CSCC51H - Numerical Approximation, Integration and Ordinary Differential Equations

©
W.H. Enright
Computer and Mathematical Sciences Division,
University of Toronto at Scarborough
(enright@cs.utoronto.ca)

Overview and Course Organization

1. General Information and Mathematical Background [1 week].

   - Administration Details:
     Grading Scheme, Course Website, Tutorials, Office hours Prerequisites etc.
   - What is Numerical Analysis ?
     The generic difficulty, Mathematical Modelling and Numerical issues, Conditioning and Stability of a problem, The need to approximate, Using existing methods as implemented in available Software libraries
   - Mathematical Preliminaries:
     Floating Point arithmetic, Relevant results/theorems from Calculus,

2. Interpolation and Approximation [3 weeks].

   - Approximation and interpolation – basic problem [3]
   - Existence and uniqueness of polynomial interpolant [3.1]
   - Representation of interpolating polynomial [3.1]
   - Newton basis (divided differences), monomial basis, Lagrange basis [3.1]
   - Properties of divided differences [3.2]
   - Errors in polynomial interpolation [3.1]
   - Interpolating with derivative constraints – Hermite interpolation (or osculatory interpolation) [3.3]
   - Difficulties with polynomial interpolation [3.4]
   - Piecewise polynomials and splines [3.4]

- Linear and Cubic Splines [3.4]
- Choice of end-condition [3.4]
- Shape preserving splines and 2-D splines

3. Least Squares Polynomial Approximation [1 week].
   - Least squares approximation –discrete and continuous [8.2]
   - Normal equations and existence/uniqueness of solution [8.2]
   - QR algorithm
4. Numerical Quadrature [4 weeks].
   - The basic Problem [4.3]
   - Interpolatory Rules [4.3]
   - Errors in Interpolatory Rules [4.3]
   - Gaussian Quadrature [4.7]
   - Composite Quadrature Rules [4.4]
   - Extrapolation – Romberg Quadrature [4.5]
   - Adaptive Quadrature Rules [4.6]
   - Special Difficulties [4.9]
   - Two Dimensional Quadrature [4.8]
5. Numerical Ordinary Differential Equations [3 weeks].
   - Mathematical Preliminaries [5.1]
   - Taylor Series Methods [5.2, 5.3]
   - Runge-Kutta Methods [5.4]
   - Higher-order Runge-Kutta Methods [5.4, 5.5]
   - Error Estimates [5.5]
   - Stepsize Control [5.5]

1 General information / Mathematical Background

1.1 What is Numerical Analysis?

- Consider the investigation of a well defined mathematical model arising in any application area. The problem is 'well defined' in the sense that there exists a 'solution' and it is unique.
- We are interested in the 'Conditioning' (or sensitivity) of the underlying mathematical problem. That is, do small changes in the data defining the problem lead to 'small' changes in the exact (unique) solution?

- For virtually all mathematical models of practical interest one cannot determine a useful 'closed form' expression for the exact solution and one must approximate the exact solution.

- In Numerical Analysis we develop and analyse algorithms to approximate the exact solution to mathematical problems.
  - Algorithms must be constructive and finite (time and space).
  - We must analyse the errors in the approximation.
  - We must also quantify the stability and efficiency of the algorithms.

- The focus of this course is on the intelligent use of existing algorithms embedded in widely used numerical software. (It is NOT focused on deriving algorithms or writing code.)
  - How to interpret the numerical (approximate) results.
  - What method (algorithm) should be used.
  - Can a 'standard' method be applied to a particular problem?
  - What methods are available in the usual 'Problem Solving Environments' that scientists, engineers and students work in. For example in MATLAB, MAPLE or Mathematica.
  - In order to appreciate the limitations of the methods we will work with (primarily in MATLAB) we must analyse and understand the underlying algorithms on which the methods are based.

1.2 Mathematical Preliminaries (A Review):

1. Floating Point Arithmetic (from CSCC50 and CSCB70):
   Recall that a floating point number system, Z, can be characterized by four parameters, (β, s, m, M), and each element of Z is defined by:

   $$z = .d_1 d_2 \cdots d_s \times \beta^e,$$

   where $d_1 \neq 0$, $0 \leq d_i \leq \beta$, and $m \leq e \leq M$.

   The floating point representation mapping, $fl(x)$, is a mapping from the Reals to Z that satisfies:

   $$fl(x) = x(1+\epsilon), \text{ with } |\epsilon| \leq \mu.$$

   where μ is the 'unit roundoff' and is defined to be $1/2\ \beta^{1-s}$.

   For any standard elementary arithmetic operation (+, -, $\times$ and /), we have the corresponding F.P. approximation (denoted by $\oplus, \ominus, \otimes$ and $\oslash$) which satisfies, for any $a, b \in Z$,

   $$a \odot b = fl(a \cdot b) = (a \cdot b)(1+\epsilon),$$

   where $|\epsilon| \leq \mu$ and $\cdot$ is any elementary operation.

For any real-valued function, $F(a_1, a_2, \cdots a_n)$, the most we can expect is that the floating point implementation $\bar{F}$, will return (when invoked) the value $\bar{y}$ satisfying:

$$\begin{aligned}
\bar{y} &= \bar{F}(fl(a_1), fl(a_2), \cdots fl(a_n)), \\
&= \bar{F}(a_1(1+\epsilon_1), a_2(1+\epsilon_2), \cdots a_n(1+\epsilon_n)), \\
&= fl(F(a_1(1+\epsilon_1), a_2(1+\epsilon_2), \cdots a_n(1+\epsilon_n)).
\end{aligned}$$

Therefor if $y = F(a_1, a_2, \cdots a_n)$ is the desired result (defined by exact arithmetic over the Reals), the computed value (computed in FP arithmetic), $\bar{y}$, will at best satisfy (for differentiable functions, F):

$$\begin{aligned}
\frac{|\bar{y} - y|}{|y|} &\leq \frac{\|(\frac{\partial F}{\partial \underline{x}})^T\| \, \|\epsilon\|}{\|F\|}, \\
&\leq \frac{\|\frac{\partial F}{\partial \underline{x}}\| \, \|\epsilon\|}{\|F\|},
\end{aligned}$$

where $\epsilon = [\epsilon_1, \epsilon_2 \cdots \epsilon_n]^T$, and

$$(\frac{\partial F}{\partial \underline{x}})^T = [\frac{\partial F}{\partial x_1}, \frac{\partial F}{\partial x_2}, \cdots \frac{\partial F}{\partial x_n}],$$

evaluated at $\underline{x} = (a_1, a_2, \cdots a_n)$. From this expression we see that we could be in trouble with an inherent (independent of the approximation used) amplification of relative error whenever

$$\frac{\|\frac{\partial F}{\partial \underline{x}}\|}{\|F\|}$$

is large.

2. Basic Notation:

We will use standard notation from calculus and analysis. For example:

- $[a, b]$ is the closed interval, ($x \in R$, such that $a \leq x \leq b$).
- (a, b) is the open interval, ($x \in R$, such that $a < x < b$).
- $f^n(x) = \frac{d^n}{dx^n} f(x)$.
- $f \in C^n[a, b] \;\Rightarrow\; f$ is n times differentiable on $[a, b]$ and $f^n(x)$ is continuous on (a, b).
- $g_x(x, y) \equiv \frac{\partial}{\partial x} g(x, y)$, $g_y(x, y) \equiv \frac{\partial}{\partial y} g(x, y)$, $g_{xy}(x, y) \equiv \frac{\partial^2}{\partial x \partial y} g(x, y)$ etc.
- $g(h) = O(h^n)$ as $h \to 0 \Leftrightarrow \exists h_0 > 0 \; and \; K > 0 \ni |g(h)| < K h^n \; \forall \; 0 < h < h_0$.

3. Useful Theorems from Calculus:

We will also frequently use the classical theorems from analysis such as:

Intermediate Value Theorem Let $f(x)$ be continuous on $[a,b]$. If $f(x_1) < \alpha < f(x_2)$ for some α and $x_1,\ x_2 \in [a,b]$, then $\alpha = f(\eta)$ for some $\eta \in [a,b]$.

Max-Min Theorem Let $f(x)$ be continuous on $[a,b]$. Then $f(x)$ assumes its maximum and minimum values on $[a,b]$. (That is, $\exists \underline{x}$ *and* $\bar{x}\ \in [a,b] \ni \forall x \in [a,b]$, we have $f(\underline{x}) \le f(x) \le f(\bar{x})$.)

Mean Value Theorem for Integrals Let $g(x)$ be a non-negative (or non-positive) integrable function on $[a,b]$. If $f(x)$ is continuous on $[a,b]$ then

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

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

Mean Value Theorem for Sums Let $f(x) \in C^1[a,b]$, let $x_1, x_2, \cdots, x_n$ be points in $[a,b]$ and let $w_1, w_2, \cdots, w_n$ be real numbers of one sign, then

$$\sum_{i=1}^{n} w_i f(x_i) = f(\eta)\sum_{i=1}^{n} w_i,$$

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

Rolle's Theorem Let $f(x) \in C^1[a,b]$. If $f(a) = f(b) = 0$ then $f'(\eta) = 0$ for some $\eta \in (a,b)$.

Mean Value Theorem for Derivatives If $f(x) \in C^1[a,b]$ then

$$\frac{f(b)-f(a)}{b-a} = f'(\eta),$$

for some $\eta \in (a,b)$.

Fundamental Theorem of Calculus If $f(x) \in C^1[a,b]$ then $\forall x \in [a,b]$ and any $c \in [a,b]$ we have

$$f(x) = f(c) + \int_c^x f'(s)ds.$$

Taylor's Theorem (with remainder) If $f(x) \in C^{n+1}[a,b]$ and c is any point in $[a,b]$, then for $x \in [a,b]$, we have

$$f(x) = f(c) + f'(c)(x-c) + f''(c)\frac{(x-c)^2}{2}\cdots + f^n(c)\frac{(x-c)^n}{n!} + R_{n+1}(x),$$

where $R_{n+1}(x) = \frac{1}{n!}\int_c^x (x-u)^n f^{n+1}(u)du$.

Note that Taylor's Theorem is particularly relevant to this course. We can observe that, since $(x-u)^n$ is of constant sign for $u \in [c,x]$ we can write

$$R_{n+1}(x) = \frac{1}{n!}\int_c^x (x-u)^n f^{n+1}(u)du = f^{n+1}(\eta)\frac{(x-c)^{n+1}}{(n+1)!},$$

for some $\eta \in [c,x]$.

We can also observe the first few terms of the Taylor Series provides an accurate approximation to $f(c+h)$ for small h since we have

$$f(c+h) = f(c) + hf'(c) + \cdots \frac{h^n}{n!} f^n(c) + \frac{h^{n+1}}{(n+1)!} f^{n+1}(\eta).$$

where the error term, $E(h)$ is $O(h^{n+1})$.

2 Interpolation and Approximation

1. **The Basic Problem:**
   Approximate a continuous function $f(x)$, by a polynomial $p(x)$, over $[a, b]$.
   - $f(x)$ may only be known in tabular form.
   - $f(x)$ may be expensive to compute.

2. **Definition:**
   A polynomial $p(x)$ <u>interpolates</u> $f(x)$ at the nodes $x_0, x_1, \cdots x_n$ if $p(x_i) = f(x_i)$ for $i = 0, 1, \cdots n$.

   (Intuitively if $f(x)$ and $p(x)$ agree at the x_i then they should be 'close' at nearby points.)

3. **Key Theorem:**
   Given distinct nodes $x_0, x_1, \cdots x_n$ and arbitrary $f_0, f_1, \cdots f_n$, there is a unique polynomial $p_n(x)$ of degree at most n that interpolates $f(x)$ at $x_0, x_1, \cdots x_n$.

   <u>Proof:</u>

   (a) Existence: (constructive)
   Let $P_n(x) = a_0 + a_1(x - x_0) + \cdots a_n(x - x_0)(x - x_1) \cdots (x - x_{n-1})$.
   For any choice of $a_0, a_1, \cdots a_n, P_n(x)$ will be of degree at most n. We will choose the a_i to ensure that $P_n(x_i) = f_i$. This results in a system of $n+1$ linear equations in the $n+1$ unknowns, $a_0, a_1, \cdots a_n$. The i^{th} equation is:

   $$a_0 + a_1(x_i - x_0) + \cdots a_n(x_i - x_0)(x_i - x_1) \cdots (x_i - x_{n-1}) = f_i,$$

   and we have (in matrix form):

   $$B\underline{a} = \underline{f},$$

   where $\underline{a}^T = [a_0, a_1, \cdots a_n]^T, \underline{f} = [f_0, f_1, \cdots f_n]^T$, and B is an $(n+1) \times (n+1)$ matrix,

   $$B = (b_{ij}), \quad b_{ij} = \begin{cases} 1 & \text{for } j = 0; i = 0, 1, \cdots n, \\ (x_i - x_0)(x_i - x_1) \cdots (x_i - x_{j-1}) & \text{for } j = 1, 2, \cdots n; i = 0, 1, \cdots n. \end{cases}$$

   Note:

i. B is lower triangular since b_{ij} contains a factor $(x_i - x_i)$ for $j > i$.

ii. $b_{ii} \neq 0$ since $x_0, x_1, \cdots x_n$ are distinct. This implies B is nonsingular and there exists a unique solution $\underline{a}$. Solving this triangular linear system by forward substitution we have:

$$\begin{aligned} a_0 &= f_0, \\ a_1 &= (f_1 - a_0)/(x_1 - x_0), \\ &\vdots \\ a_n &= [f_n - (a_0 + a_1(x_n - x_0) + \cdots a_{n-1}(x_n - x_0)(x_n - x_1)\cdots(x_n - x_{n-2}))] \\ &\quad /[(x_n - x_0)(x_n - x_1)\cdots(x_n - x_{n-1})]. \end{aligned}$$

iii. The first $r+1$ terms, $a_0, a_1, \cdots a_r$, determine a polynomial of degree at most r that interpolates $f(x)$ at $x_0, x_1, \cdots x_r$.

iv. The resulting $P_n(x)$ satisfies the interpolation conditions and the coefficients $a_0, a_1, \cdots a_n$ define the Newton Form of this polynomial. (The Newton Form depends on the nodes and their order.)

(b) Uniqueness:
Let $q_n(x)$ be a polynomial of degree at most n such that $q_n(x_i) = f_i$ for $i = 0, 1, \cdots n$. Then with $r(x) \equiv P_n(x) - q_n(x)$ we observe that $r(x)$ is a polynomial of degree at most n such that $r(x_i) = 0$ for $i = 0, 1, \cdots n$. Then, by Rolle's theorem, $r'(x)$ has n distinct zeros, $r''(x)$ has $n-1$ distinct zeros, ... and $r^n(x)$ has one zero in the interval containing the nodes.

But this is impossible unless $r(x) \equiv 0$, in which case $q_n(x)$ must equal $P_n(x)$.

4. **Representation of Interpolating Polynomials:**
The interpolating polynomial $P_n(x)$ is unique but it can be represented in different ways. Consider the following representations of $P_n(x)$:

(a) Monomials (powers of x or $(x - w)$):

$$P_n(x) = c_0 + c_1 x + \cdots c_n x^n,$$

(In this case $P_n(x)$ is represented by the coefficients $c_0, c_1, \cdots c_n$ and we do not need to know the nodes or their order to evaluate or use this polynomial.)

