Fortran & Numerical Computations

Rasyid Sulaeman

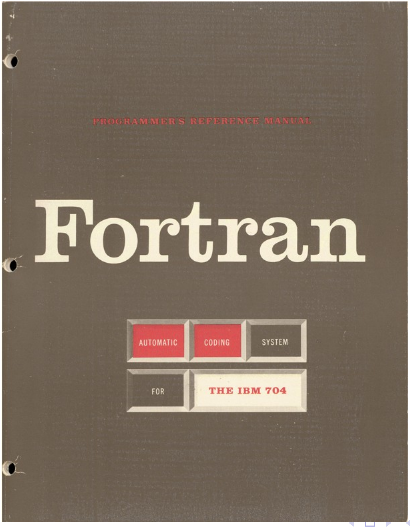
University of Indonesia

rasyid.sulaeman@ui.ac.id

July 2, 2019

Overview

What is Fortran?



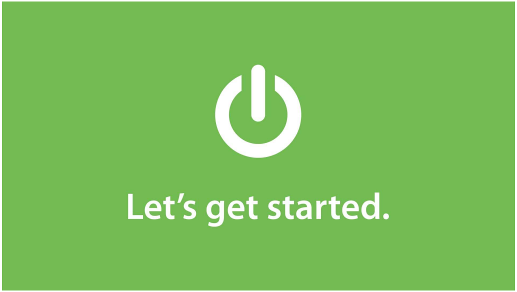
What is Fortran?

- Programming language that is extensively used in numerical and scientific computing.
- Differ from other languages, Fortran have to be *compiled* before run it on a computer.

Getting a Feel for Fortran

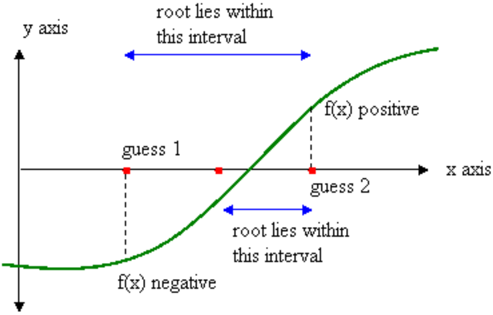
```
program belajar

implicit none
REAL :: a, b, c, D
REAL :: x1, x2
REAL :: Discriminant


write(*,*) 'Enter number a, b, c ='
read (*,*) a, b, c


D = Discriminant(a,b,c)
write(*,*) 'Discriminant = ', D

if (D .ge. 0.0) then
        call roots(a, b, c, x1, x2)
        write(*,*) 'Roots x1 = ', x1, 'x2 =', x2

    else if (D == 0.0) then
        call roots(a, b, c, x1, x2)
        write(*,*) 'Equal roots = x1', x1

    else
        call roots(a, b, c, x1, x2)
        write(*,*) 'Doesnt have a roots'
end if

end program belajar
```

Let's get started



Roots of Functions

Bisection Method



Roots of Functions

Bisection Method

Principles

Confine roots of a function in between two boundary, then **take a midpoint of the interval** continously in such a way we get a good approximation about the roots.

- Guess the root of a functions
- Determine the initial boundary that confine the root
- Take a midpoint of the interval
- Determine a region that have root
- Repeat step 3 and 4 until **convegence**
- You get your roots

Solution of Linear System Equation

Linear system equations has a form:

$$
\begin{array}{cccccc}
a_{11}x_1 & + a_{12}x_2 & + a_{13}x_3 & \dots & + a_{1n}x_n = & b_1 \\
a_{21}x_1 & + a_{22}x_2 & + a_{23}x_3 & \dots & + a_{2n}x_n = & b_2 \\
a_{31}x_1 & + a_{32}x_2 & + a_{33}x_3 & \dots & + a_{3n}x_n = & b_3 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
a_{n1}x_1 & + a_{n2}x_2 & + a_{n3}x_3 & \dots & + a_{nn}x_n = & b_n
\end{array} \tag{1}
$$

