_1(h_1+h_2)}$$

$$l_2(x) = \frac{(x-x_3)(x-x_1)}{(x_2-x_3)(x_2-x_1)} = \frac{(x-h_2)(x+h_1)}{(0-h_2)(0-(-h_1))} =$$
$$= \frac{(x+h_1)(x-h_2)}{-h_1h_2}$$

$$l_3(x) = \frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)} = \frac{(x+h_1)(x-0)}{(h_2-(-h_1))(h_2-0)}$$
$$= \frac{x(x+h_1)}{h_2(h_1+h_2)}$$

After 1 differentiation we have:

$$l_1'(x) = \frac{2x - h_2}{h_1(h_1+h_2)}$$

$$l_2'(x) = \frac{2x + (h_1 - h_2)}{-h_1 h_2}$$

$$l_3'(x) = \frac{2x + h_1}{h_2(h_1+h_2)}$$

and we get:

1) <u>Centered scheme</u> : $x = 0$

$$f'(0) \approx y_1 l_1'(0) + y_2 l_2'(0) + y_3 l_3'(0)$$

$$= \frac{-h_2}{h_1(h_1+h_2)} y_1 - \frac{h_1 - h_2}{h_1 h_2} y_2 + \frac{h_1}{h_2(h_1+h_2)} y_3 .$$

If $h_1 = h_2 = \Delta x$ then

$$f'(0) \approx \frac{y_3 - y_1}{2\Delta x} = \frac{f(\Delta x) - f(-\Delta x)}{2\Delta x}$$

2) Left edge scheme : $x=-h_1$
(also called : forward scheme)

$$l_1'(-h_1) = \frac{2(-h_1)-h_2}{h_1(h_1+h_2)} = -\frac{2h_1+h_2}{h_1(h_1+h_2)}$$

$$l_2'(-h_1) = \frac{2(-h_1)+(h_1-h_2)}{-h_1h_2} = \frac{-h_1-h_2}{-h_1h_2} = \frac{h_1+h_2}{h_1h_2}$$

$$l_3'(-h_1) = \frac{2(-h_1)+h_1}{h_2(h_1+h_2)} = \frac{-h_1}{h_2(h_1+h_2)}$$

Consequently:

$$f'(-h_1) = -\frac{2h_1+h_2}{h_1(h_1+h_2)}\, y_1 + \frac{h_1+h_2}{h_1h_2}\, y_2 + \frac{-h_1}{h_2(h_1+h_2)}\, y_3 .$$

For $h_1 = h_2 = \Delta x$

$$f'(-\Delta x) = -\frac{3}{2\Delta x} f(-\Delta x) + \frac{2}{\Delta x} f(0) - \frac{1}{2\Delta x} f(\Delta x)$$

By symmetry $\Delta x \rightarrow -\Delta x$ we have

$$f'(\Delta x) = +\frac{1}{2\Delta x} f(-\Delta x) - \frac{2}{\Delta x} f(0) + \frac{3}{4\Delta x} f(\Delta x)$$

which is the right-edge scheme (backward scheme).

Summary 1st derivative

We have three schemes

$$f'(t) = [-3f(t) + 4f(t+\Delta t) - f(t+2\Delta t)]/(2\Delta t)$$
$$f'(t) = [f(t+\Delta t) - f(t-\Delta t)]/(2\Delta t)$$
$$f'(t) = [3f(t) - 4f(t-\Delta t) + f(t-2\Delta t)]/(2\Delta t)$$

Schemes for 2nd derivative

Using the points $x=-h_1$, $x=0$, $x=h_2$ we have

$$l_1''(x) = \frac{2}{h_1(h_1+h_2)}$$
$$l_2''(x) = \frac{-2}{h_1 h_2}$$
$$l_3''(x) = \frac{2}{h_2(h_1+h_2)}$$

Thus $f''(0) = y_1 l_1''(x) + y_2 l_2''(x) + y_3 l_3''(x)$

$$= \frac{2}{h_1(h_1+h_2)} f(-h_1) - \frac{2}{h_1 h_2} f(0) + \frac{2}{h_2(h_1+h_2)} f(h_2)$$

For $h_1 = h_2 = \Delta t$ we get

$$f''(t) = \frac{f(t+\Delta t) - 2f(t) + f''(t-\Delta t)}{\Delta t^2}$$

The same scheme can also be used as a forward or backward scheme. One can also introduce a scheme with 4 points.