(b) Newton Basis (or divided differences):

$$P_n(x) = a_0 + a_1(x - x_0) + \cdots a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}),$$

(In this case $P_n(x)$ is represented by the a_i's and one must know the nodes and their order to use this polynomial.)

(c) Lagrange Basis:
We introduce the j^{th} Lagrange basis function $\ell_j(x)$ (associated with the nodes $[x_i]_{i=0}^n$) by

$$\ell_j(x) = \prod_{i=0, i\neq j}^{n} (x - x_i) / \prod_{i=0, i\neq j}^{n} (x_j - x_i) = \prod_{i=0, i\neq j}^{n} \frac{(x - x_i)}{(x_j - x_i)},$$

for $j = 0, 1, \cdots n$.
Then it is clear that $\ell_j(x)$ is a polynomial of degree n such that

$$\ell_j(x_i) = \begin{cases} 0 & \text{for } j = 0, 1, \cdots n; j \neq i, \\ 1 & \text{for } j = i. \end{cases}$$

It is also clear that any linear combination of the $\ell_j(x)$ will be a polynomial of degree at most n. In particular,

$$P_n(x) = \sum_{j=0}^{n} f_j \ell_j(x),$$

since $P_n(x)$ is unique and the polynomial $\sum_{j=o}^{n} f_j \ell_j(x)$ satisfies,

$$\sum_{j=0}^{n} f_j \ell_j(x_i) = f_i, \quad \text{for } i = 0, 1, \cdots n.$$

(In this case $P_n(x)$ is represented by the function values, $f_0, f_1, \cdots f_n$ and we must know the nodes to use $P_n(x)$.)
The particular choice of what representation to use will depend on the application. Often we will use the Lagrange form in our analysis but other forms in our implementations.
For example, with the Lagrange representation of $P_n(x)$ one cannot easily determine $P_n^{'}(x)$ while this task is easy with the monomial representation.
An Example:
Consider the unique quadratic defined by the three interpolating conditions, $P_2(-1) = 7, P_2(0) = 2$, and $P_2(1) = 1$. We then have the nodes $x_0 = -1, x_1 = 0$, and $x_2 = 1$ with the corresponding function values $f_0 = 7, f_1 = 2$ and $f_2 = 1$. The Lagrange basis set is,

$$\begin{aligned} \ell_0(x) &= \frac{(x-0)(x-1)}{(-1-0)(-1-1)} = \frac{x^2-x}{2}, \\ \ell_1(x) &= \frac{(x+1)(x-1)}{(0+1)(0-1)} = \frac{x^2-1}{-1} = 1 - x^2, \\ \ell_2(x) &= \frac{(x+1)x}{(1+1)(1-0)} = \frac{x^2+x}{2}. \end{aligned}$$

In Lagrange form $P_2(x)$ is

$$\begin{aligned} P_2(x) &= f_0\ell_0(x) + f_1\ell_1(x) + f_2\ell_2(x), \\ &= 7(\frac{x^2-x}{2}) + 2(1-x^2) + \frac{x^2+x}{2}. \\ (&= 2x^2 - 3x + 2 \quad \text{(in monomial form))} \end{aligned}$$

5. **Divided Differences**

(a) Definition: The divided difference, $f[x_0, x_1, \cdots x_i]$ is defined to be a_i the $(i+1)^{st}$ coefficient of the interpolating polynomial $P_n(x)$ written in Newton form.

$$P_n(x) = a_0 + a_1(x - x_0) + \cdots a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}).$$

(b) Recall that the first $r + 1$ coefficients of the Newton form of $P_n(x)$ determine a polynomial of degree at most r that interpolates $f(x)$ at the nodes $(x_0, x_1, \cdots x_r)$.

(c) Properties of Divided Differences:

i. $f[x_0, x_1, \cdots x_i] = f[x_{j_0}, x_{j_1}, \cdots x_{j_i}]$, where $j_0, j_1, \cdots j_i$ is a permutation of $0, 1, \cdots i$.

ii. $f[x_0, x_1, \cdots x_i] = \frac{f[x_0, x_1, \cdots x_{i-1}] - f[x_1, x_2, \cdots x_i]}{x_0 - x_i}$.

Proof of these properties:

i. The polynomial of degree at most i interpolating $f(x)$ at $(x_0, x_1, \cdots x_i)$ is

$$P_i(x) = f[x_0] + f[x_0, x_1](x - x_0) + \cdots f[x_0, x_1, \cdots x_i](x - x_0)(x - x_1)\cdots(x - x_{i-1}),$$

Note that the coefficient of x^i in this polynomial is $f[x_0, x_1 \cdots x_i]$. Similarly the polynomial $\bar{P}_i(x)$ interpolating $f(x)$ at $(x_{j_0}, x_{j_1}, \cdots x_{j_i})$ is

$$\bar{P}_i(x) = f[x_{j_0}] + f[x_{j_0}, x_{j_1}](x - x_{j_0}) + \cdots f[x_{j_0}, x_{j_1}, \cdots x_{j_i}](x - x_{j_0})(x - x_{j_1})\cdots(x - x_{j_{i-1}}).$$

Now the sets $[x_0, x_1, \cdots x_i]$ and $[x_{j_0}, x_{j_1}, \cdots x_{j_i}]$ are identical (only the order may be different) so by uniqueness we have $P_i(x) = \bar{P}_i(x)$. In particular the respective coefficients of x^i must agree and we have,

$$f[x_0, x_1 \cdots x_i] = f[x_{j_0}, x_{j_1}, \cdots x_{j_i}].$$

ii. Consider the orderings $(x_1, x_2, \cdots x_i, x_0)$ and $(x_0, x_1, \cdots x_i)$. Writing $P_i(x)$ using the two different Newton Forms (corresponding to these different orderings) we have,

$$\begin{aligned}
P_i(x) &= f[x_0] + f[x_0, x_1](x - x_0) + \cdots f[x_0, x_1, \cdots x_i](x - x_0)(x - x_1)\cdots(x - x_{i-1}), \\
&= f[x_1] + f[x_1, x_2](x - x_1) + \cdots f[x_1, x_2, \cdots x_i, x_0](x - x_1)(x - x_2)\cdots(x - x_i).
\end{aligned}$$

Multiplying the first equation by $(x - x_i)$, the second equation by $(x - x_0)$ and subtracting we obtain,

$$\begin{aligned}
(x - x_i)P_i(x) &- (x - x_0)P_i(x) = (x_0 - x_i)P_i(x), \\
&= f[x_0](x - x_i) - f[x_1](x - x_0)\cdots \\
&+ f[x_0, x_1, \cdots x_{i-1}](x - x_0)(x - x_1)\cdots(x - x_{i-2})(x - x_i) \\
&\quad - f[x_1, x_2, \cdots x_i](x - x_0)(x - x_1)\cdots(x - x_{i-1}) \\
&+ f[x_0, x_1, \cdots x_i](x - x_0)(x - x_1)\cdots(x - x_i) \\
&\quad - f[x_1, x_2, \cdots x_i, x_0](x - x_0)(x - x_1)\cdots(x - x_i).
\end{aligned}$$

But the last term in this expansion vanishes and this implies the coefficient of x^i in the polynomial $(x_0 - x_i)P_i(x)$ is $f[x_0, x_1, \cdots x_{i-1}] - f[x_1, x_2, \cdots x_i]$. But we have shown before that the coefficient of x^i in $P_i(x)$ is $f[x_0, x_1 \cdots x_i]$ so we can conclude,

$$f[x_0, x_1, \cdots x_i] = \frac{f[x_0, x_1 \cdots x_{i-1}] - f[x_1, x_2, \cdots x_i]}{x_0 - x_i}.$$

6. **Error in Interpolation**

(a) Let $E_n(x) = f(x) - P_n(x)$. To investigate the behaviour of $E_n(x)$ consider fixing x and determine the interpolating polynomial $p_{n+1}(z)$ of degree at most $n+1$ (in z), interpolating $f(z)$ at the $n+2$ nodes $(x_0, x_1, \cdots x_n, x)$. We then have,

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

Evaluating this expression at $z = x$ and using the fact that $p_{n+1}(x) = f(x)$ we have,

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

(b) If $f[x_0, x_1, \cdots x_n, x]$ (as a function of x) is 'slowly varying', then we can estimate $E_n(x)$ by

$$\boxed{est_n(x) = \prod_{i=0}^{n}(x - x_i) f[x_0, x_1, \cdots x_n, x_{n+1}]}$$

for some x_{n+1}.

(c) 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 this implies

$$E_r'(x) \text{ has at least } r \text{ distinct zeros in } [a, b],$$

$$\Rightarrow \quad E_r''(x) \text{ has at least } r-1 \text{ distinct zeros in } [a,b],$$

$$\vdots$$

$$\Rightarrow \quad E_r^{(r)}(x) \text{ has at least 1 zero in } [a,b].$$

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:

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

(d) 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,

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

NOTE:

i. This is an expression for the exact error in interpolation. We can use it to derive an overall error bound and to provide guidance in the choice of interpolation points.

ii. Increasing n may not imply $|E_n(x)| \to 0$ since $|f^{(n+1)}|$ may grow faster than $(n+1)!$ as n increases.

iii. One can show (a classical theorem in approximation theory) that for any continuous function, $f(x)$ defined on $[a,b]$, and $\epsilon > 0$ there is an $n > 0$ and set of interpolation points $(x_0, x_1, \cdots x_n)$ such that $|E_n(x)| < \epsilon \ \forall x \in [a,b]$. (The x_i's will depend on f and ϵ.)

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

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

(e) 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, $d_{i,j},\ i = 0, 1, \cdots n;\ \ j = 0, 1, \cdots i$ where

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

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

7. **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. The associated tableau of divided differences is presented in Table 1.

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	$\frac{0-2}{1-2} = 2$	$\frac{-1-2}{0-2} = 3/2$	$\frac{2-3/2}{-1-2} = -1/6$	$\frac{11/6+1/6}{-2-2} = -1/2$

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,

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

8. **Hermite (or osculatory) Interpolation**

In many applications one is interested in deriving interpolating 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.

Recall that for the standard case of interpolating solution values only at $n + 1$ distinct interpolation points $(x_0, x_1, \cdots, x_n)$ we can determine the Newton form of $P_n(x)$,

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

using the diagonal entries of the divided difference 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	$f[x_n] = f(x_n)$	$\frac{f(x_{n-1})-f(x_n)}{x_{n-1}-x_n} = f[x_{n-1}, x_n]$	$\frac{f[x_{n-2},x_{n-1}]-f[x_{n-1},x_n]}{x_{n-2}-x_n}$

In the limit as two interpolation nodes coalesce (ie, $x_m \to x_{m+1}$), the corresponding entries of the divided difference 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	$f[x_m] = f(x_m)$	$f^{'}(x_m)$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$

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.

9. **Piecewise Polynomial Approximation:**

   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 $(\xi_{i,0}, \xi_{i,1}, \cdots \xi_{i,n})$, and then approximate $f(x)$ by the interpolating polynomial $P_{i,n}(x)$ of degree at most n. The resulting 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 <u>piecewise polynomial</u>.

   Note:

   (a) The evaluation of $S(x)$ requires an initial search to locate the correct interval such that $x \in [x_{i-1}, x_i]$.

   (b) The error in this approximation satisfies,

   $$\begin{aligned} |f(x) - S(x)| &= \left|\frac{f^{n+1}(\eta)}{(n+1)!} \prod_{j=0}^{n} (x - \xi_{i,j})\right| \\ &\leq \frac{L}{(n+1)!} h_i^{n+1}, \end{aligned}$$

   where L is a bound on $|f^{n+1}(x)|$ and $h_i = (x_i - x_{i-1})$ (usually a constant $= \frac{(b-a)}{M}$).

   (c) $S(x)$ will be continuous if the endpoints of each subinterval are interpolation points. That is, the set of points $[\xi_{i,j}]_{j=0}^{n}$ must include x_i and x_{i-1}. $S(x)$ will not in general be differentiable.

10. **An Example – 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 (usually $h = (b - a)/M$).

11. **Piecewise Cubic Approximations (3 possibilities):**

(a) Piecewise Cubic Interpolant: On each $[x_{i-1}, x_i]$, introduce $\xi_{i,0} = x_{i-1}, \xi_{i,3} = x_i$ and two additional points $\xi_{i,1}, \xi_{i,2}$ to define $P_{i,3}(x)$. We then have,

$$|f(x) - S(x)| \leq \frac{L}{4!} h^4, \quad h = \max_{i=1}^{M} |x_i - x_{i-1}|.$$

Note that $S(x)$ is continuous but not 'smooth'.

(b) Hermite Cubic: Define $\bar{P}_{i,3}(x)$ to be the unique cubic polynomial associated with $[x_{i-1}, x_i]$ such that $\bar{P}_{i,3}(x_{i-1}) = f_{i-1}$, $\bar{P}'_{i,3}(x_{i-1}) = f'_{i-1}$, $\bar{P}_{i,3}(x_i) = f_i$ and $\bar{P}'_{i,3}(x_i) = f'_i$. Recall that this can be viewed as the limiting case of piecewise cubic interpolation where $\xi_{i,1} \to x_{i-1}$ and $\xi_{i,2} \to x_i$ with $f[x_{i-1}, \xi_{i,1}] \to f'_{i-1}$ and $f[x_i, \xi_{i,2}] \to f'_i$.

Note:

i. The resulting $\bar{S}(x)$ will be continuous and differentiable since $\bar{P}'_{i,3}(x_{i-1}) = f'_{i-1} = \bar{P}'_{i-1,3}(x_{i-1})$.

ii. The error in $\bar{S}(x)$ satisfies

$$\begin{aligned} |f(x) - \bar{S}(x)| &\leq \frac{L}{4!}(x - x_{i-1})^2 (x - x_i)^2, \\ &\leq \frac{L}{4!}(h/2)^4, \end{aligned}$$

where $L \geq \max_{a \leq x \leq b}\{|f^4(x)|\}$ and $h = \max_{i=1}^{M} |x_i - x_{i-1}|$.

iii. In some applications f'_i may not be available or may be too expensive to compute.

iv. The resulting $\bar{S}(x)$ will be continuous and 'smoother' than Piecewise cubic interpolants, but $\bar{S}''(x)$ will usually be discontinuous.

(c) Cubic Splines (concentrate on smoothness of $S(x)$):

The basic idea is to determine the polynomial, $\hat{P}_{i,3}(x)$, associated with $[x_{i-1}, x_i]$, by requiring $\hat{P}_{i,3}(x)$ to interpolate $f_{i-1},\ f_i$ and have continuous first and second derivatives at x_{i-1}.

Consider the following representation of $\hat{P}_{i,3}(x)$,

$$\hat{P}_{i,3}(x) = c_{i,0} + c_{i,1}(x - x_{i-1}) + c_{i,2}(x - x_{i-1})^2 + c_{i,3}(x - x_{i-1})^3,$$

which can be stored as a $4 \times M$ array.

The piecewise polynomial, $\hat{S}(x)$ is determined by specifying $4M$ linear equations which uniquely determine the $c_{i,j}$'s. To do this we let $h_i = x_i - x_{i-1}$ and we associate 'interpolation' and 'smoothness' constraints with each subinterval.

