BM 593 Numerical Methods & C Programming

4th week Computer Arithmetic & Number Representations

Float representation : 4 bytes

$x_f = (-1)^s \times mantissa \times 2^{exp-bias}$

$(0.5)_f$: 0 0111 1111 1000 0000 0000 0000 0000 000

The bias : 0111 1111

$(\text{Maximum Number})_f$: 0 1111 1111 1111 1111 1111 1111 1111 111 : $2^{128} = 3.4 \times 10^{38}$

$(\text{Minimum Number})_f$: 0 0000 0000 1000 0000 0000 0000 0000 000 : $2^{-128} = 2.9 \times 10^{-39}$

Machine Precision

$7 + 1.0 \times 10^{-7}$

$(7)_f$: 0 1000 0010 1110 0000 0000 0000 0000 000

$(10^{-7})_f$: 0 0110 0000 1101 0110 1011 1111 1001 010

Shift right to align the exponents before adding

$(10^{-7})_f$: 0 1000 0010 0000 0000 0000 0000 0000 000 (0001101...)

$7 + 1.0 \times 10^{-7} = 7$

If 24th bit is 1, round up makes an error of $2^{-23} \approx 10^{-7}$

Float precision is no more reliable after 7 decimal digits

Relative Error

$x = (-39.9)_{10}$: 1 1000 0101 1001 1111 1001 1001 1001 100 $\overline{1100}$

$x_f =$: 1 1000 0101 1001 1111 1001 1001 1001 101 rounded up

$x_f = (-39.90000152587890625)_{10}$

Relative Error= $\epsilon_r = |x_f - x|/|x|$

Maximum Relative Error occurs at $x = 1$ with $\epsilon_r^{max} = 2^{-23}$

Double precision representation : 8 bytes

52 bits : mantissa

11 bits : exponent

Double Precision Round up error : $2^{-52} \approx 10^{-16}$

Magnitude Range of Double Precision : $2.225074 \times 10^{-308}$: 1.799693×10^{308}

Numerical Evaluation

$\log 2 \approx ?$

$\log 3 \approx ?$

$e \approx ?$

$\ln 2 \approx ?$

Series Expansion

$(a+b)^n = a^n + na^{n-1}/1! + n(n-1)a^{n-1}b^2/2! + \dots$

$(1+x)^n = 1 + nx/1! + n(n-1)x^2/2! + \dots$

Taylor Expansion Theorem

$f(x-x_0) = f(x_0) + f'(x_0)(x-x_0)/1! + f''(x_0)(x-x_0)^2/2! + \dots$

The 1st derivative

$f'(x) =_{\lim \Delta x \to 0} \frac{f(x+\Delta x)-f(x)}{\Delta x}$

$(sin(x))' =?$

Chain Rule

$d(f(g(x)))/dx = d(f(x))/dx \quad d(g(x))/dx$

$(sin^2(x))' =?$

The second Derivative

$f''(x) =_{\lim \Delta x \to 0} \frac{f'(x+\Delta x)-f'(x)}{\Delta x}$

$(x^2)'' =?$

Differential equation

$f''(x) + \alpha f'(x) = x$

SOLUTION of NONLINEAR EQUATIONS $f(x) = 0$

Fixed Point Iteration

p_0 : Initial Point

$p_1 = f(p_0)$

$\vdots$

$p_k = f(p_{k-1})$

$p_{k+1} = f(p_k)$

until $x = f(x)$

Bracketing Methods

Bisection Method

Find an interval defined by $[a, b]$ determine $c = (a+b)/2$ and analyze the three possibilities

If $f(a)$ and $f(c)$ have opposite signs, a root lies in $[a, c]$.

If $f(c)$ and $f(b)$ have opposite signs, a root lies in $[c, b]$.

If $f(c) = 0$ we found a root at $x = c$.

