Rectangle Rule for Integration

$$\int_{x_{i-1}}^{x_i} f(x)\mathrm{d}x \approx \sum_{i=1}^{n} h\, f\left(\frac{x_{i-1}+x_i}{2}\right)$$

Where:

$$\begin{aligned} n &= \text{Number of Subintervals} \\ h &= \frac{x_{\text{final}} - x_{\text{initial}}}{n} \\ &= \text{Length of a Single Subinterval} \end{aligned}$$

Trapezoidal Rule for Integration

$$\int_{x_{i-1}}^{x_i} f(x)\mathrm{d}x \approx \sum_{i=1}^{n} \frac{h}{2}\left(f(x_{i-1}) + f(x_i)\right)$$

Where:

$$\begin{aligned} n &= \text{Number of Intervals} \\ h &= \frac{x_{\text{final}} - x_{\text{initial}}}{n} \\ &= \text{Length of a Single Subinterval} \end{aligned}$$

Simpson Rule for Integration

$$\int_{x_{i-1}}^{x_i} f(x)\mathrm{d}x \approx \frac{2}{3}\, I_{\text{rectangular}} + \frac{1}{3}\, I_{\text{trapezoidal}}$$

Where:

$$\begin{aligned} I_{\text{rectangular}} &= \text{Rectangle Rule Estimate} \\ I_{\text{trapezoidal}} &= \text{Trapezoidal Rule Estimate} \end{aligned}$$

Gauss Quadrature Rule for Integration

$$\int_{-1}^{1} f_n(\xi)\,\mathrm{d}\xi = \sum_{k=1}^{q} f_n(\xi_k)\,w_k$$

Basic Idea: We will try to minimize the number of sampling points needed to integrate exactly a polynomial of a chosen degree n on the domain [-1, 1].

Where:

w_k = Weighting Function

- **To integrate exactly (n=1)** $f_1(\xi) = \alpha_0 + \alpha_1\,\xi$

$$\begin{cases} w_1 = 2 \\ \xi_1 = 0 \end{cases}$$

- **To integrate exactly (n=3)** $f_3(\xi) = \alpha_0 + \alpha_1\,\xi + \alpha_2\,\xi^2 + \alpha_3\,\xi^3$

  We need 2 sampling points (and 2 weights)

$$\begin{cases} w_1 = 1 & w_2 = 1 \\ \xi_1 = -\frac{\sqrt{3}}{3} & \xi_2 = \frac{\sqrt{3}}{3} \end{cases}$$

- **To integrate exactly (n=5)** $f_5(\xi) = \alpha_0 + \alpha_1\,\xi + \alpha_2\,\xi^2 + \alpha_3\,\xi^3 + \alpha_4\,\xi^4 + \alpha_5\,\xi^5$

  We need 3 sampling points (and 3 weights):

$$\begin{cases} w_1 = \frac{8}{9} & w_2 = \frac{5}{9} & w_3 = \frac{5}{9} \\ \xi_1 = 0 & \xi_2 = \sqrt{\frac{3}{5}} & \xi_3 = -\sqrt{\frac{3}{5}} \end{cases}$$

Coordinate Transformation

$$\xi = \frac{2x}{b-a} + \frac{a+b}{a-b} = \frac{2x-(a+b)}{b-a}$$

such that

$$\int_a^b f(x)\,\mathrm{d}x = \frac{b-a}{2}\int_{-1}^{1} f\left(\frac{(b-a)\xi+(a+b)}{2}\right)\mathrm{d}\xi$$

Forward Difference Scheme

$$f'(x) = \frac{f(x+h)-f(x)}{h} - \frac{h}{2}f''(\xi) \qquad \text{with } \xi \in (x, x+h)$$

Central Difference Scheme

$$f'(x) = \frac{f(x+h)-f(x-h)}{2h} - \frac{h^2}{6}f'''(\xi) \qquad \text{with } \xi \in (x, x+h)$$

Richardson Extrapolation

$$a_o = F(h) + \frac{F(h)-F(qh)}{q^p-1} + O(h^r) \qquad \text{or} \qquad a_o = \frac{F(qh)-q^p\,F(h)}{1-q^p} + O(h^r)$$

Where:

$p = 1$ For Forward Difference Scheme since error is linear.