- On the first subinterval (3 interpolation constraints),

$$\begin{aligned} \hat{P}'_{1,3}(x_0) = ? &\quad\Rightarrow\quad c_{1,1} = ?\ , \\ \hat{P}_{1,3}(x_0) = f_0 &\quad\Rightarrow\quad c_{1,0} = f_0, \\ \hat{P}_{1,3}(x_1) = f_1 &\quad\Rightarrow\quad c_{1,0} + c_{1,1}h_1 + c_{1,2}h_1^2 + c_{1,3}h_1^3 = f_1, \end{aligned}$$

- On the second subinterval (2 interpolation and 2 smoothness constraints),

$$\begin{aligned}
\hat{P}_{2,3}(x_1) = f_1 &\Rightarrow c_{2,0} = f_1, \\
\hat{P}'_{2,3}(x_1) = \hat{P}'_{1,3}(x_1) &\Rightarrow c_{2,1} - c_{1,1} - 2c_{1,2}h_1 - 3c_{1,3}h_1^2 = 0, \\
\hat{P}''_{2,3}(x_1) = \hat{P}''_{1,3}(x_1) &\Rightarrow 2c_{2,2} - 2c_{1,2} - 6c_{1,3}h_1 = 0, \\
\hat{P}_{2,3}(x_2) = f_2 &\Rightarrow c_{2,0} + c_{2,1}h_2 + c_{2,2}h_2^2 + c_{2,3}h_2^3 = f_2,
\end{aligned}$$

- In general on the i^{th} subinterval (2 interpolation and 2 smoothness constraints),

$$\begin{aligned}
\hat{P}_{i,3}(x_{i-1}) = f_{i-1} &\Rightarrow c_{i,0} = f_{i-1}, \\
\hat{P}'_{i,3}(x_{i-1}) = \hat{P}'_{i-1,3}(x_{i-1}) &\Rightarrow c_{i,1} - c_{i-1,1} - 2c_{i-1,2}h_{i-1} - 3c_{i-1,3}h_{i-1}^2 = 0, \\
\hat{P}''_{i,3}(x_{i-1}) = \hat{P}''_{i-1,3}(x_{i-1}) &\Rightarrow 2c_{i,2} - 2c_{i-1,2} - 6c_{i-1,3}h_{i-1} = 0, \\
\hat{P}_{i,3}(x_i) = f_i &\Rightarrow c_{i,0} + c_{i,1}h_i + c_{i,2}h_i^2 + c_{i,3}h_i^3 = f_i,
\end{aligned}$$

- And finally on the last subinterval we impose an additional interpolation constraint:

$$\hat{P}'_{M,3}(x_M) = c_{M,1} + 2c_{M,2}h_M + 3c_{M,3}h_M^2 = ?.$$

Note:

i. The two extra interpolation constraints imposed at x_0 and x_M can be set by specifying f_0' and f_M' or by using the respective approximating values, $f[x_0, x_1]$ and $f[x_{M-1}, x_M]$ or by replacing these constraints with $\hat{P}''_{1,3}(x_0) = \hat{P}''_{M,3}(x_M) = 0$. The latter choice leads to what are called 'natural splines'.

ii. The total number of linear equations is then $3+4(M-1)+1 = 4M$. This system can be shown to be nonsingular as long as the x_i's are distinct.

iii. The error bounds for splines are similar to that for cubic Hermite. That is, it can be shown that, in most cases

$$|f(x) - \hat{S}(x)| \leq \frac{5L}{4!}(h/2)^4$$

(although for natural splines we only have $O(h^2)$ accuracy).

iv. The choice of the 'knots' (the x_i's) can be significant.

v. For an alternative derivation of splines see Johnson and Riess.

(d) **Difficulties with Piecewise Cubics:**

- They do not preserve monotone data. That is if the data is monotone (increasing or decreasing) then, in some applications, so should the interpolant. This leads to the notion of 'Splines under tension' and results in additional (nonlinear) constraints on the coefficients.

- They do not preserve discontinuous derivatives. For example in representing 'corners' in parametric curves.
- They assume accurate data. In many applications (in particular in those arising in CAD and computer graphics) one often wants to represent the 'general shape' of the function or curve rather than insisting on strict interpolation. This leads to the notion of Bezier Curves/approximation which is useful in interactive design using curves and surfaces. Another approach to cope with this difficulty is to use 'data fitting' or linear least squares.

12. **Interpolation in 2 Dimensions:**

Consider the problem of approximating $u(x, y)$ in a finite region of R^2 or approximating a surface in 3-dimensions. We can generalize the notion of piecewise polynomials to higher dimensions in a natural way:

(a) The original region (or domain) is first decomposed (or partitioned) into a collection of regularly-shaped subregions. Usually rectangles or triangles are used. If the original domain is not a simple shape there may be some 'approximation' introduced in this partitioning process.

(b) Over each subregion (the triangular or rectangular mesh element) one can define a bi-variate polynomial (in x and y) and use the total collection of such polynomials to define the bivariate piecewise polynomial, $S(x, y)$.

(c) As in the one dimensional problem one can define the coefficients of the polynomials associated with each element by imposing interpolation constraints, solution continuity constraints at the boundaries, and/or smoothness of derivatives at the boundaries.

3 Least Square Polynomial Approximation

1. **The basic Problem– Data Fitting:**

Given data $\{x_i, f_i\}_{i=0}^m$ find the polynomial $p_n(x)$ of degree at most n represented by,

$$p_n(x) = c_0 + c_1 x + \cdots c_n x^n,$$

such that $G(\underline{c})$ is minimized, where $G(\underline{c})$ is defined by,

$$G(\underline{c}) = \sum_{i=0}^{m} (p_n(x_i) - f_i)^2.$$

- Other norms (measures of the error) could be minimized. For example we could use,

$$\hat{G}(\underline{c}) = \sum_{i=0}^{m} |p_n(x_i) - f_i| \quad \text{or} \quad \tilde{G}(\underline{c}) = \max_{i=0}^{m} |p_n(x_i) - f_i|,$$

but these measures are not differentiable and the resulting algorithms are more complex.

- We will assume that the x_i's are distinct and $m > n$. From standard results in linear algebra we know that these assumptions will guarantee a full rank problem with a unique solution.

2. **Characterization of the 'best fit',** $p_n(x)$

Let $r_i(\underline{c}) = p_n(x_i) - f_i$ for $i = 0, 1, \cdots m$. We then have,

$$\begin{aligned} r_i(\underline{c}) &= c_0 + c_1 x_i + \cdots c_n x_i^n - f_i, \\ &= \left[A\underline{c} - \underline{f}\right]_i, \end{aligned}$$

where A is the $(m+1) \times (n+1)$ matrix,

$$A = \begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_m & \cdots & x_m^n \end{bmatrix},$$

$\underline{c}$ is the $n{+}1$ vector, $\underline{c} = (c_0, c_1, \cdots c_n)^T$ and $\underline{f}$ is the $m{+}1$ vector, $\underline{f} = (f_0, f_1, \cdots f_m)^T$.

From standard results in calculus we know that $G(\underline{c})$ is a minimum when,

$$\frac{\partial G}{\partial c_j} = 0 \text{ for } \quad j = 0, 1, \cdots n.$$

But since $G(\underline{c}) = \sum_{i=0}^m r_i^2(\underline{c})$ we have,

$$\begin{aligned} \frac{\partial G}{\partial c_j} &= \frac{\partial}{\partial c_j}\left[\sum_{i=0}^m r_i^2(\underline{c})\right] && (1) \\ &= \sum_{i=0}^m \frac{\partial}{\partial c_j}(r_i^2(\underline{c})), && (2) \\ &= 2\sum_{i=0}^m r_i(\underline{c})\frac{\partial r_i(\underline{c})}{\partial c_j}. && (3) \end{aligned}$$

From the definition of $r_i(\underline{c})$ we have,

$$\begin{aligned} \frac{\partial r_i(\underline{c})}{\partial c_j} &= \frac{\partial}{\partial c_j}\left[A\underline{c} - \underline{f}\right]_i, \\ &= (a_{i,j}) = x_i^j, \end{aligned}$$

for $i = 0, 1, \cdots m; j = 0, 1, \cdots n$.

It then follows that

$$\frac{\partial G}{\partial c_j} = 2\sum_{i=0}^m r_i(\underline{c})a_{i,j} = 2\left(A^T\underline{r}\right)_j.$$

Therefore to achieve $\frac{\partial G}{\partial c_j} = 0$ for $j = 0, 1, \cdots n$ we must have,

$$\left(A^T \underline{r}\right)_j = 0, \text{ for } j = 0, 1, \cdots m.$$

This is equivalent to asking that,

$$A^T \underline{r} = \underline{0} \text{ or } A^T \left(A\underline{c} - \underline{f}\right) = \underline{0}.$$

Note that the matrix $A^T A$ is a square $(n+1) \times (n+1)$ nonsingular matrix and we can therefore compute the best fit least squares polynomial, $p_n(x)$ by solving the linear system:

$$\boxed{\boldsymbol{A^T A \underline{c} = A^T \underline{f}}}$$

These linear equations are called the <u>Normal Equations</u>.

3. **An Efficient Algorithm for solving the Normal Equations:**

To solve the Normal Equations efficiently we can use a QR based algorithm (from Numerical linear algebra) that doesn't require the explicit computation of the (possibly ill-conditioned) matrix $A^T A$.

Consider forming the $\mathcal{QR}$ factorization (or Schur decomposition) of the $(m+1) \times (n+1)$ matrix A,

$$A = \mathcal{QR} = (Q_1 Q_2 \cdots Q_{n+1})\mathcal{R},$$

where $\mathcal{Q}$ is an orthogonal matrix and $\mathcal{R}$ is an upper triangular matrix. This is a standard factorization in numerical linear algebra and is usually accomplished using a modified Gram Schmidt algorithm or an algorithm based on the use of a sequence of 'Householder reflections'. We will consider the latter approach.

That is, we will determine $\mathcal{Q}$ as a product of $n+1$ Householder reflections:

$$\mathcal{Q}A = \mathcal{R} \Leftrightarrow Q_{n+1}^T(Q_n^T \cdots Q_1^T A)) \cdots) = \mathcal{R},$$

where each $Q_i = Q_i^T$ is an $(m+1) \times (m+1)$ Householder reflection and $\mathcal{R}$ is an $(m+1) \times (n+1)$ upper triangular matrix,

$$\mathcal{R} = \begin{bmatrix} x & x & \cdots & x \\ 0 & x & \cdots & x \\ 0 & 0 & \cdots & x \\ \vdots & \vdots & \vdots & x \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \equiv \begin{bmatrix} R \\ 0 \end{bmatrix},$$

and R is a square $(n+1) \times (n+1)$ upper triangular matrix.

With such a factorization of A we have,

$$A^TA = (\mathcal{QR})^T\mathcal{QR} = \mathcal{R}^T\mathcal{Q}^T\mathcal{QR} = \mathcal{R}^T\mathcal{R},$$

where

$$\mathcal{R}^T\mathcal{R} = \begin{bmatrix} x & 0 & 0 & \cdots & 0 \\ x & x & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & 0 \\ x & x & x & \cdots & 0 \end{bmatrix} \begin{bmatrix} x & x & \cdots & x \\ 0 & x & \cdots & x \\ 0 & 0 & \cdots & x \\ \vdots & \vdots & \cdots & x \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} = \begin{bmatrix} x & x & \cdots & x \\ x & x & \cdots & x \\ \vdots & \vdots & \cdots & \vdots \\ x & x & \cdots & x \end{bmatrix}$$

$$= \begin{bmatrix} R^T & 0 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix} = R^TR.$$

Now solving the Normal Equations to determine $\underline{c}$ can be done by solving the equivalent linear system,

$$R^TR\underline{c} = A^T\underline{f}.$$

This requires the computation of $A^T\underline{f}$ (at a cost of $(n+1)(m+1)$ flops) and two triangular linear systems (at a cost of n^2 flops). The cost of determining the $\mathcal{QR}$ factorization of A is $n^2m + O(nm)$ and therefore the total cost of this algorithm to determine $\underline{c}$ is $n^2(m+1) + O(nm)$ flops.

Note that in most applications m is much larger than n and n is often less than 4.

We can define a similar algorithm for the continuous least squares problem, where we are asked to minimize $\bar{G}(\underline{c})$,

$$\bar{G}(\underline{c}) \equiv \int_a^b (p_n(x) - f(x))^2 dx.$$

In this case the definitions of $a_{i,j}$ and f_i are modified but a similar $\mathcal{QR}$-based algorithm is available.

4 Numerical Quadrature

1. **The basic Problem – approximation of integrals:**

   We will investigate methods for computing an approximation to the definite integral:

   $$I(f) \equiv \int_a^b f(x)dx.$$

   The obvious generic approach is to approximate the integrand $f(x)$ on the interval $[a, b]$ by a function that can be integrated exactly (such as a polynomial) and then take the integral of the approximating function to be an approximation to $I(f)$.

2. **Interpolatory Rules:**

   When the approximating function is an interpolating polynomial, $P_n(x)$, the corresponding approximation $I(P_n(x))$ is called an interpolatory rule. We will investigate several widely used interpolatory rules.

   Consider writing $P_n(x)$ in Lagrange form,

$$P_n(x) = \sum_{i=0}^{n} f(x_i) l_i(x),$$

   where $l_i(x)$ is defined by

$$l_i(x) = \prod_{j=0, j\neq i}^{n} \left(\frac{x - x_j}{x_i - x_j} \right).$$

   We then have

$$\begin{aligned} \int_a^b P_n(x)dx &= \int_a^b \sum_{i=0}^{n} f(x_i) l_i(x) dx \\ &= \sum_{i=0}^{n} f(x_i) \int_a^b l_i(x) dx \\ &= \sum_{i=0}^{n} \omega_i f(x_i). \end{aligned}$$

   Note:

   - The 'weights' (the ω_i's) depend only on the interval (the value of a and b) and on the x_i's. In particular these weights are independent of the integrand.
   - The interpolatory rules then approximate $I(f)$ by a linear combination of sampled integrand evaluations.
   - If $a = x_0 < x_1 < \cdots x_n = b$ are equally spaced the corresponding interpolatory rule is called a Newton-Coates quadrature rule.

3. **Errors in Interpolatory Rules:**

   The error associated with an interpolatory rule is $E(f) = I(f) - I(P_n)$ and satisfies,

$$\begin{aligned} E(f) &= \int_a^b f(x)dx - \int_a^b P_n(x)dx = \int_a^b [f(x) - P_n(x)]dx, \\ &= \int_a^b E_n(x)dx, \end{aligned}$$

   where $E_n(x)$ is the error in polynomial interpolation and satisfies,

$$\begin{aligned} E_n(x) &= (x - x_0)(x - x_1) \cdots (x - x_n) f[x_0, x_1, \cdots x_n, x], \\ &= \Pi_n(x) f[x_0, x_1, \cdots x_n, x]. \end{aligned}$$

   This expression for the error is valid for all interpolatory rules. In some special cases we can simplify this expression to obtain estimates and/or more insight into the behaviour of the error.

