Theorems

Mean Value Theorem: between two points of a function at heights $f(a)$ and $f(b)$, there is at least one point with slope $f'(c) = [f(b) - f(a)]/(b-a)$.

Extreme Value Theorem: a continuous function on a bounded interval, has a non-infinite maximum and a minimum.

Intermediate Value Theorem: between two points at heights $f(a)$ and $f(b)$, any continuous function passes through all intermediate heights.

Taylor's Theorem, for a $\mathscr{C}^n$ function:

$$f(x) = \left[\sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k\right] + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)^{n+1}$$

Machine Numbers

IEEE 754 is a standard for machine numbers. It encodes a sign `s`, a power `p`, and a mantissa `m`. Numbers are:

$$x = (-1)^{\mathtt{s}} 2^{\mathtt{p}-1023}(1+\mathtt{m})$$

Numbers are encoded to finite precision, which can lead to overflow, underflow, and chopping/rounding errors.

Numeric errors may be quantified as actual, absolute, or relative errors.

Machine epsilon is the largest number such that $1+\epsilon = 1$, ex. to five digits, $1+\epsilon = 0.10000|\bar{9} \times 10^1$, so $\epsilon = 0.0001$.

Subtraction can sometimes lead to a "catastrophic" loss in significance. Addition/multiplication/division are nice.

Bisection Method

Given an interval with a root $f(x^*)=0$, bisect the interval and see which half contains the root and update the interval. For continuous functions, if $f(a)f(b)<0$, then $[a,b]$ contains a root.

The error, $e_k = x_k - x^*$ is bounded by:

$$|x_k - x^*| \le \frac{|b_k - a_k|}{2} = \frac{|b_0 - a_0|}{2 \cdot 2^k}$$

The bisection method converges with order $\alpha = 1$ and rate constant $\lambda = 1/2$.

Newton's Method

A less stable, but much faster method of finding roots takes the derivative:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Newton's method converges with order $\alpha = 2$ and rate constant:

$$\lambda = \left|\frac{f''(x^*)}{2f'(x^*)}\right|$$

Convergence Order & Rate

There are many rootfinding methods, but iterative root methods often fulfill:

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^\alpha} = \lambda$$

Where α is the convergence order and λ is the convergence rate. For convergence we need $\lambda < 1$, so that $\lambda^k \to 0$.

Fixed Point Convergence

Assume x^* is a solution of $g(x) = x$ and $g(x)$ is α times continuously differentiable for all x near x^* for some $\alpha \ge 2$. Furthermore assume $g'(x^*) = g''(x^*) = \dots g^{(\alpha-1)}(x^*) = 0$. Then if x_0 is chosen sufficiently close to x^*, the iteration $x_{k+1} = g(x_k)$ will have:

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^\alpha} = \left|\frac{g^{(\alpha)}(x^*)}{\alpha!}\right| = \lambda$$

Also, if $g'(x^*) \ne 0$, and $|g'(x^*)| < 1$, and x_0 is sufficiently close to x^*, then:

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^1} = |g'(x^*)|$$

Stopping Conditions

"Do you want a small residual or the actual result?" This will determine the stopping condition you choose. Three common metrics for stopping are:

error	$\lvert x_k - x^*\rvert$
residual	$\lvert f(x_k)\rvert$
difference	$\lvert x_{k+1} - x_k\rvert$

Which of the metrics work for a problem should be carefully considered.

For the Newton Method, with assumptions, a simple bound on the error is:

$$\frac{|f(x_k)|}{|f'(x_k)|} < \text{tol} \implies |x_k - x^*| < \text{tol}$$

Polynomial Interpolation

For n points (x_i, f_i), there is a unique polynomial of degree $\le n-1$ which interpolates the points. One method to find this is Newton divided differences:

$$\begin{array}{lll} x_0 \ \boxed{f_0} & \boxed{\dfrac{f_1 - f_0}{x_1 - x_0}} & \boxed{\dfrac{\frac{f_2-f_1}{x_2-x_1} - \frac{f_1-f_0}{x_1-x_0}}{x_2 - x_0}} \\ x_1 \ f_1 & \dfrac{f_2 - f_1}{x_2 - x_1} & \\ x_2 \ f_2 & & \end{array}$$