$p = 2$ For Central Difference Scheme since error is squared.

Rate of Convergence

Let e_k be the "error" associated with iteration k

(on $||x_k - x^*||, ||f(x_k)||, size k^{th}$ bracketing interval,...)

$$\text{Rate of Convergence } = r \qquad \text{if} \qquad \lim_{k\to\infty} \frac{||e_{k+1}||}{||e_k||^2} = C$$

If

$r = 1$ we have **Linear Convergence.**

$r > 1$ we have **Superlinear Convergence.**

$r = 2$ we have **Quadratic Convergence.**

Bisection Method

Basic Idea: Bracket the root until the interval size is small enough.

```
function x = bisection(f, x_low, x_high, x_tol)
% Isolate a root of f(x) using bisection.
while x_high-x_low > x_tol
    c = (x_low+x_high)/2; % Find mid-point between interval brackets
    if fn(x_low)*fn(c) > 0 % If midpoint above 0, set lower bracket to midpoint, repeat
        x_low = c;
    else
        x_high = c; % If midpoint below 0, set upper bracket to midpoint, repeat.
    end
end
x = c;
end
```

$$\text{Rate of Convergence } = r \qquad \text{if} \qquad \lim_{k\to\infty} \frac{||e_{k+1}||}{||e_k||^r} = C$$

$$\text{Number of Iterations} = n = \log_2\left(\frac{b-a}{tolerance}\right)$$

Fixed-Point Iteration Method

Newton-Raphson Method **Linear Equations**

- Start from an initial guess of the root x_1
- For each iteration step, $k = 1, 2, 3, ...$
  - check for convergence : $||f(x_k)|| < Tolerance$
  - if so, exit: you have found the approximate root
  - if not, compute the correction increment : $\Delta x_k = -\frac{f(x_k)}{f'(x_k)}$
  - then update the solution: $x_{k+1} = x_k + \Delta x_k$

```
x=x_init
while abs(fn(x)) > tolerance
        dx = -fn(x)/fn_prime(x);
        x = x + dx
end
```

Secant Method

Secant method allowes for a successive $f(x_k)$ to approach the derivative such that:

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

```
% Isolate a root of f(x) using the secant method.
x = x_0; % Initial x
while abs(fn(x))>tolerance
    x_kp1 = x_1 - fn(x_1)*((x_1 - x)/(fn(x_1)-fn(x))); % x_{k+1}
    x = x_1;
    x_1 = x_kp1;
end
```

Newton-Raphson Method **Nonlinear Equations**

- Start from an initial guess of the root x_1
- For each iteration step, $k = 1, 2, 3, ...$
  - check for convergence : $||f(x_k)|| < Tolerance$
  - if so, exit: you have found the approximate root
  - if not, compute the correction increment : $\nabla f(x_k) \cdot \Delta x_k = -f(x_k)$
  - then update the solution: $x_{k+1} = x_k + \Delta x_k$

See Homework 2 Problem 2 for example.

ODE Stability

Forward Euler Scheme

Let h denote the time step size. Using Taylor series expansion $y(t+h)$ becomes:

$$y(t+h) \quad = \quad y(t) + h\,\frac{\mathrm{d}y(t)}{\mathrm{d}t} \quad = \quad y(t) + h\,f(t, y(t)) + O(h^2)$$

The iterate scheme thus is:

1) Start from t_0 and $y(t_0) = y_0$

2) $y(h) = y_1 = y_0 + h\,f(t_0, y_0)$

3) $y(2h) = y_2 = y_1 + h\,f(t_1, y_1)$

...

4) $y((n+1)h) = y_{n+1} = y_n + h\,f(t_n, y_n)$

Use Forward Euler for initial guess

$$y_{k+1}^{(0)} = y_k + h\, f(t_k, y_k)$$

Then iterate using the fixed-point scheme until convergence

$$y_{k+1}^{(i)} = y_k + h\, f(t_{k+1}, y_{k+1}^{(i-1)})$$

Once convergence is achieved, move to t_{k+2}.