- First special case – If $\Pi_n(x)$ is of one sign (on $[a,b]$) then the Mean Value Theorem for Integrals implies,

$$\begin{aligned} E(f) &= \int_a^b f[x_0,x_1,\cdots x_n,x]\Pi_n(x)dx, \\ &= f[x_0,x_1,\cdots x_n,\xi]\int_a^b \Pi_n(x)dx, \end{aligned}$$

for some $\xi \in [a,b]$. Also since $f[x_0,x_1,\cdots x_n,\xi] = \frac{f^{(n+1)}(\eta)}{(n+1)!}$ for some $\eta \in (a,b)$, we have shown that if $\Pi_n(x)$ is of one sign then,

$$\boxed{E(f) = \tfrac{1}{(n+1)!}f^{(n+1)}(\eta)\textstyle\int_a^b \Pi_n(x)dx}$$

- Second special case – If $\int_a^b \Pi_n(x)dx = 0$ we have, for arbitrary x_{n+1},

$$f[x_0,x_1,\cdots x_n,x] = f[x_0,x_1,\cdots x_n,x_{n+1}] + f[x_0,x_1,\cdots x_{n+1},x](x-x_{n+1}),$$

and therefore,

$$\begin{aligned} E(F) &= \int_a^b f[x_0,x_1,\cdots x_n,x]\Pi_n(x)dx, \\ &= \int_a^b f[x_0,x_1,\cdots x_{n+1}]\Pi_n(x)dx + \int_a^b f[x_0,x_1,\cdots x_{n+1},x]\Pi_{n+1}(x)dx, \\ &= \int_a^b f[x_0,x_1,\cdots x_{n+1},x]\Pi_{n+1}(x)dx. \end{aligned}$$

As a result, if $\int_a^b \Pi_n(x)dx = 0$ and we can choose x_{n+1} so that $\Pi_{n+1}(x)$ is of one sign, then using a similar argument to that presented in the first special case, it follows that, if $\int_a^b \Pi_n(x)dx = 0$ and $\Pi_{n+1}(x)$ is of one sign,

$$\boxed{E(f) = \tfrac{1}{(n+2)!}f^{(n+2)}(\eta)\textstyle\int_a^b \Pi_{n+1}(x)dx}$$

4. **Examples of Interpolatory Rules:**

(a) Trapezoidal Rule (an example of the first special case):

$$T(f) \equiv \int_a^b P_1(x)dx,$$

where $x_0 = a$ and $x_1 = b$. We then have,

$$P_1(x) = l_0(x)f_0 + l_1(x)f_1 = \frac{x-x_1}{x_0-x_1}f_0 + \frac{x-x_0}{x_1-x_0}f_1.$$

Therefore we have

$$\begin{aligned} T(f) &= \int_a^b \frac{x-b}{a-b}dxf(a) + \int_a^b \frac{x-a}{b-a}dxf(b), \\ &= \left(\frac{b-a}{2}\right)f(a) + \left(\frac{b-a}{2}\right)f(b), \\ &= \left(\frac{b-a}{2}\right)[f(a)+f(b)]. \end{aligned}$$

We also have that $\Pi_1(x) = (x-a)(x-b)$ is negative for $x \in [a,b]$ and $\int_a^b \Pi_1(x)dx = -\frac{(b-a)^3}{6}$. We therefore have satisfied the conditions of the first special case and this implies,

$$\boxed{T(f) = (\tfrac{b-a}{2})[f(a)+f(b)], \;\; E^T(f) = \tfrac{-f''(\eta)}{12}(b-a)^3.}$$

(b) Simpsons Rule (an example of the second special case):

$$S(f) \equiv \int_a^b P_2(x)dx,$$

with $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$.
Exercise: Using

$$P_2(x) = l_0(x)f(a) + l_1(x)f\left(\frac{a+b}{2}\right) + l_2(x)f(b),$$

where

$$\begin{aligned} l_0(x) &= \frac{(x-\frac{a+b}{2})(x-b)}{(a-\frac{a+b}{2})(a-b)}, \\ l_1(x) &= \frac{(x-a)(x-b)}{(\frac{a+b}{2}-a)(\frac{a+b}{2}-b)}, \\ l_2(x) &= \frac{(x-a)(x-\frac{a+b}{2})}{(b-a)(b-\frac{a+b}{2})}. \end{aligned}$$

Simplify and verify (after some tedious algebra) that,

$$\begin{aligned} \int_a^b P_2(x)dx &= [\int_a^b l_0(x)dx]f(a) + [\int_a^b l_1(x)dx]f(\frac{a+b}{2}) + [\int_a^b l_2(x)dx]f(b), \\ \vdots & \;\; \vdots \\ &= \left(\frac{b-a}{6}\right)\left[f(a) + 4f(\frac{a+b}{2}) + f(b)\right]. \end{aligned}$$

Note that for $x \in [a,b]$, $\Pi_2(x)$ is antisymmetric about $\frac{a+b}{2}$ and this implies $\int_a^b \Pi_2(x)dx = 0$. Furthermore by choosing $x_3 = \frac{a+b}{2}$ we have

$$\Pi_3(x) = (x-a)(x-\frac{a+b}{2})^2(x-b),$$

is of one sign and this implies,

$$E^S(f) = I(f) - S(F) = \frac{1}{4!}f^{(4)}(\eta)\int_a^b \Pi_3(x)dx.$$

But $\int_a^b \Pi_3(x)dx = -\frac{4}{15}(\frac{b-a}{2})^5$ so we have,

$$\boxed{S(f) = (\tfrac{b-a}{6})[f(a) + 4f(\tfrac{a+b}{2}) + f(b)], \quad E^S(f) = \tfrac{-f^{(4)}(\eta)}{90}(\tfrac{b-a}{2})^5}$$

5. **Gaussian Quadrature (choosing the x_i's to maximize accuracy):**

   (a) Recall that the error in interpolatory rules satisfies,

   $$E(f) = \int_a^b f[x_0, x_1, \cdots x_n, x]\Pi_n(x)dx,$$

   and if $\int_a^b \Pi_n(x)dx = 0$ we have,

   $$E(f) = \int_a^b f[x_0, x_1, \cdots x_{n+1}, x]\Pi_{n+1}(x)dx,$$

   for <u>any</u> choice of x_{n+1}.
   Now if $\int_a^b \Pi_{n+1}(x) = 0$ as well we can repeat this argument and obtain,

   $$E(f) = \int_a^b f[x_0, x_1, \cdots x_{n+2}, x]\Pi_{n+2}(x)dx.$$

   For the general case, let $q_0(x) \equiv 1$ and $q_i(x) \equiv (x - x_{n+1})(x - x_{n+2}) \cdots (x - x_{n+i})$ for $i = 1, 2, \cdots (m-1)$. We can then show that if $\int_a^b \Pi_n(x)q_i(x)dx = 0$, for $i = 0, 1, \cdots (m-1)$ then,

   $$E(f) = \int_a^b f[x_0, x_1, \cdots x_{n+m}, x]\Pi_{n+m}(x)dx.$$

   (b) The key idea of Gaussian Quadrature is to choose the nodes or interpolation points, $(x_0, x_1, \cdots x_n)$ such that $\int_a^b \Pi_n(x)q(x)dx = 0$ for all polynomials, $q(x)$, of degree at most n. Therefore, in particular for the choice $q(x) = q_i(x)$ for $i = 0, 1, \cdots n$ we have $\int_a^b \Pi_n(x)q_i(x)dx = 0$, and from the above observation,

   $$E(f) = \int_a^b f[x_0, x_1, \cdots x_{2n+1}, x]\Pi_{2n+1}(x)dx.$$

   To ensure that $\Pi_{2n+1}(x)$ is of one sign for $x \in [a\ b]$ we can choose $x_{n+i} = x_i$ for $i = 1, 2, \cdots n+1$ and we then have $\Pi_{2n+1}(x) = \Pi_n^2(x)$ with the corresponding error expression,

   $$\begin{aligned} E(f) &= f[x_0, x_1, \cdots x_{2n+1}, \xi] \int_a^b \Pi_n^2(x)dx, \\ &= \frac{1}{(2n+2)!} f^{(2n+2)}(\eta)s_{n+1}, \end{aligned}$$

   where $s_{n+1} = \int_a^b \Pi_n^2(x)dx$.
   Note that such rules will be exact for all polynomials of degree at most $2n+1$. That is, if the integrand is a polynomial of degree less than $2n+2$ the corresponding Gaussian Quadrature interpolatory rule (based on the n carefully chosen points) will give the exact answer.

(c) How do we choose the x_i's to ensure that $\int_a^b \Pi_n(x)q(x)dx = 0$ for all polynomials, $q(x)$ of degree at most n ?

This question leads to the study of orthogonal polynomials.

- Definition: The set of polynomials $\{r_0(x), r_1(x), \cdots r_k(x)\}$ is orthogonal on $[-1,1]$ iff the following two conditions are satisfied:
  i. $\int_{-1}^1 r_i(x)r_j(x)dx = 0,$ for $i \neq j$,
  ii. The degree of $r_i(x)$ is i for $i = 0, 1, \cdots k$.
- Properties of orthogonal polynomials:
  i. Any polynomial $q_s(x)$ of degree $s \leq k$ can be expressed as.

$$q_s(x) = \sum_{j=0}^{s} c_j r_j(x).$$

  ii. $r_k(x)$ is orthogonal to all polynomials of degree less than k. (This follows from the previous property.)

  iii. $r_k(x)$ has k simple zeros all in the interval $[-1,1]$.
  Proof:
  For $r_k(x)$, let $\{\mu_1, \mu_2, \cdots \mu_m\}$ be the set of points in $[-1,1]$ where $r_k(x)$ changes sign. It is clear that each μ_j is a zero of $r_k(x)$ and all simple zeros of $r_k(x)$ in $[-1,1]$ must be in this set.
  We then have $m \leq k$ as the maximum number of zeros of a polynomial of degree k is k. To show that $m \geq k$ (and hence $m = k$) assume the contrary, ie. $m < k$. With this assumption we have that

$$\hat{q}_m(x) \equiv \prod_{i=1}^{m}(x - \mu_i),$$

  is a polynomial of degree $m < k$ that changes sign at each μ_i and,

$$\int_{-1}^{1} \hat{q}_m(x) r_k(x) dx = 0,$$

  but $\hat{q}_m(x)$ and $r_k(x)$ have the same sign for all x in $[-1,1]$ (they change sign at the same locations) and this implies a contradiction (the integrand is of one sign but the integral is zero) and therefore our assumption must be false and $m \geq k$.

  iv. The $r_k(x)$ satisfy a 3-term recurrence,

$$r_{s+1}(x) = a_s(x - b_s)r_s(x) - c_s r_{s-1}(x),$$

  for $s = 1, 2, \cdots k$, where the a_s are normalization constants, $r_{-1}(x) = 0$, and if $t_s = \int_{-1}^1 r_s^2(x)dx$ then,

$$\begin{aligned} b_s &= \frac{1}{t_s}\int_{-1}^{1} x r_s^2(x)dx, \\ c_s &= \frac{a_s t_s}{a_{s-1}t_{s-1}}. \end{aligned}$$

For example, we obtain the classical Legendre polynomials if we normalise so $r_s(-1) = 1$. This leads to,

$$a_s = \frac{2s+1}{s+1}, \quad b_s = 0, \quad c_s = \frac{s}{s+1}.$$

- Orthogonal Polynomials on arbitrary intervals $[a, b]$.
  To transform orthogonal polynomials defined on $[-1, 1]$ to $[a, b]$ consider the linear mapping from $[-1, 1] \to [a, b]$ defined by $x = \frac{b-a}{2}y + \frac{a+b}{2}$. The corresponding inverse mapping is $y = \frac{1}{b-a}[2x - b - a]$ and from calculus we know,

$$\int_a^b g(x)dx = (\frac{b-a}{2}) \int_{-1}^1 g(\frac{b-a}{2}y + \frac{a+b}{2})dy.$$

This relationship, combined with the properties of Legendre polynomials (that we have identified above) give a prescription for the selection of the set of interpolation points (the x_i's) that define Gaussian Quadrature: For $i = 0, 1, \cdots n$, set y_i to the i^{th} zero of the Legendre Polynomial, $r_{n+1}(y)$. With this choice we note that $\prod_{j=0}^n (y - y_j) = K\ r_{n+1}(y)$ for some constant $K \neq 0$. Then with $x_i = \frac{b-a}{2}y_i + \frac{b+a}{2}$ we have,

$$\begin{aligned}
\Pi_n(\frac{b-a}{2}y + \frac{a+b}{2}) &= \prod_{j=0}^n (\frac{b-a}{2}y + \frac{a+b}{2} - x_j), \\
&= \prod_{j=0}^n (\frac{b-a}{2}y + \frac{a+b}{2} - (\frac{b-a}{2}y_j + \frac{a+b}{2})), \\
&= \prod_{j=0}^n \left[\frac{b-a}{2}(y - y_j)\right], \\
&= (\frac{b-a}{2})^{n+1} \prod_{j=0}^n (y - y_j), \\
&= (\frac{b-a}{2})^{n+1} K\ r_{n+1}(y),
\end{aligned}$$

and therefore for any polynomial, $q(x)$ of degree at most n,

$$\begin{aligned}
\int_a^b \Pi_n(x)q(x)dx &= (\frac{b-a}{2}) \int_{-1}^1 \Pi_n(\frac{b-a}{2}y + \frac{b+a}{2})q(\frac{b-a}{2}y + \frac{b+a}{2})dy, \\
&= (\frac{b-a}{2}) \int_{-1}^1 \Pi_n(\frac{b-a}{2}y + \frac{b+a}{2})\hat{q}(y)dy,
\end{aligned}$$

(where $\hat{q}(y)$ is a polynomial of degree at most n since the degree of $q(x)$ is at most n)

$$\begin{aligned}
&= (\frac{b-a}{2})^{n+2} K \int_{-1}^1 r_{n+1}(y)\hat{q}(y)dy, \\
&= 0.
\end{aligned}$$

That is with the x_i's chosen as the 'transformed zeros' of the Legendre polynomial, $r_{n+1}(y)$, we have the interpolation points satisfying our desired property.

6. **Composite Quadrature Rules:**

Approximating the integrand with a piecewise polynomial leads to the class of <u>Composite Rules</u>. Let $a = x_0 < x_1 < \cdots x_M = b$ and $S(x)$ be a piecewise polynomial approximation to $f(x)$ for $x \in [a,b]$. We can then use $\int_a^b S(x)dx$ as the approximation to $I(f) = \int_a^b f(x)dx$. Recall that $S(x) \equiv p_{i,n}(x)$ for $x \in [x_{i-1}, x_i]$ $i = 1, 2, \cdots M$. From calculus we have,

$$\begin{aligned} \int_a^b S(x)dx &= \sum_{i=1}^{M} \int_{x_{i-1}}^{x_i} S(x)dx, \\ &= \sum_{i=1}^{M} \int_{x_{i-1}}^{x_i} p_{i,n}(x)dx, \end{aligned}$$

–A <u>sum</u> of basic interpolatory rules.