Solution of Linear System Equation

Gauss Elimination

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}$$

$$\boxed{\begin{array}{l} a_{ij}^{(0)} = a_{ij}, \quad b_i^{(0)} = b_i \\ a_{ij}^{(k)} = a_{ij}^{(k-1)} - \dfrac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} a_{kj}^{(k-1)} \\ b_i^{(k)} = b_i^{(k-1)} - \dfrac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} b_k^{(k-1)} \\ (k = 1, \ldots, n-1; \\ i = k+1, \ldots, n; j = k, \ldots, n) \end{array}}$$

$a_{ij}^{(m)}, b_i^{(m)} \equiv a_{ij}, b_i$
pada langkah ke m

$$\begin{pmatrix} a_{11}^{(0)} & a_{12}^{(0)} & a_{13}^{(0)} & \cdots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn}^{(n-1)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^{(0)} \\ b_2^{(1)} \\ b_3^{(2)} \\ \vdots \\ b_n^{(n-1)} \end{pmatrix}$$

$\longrightarrow$ halaman berikut

Solution of Linear System Equation

Gauss Elimination

$$\begin{pmatrix} a_{11}^{(0)} & a_{12}^{(0)} & a_{13}^{(0)} & \cdots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn}^{(n-1)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^{(0)} \\ b_2^{(1)} \\ b_3^{(2)} \\ \vdots \\ b_n^{(n-1)} \end{pmatrix}$$

$$\downarrow$$

$$\boxed{\begin{aligned} x_n &= \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}} \\ x_{n-j} &= \frac{b_{n-j}^{(n-j-1)} - \sum_{k=n-j+1}^{n} a_{n-j,k}^{(n-j-1)} x_k}{a_{n-j,n-j}^{(n-j-1)}} \qquad (j=1,\ldots,n-1) \end{aligned}}$$

Solution of Linear System Equation

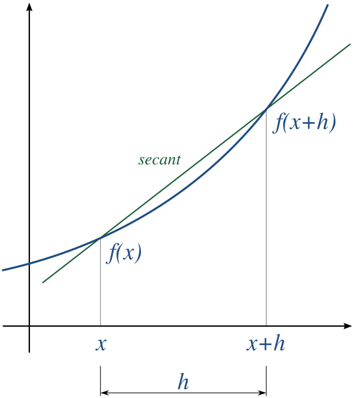
Gauss Elimination

So, Gauss elimination method consist of two steps:

1. Triangulation. Change matrix A into triangular matrix (so as matrix B)
2. Do backwards substitution. Calculate x by reverser order, from the last x_n to x_1.

Numerical Differentiation

Numerical Differentiation

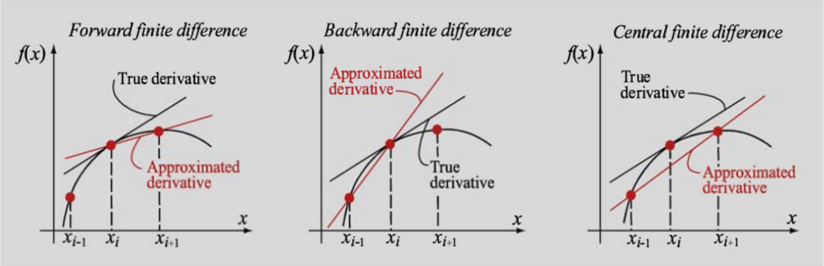


A simple approximation of the first derivative is

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

Numerical Differentiation

Forward, Backward, and Central Finite Difference



Numerical Differentiation

Forward, Backward, and Central Finite Difference

First Derivative		
Method	**Formula**	**Truncation Error**
Two-point forward difference	$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h}$	$O(h)$
Three-point forward difference	$f'(x_i) = \frac{-3f(x_i) + 4f(x_{i+1}) - f(x_{i+2})}{2h}$	$O(h^2)$
Two-point backward difference	$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{h}$	$O(h)$
Three-point backward difference	$f'(x_i) = \frac{f(x_{i-2}) - 4f(x_{i-1}) + 3f(x_i)}{2h}$	$O(h^2)$
Two-point central difference	$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$	$O(h^2)$
Four-point central difference	$f'(x_i) = \frac{f(x_{i-2}) - 8f(x_{i-1}) + 8f(x_{i+1}) - f(x_{i+2})}{12h}$	$O(h^4)$

Terima Kasih