Accuracy

For 1st derivative schemes, the error is controlled by

$$W'_k(x) = \prod_{a \in [n] - \{k\}} (x_k - x_a).$$

Rescaling $x_k \longrightarrow \lambda x_k$, $\forall k \in [n]$ will rescale $W'(x_k) \longrightarrow \lambda^{n-1} W(x_k)$, $\forall k \in [n]$

Thus for $n = 3$, the error for our schemes scales as $O(\Delta t^2)$.

Alternative derivation possible via Taylor Expansion.

$$f(t+\Delta t) = f(t) + \Delta t\, f'(t) + \frac{\Delta t^2}{2!} f''(t) + \frac{\Delta t^3}{3!} f'''(t) + o(\Delta t^4)$$

$$f(t-\Delta t) = f(t) - \Delta t\, f'(t) + \frac{\Delta t^2}{2!} f''(t) - \frac{\Delta t^3}{3!} f'''(t) + o(\Delta t^4)$$

and therefore:

$$\frac{f(t+\Delta t) - f(t-\Delta t)}{2\Delta t} = f'(t) + \underbrace{\frac{\Delta t^2}{3!} f'''(t)}_{\text{truncation error.}} + O(\Delta t^4)$$

Roundoff error vs. Truncation error

Decreasing Δt decreases truncation error but increases round-off error! If

$$f(t+\Delta t) = F(t+\Delta t) + e(t+\Delta t)$$
$$f(t-\Delta t) = F(t-\Delta t) + e(t-\Delta t)$$

with f = floating point representation
F = exact value
e = round-off error

then the round-off error for

$$\frac{f(t+\Delta t) - f(t-\Delta t)}{2\Delta t}$$

is estimated as

$$|E_R| = \Big| \, f(t+\Delta t) - f(t$$

$$|E_R| = \left| \frac{e(t+\Delta t) - e(t+\Delta t)}{2\Delta t} \right| \leq \frac{1}{2\Delta t} \Big[|e(t+\Delta t)| + |e(t-\Delta t)| \Big]$$

$$\leq \frac{2\,eps}{2\Delta t} = \frac{eps}{\Delta t}$$

The truncation error is estimated as

$$|E_T| = \left| \frac{\Delta t^2}{6} f''(\xi) \right| \leq \frac{M\,\Delta t^2}{6}$$

For $M \approx 1$ we estimate cross-over when

$$\frac{eps}{\Delta t} \sim \frac{\Delta t^2}{6} \Leftrightarrow \Delta t \sim (eps)^{1/3}$$

For $eps \approx 10^{-16} \Rightarrow \underline{\underline{\Delta t \sim 10^{-5}}}$

- Higher-Order schemes have smaller truncation error. However at the optimal time-step Δt one should not expect error improvement! In fact for large enough orders one should see deterioration.

② Newton-Cotes integration

Let $f : [a,b] \longrightarrow \mathbb{R}$ be a function. Define a grid $x_k = a + kh$ such that $x_0 = a$ and $x_n = b$. It follows that

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

Let $P_n(x)$ be an interpolating polynomial such that

$$\forall k \in \{0, \dots, n\} : P_n(x_k) = f(x_k)$$

We already know that

$$\forall x \in [a,b] : \exists\, \xi(x) \in [a,b] :$$

$$: f(x) = P_n(x) + \frac{1}{(n+1)!} f^{(n+1)}(\xi(x))\, w(x)$$

with

$$w(x) = \prod_{k=0}^{n} (x - x_n)$$

We may use this relation to approximate the integral of $f(x)$:

$I = I_n[f] + E_n[f]$ with

$$I_n[f] = \int_a^b P_n(x)\,dx$$

$$E_n[f] = \frac{1}{(n+1)!}\int_a^b f^{(n+1)}(\xi(x))\,w(x)\,dx.$$

$\rightarrow$ Calculate $I_n[f]$

To calculate $I_n[f]$ we use Lagrange representation of $P_n(x)$:

$$I_n[f] = \int_a^b P_n(x)\,dx = \int_a^b \sum_{k=1}^{n} f(x_k)\,l_k(x)\,dx$$

$$= \sum_{k=1}^{n} f(x_k) \int_a^b l_k(x)\,dx$$

with

$$\int_a^b l_k(x)\,dx = \int_a^b \prod_{\substack{\gamma=0 \\ \gamma\neq k}}^{n} \left[\frac{x - x_\gamma}{x_k - x_\gamma}\right] dx$$

$$= \int_0^1 \prod_{\substack{\gamma=0 \\ \gamma\neq k}}^{n} \left[\frac{[a+tnh]-[a+\gamma h]}{[a+kh]-[a+\gamma h]} \right] (b-a)\,dt$$

(change of variables
$x = a+tnh$)

$$= \int_0^1 \prod_{\substack{\gamma=0 \\ \gamma\neq k}}^{n} \left[\frac{t n-\gamma}{(k-\gamma)} \right] (b-a)\,dt$$

$$= (b-a) \int_0^1 \prod_{\substack{\gamma=0 \\ \gamma\neq k}}^{n} \left[\frac{tn-\gamma}{k-\gamma} \right] dt$$

It follows that

$$\boxed{\begin{aligned} I_n[f] &= (b-a) \sum_{k=0}^{n} \frac{\sigma_k}{s} f(x_k) \\ \frac{\sigma_k}{s} &= \int_0^1 \prod_{\substack{\gamma=0 \\ \gamma\neq k}}^{n} \left[\frac{tn-\gamma}{k-\gamma} \right] dt \end{aligned}}$$

For $f(x)=1,\ \forall x\in[a,b] \Rightarrow \sum_{k=1}^{n} \frac{\sigma_k}{s} = 1$

The first three evaluations give:

$n=1$: $\quad I_n[f] = \frac{1}{2}(f_0 + f_1) \qquad$ (Trapezoidal)

$n=2$: $\quad I_2[f] = \frac{1}{6}(f_0 + 4f_1 + f_2) \qquad$ (Simpson's)

$n=3$: $\quad I_3[f] = \frac{1}{8}(f_0 + 3f_1 + 3f_2 + f_3) \qquad$ (3/8 rule)

→ <u>Calculating the error term</u>

A naive argument for estimating the error is:

$$|E_n[f]| = \frac{1}{(n+1)!}\left|\int_a^b f^{(n+1)}(\xi(x))\, w(x)\, dx\right|$$

$$\leq \frac{1}{(n+1)!}\int_a^b |f^{(n+1)}(\xi(x))|\,|w(x)|\,dx$$

$$= \frac{|f^{(n+1)}(\xi)|}{(n+1)!}\int_a^b |w(x)|\,dx$$

$$= \frac{|f^{(n+1)}(\xi)|}{(n+1)!}\int_a^b \prod_{j=0}^{n} |x - x_j|\,dx$$

Change variable $x = a + tnh$

$$|E_n[f]| \leq \frac{|f^{(n+1)}(\xi)|}{(n+1)!} \int_0^1 \prod_{\gamma=0}^{n} |(a+tnh) - (a+\gamma h)|\, dt$$

$$= \frac{h^{n+1} |f^{(n+1)}(\xi)|}{(n+1)!} \int \prod_{\gamma=0}^{n} (tn - \gamma)\, dt.$$

Thus the error is at least $O(h^{n+1})$.
In fact it is better!