If we use equally spaced x_i's and low degree interpolation we obtain familiar rules:

- The composite trapezoidal rule, $T_M(f)$:

$$\begin{aligned} \int_{x_{i-1}}^{x_i} f(x)dx &= \frac{x_i - x_{i-1}}{2}\left[f(x_{i-1}) + f(x_i)\right] - \frac{f''(\eta_i)(x_i - x_{i-1})^3}{12}, \\ &= \frac{h}{2}\left[f_{i-1} + f_i\right] - \frac{f''(\eta_i)h^3}{12}. \end{aligned}$$

Summing over all sub-intervals we obtain,

$$T_M(f) \equiv h \sum_{i=1}^{M-1} f_i + \frac{h}{2}(f_0 + f_M),$$

with the corresponding error expression,

$$\begin{aligned} E_M^T(f) &\equiv I(f) - T_M(f), \\ &= -\sum_{i=1}^{M} \frac{h^3}{12} f''(\eta_i). \end{aligned}$$

If $f''(x)$ is continuous we can apply the MVT for sums to obtain the expression,

$$\begin{aligned} E_M^T(f) &= -f''(\eta) \sum_{i=1}^{M} \frac{h^3}{12} \quad \text{for some } \eta \in (a,b), \\ &= -f''(\eta)\, M \frac{h^3}{12}, \quad (\text{but } h = (b-a)/M\ ) \\ &= -f''(\eta)(b-a)\frac{h^2}{12}. \end{aligned}$$

We therefore have:

$$\boxed{T_M(f) = h\sum_{i=1}^{M-1} f_i + h/2(f_0 + f_M)}$$

with

$$\boxed{E_M^T = -f''(\eta)(b-a)\tfrac{1}{3}(\tfrac{h}{2})^2}.$$

- The composite Simpsons rule, $S_M(f)$:
  Similarly we can derive the Composite Simpsons Rule:

$$\boxed{S_M(f) = h/6\left[f_0 + f_M + 2\sum_{i=1}^{M-1} f_i + 4\sum_{i=1}^{M} f_{i-1/2}\right]}$$

with the corresponding error expression,

$$\boxed{E_M^S(f) = \tfrac{-f^{(4)}(\eta)}{180}(b-a)(\tfrac{h}{2})^4}$$

7. **Error estimates for composite rules:**

- Trapezoidal rule: Consider the contribution to the overall error that comes from the i^{th} sub-interval,

$$\begin{aligned} E^{(i)} &\equiv I^{(i)} - T_1^{(i)}, \\ &= \int_{x_{i-1}}^{x_i} f(x)dx - \frac{h}{2}(f_{i-1} + f_i), \\ &= -(\frac{1}{12})h_i^3 f''(\xi_i). \end{aligned}$$

Subdividing $[x_{i-1}, x_i]$ into two subintervals leads to,

$$\int_{x_{i-1}}^{x_{i-1/2}} f(x)dx \approx \frac{h_i}{4}(f_{i-1} + f_{i-1/2}) \text{ with error } = \frac{-1}{12}(\frac{h_i}{2})^3 f''(\bar{\xi}_i),$$

$$\int_{x_{i-1/2}}^{x_i} f(x)dx \approx \frac{h_i}{4}(f_{i-1/2} + f_i) \text{ with error } = \frac{-1}{12}(\frac{h_i}{2})^3 f''(\hat{\xi}_i).$$

Summing these two terms we obtain,

$$I^{(i)}(f) \approx \frac{h_i}{4}(f_{i-1} + 2f_{i-1/2} + f_i) \equiv T_2^{(i)}(f),$$

with an associated error expression,

$$I^{(i)} - T_2^{(i)} = \frac{-1}{12}(\frac{h_i}{2})^3\left[f''(\bar{\xi}_i) + f''(\hat{\xi}_i)\right] = \frac{-h_i^3}{48}f''(\tilde{\xi}_i).$$

Now if we assume that h is 'small enough' so that f'' doesn't change much on the i^{th} subinterval (ie, $f''(\bar{\xi}_i) \approx f''(\hat{\xi}_i) \approx f''(\tilde{\xi}_i)$) then subtracting these two error expressions we obtain,

$$(I^{(i)} - T_1^{(i)}) - (I^{(i)} - T_2^{(i)}) = T_2^{(i)} - T_1^{(i)} \approx \frac{-h_i^3}{12}f''(\xi_i)\left[1 - 1/4\right] = \frac{-3}{48}h_i^3 f''(\xi_i).$$

We can then estimate the error associated with the approximation, $T_2^{(i)}$ as $1/3\left[T_2^{(i)} - T_1^{(i)}\right]$ and, after summing over all M subintervals, we obtain

$$\begin{aligned}
EST_{2M} &\equiv \sum_{i=1}^{M} \frac{1}{3}(T_2^{(i)} - T_1^{(i)}), \\
&= \frac{1}{3}\left[\sum_{i=1}^{M} T_2^{(i)} - \sum_{i=1}^{M} T_1^{(i)}\right], \\
&= \frac{1}{3}\left[T_{2M} - T_M\right].
\end{aligned}$$

Furthermore by applying the MVT for sums, it can easily be shown that EST_{2M} also equals $\frac{-f''(\xi)}{24}(\frac{h}{2})^2(b-a)$ for some $\xi \in (a,b)$.

Note:

(a) This estimate is only justified for h_i sufficiently small so that f'' is almost constant over each subinterval.

(b) The computation of EST_{2M} can be subject to large relative error as it involves the subtraction of 'near equals'.

(c) A validity check is available based on the monitoring the ratio $|EST_M/EST_{2M}|$ which should be close to 4.

- A similar analysis for Simpsons rule yields,

$$\begin{aligned}
EST_{2M}^{S} &= \frac{1}{15}\left[S_{2M} - S_M\right], \\
&= \frac{f^{(4)}(\eta)}{180}(\frac{h}{4})^4(b-a).
\end{aligned}$$

where the corresponding validity check is that $|EST_M^S/EST_{2M}^S| \approx 16$.

- Exercise: Show that $T_{2M} + EST_{2M} = S_M$.

8. **An Example:** Consider applying the Composite Trapezoidal and Composite Simpsons rules in single and double precision floating point arithmetic ($\beta = 16$ and $s = 5,\ s = 12$, respectively) to approximate $\int_0^1 e^{-x^2}dx$. The numerical results are presented in Table 1 and they clearly indicate the ability of the validity check to reflect when the error estimate can be trusted. Note that we have only justified it in the limit as $h \to 0$ and in the situation where truncation error dominates round-off error.

9. **Extrapolation – Romberg Quadrature:**

- Recall, for the Trapezoidal rule we have established,

$$\begin{aligned}
T_{2M} + EST_{2M} &= \sum_{i=1}^{M} T_2^{(i)} + \frac{1}{3}(T_2^{(i)} - T_1^{(i)}), \\
&= \sum_{i=1}^{M} \frac{4}{3}T_2^{(i)} - \frac{1}{3}T_1^{(i)},
\end{aligned}$$

M	Approximation	EST	True Err	Ratio
I) Composite Trapezoidal, T_M – single precision				
2	0.7313699000000	-.158E-01	-.155E-01	4.07
4	0.7429835000000	-.387E-02	-.384E-02	4.02
8	0.7458651000000	-.961E-03	-.959E-03	4.00
16	0.7465841000000	-.240E-03	-.240E-03	4.00
32	0.7467608000000	-.589E-04	-.632E-04	3.79
64	0.7468032000000	-.141E-04	-.208E-04	3.04
128	0.7468135000000	-.342E-05	-.106E-04	1.97
256	0.7468156000000	-.715E-06	-.840E-05	1.26
512	0.7467660000000	.165E-04	-.580E-04	0.15
II) Composite Trapezoidal, T_M – double precision				
2	0.7313702518257	-.158E-01	-.155E-01	4.07
4	0.7429840977975	-.387E-02	-.384E-02	4.02
8	0.7458656148428	-.961E-03	-.959E-03	4.01
16	0.7465845967854	-.240E-03	-.240E-03	4.00
32	0.7467642546494	-.599E-04	-.599E-04	4.00
64	0.7468091636350	-.150E-04	-.150E-04	4.00
128	0.7468203905388	-.374E-05	-.374E-05	4.00
256	0.7468231972433	-.936E-06	-.936E-06	4.00
512	0.7468238989181	-.234E-06	-.234E-06	4.00
III) Composite Simpson, S_M – single precision				
2	0.7468550000000	.217E-04	.310E-04	11.5
4	0.7468255000000	.197E-05	.143E-05	21.7
8	0.7468219000000	.234E-06	-.209E-05	.7
16	0.7468218000000	.795E-08	-.221E-05	.9
32	0.7468217000000	.119E-07	-.238E-05	.9
64	0.7468134000000	.552E-06	-.107E-04	.2
128	0.7467956000000	.118E-05	-.284E-04	.4
256	0.7467887000000	.457E-06	-.353E-04	.8
512	0.7467867000000	.139E-06	-.374E-04	.9
IV) Composite Simpson, S_M – double precision				
2	0.7468553797881	.217E-04	.312E-04	11.4
4	0.7468261205246	.195E-05	.199E-05	15.7
8	0.7468242574329	.124E-06	.125E-06	15.9
16	0.7468241406041	.779E-08	.779E-08	16.0
32	0.7468241332968	.487E-09	.487E-09	16.0
64	0.7468241328400	.305E-10	.305E-10	16.0
128	0.7468241328115	.190E-11	.196E-11	15.6
256	0.7468241328097	.119E-12	.173E-12	11.3
512	0.7468241328096	.748E-14	.611E-13	2.8

Table 1: Numerical results for Composite rules applied to approximate $\int_0^1 e^{-x^2} dx$

$$\begin{aligned}
&= \sum_{i=1}^{M} \left\{ \frac{h}{3}(f_{i-1} + 2f_{i-1/2} + f_i) - \frac{h}{6}(f_{i-1} + f_i) \right\}, \\
&= \sum_{i=1}^{M} \frac{h}{6} \left\{ f_{i-1} + 4f_{i-1/2} + f_i \right\}, \\
&= S_M, \\
&= I(f) + O(h^4),
\end{aligned}$$

(A fourth order approximation to $I(f)$).

- This process of taking a basic quadrature rule, applying it with a sequence of 'stepsizes' $h, h/2, h/4, \cdots h/2^k$ and then using a linear combination of the resulting approximations, $A_0, A_1, \cdots A_k$ to obtain a <u>higher order</u> approximation is called <u>extrapolation</u>.

  When the Trapezoidal rule is used as the basic rule this is called Romberg quadrature (or Romberg Integration).

- To justify extrapolation for the Trapezoidal rule we must show that whenever $f(x)$ has $(2k+2)$ continuous derivatives, then the true error satisfies,

$$E_M^T = c_1 h^2 + c_2 h^4 + \cdots c_k h^{2k} + O(h^{2k+2}),$$

  where the c_i's are independent of h.

  (a) One can then 'eliminate' the h^2 term in the error by taking a linear combination of T_M and T_{2M} (two different approximations to $I(f)$). Let h be the interval width associated with T_{2M}. We then have,

$$\begin{aligned}
T_{2M} &= I(f) + c_1 h^2 + c_2 h^4 + \cdots c_k h^{2k} + O(h^{2k+2}), \\
T_M &= I(f) + c_1 (2h)^2 + c_2 (2h)^4 + \cdots c_k (2h)^{2k} + O((2h)^{2k+2}).
\end{aligned}$$

  Defining T_{2M}^1 by,

$$\begin{aligned}
T_{2M}^1 &\equiv \frac{4T_{2M} - T_M}{3}, \\
&= T_{2M} + \frac{1}{3}(T_{2M} - T_M), \\
&= I(f) + c_2^1 h^4 + c_3^1 h^6 + \cdots c_k^1 h^{2k} + O(h^{2k+2}),
\end{aligned}$$

  we have derived an error expansion for this fourth order approximation (which is Simpson's rule).
  Similarly, by considering the resulting error expressions for T_{2M}^1 and T_{4M}^1 we can 'eliminate' the $O(h^4)$ term to obtain,

$$\begin{aligned}
T_{4M}^2 &\equiv \frac{16T_{4M}^1 - T_{2M}^1}{15}, \\
&= T_{4M}^1 + \left(\frac{T_{4M}^1 - T_{2M}^1}{15}\right), \\
&= S_{2M} + EST_{2M}^S, \\
&= I(f) + c_3^2 h^6 + c_4^2 h^8 + \cdots c_k^2 h^{2k} + O(h^{2k+2}).
\end{aligned}$$

(b) This process can continue and we have, in general,

$$T^m_{2^m M} \equiv T^{m-1}_{2^m M} + \frac{T^{m-1}_{2^m M} - T^{m-1}_{2^{m-1} M}}{4^m - 1},$$

where we have the following expansion of the error,

$$T_{2^m M} = I(f) + c^m_{m+1} h^{2(m+1)} + c^m_{m+2} h^{2(m+2)} + \cdots c^m_k h^{2k} + O(h^{2k+2}).$$

(c) This technique gives high order approximations but round-off limits the accuracy that can be achieved. In practice we usually have $m \leq 6$ or 7. Note that we can obtain more accuracy by increasing m or M since the error term associated with $T^m_{2^m M}$ is $O(h^{2(m+1)})$ which is $O((\frac{b-a}{M})^{2(m+1)})$. The 'cost' of computing this approximation is $2^m M$ evaluations of the integrand.

10. **Error estimates for Gaussian Quadrature Rules:**

- Let $G_n(f) = \sum_{i=0}^n \omega_i f(x_i)$ denote the $(n+1)$ – point Gaussian quadrature rule.

  (a) We have shown,

  $$I(f) - G_n(f) = O(b-a)^{2n+3}, \text{ as } (b-a) \to 0,$$

  and this is optimal.

  (b) The rules $G_{n+1}, G_{n+2}, \cdots$, are higher order and therefore asymptotically more accurate (as $(b-a) \to 0$) so we could form an error estimate from one of these. That is, we could use,

  $$\widehat{EST}_{G_n} \equiv G_{n+k}(f) - G_n(f) = E_{G_n} + O(b-a)^{2(n+k)+3}.$$

  (c) The rules G_{n+k} and G_n have at most one common interpolation point so the computation of this error estimate more than <u>doubles</u> the cost ($2n + k + 2$ integrand evaluations).

- An alternative (to forming an error estimate based on G_{n+k}) is is to use the integrand evaluations already available (for the computation of $G_n(f)$) and introduce only the minimum number of extra evaluations required to obtain an effective error estimate. This approach leads to a class of quadrature rules called Kronrod quadrature rules, $K_{n+k}(f)$. The error estimate for $G_n(f)$, is then $K_{n+k}(f) - G_n(f)$, where $K_{n+k}(f)$ is more accurate <u>and</u> less expensive to compute than is $G_{n+k}(f)$. Kronrod proposed a particularly effective class of such rules where $k = n + 1$,

$$K_{2n+1}(f) \equiv \sum_{i=0}^n a_i f(x_i) + \sum_{j=0}^{n+1} b_j f(y_j),$$

where the $x_i's$ are the interpolation points associated with $G_n(f)$, and the y_i's are the extra interpolation points necessary to define an accurate approximation to $I(f)$. Kronrod derived these weights (the a_i's and the b_i's) and the extra interpolation points $(y_0, y_1, \cdots y_n)$ so that the resulting rule is order $3n+3$. The resulting error estimate is then,

$$EST_{G_n} \equiv K_{2n+1}(f) - G_n(f),$$

with an associated cost of $2n+3$ integrand evaluations and an order of accuracy of $O((b-a)^{3n+4})$.

- The resulting Gauss-Kronrod pairs of rules can be the basis for composite quadrature rules and adaptive methods. These methods are widely used and implemented in numerical libraries.