The polynomial is for boxed constants:

$$p(x) = c_0 + c_1(x-x_0) + c_2(x-x_0)(x-x_1)$$

For equispaced data, error is bounded:

$$\max_{[x_0,x_n]} |f(x) - p(x)| \le \frac{f^{(n+1)}}{4(n+1)} \frac{1}{n^{n+1}}$$

This can blow up as in Runge's phenomenon, so Chebyshev spacing and piecewise interpolation are often used.

Vandermonde Matrix

An efficient way to determine $p(x) = \sum_i a_i x^i$ is to solve the matrix equation:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{pmatrix}$$

Lagrange Polynomials

Another rote way to determine $p(x) = \sum_{i=0}^n f_i l_i(x)$, is Lagrange Polynomials:

$$l_i(x) = \frac{\prod_{k=0,k\ne i}^n (x - x_k)}{\prod_{k=0,k\ne i}^n (x_i - x_k)}$$

Chebyshev Spacing

The solution of a minimax problem gives the Chebyshev spacing which has the smallest maximal error for $p(x)$. This spacing is for n points on $[0,1]$:

$$x_i = \cos\left(\frac{2i+1}{2n+2}\pi\right)$$

The error bound is:

$$\max_{[x_0,x_n]} |f(x) - p(x)| \le \frac{f^{(n+1)}}{(n+1)!} \frac{1}{2^n}$$

Piecewise Interpolation

Interpolating intervals piecewisely of size h/N, has an error of $N^{-(n+1)}$ compared to intervals of size h, as expected.

Numerical Differentiation

The idea is to interpolate a set of points with a polynomial, and approximate:

$$f'(x) \approx p'(x)$$

Linear interpolation, with $f_i = f(x_i)$:

$$p'(x) = \frac{f_1 - f_0}{x_1 - x_0}$$

For quadratic, centered, equispaced:

$$p'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

By Taylor expansion, find the error:

$$f'(x_0) - \frac{f(x_0+h) - f(x_0)}{h} \approx -\frac{f''(x_0)}{2!}h$$

In terms of $M_p = \max f^{(p)}(x)/p!$:

linear	$M_1 h^1$
quadratic	$M_2 h^2$

With an asymptotic error expansion:

$$e_h = c_p h^p + c_{p+1} h^{p+1} + \dots$$

It naturally follows that in $\lim_{h\to 0}$:

$$p = \log_2\left(\frac{|e_h|}{|e_{h/2}|}\right)$$

For large h, $h^{p+1} \not\approx 0$, and for small h there may be catastophic cancellation. In fact, the optimal error is at $h = \mathcal{O}(\epsilon^{1/(p+1)})$, which is seen from:

$$e_h \le M_p h^p + \frac{\epsilon}{h}$$

Newton-Cotes Integration

The Newton-Cotes method integrates polynomial interpolants to functions:

$$\int_a^b dx\ f(x) \approx \int_a^b dx\ p(x)$$

For a linear interpolant (trapezoid):

$$\int_a^b dx\ p(x) = h \cdot \frac{f_a + f_b}{2}$$

For a quadratic interpolant (Simpson):

$$\int_a^b dx\ p(x) = h \cdot \frac{f_0 + 4f_1 + f_2}{3}$$

For a quartic interpolant:

$$\int_a^b dx\ p(x) = h \cdot \frac{3(f_0 + 3f_1 + 3f_2 + f_3)}{8}$$

These have associated errors of:

linear	$-(h^3/12)f''(\xi)$
quadratic	$-(h^5/90)f''''(\xi)$
quartic	$-(3h^5/80)f''''(\xi)$

Composite Integration

Composite methods use the relation:

$$\int_{x_0}^{x_n} dx\ f(x) = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} dx\ f(x)$$