Method of False Position (Regula Falsi)

$m = (f(b) - f(a))/(b-a)$

$m = (0 - f(a))/(c-a)$

$c = b - f(b)(b-a)/(f(b) - f(a))$

$c_n = b_n - f(b_n)(b_n - a_n)/(f(b_n) - f(a_n))$

Newton–Raphson Method

$m = (0 - f(p_0))/(p_1 - p_0) = f'(p_0)$

$p_1 = p_0 - f(p_0)/f'(p_0)$

$p_k = p_{k-1} - f(p_{k-1})/f'(p_{k-1})$ for $k = 1, 2, \dots$

Secant Method

$m = (f(p_1) - f(p_0))/(p_1 - p_0) = (0 - f(p_1))/(p_2 - p_1)$

$p_2 = p_1 - f(p_1)(p_1 - p_0)/(f(p_1) - f(p_0))$

$p_{k+1} = p_k - f(p_k)(p_k - p_{k-1})/(f(p_k) - f(p_{k-1}))$

Iteration for nonlinear systems

$f1(x, y) = x^2 - 2x - y + 0.5 = 0,$

$f2(x, y) = x^2 + 4y^2 - 4 = 0,$

$x = (x^2 - y + 0.5)/2,$

$y = (-x^2 - 4y^2 + 8y + 4)/8,$

$p_{k+1} = (p_k^2 - q_k + 0.5)/2,$

$p_{k+1} = (-p_k^2 - 4q_k^2 + 8q_k + 4)/8,$

Jacobian Matrix

$$\mathbf{J}(x, y) = \begin{bmatrix} \partial f_1/\partial x & \partial f_1/\partial y \\ \partial f_2/\partial x & \partial f_2/\partial y \end{bmatrix}$$

Convergence for fixed point iterations

$|\partial g_1(p, q)/\partial x| + |\partial g_1(p, q)/\partial y| < 1$

$|\partial g_2(p, q)/\partial x| + |\partial g_2(p, q)/\partial y| < 1$

Seidel Iteration

$$\begin{aligned} x &= g1(x, y, z) \\ y &= g2(x, y, z) \\ z &= g3(x, y, z) \end{aligned}$$

$p_{k+1} = g_1(p_k, q_k, r_k),\ q_{k+1} = g_2(p_{k+1}, q_k, r_k),\ r_{k+1} = g_3(p_{k+1}, q_{k+1}, r_k)$

Newton's Method based on linear approximation

$$\mathbf{f}(x, y) = \begin{bmatrix} f_1(x, y) \\ f_2(x, y) \end{bmatrix} = \mathbf{f}(x_0, y_0) + (x - x_0)\partial\mathbf{f}(x, y)/\partial x + (y - y_0)\partial\mathbf{f}(x, y)/\partial y + \dots$$

$\mathbf{f}(x, y) = \mathbf{f}(x_0, y_0) + \mathbf{J}(x, y)[(x - x_0) \quad (y - y_0)]' + \dots$

Based on iteration

$\mathbf{P}_{k+1} = \mathbf{P}_k + \Delta\mathbf{P} = \mathbf{P}_k - \mathbf{J}(p_k, q_k)^{-1}\mathbf{f}(p_k, q_k)$ where $\mathbf{P}_k = [p_k \quad q_k]'$

Outline of Newton's method:

1. Evaluate the function $\mathbf{f}(\mathbf{P}_k) = \begin{bmatrix} f_1(p_k, q_k) \\ f_2(p_k, q_k) \end{bmatrix}$,
2. Evaluate the Jacobian $\mathbf{J}(\mathbf{P}_k)$,
3. Solve the linear system $\mathbf{J}(\mathbf{P}_k)\Delta\mathbf{P} = -\mathbf{f}(\mathbf{P}_k) \quad for \quad \Delta\mathbf{P}$,
4. Compute the next point $\mathbf{P}_{k+1} = \mathbf{P}_k + \Delta\mathbf{P}$,
5. Repeat 1 until convergence.