11. **Adaptive Quadrature Rules:**

- A straightforward implementation of a numerical quadrature method based on a basic quadrature rule with error estimate would accept, as input parameters:
  (a) The integrand function, $f(x)$.
  (b) The upper and lower limits of integration, a and b.
  (c) The desired accuracy, TOL, where the method would attempt to provide an approximation, A, that satisfies $|I(f) - A| < TOL$.

  and proceed, as in the case of the composite rules (introduced earlier). That is after applying the basic rule, if the magnitude of the associated error estimate exceeds TOL, the interval $[a, b]$ is subdivided (by interval halving) and the basic rule applied to each sub-interval (with an associated overall error estimate obtained by summing the magnitudes of the estimates from each subinterval). This process of interval halving and updating the approximation to $I(f)$ and the associated error estimate $EST(f)$ continues until $|EST(f)| < TOL$ with some failure condition possible if no convergence is achieved after a reasonable amount of effort (for example after $8-10$ subdivisions).

  Such an implementation will work well for functions that are smooth and relatively well behaved over the interval of integration. On the other hand, such a method can be inefficient if the integrand is badly behaved (rapidly varying for example) on only a small part of the interval of integration. In such cases it would be more effective to concentrate the effort (the integrand evaluations) in the neighborhood where the integrand is changing rapidly. This is the key idea behind 'adaptive' quadrature methods.

- In adaptive quadrature methods, a basic rule with an associated error estimate is implemented in a similar fashion to the straightforward implementation discussed above except that uniform interval halving is not used when more accuracy is needed. Rather than doubling the total number of integrand evaluations on each step of the method (as is the case with uniform interval halving) we selectively refine or subdivide only those subintervals whose approximations have an associated error estimate that is too large. That is, on each step we use interval halving to update the approximations and corresponding overall error estimate for only a subset of the subintervals. At each step of the method we maintain a partitioning of the interval $a = X_0 < X_1 < \cdots X_N = b$ and we have the associated approximation, $A(f)$ and associated error estimate, $EST(f)$,

$$\begin{aligned} A(f) &\equiv \sum_{r=1}^{N} A_r, \\ EST(f) &\equiv \sum_{r=1}^{N} |EST_r|, \end{aligned}$$

where A_r and EST_r are the approximation and error estimate associated with applying the basic rule to the r^{th} subinterval.

The effectiveness of this approach depends largely on how one decides which interval to subdivide next (when $|EST(f)|$ exceeds TOL).

- Several possible refinement strategies (strategies for choosing which subinterval to halve) are possible. We will consider two alternatives. Note that in each case one must choose a suitable data structure carefully to ensure that the potential advantages of the strategy can be realized.

  (a) Equal distribution of the error to each subinterval:
  Each interval is allowed to contribute an amount to the total error that is proportional to its width. The maximum allowable error on the i^{th} subinterval is then $\frac{x_i - x_{i-1}}{b-a} TOL$ and this will guarantee,

$$\begin{aligned} |EST(f)| &\equiv \sum_{i=1}^{N} |EST_i|, \\ &\leq \sum_{i=1}^{N} (\frac{x_i - x_{i-1}}{b-a}) TOL, \\ &= (\frac{TOL}{b-a}) \sum_{i=1}^{N} (x_i - x_{i-1}), \\ &= TOL. \end{aligned}$$

  Note that this strategy can be effectively implemented recursively or using stacks.

  (b) Refine where error contribution is largest:
  On each step subdivide only the subinterval with the estimate of largest magnitude. Such a strategy can be effectively implemented using an ordered linked data structure, where the ordering is determined by the magnitude of the corresponding estimate, EST_r.

12. **Special Difficulties:**

One can often introduce a mathematical transformation of a difficult quadrature problem to ‘reduce’ it to an equivalent ‘standard problem’.

- Infinite range problems (improper integrals):

  The infinite integration, $\int_0^\infty f(x)dx$ or $\int_{-\infty}^\infty f(x)dx$ is well-defined only if $\lim_{R\to\infty} \int_0^R f(x)dx$ exists.

  Some possible approaches for approximating $I(f) = \int_0^\infty f(x)dx$ when it exists:

  (a) For $0 = R_0 < R_1 < \cdots R_J < \cdots$, define A_i as the approximation to $\int_{R_{i-1}}^{R_i} f(x)dx$ associated with a standard quadrature rule. We then have that,
  $$S_j \equiv \sum_{i=0}^{j} A_i,$$
  can be used as an approximation to $I(f)$ if $R_i \to \infty$. This process can halt (at a fixed value of j) when $|A_j| < TOL$.

  (b) If $f(x) = \omega(x)g(x)$ with $\omega(x)$ positive, then one can apply a generalized Gaussian rule. For example, if $\omega(x) = e^{-x}$ we obtain Gauss Laquerre rules. In this case we have,
  $$\int_0^\infty \omega(x)g(x)dx \approx \sum_{i=0}^{n} \omega_i g(x_i),$$
  where the x_i's are the zeros of polynomials orthogonal on $[0,\infty)$ with respect to $\omega(x)$.

  (c) Special transformation of variable. Let $x = \rho(t)$ for some differentiable function $\rho(t)$. We then have,
  $$I(f) = \int_0^\infty f(x)dx = \int_{\rho^{-1}(0)}^{\rho^{-1}(\infty)} f(\rho(t))\rho^{'}(t)dt.$$
  For example, if $x = -\ln(t)$, $\Rightarrow t = e^{-x}$ and we have
  $$I(f) = \int_0^1 f(-\ln(t))/tdt.$$

- Singularities of the integrand: Consider the approximation of $I(f) = \int_a^b f(x)dx$ where $f(a)$ or $f(b)$ is undefined. For example,
  $$I = \int_0^1 \frac{1}{x^{1/2} + x^{1/3}} dx.$$
  For such problems we can attempt to 'remove' the singularity by the following procedure,

  (a) Determine the 'type' of the singularity at $t = t^*$, choose $s(x)$ where $\int_a^b s(x)dx$ can be computed $\underline{\text{analytically}}$ and where $(f(x) - s(x))$ is not singular at t^*.

  (b) Replace $\int f(x)dx$ by $\int (f(x)-s(x))dx + \int s(x)dx$ where standard methods can be used to approximate the first integral and the analytic formula used for the second.

For the above example, with $f(x) = \frac{1}{x^{1/2}+x^{1/3}}$ consider what happens as $x \to 0$,

$$\begin{aligned}\frac{1}{x^{1/2}+x^{1/3}} &= \frac{1}{x^{1/3}(x^{1/6}+1)},\\ &= \frac{1}{1+x^{1/6}} - \frac{x^{1/6}-1}{x^{1/3}}.\end{aligned}$$

We therefore have,

$$\int_0^1 \frac{1}{x^{1/2}+x^{1/3}}dx = \int_0^1 \frac{1}{1+x^{1/6}}dx + \int_0^1 \frac{1-x^{1/6}}{x^{1/3}}dx.$$

The first integral on the right hand side can then be approximated by standard methods while the second is equal to 3/10.

For the general case the key step is to perform an expansion of the integrand about the point of singularity ($t^* = a$ or $t^* = b$) to allow one to 'remove' it.

13. **Two Dimensional Quadrature:**

Consider the problem of approximating integrals in two dimensions,

$$I(f) = \int\int_D f(x,y)dxdy,$$

This problem is more complicated than the one dimensional case since D can take many forms.

- One can develop the analogs of Gaussian rules or interpolatory rules but the weights and nodes will depend on the region D. Such rules can be determined and tabulated for simple regions such as rectangles, triangles and circles. An arbitrary region must then be transformed onto one of these simple regions before the rule can be used. Such a transformation will generally be nonlinear and may introduce an approximation error as well.
- One can apply a 'product rule' where one reduces the $2D$-integral to a sequence of two $1D$-integrals:

$$\int_a^b \int_{\alpha(y)}^{\beta(y)} f(x,y)dxdy = \int_a^b g(y)dy,$$

where

$$g(y) \equiv \int_{\alpha(y)}^{\beta(y)} f(x,y)dx$$

is approximated, for a fixed value of y, by a standard method (for example, $\approx \sum_{j=0}^{M} \omega_j f(x_j,y)$, and $\int_a^b g(y)dy$ is also approximated by a standard (possibly different) standard method. That is

$$\int_a^b g(y)dy \approx \sum_{r=0}^{M'} \hat{\omega}_r g(y_r),$$

$$\begin{aligned} &\approx \sum_{r=0}^{M'} \hat{\omega}_r \left(\sum_{j=0}^{M} \omega_j f(x_j, y_r) \right), \\ &= \sum_{r=0}^{M'} \sum_{j=0}^{M} (\hat{\omega}_r \omega_j) f(x_j, y_r). \end{aligned}$$

Note that error estimates for product rules are not easy to develop since the function $g(y) \approx \sum_{j=0}^{M} \omega_j f(x_j, y)$ will not be a 'smooth' function of y unless M and the x_j's are fixed. In particular this 'inner rule' cannot be adaptive.

5 Numerical ODEs

1. **Mathematical Preliminaries:**

- Definition: A first-order ordinary differential equation is specified by:

$$y' = f(x, y),$$

over a finite interval $[a, b]$.

Note that a solution of this ODE, $y(x)$, is a function of one variable (this is the reason for the term 'ordinary' as opposed to 'partial'). When the solution depends on more than one variable (ie a multivariate function) it is called a partial differential equation – PDE). The term first-order refers to the highest derivative that appears in the equation. We will consider higher-order equations later. For ODEs the variable x is called the independent variable while y (which depends on x) is called the dependent variable. 'Solving' the ODE is interpreted as determining a technique for expressing y as a function of x in some explicit way.

- A function $\Phi(x)$ is a solution of this ODE if $\Phi(x) \in C^1[a, b]$ and $\forall x \in [a, b]$ we have $\Phi^{'}(x) = f(x, \Phi(x))$. (Note that this condition is often easy to check or verify).

–An Example:
$y' = \lambda y$, has solutions $\Phi(x) = c\ e^{\lambda x}$ for any constant c. In particular this ODE does not have a unique solution but rather a whole family of solutions (characterized by the parameter c).

- To determine a unique mathematical solution we must add an additional constraint as part of the problem specification. This can be done in many ways. The most common is to prescribe the value of the solution at the initial point of the interval. That is we specify,

$$y(a) = y_0.$$

–Definition: An ODE together with the initial conditions specifies an initial value problem for an ordinary differential equation (IVP for an ODE).

- Before we can attempt to approximate a solution to an IVP we must consider some essential mathematical questions:
  (a) Does a solution exist?
  (b) If a solution exists, is it unique?
  (c) Can the problem be solved analytically? (If so, is it worth it?)

- Definition: The function $f(x, y)$ (a function of two variables that defines the ODE) satisfies a Lipschitz condition in y (ie, wrt its second argument) if $\exists L > 0$ such that $\forall x \in [a, b]$ and $\forall\, u, v$ we have

$$|f(x, u) - f(x, v)| \leq L|u - v|.$$

  In particular, if $f(x, y)$ has a continuous partial derivative with respect to y and this derivative is bounded for all y, then f satisfies a Lipschitz condition in y since,

$$|f(x, u) - f(x, v)| = |\frac{\partial f}{\partial y}(x, \eta)|\,|u - v|,$$

  for some η between u and v.

- Theorem:
  Let $f(x, y)$ be continuous for $x \in [a, b]$ and $\forall y$ and satisfy a Lipschitz condition in y, then for any initial condition y_0 the IVP,

$$y' = f(x, y), \;\; y(a) = y_0, \;\; \text{over } [a, b],$$

  has a unique solution, $y(x)$ defined for all $x \in [a, b]$.

- Extension to systems of equations:
  Often one must deal with a system of n 'unknown' dependent variables of the form:

$$\begin{aligned} y_1' &= f_1(x, y_1, y_2, \cdots y_n), \\ y_2' &= f_2(x, y_1, y_2, \cdots y_n), \\ \vdots \;&\vdots\; \vdots \\ y_n' &= f_n(x, y_1, y_2, \cdots y_n), \end{aligned}$$

  with initial conditions all specified at the same point,

$$\begin{aligned} y_1(a) &= c_1, \\ y_2(a) &= c_2, \\ \vdots \;&\vdots\; \vdots \\ y_n(a) &= c_n, \end{aligned}$$

In vector notation, this system of IVPs in ODEs can be written

$$Y' = F(x, Y), \;\; Y(a) = Y_0,$$

where $Y(x) = [y_1(x), y_2(x), \cdots y_n(x)]^T$, $Y_0 = [c_1, c_2, \cdots c_n]^T$ and $F(x, Y)$ is a vector-valued function,

$$F(x, Y) = \begin{bmatrix} f_1(x, Y) \\ f_2(x, Y) \\ \vdots \\ f_n(x, Y) \end{bmatrix}.$$

The theory and the investigation of numerical methods that we present will be the same for systems as for scalar IVPs. In particular, the main mathematical Theorem quoted above holds for systems.

–Examples of systems:

(a) From Biology:
A predator-prey relationship can be modeled by the IVP:

$$y_1^{'} = y_1 - 0.1 y_1 y_2 + 0.02x$$

$$y_2^{'} = -y_2 + 0.02 y_1 y_2 + 0.008x$$

with

$$y_1(0) = 30, \;\; y_2(0) = 20.$$

Here $y_1(x)$ represents the 'prey' population at time x and $y_2(x)$ represents the 'predator' population at time x. The solution can then be visualized as a standard x/y solution plot or by a 'phase plane' plot. Figure 1 illustrates the solution to this system. We know that for different initial conditions solutions to this problem exhibit oscillatory behaviour as x increases.
A biologist may be interested in whether the solutions to this equation are 'almost periodic' (in the sense that the difference between successive maximum is constant) and whether the local maxima approach a steady state exponentially. (See Figure 2).

(b) From Chemistry:
The chemical reaction involving the combination of two chemicals C_1 and C_2, to yield a product C_3 is represented (in chemistry) by the mechanism (or notation)

$$C_1 + C_2 \;\; \underset{K_1}{\overset{K_2}{\rightleftharpoons}} \;\; C_3 \; .$$

(a) Solution Plot (b) Phase Plane Plot

Figure 1: Solutions to the predator prey problem for x in [0, 20]

Figure 2: Typical behaviour of prey or predator population and decay to steady state

We can model this chemical reaction with the system of 3 ODEs, where $y_1(x) = [C_1]$ the concentration of the chemical C_1 at time x, $y_2(x) = [C_2]$ and $y_3(x) = [C_3]$. The resulting system of IVPs whose solution for $x \in [a, b]$ describes the change in concentrations over time as the reaction takes place (possibly approaching a steady state) is,

$$\begin{aligned} y_1' &= K_1 y_3 - K_2 y_1 y_2, \\ y_2' &= K_1 y_3 - K_2 y_1 y_2, \\ y_3' &= K_2 y_1 y_2 - K_1 y_3. \end{aligned}$$

- Extension to Second (and higher) order ODEs:

  Often physical or biological systems are best described by second or higher-order ODEs. That is, second or higher order derivatives appear in the mathematical model of the system. For example, from physics we know that Newtons laws of motion describe trajectory or gravitational problems in terms of relationships between velocities, accelerations and positions. These can often be described as IVPs where the ODE has the form $y''(x) = f(x, y)$ or $y''(x) = f(x, y, y')$.