For composite linear (trapezoid):

$$\int_{x_0}^{x_n} dx\ f(x) \approx \sum_{i=0}^{n-1} h \cdot \frac{f_i + f_{i+1}}{2}$$

For composite quadratic (Simpson):

$$\frac{h}{3}\sum_{i=1}^{n/2}[f(x_{2i-2}) + f(x_{2i-1}) + f(x_{2i})]$$

These have associated errors of:

linear	$-\frac{b-a}{12}h^2 f''(\mu)$
quadratic	$-\frac{b-a}{180}h^4 f''''(\mu)$

Aitken Order Estimation

For an integral with asymptotic error expansion of $I_{\text{exact}} - I_h$, in the $\lim_{h\to 0}$:

$$p = \log_2\left(\frac{I_{h/2} - I_h}{I_{h/4} - I_{h/2}}\right)$$

Richardson Extrapolation

If there exists an asymptotic error expansion, then through clever addition, one may increase the order. With two results and step sizes $h_1/h_2 = 2$:

$$B_h = \frac{2^2 A_{h/2} - A_h}{2^2 - 1}; \quad C_h = \frac{2^4 B_{h/2} - B_h}{2^4 - 1}$$

Expressed in tabular form with order:

$\mathcal{O}(h^2)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^6)$
A_h	$B_h = \frac{4A_{h/2} - A_h}{3}$	$\frac{16B_{h/2} - B_h}{15}$
$A_{h/2}$	$B_{h/2} = \frac{4A_{h/4} - A_{h/2}}{3}$	
$A_{h/4}$		

Gaussian Quadrature

Specify positions $\{x_i\}$ and weights $\{w_i\}$ that ensure exact integration of all polynomials of deg $\le 2n-1$ with:

$$\int_a^b dx\ p(x) = \sum_{i=1}^{n} w_i f(x_i)$$

This works because there is a best polynomial of deg $\le 2n-1$ which is very close to $f(x)$ in the $\lim_{n\to\infty}$.

If given coefficients for the interval $[-1, 1]$, they can be transformed to $[a, b]$ by a linear scaling.

LU Factorization

LU factorization of a matrix into a lower triangular and an upper triangular matrix is useful for solving systems of linear equations, inverting matrices, and computing determinants. How the method works is shown by an example:

$$A = \begin{pmatrix} 2 & 4 & 0 \\ 2 & 6 & 6 \\ 1 & 8 & 4 \end{pmatrix}$$

$$L_1 = \begin{pmatrix} 1 & 0 & 0 \\ -2/2 & 1 & 0 \\ -1/2 & 0 & 1 \end{pmatrix}$$

$$L_1 A = \begin{pmatrix} 2 & 4 & 0 \\ 0 & 2 & 6 \\ 0 & 6 & 4 \end{pmatrix}$$

$$L_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -6/2 & 1 \end{pmatrix}$$

$$L_2 L_1 A = \begin{pmatrix} 2 & 4 & 0 \\ 0 & 2 & 6 \\ 0 & 0 & -14 \end{pmatrix} = U$$

$$L_1^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 2/2 & 1 & 0 \\ 1/2 & 0 & 1 \end{pmatrix}$$

$$L_2^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 6/2 & 1 \end{pmatrix}$$

$$L = L_1^{-1} L_2^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1/2 & 3 & 1 \end{pmatrix}$$

$$LU = A$$

Note: each row/column is iteratively determined so this generalizes.

Splines

Splines are piecewise polynomial interpolants. An n-th degree spline is determined by imposing interpolation, and continuity of the up to $n-1$th derivatives at each interior data point, as well as specifying conditions on the first point. For $n = 2$, we find (iteratively):

$$S_i(x) = a_i + b_1(x - x_{i-1}) + c_i(x - x_{i-1})^2$$

Where $a_i = F(x_{i-1})$, $b_i = S'(x_{i-1})$, and $c_i = (S'(x_i) - S'(x_{i-1}))/2h$, and:

$$S'(x_i) = \frac{2}{h}(F(x_i) - F(x_{i-1})) - S'(x_{i-1})$$

This works in matrix form too.