<u>Thm</u>: If <u>n even</u> then

$$\exists \xi \in [a,b] : E_n(f) = \frac{f^{(n+2)}(\xi)}{(n+2)!} K_n$$

with

$$K_n = \int_a^b x \prod_{\gamma=0}^{n} (x - x_\gamma)\, dx$$

<u>Thm</u>: If <u>n odd</u> then

$$\exists \xi \in [a,b] : E_n[f] = \frac{f^{(n+1)}(\xi)}{(n+1)!} \int_a^b \prod_{\gamma=0}^{n} (x - x_\gamma)\, dx$$

A few Newton-Cotes schemes

n	σ_k							s	p	k
1	1	1						2	2	1/12
2	1	4	1					6	4	1/90
3	1	3	3	1				8	4	3/80
4	7	32	12	32	7			90	6	8/945
5	19	75	50	50	75	19		288	6	275/12096
6	41	216	27	272	27	216	41	840	8	9/1400

$$\int_a^b f(x)\,dx = (b-a)\sum_{k=0}^{n} \frac{\sigma_k}{s} f(x_k) + h^{p+1} k f^{(p)}(\xi)$$

with $\xi \in [a,b]$ and p, k functions of n.

$n=1$: Trapezoidal
$n=2$: Simpson's
$n=3$: 3/8 rule
$n=4$: Milne's Rule
$n=5$: (Bill) Clinton Rule
$n=6$: Weddle's rule.

Composite Newton-Cotes schemes

Let $f_0, f_1, f_2, \ldots, f_n$ be a sampling of the function f at gridpoints $x_k = a + kh$. We can approximate the integral

$$I = \int_a^b f(x)\,dx$$

with the Newton-Cotes schemes as follows:

1) <u>For $n=1$:</u> $\displaystyle\int_a^b f(x)\,dx \cong \sum_{k=0}^{n-1} \int_{x_k}^{x_{k+1}} f(x)\,dx$

with

$$\int_{x_k}^{x_{k+1}} f(x)\,dx = \frac{h}{2}(f_k + f_{k+1})$$

It follows that

$$\int_a^b f(x)\,dx = \frac{h}{2}(f_0+f_1) + \frac{h}{2}(f_1+f_2) + \cdots + \frac{h}{2}(f_{n-1}+f_n)$$

$$= \frac{h}{2}(f_0 + 2f_1 + 2f_2 + \cdots + 2f_{n-1} + f_n)$$

(composite trapezoidal rule)

n = multiple of 2. Error is $O(h^2)$

2) For $n=2$

We use Simpson's rule:

$$\int_{x_k}^{x_{k+2}} f(x)\,dx = \frac{h}{3}\left(f_k + 4f_{k+1} + f_{k+2}\right)$$

thus

$$\int_a^b f(x)\,dx = \frac{h}{3}\left(f_0 + 4f_1 + f_2\right) + \frac{h}{3}\left(f_3 + 4f_4 + f_5\right)$$

$$+ \cdots + \frac{h}{3}\left(f_{n-2} + 4f_{n-1} + f_n\right)$$

$$= \frac{h}{3}\left(f_0 + 4f_1 + 2f_2 + 4f_3 + 2f_4 + \cdots + f_n\right)$$

Here n = multiple of 3
error is $O(h^3)$.

Richardson extrapolation

These schemes can be improved upon as follows:

Let I_h be the approximation of the integral I by the trapezoidal rule. Then we have

$$I = I_h + c_1 h^2 + c_2 h^4 + O(h^6).$$

For $h/2$ we have

$$I = I_{h/2} + c_1 \frac{h^2}{4} + c_2 \frac{h^4}{16} + \dots$$

and:

$$\cancel{\frac{4 I_h - I_{h/2}}{3} = O(h^4)}$$

Then, note that

$$I = \frac{4 I_h - I_{h/2}}{3} + \underline{\underline{O(h^4)}} \quad !!!!$$

This idea can be generalized for all composite Newton-Cotes schemes!