  (a) A Second-order ODE can be reduced to an <u>equivalent</u> system of first-order ODEs as follows: With $y'' = f(x, y, y')$ we let $Z(x)$ be defined by,
  $$Z(x) = [z_1(x), z_2(x)]^T,$$
  where $z_1(x) = y(x)$ and $z_2(x) = y'(x)$. It is then clear that $Z(x)$ is the solution of the first order system of IVPs:
  $$\begin{aligned} Z' &= \begin{bmatrix} z_1'(x) \\ z_2'(x) \end{bmatrix}, \\ &= \begin{bmatrix} y'(x) \\ y''(x) \end{bmatrix}, \\ &= \begin{bmatrix} z_2(x) \\ f(x, y, y') \end{bmatrix}, \\ &= \begin{bmatrix} z_2(x) \\ f(x, z_1, z_2) \end{bmatrix}, \\ &= F(x, Z). \end{aligned}$$

  - Note that in solving this 'equivalent' system for $Z(x)$, we actually determine an approximation to $y'(x)$ as well as to $y(x)$. This has implications for numerical methods as, when working with this equivalent system, we will actually be trying to accurately approximate $y'(x)$ and this may be a more difficult problem than just approximating $y(x)$.
  - Note also that to determine a unique solution to our problem we must prescribe initial conditions for $Z(a)$, that is for both $y(a)$ and $y'(a)$.
  - Higher order (greater than second order) equations can be reduced to first order systems in a similar way.

2. **Taylor Series Methods**:

- If $f(x, y)$ is sufficiently differentiable wrt x and y then we can determine the Taylor series expansion of the unique solution $y(x)$ to
$$y' = f(x, y), \quad y(a) = y_0,$$

by differentiating the ODE at the point $x_0 = a$. That is, for x near $x_0 = a$ we have,

$$y(x) = y(x_0) + (x - x_0)y'(x_0) + \frac{(x - x_0)^2}{2} y''(x_0) + \cdots,$$

- To generate the TS coefficients, $y^{(n)}(x_0)/n!$, we differentiate the ODE and evaluate at $x = x_0 = a$. The first few terms are computed from the expressions,

$$\begin{aligned} y'(x) = f(x, y) &= f, \\ y''(x) = \frac{d}{dx} f(x, y) &= f_x + f_y y' \\ &= f_x + f_y f. \\ y'''(x) = \frac{d}{dx}[y''(x)] &= (f_{xx} + f_{xy} f) + (f_{yx} + f_{yy} f)f + f_y(f_x + f_y f), \\ &= f_{xx} + 2f_{xy} f + f_{yy} f^2 + f_y f_x + f_y^2 f. \end{aligned}$$

- In general, if $f(x, y)$ is sufficiently differentiable, we can use the first $(k + 1)$ terms of the Taylor series as an approximation to $y(x)$ for $|(x - x_0)|$ 'small'. That is, we can approximate $y(x)$ by $\hat{z}_{k,0}(x)$,

$$\hat{z}_{k,0}(x) \equiv y_0 + (x - x_0)y_0' + \cdots + \frac{(x - x_0)^k}{k!} y_0^{(k)}.$$

  Note that the derivatives of y become quite complicated so one usually chooses a small value of k (for example $k \leq 6$).

- One can use $\hat{z}_{k,0}(x_1)$ as an approximation, y_1, to $y(x_1)$. We can then evaluate the derivatives of $y(x)$ at $x = x_1$ to define a new polynomial $\hat{z}_{k,1}(x)$ as an approximation to $y(x)$ for $|(x - x_1)|$ 'small' and repeat the procedure.

  Note:

  (a) The resulting $\hat{z}_{k,j}(x)$ for $j = 0, 1, \cdots$ define a piecewise polynomial approximation to $y(x)$ that is continuous on $[a, b]$.

  (b) How do we effectively choose $h_j = (x_j - x_{j-1})$ and k?

- Let $T_k(x, y_{j-1})$ denote the first $k + 1$ terms of the Taylor series expanded about the discrete approximation, (x_{j-1}, y_{j-1}), and $\hat{z}_{k,j}(x)$ be the polynomial approximation (to $y(x)$) associated with this truncated Taylor series. That is,

$$\begin{aligned} \hat{z}_{k,j}(x) &= y_{j-1} + h\, T_k(x, y_{j-1}), \\ T_k(x, y_{j-1}) &\equiv f(x_{j-1}, y_{j-1}) + \frac{h}{2} f'(x_{j-1}, y_{j-1}) + \cdots \frac{h^{k-1}}{k!} f^{(k-1)}(x_{j-1}, y_{j-1}), \end{aligned}$$

  where $h = (x - x_{j-1})$.

  A simple, constant stepsize (fixed h) numerical method is then given by:

```
-Set $h = (b-a)/N$;
-for $j = 1, 2, \cdots N$
        $x_j = x_{j-1} + h$;
        $y_j = y_{j-1} + h\ T_k(x_{j-1}, y_{j-1})$;
-end
```

3. **Local and Global Errors:**

   Note that, strictly speaking, $z_{k,j}(x)$ is not a <u>direct</u> approximation to $y(x)$ but to the solution of the 'local' IVP:

$$z_j^{'} = f(x, z_j), \quad z_j(x_{j-1}) = y_{j-1}.$$

   Since y_{j-1} will not be equal to $y(x_{j-1})$ in general, the solution to this local problem, $z_j(x)$, will not then be the same as $y(x)$.

   To understand and appreciate the implications of this observation we distinguish between the 'local' and 'global' errors.

   Definitions:

   - The <u>local error</u> associated with step j is $z_j(x_j) - y_j$.
   - The <u>global error</u> at x_j is $y(x_j) - y_j$.

4. **The Classical Approach:**

   A Classical (pre 1965) numerical method approximates $y(x)$ by dividing $[a, b]$ into equally spaced subintervals, $x_j = a + j\ h$, where $h = (b-a)/N$ and (proceeding in a step-by-step fashion), generates y_j after $y_1, y_2, \cdots y_{j-1}$ have been determined.

   - If the Taylor series method is used in this way, then the TS theorem with remainder shows that the local error on step j (for the TS method of order k) is:

$$E_j = \frac{h^{k+1} f^{(k)}(\eta_j, z_j(\eta_j))}{(k+1))!} = \frac{h^{k+1} z_j^{(k+1)}(\eta_j)}{(k+1)!}.$$

   - An Example:
     If $k = 1$ we have <u>Eulers Method</u> where

$$y_j = y_{j-1} + h\ f(x_{j-1}, y_{j-1}),$$

     and the associated local error satisfies,

$$LE_j = \frac{h^2}{2} y^{''}(\eta_j).$$

- Classical convergence result (for a fixed-step method):
  Definition: A method is said to converge if and only if,

$$\lim_{h\to 0,(N\to\infty)} \left\{ \max_{j=1,2,\cdots N} |y(x_j) - y_j| \right\} \to 0.$$

- **Theorem:** (typical of classical convergence results)
  Let $[x_j, y_j]_{j=0}^N$ be the approximate solution of the IVP, $y^{'} = f(x,y),\ y(a) = y_0$ over $[a,b]$ generated by Euler's method with constant stepsize $h = (b-a)/N$. If the exact solution, $y(x)$, has a continuous second derivative and $|f_y| < L,\ |y^{''}(x)| < Y$ then the associated global error, $e_j = y(x_j) - y_j$, at the points $x_j = a + j\ h$ satisfies,

$$\begin{aligned} |e_j| &\leq \frac{hY}{2L}(e^{(x_j-x_0)L} - 1) + e^{(x_j-x_0)L}|e_0|, \\ &\leq \frac{hY}{2L}(e^{(b-a)L} - 1) + e^{(b-a)L}|e_0|. \end{aligned}$$

  Note:

  (a) e_0 will usually be equal to zero.
  (b) This bound is generally pessimistic as it is exponential in $(b-a)$ where linear error growth is often observed.
  (c) In the general case one can show that when local error is $O(h^{p+1})$ the global error is $O(h^p)$.

  Proof of this Theorem (Outline only): Eulers Method satisfies,

$$y_j = y_{j-1} + hf(x_{j-1}, y_{j-1}).$$

  A Taylor series expansion of $y(x)$ about $x = x_{j-1}$ implies

$$y(x_j) = y(x_{j-1}) + hf(x_{j-1}, y(x_{j-1})) + \frac{h^2}{2}y^{''}(\eta_j).$$

  Subtracting the first equation from the second we obtain,

$$y(x_j) - y_j = y(x_{j-1}) - y_{j-1} + h\left[f(x_{j-1}, y(x_{j-1})) - f(x_{j-1}, y_{j-1})\right] + \frac{h^2}{2}y^{''}(\eta_j).$$

  If $Y = \max_{x\in[a,b]} |y^{''}(x)|$ and $|f_y| \leq L$, then, from the definition of e_j and the observation that $f(x,y)$ satisfies a Lipschitz condition with respect to y, we have

$$\begin{aligned} |e_j| &\leq |e_{j-1}| + hL|y(x_{j-1}) - y_{j-1}| + |\frac{h^2}{2}y^{''}(\eta_j)|, \\ &\leq |e_{j-1}| + hL|e_{j-1}| + \frac{h^2}{2}Y, \\ &= |e_{j-1}|(1 + hL) + \frac{h^2}{2}Y. \end{aligned}$$

This is a linear recurrence relation (or inequality) which after some work (straightforward) can be shown to imply our desired result,

$$|e_j| \leq \frac{hY}{2L}(e^{(b-a)L} - 1) + e^{(b-a)L}|e_0|.$$

Note that this is only an upper bound on the global error and it may not be sharp.

5. **An Example:**

   Consider the following equation,

   $$y' = y, \;\; y(0) = 1, \;\; \text{on } [0,1].$$

   Now since $\frac{\partial f}{\partial y} = 1$, $L = 1$ and since $y(x) = e^x$, we have $Y = e$ and $e_0 = 0$.

   Applying our error bound with $h = 1/N$ and $y_N \approx y(1) = e$ we obtain,

   $$|GE_N| = |Y_N - e| \leq \frac{he}{2}(e-1) < 2.4h.$$

   But for $h = .1$ we observe that $y_{10} = 2.5937..$ with an associated true error of .1246... This error bound is .24.

6. **Limitations and Difficulties with Classical Approach:**

   - Analysis is valid only in the limit as $h \to 0$.
   - Bounds are usually very pessimistic (can overestimate the error by several orders of magnitude).
   - Analysis does not consider the affect of f.p. arithmetic.

     To extend the analysis, consider applying Eulers method with roundoff error:
     Assume $fl(f(x_{j-1}, y_{j-1})) = f(x_{j-1}, y_{j-1}) + \epsilon_j$ and

     $$\begin{aligned} y_j &= y_{j-1} \oplus h \otimes fl(f(x_{j-1}, y_{j-1})), \\ &= y_{j-1} + hf(x_{j-1}, y_{j-1}) + h\epsilon_j + \rho_j, \end{aligned}$$

     where $|\epsilon_j|, \; |\rho_j| \; < \; \mu$.

     Then, proceeding as before we obtain,

     $$|e_j| < |e_{j-1}|(1 + hL) + \frac{h^2}{2}\bar{M},$$

     where $\bar{M} = Y + \mu/h + \mu/(h^2)$.

     Therefore the revised error bound becomes:

     $$\begin{aligned} |e_j| &\leq e^{(b-a)L}|e_0| + \frac{h\bar{M}}{2L}(e^{(b-a)L} - 1), \\ &= e^{(b-a)L}|e_0| + (e^{(b-a)L} - 1)(\frac{hY}{2L} + \frac{\mu}{2L} + \frac{\mu}{2hL}). \end{aligned}$$

     So, as $h \to 0$, the term $\frac{\mu}{2hL}$ will become unbounded (unless the precision changes) and we will not observe convergence.

- Special difficulties with fixed-h Euler:
  - The low order results in requiring a small stepsize, which leads to a large number of derivative evaluations and excessive amount of computer time.
  - The use of a constant stepsize can be inappropriate if the solution behaves differently on parts of the interval of interest. For example in integrating satellite orbits 'close approaches' typically requires a smaller stepsize to ensure accuracy.

7. **Runge-Kutta Methods:**

   (a) We will consider a general class of one-step formulas of the form:

   $$y_j = y_{j-1} + h\Phi(x_{j-1}, y_{j-1}). \tag{4}$$

   where Φ satisfies a Lipschitz condition with respect to y. That is,

   $$|\Phi(x,u) - \Phi(x,v)| \le \mathcal{L}|u - v|.$$

   We will consider a variety of choices for Φ and will observe that, in each case considered, Φ will be Lipschitz if f is.

   Two examples of such formulas are:

   Euler: $\Phi \equiv f$.

   Taylor Series: $\Phi \equiv T_k(x, y)$.

   (b) **Definition:** A formula (4) is of <u>order p</u> if for all sufficiently differentiable functions $y(x)$ we have,

   $$y(x_j) - y(x_{j-1}) - h\Phi(x_{j-1}, y(x_{j-1})) = O(h^{p+1}). \tag{5}$$

   Note that:

   i. The LHS of (5) is defined to be the <u>Local Truncation Error (LTE)</u> of the formula.

   ii. Order p implies that both the LE and the LTE are $O(h^{p+1})$. (This follows by substituting $z_j(x)$ for $y(x)$ in the definition.)

   (c) **Main Result:**

   <u>Theorem:</u> A p^{th} order formula applied to an initial value problem with constant stepsize h satisfies,

   $$|y(x_j) - y_j| \le |e_0|e^{\mathcal{L}(b-a)} + \frac{Ch^p}{\mathcal{L}}(e^{\mathcal{L}(b-a)} - 1).$$

   This result can be proved using a similar argument to that used in the Euler convergence theorem.

(d) We wish to consider formulas Φ that are less 'expensive' than higher order Taylor Series and yet are higher order than Euler's formula. Consider a formula Φ based on $\underline{2}$ derivative evaluations. That is,

$$\Phi(x_{j-1}, y_{j-1}) = \omega_1 k_1 + \omega_2 k_2,$$

where,

$$\begin{aligned} k_1 &= f(x_{j-1}, y_{j-1}), \\ k_2 &= f(x_{j-1} + \alpha h, y_{j-1} + h\beta k_1). \end{aligned}$$

We determine the parameters $\omega_1, \omega_2, \alpha, \beta$ to obtain as high an order formula as possible. From the definition of order we have order p if

$$y(x_j) = y(x_{j-1}) + h(\omega_1 k_1 + \omega_2 k_2) + O(h^{p+1}) \qquad (6)$$

for all sufficiently differentiable functions $y(x)$. To derive such a formula we expand $y(x_j), k_1, k_2$ in Taylor Series about the point (x_{j-1}, y_{j-1}), equate like powers of h on both sides of (6), and set $\alpha, \beta, \omega_1, \omega_2$ accordingly.

In what follows we omit arguments when they are evaluated at the point (x_{j-1}, y_{j-1}). The expansion of the LHS of (6) is:

$$\begin{aligned} LHS &= y(x_j), \\ &= y(x_{j-1}) + hy^{'}(x_{j-1}) + \frac{h^2}{2}y^{''}(x_{j-1}) + \frac{h^3}{6}y^{'''}(x_{j-1}) + O(h^4), \\ &= y(x_{j-1}) + hf + \frac{h^2}{2}(f_x + f_y f) \\ &\quad + \frac{h^3}{6}(f_{xx} + 2f_{xy}f + f_{yy}f^2 + f_y f_x + f_y^2 f) + O(h^4). \end{aligned}$$

The expansion of the RHS of (6) is more complicated and first requires the expansions of k_1 and k_2,

$$\begin{aligned} k_1 &= f, \\ k_2 &= f(x_{j-1} + \alpha h, y(x_{j-1}) + \beta h k_1), \\ &= f(x_{j-1}, y(x_{j-1}) + \beta h f) + (\alpha h) f_x(x_{j-1}, y(x_{j-1}) + \beta h f) \\ &\quad + \frac{\alpha^2 h^2}{2} f_{xx}(x_{j-1}, y(x_{j-1}) + \beta h f) + O(h^3), \\ &= \left[f + \beta h f f_y + \frac{(\beta h f)^2}{2} f_{yy} + O(h^3) \right] \\ &\quad + \left[\alpha h f_x + \alpha\beta h^2 f f_{xy} + O(h^3) \right] + \left[\frac{\alpha^2 h^2}{2} f_{xx} + O(h^3) \right], \\ &= f + (\beta f f_y + \alpha f_x)\, h + (\frac{\beta^2}{2} f^2 f_{yy} + \alpha\beta f f_{xy} + \frac{\alpha^2}{2} f_{xx})\, h^2 + O(h^3). \end{aligned}$$

The expansion of the RHS of (6) then is (with these substitutions for k_1 and k_2)

$$\begin{aligned}
RHS &= y(x_{j-1}) + h(\omega_1 k_1 + \omega_2 k_2), \\
&= y(x_{j-1}) + h\omega_1 f + h\omega_2 [\cdots] + O(h^4), \\
&= y(x_{j-1}) + [(\omega_1 + \omega_2) f]\ h + [\omega_2(\beta f f_y + \alpha f_x)]\ h^2 \\
&+ \left[\omega_2(\frac{\beta^2}{2} f^2 f_{yy} + \alpha\beta f f_{xy} + \frac{\alpha^2}{2} f_{xx})\right]\ h^3 + O(h^4).
\end{aligned}$$

Finally these expansions are true for all values of h, so equating like powers of h, in the LHS and RHS expansions, we observe the following:

For order 0 : The coefficients of h^0 always agree and we have order at least zero for any choice of the parameters.

For order 1: If $\omega_1 + \omega_2 = 1$ the coefficients of h^1 agree and we have at least order 1.

For order 2: In addition to satisfying the order 1 constraints we must have the coefficient of h^2 the same. That is $\alpha\omega_2 = 1/2$ and $\beta\omega_2 = 1/2$.

For order 3: In addition to satisfying the order 2 constraints we must have the coefficients of h^3 the same. That is we must satisfy the equations,

$$\begin{aligned}
\omega_2 \alpha^2 &= \frac{1}{3}, \\
\omega_2 \alpha\beta &= \frac{1}{3}, \\
\omega_2 \beta^2 &= \frac{1}{3}, \\
\frac{1}{6} f_{xy} &= ?, \\
\frac{1}{6} f_y^2 &= ?.
\end{aligned}$$

Note that there are not enough terms in the coefficient of h^3 in the expansion of the RHS to match the expansion of the LHS. We cannot therefore equate the coefficients of h^3 and the maximum order we can obtain is order 2. Our formula will be order 2 for any choice of $\omega_2 \neq 0$, with $\omega_1 = 1 - \omega_2$ and $\alpha = \beta = \frac{1}{2\omega_2}$. This is a one-parameter family of 2^{nd}-order Runge-Kutta formulas.

Three popular choices from this family are:

Modified Euler: $\omega_2 = 1/2$

$$\begin{aligned}
k_1 &= f(x_{j-1}, y_{j-1}), \\
k_2 &= f(x_{j-1} + h, y_{j-1} + hk_1), \\
y_j &= y_{j-1} + \frac{h}{2}(k_1 + k_2).
\end{aligned}$$

Midpoint: $\omega_2 = 1$

$$\begin{aligned} k_1 &= f(x_{j-1}, y_{j-1}), \\ k_2 &= f(x_{j-1} + \frac{h}{2}, y_{j-1} + \frac{h}{2}k_1), \\ y_j &= y_{j-1} + hk_2. \end{aligned}$$

Heun's Formula: $\omega_2 = 3/4$

$$\begin{aligned} k_1 &= f(x_{j-1}, y_{j-1}), \\ k_2 &= f(x_{j-1} + \frac{2}{3}h, y_{j-1} + \frac{2}{3}hk_1), \\ y_j &= y_{j-1} + \frac{h}{4}(k_1 + 3k_2). \end{aligned}$$

8. **Higher-order Runge-Kutta formulas:**

An s-stage explicit Runge-Kutta formula uses s derivative evaluations and has the form:

$$y_j = y_{j-1} + h(\omega_1 k_1 + \omega_2 k_2 \cdots + \omega_s k_s),$$

where

$$\begin{aligned} k_1 &= f(x_{j-1}, y_{j-1}), \\ k_2 &= f(x_{j-1} + \alpha_2 h, y_{j-1} + h\beta_{21} k_1), \\ &\vdots \\ k_s &= f(x_{j-1} + \alpha_s h, y_{j-1} + h\sum_{r=1}^{s-1} \beta_{sr} k_r). \end{aligned}$$

This formula is represented by the tableau,

$$\begin{array}{c|ccccc} - & - & & & & \\ \alpha_2 & \beta_{21} & - & & & \\ \vdots & \vdots & & & & \\ \alpha_s & \beta_{s1} & \beta_{s2} & \dots & \beta_{s-1,s} & - \\ \hline & \omega_1 & \omega_2 & \dots & & \omega_s \end{array}$$

These $\frac{s(s-1)}{2} + (s-1) + s$ parameters are usually chosen to maximise the order of the formula.

The maximum attainable order for an s-stage Runge-Kutta formula is given by the following table:

s	1	2	3	4	5	6
max order	1	2	3	4	4	5

Note that the derivations of these maximal order formulas can be very messy and tedious, but essentially they follow (as outlined above for the case $s = 2$) by expanding each of the k_r in a 2-dimensional Taylor series.

An Example – The Classical Fourth Order Runge Formula (1895)

$$\begin{array}{c|cccc}
- & - & & & \\
1/2 & 1/2 & - & & \\
1/2 & 0 & 1/2 & - & \\
1 & 0 & 0 & 1 & - \\
\hline
 & 1/6 & 1/3 & 1/3 & 1/6
\end{array}$$

9. **Error Estimates:**

   (a) Ideally a method would estimate a bound on the global error and adjust the stepsize, h, to keep the magnitude of the global error less than a tolerance. Such computable bounds are possible but are usually pessimistic and inefficient to implement.

   (b) On the other hand, <u>local errors</u> can be reliably and efficiently estimated and controlled. Consider a method which keeps the magnitude of the local error less than $h\,TOL$ on each step.

   That is, if $z_{j-1}(x)$ is the local solution on step j,

$$z_{j-1}^{'} = f(x, z_{j-1}),\ z_{j-1}(x_{j-1}) = y_{j-1},$$

   then a method will adjust $h = x_j - x_{j-1}$ to ensure that $|z_{j-1}(x_j) - y_j| \leq h\,TOL$, for $j = 1, 2 \cdots N_{TOL}$.

   (c) With this type of error control one can show that, for the resulting $(x_j, y_j)_{j=0}^{N_{TOL}}$ there exists a piecewise polynomial, $Z(x) \in C^1[a, b]$ such that $Z(x_j) = y_j$ for $j = 0, 1, \cdots N_{TOL}$ and for $x \in [a, b]$,

$$|Z^{'}(x) - f(x, Z)| \leq TOL.$$

   We can also show that,

$$|y(x_j) - y_j| \leq \frac{TOL}{L}(e^{L(x_j - a)} - 1).$$

   (d) Derivation of Local Error Estimates for Runge-Kutta formulas:

   Consider the Modified Euler Formula:

$$\begin{array}{c|cc}
- & - & \\
1 & 1 & - \\
\hline
 & 1/2 & 1/2
\end{array}$$

We have shown

$$\begin{aligned}
z_{j-1}(x_j) &= y_{j-1} + \frac{h}{2}(k_1 + k_2) \\
&\quad + \left[\frac{1}{4}f^2 f_{yy} + \frac{1}{2}f f_{xy} + \frac{1}{4}f_{xx} - y^{'''}(x_j)\right] h^3 + O(h^4), \\
&= y_j + \left[\frac{1}{12}f_{yy}f^2 + \frac{1}{6}f f_{xy} + \frac{1}{12}f_{xx} - f_{xy} - f_y^2 f\right] h^3 + O(h^4), \\
&\equiv y_j + c(f)h^3 + O(h^4).
\end{aligned}$$

It then follows that the local error, LE, satisfies

$$LE = c(f)h^3 + O(h^4),$$

where $c(f)$ is a complicated function of f. There are two general strategies for estimating $c(f)$ – the use of "step halving" and the use of a 3rd order "companion formula".

Step Halving: let $\hat{y}_j$ be the approximation to $z_{j-1}(x_j)$ computed with two steps of size $h/2$. If $c(f)$ is almost constant the we can show

$$z_{j-1}(x_j) = \hat{y}_j + 2c(f)(\frac{h}{2})^3 + O(h^4)$$

and from above

$$z_{j-1}(x_j) = y_j + c(f)h^3 + O(h^4).$$

Therefore the local error associated with $\hat{y}_j$, $\widehat{LE}$, is

$$\begin{aligned}
\widehat{LE} &= 2c(f)(\frac{h}{2})^3 + O(h^4), \\
&= \frac{1}{3}(y_j - \hat{y}_j) + O(h^4).
\end{aligned}$$

The method could then compute $\hat{y}_j, y_j$ and accept $\hat{y}_j$ only if $\frac{1}{3}|y_j - \hat{y}_j| < h\,TOL$.

Note that this strategy requires five derivative evaluations on each step and assumes that each of the components of $c(f)$ is <u>slowly</u> varying.

Third Order Companion Formula: To estimate the local error associated with the Modified Euler formula consider the use of a 3-stage, 3rd order Runge-Kutta formula,

$$\begin{aligned}
\hat{y}_j &= y_{j-1} + h(\hat{\omega}_1 k_1 + \hat{\omega}_2 k_2 + \hat{\omega}_3 k_3), \\
&= z_{j-1}(x_j) + O(h^4),
\end{aligned}$$

We also have

$$\begin{aligned}
y_j &= y_{j-1} + \frac{h}{2}(k_1 + k_2), \\
&= z_{j-1}(x_j) - c(f)h^3 + O(h^4).
\end{aligned}$$

Subtracting these two equations we have the local error estimate,

$$est_j \equiv (\hat{y}_j - y_j) = c(f)h^3 + O(h^4).$$

Note that, for any 3rd order formula, $k_1 = \hat{k}_1$, so we require at most 4 derivative evaluations per step to compute both y_j and est_j. Furthermore, if $\hat{\alpha}_2 = \alpha_2 = 1$ and $\hat{\beta}_{21} = \beta_{21} = 1$, we have $\hat{k}_2 = k_2$ and the cost is only three derivative evaluations per step. The obvious question is can one derive such a 3-stage 3rd order Runge-Kutta formula ? The answer is yes and the following tableau with $\hat{\alpha}_3 \neq 1$ defines a one-parameter family of such "companion formulas" for the Modified Euler formula:

$$\begin{array}{c|ccc}
- & - & & \\
1 & 1 & - & \\
\hat{\alpha}_3 & \hat{\beta}_{31} & \hat{\beta}_{32} & - \\
\hline
 & \hat{\omega}_1 & \hat{\omega}_2 & \hat{\omega}_3
\end{array}$$

with

$$\hat{\beta}_{31} = \hat{\alpha}_3^2, \hat{\beta}_{32} = \hat{\alpha}_3 - \hat{\alpha}_3^2,\ \hat{\omega}_2 = \frac{1}{6(\hat{\alpha}_3 - 1)},\ \hat{\omega}_3 = \frac{-1}{6(\hat{\alpha}_3 - 1)},\ \hat{\omega}_1 = 1 - (\frac{1 + 3\hat{\alpha}_3}{6\hat{\alpha}_3}).$$

Generalization to Higher Order: This idea of using a "companion formula" of order $p+1$ to estimate the local error of a p^{th} order formula leads to the derivation of s-stage, order $(p, p+1)$ formula pairs with the fewest number of stages. Such formula pairs can be characterized by the tableau:

$$\begin{array}{c|ccccc}
- & - & & & & \\
\alpha_2 & \beta_{21} & - & & & \\
\vdots & \vdots & & & & \\
\alpha_s & \beta_{s1} & \dots & & \beta_{s-1,s} & - \\
\hline
 & \omega_1 & \omega_2 & \dots & & \omega_s \\
 & \hat{\omega}_1 & \hat{\omega}_2 & \dots & & \hat{\omega}_s
\end{array}$$

where

$$\begin{aligned}
y_j &= y_{j-1} + h\sum_{r=1}^{s} \omega_r k_r \\
&= z_{j-1}(x_j) - c(f)h^{p+1} + O(h^{p+2}), \\
\hat{y}_j &= y_{j-1} + h\sum_{r=1}^{s} \hat{\omega}_r k_r \\
&= z_{j-1}(x_j) + O(h^{p+2}), \\
est_j &= (\hat{y}_j - y_j) \\
&= c(f)h^{p+1} + O(h^{p+2}).
\end{aligned}$$

Note that the error estimate is a reliable estimate of the local error associated with the lower order (order p) formula. The following table gives the fewest number of stages required to generate formula pairs of a given order.

order pair	(2,3)	(3,4)	(4,5)	(5,6)	(6,7)
fewest stages	3	4	6	8	10

10. **Stepsize Control:**

- Step is accepted only if $|est_j| < hTOL$.
- If h is too large, the step will be rejected and the derivative evaluations will be wasted.
- If h is too small, there will be many steps and more function evaluations than necessary.

The usual strategy for choosing the attempted stepsize, h, for the next step is based on 'aiming' at the largest h which will result in an accepted step on the current step. If we assume that $c(f)$ is slowly varying then,

$$|est_j| = |c(f)|h_j^{p+1} + O(h^{p+2}),$$

and on the next step attempted step, $h_{j+1} = \gamma h_j$, we want

$$|est_{j+1}| \approx TOL\ h_{j+1}.$$

But

$$\begin{aligned} |est_{j+1}| &\approx |c(f)|(\gamma h_j)^{p+1}, \\ &= \gamma^{p+1}|est_j|. \end{aligned}$$

We can then expect

$$|est_{j+1}| \approx TOL\ h_{j+1},$$

if

$$\gamma^{p+1}|est_j| \approx TOL\ (\gamma h_j),$$

which is equivalent to

$$\gamma^p|est_j| \approx TOL\ h_j.$$

The choice of γ to satisfy this heuristic is then,

$$\gamma = \left(\frac{TOL\ h_j}{|est_j|}\right)^{1/p}.$$

A typical step-choosing heuristic, justified by the above discussion, is to use the formula,

$$h_{j+1} = .9\left(\frac{TOL\ h_j}{|est_j|}\right)^{1/p} h_j,$$

where .9 is a 'safety factor'. The formula works for use after a rejected step as well but must be modified slightly when round-off errors are significant.