METODE SECANT

"METODE SECANT"

Waktu di SMA, kita sering menyelesaikan persamaan kuadrat yaitu berbentuk

$f(x) = a.\ x^2 + b.x + c$

misalnya persamaan kuadrat:

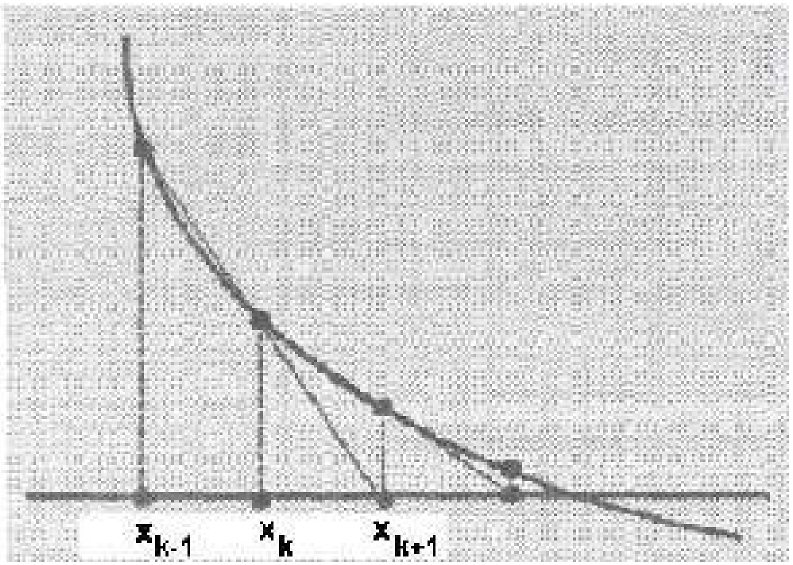
$x^2 - 9 = 0$, maka akar-akarnya dapat ditentukan dengan persamaan abc

$x = (-b \pm \sqrt{b^2 - 4.ac})/2a$

Maka akar $x^2 - 9 =$

adalah $x1 = +3$ dan $x2 = -3$

Metode Secant merupakan perbaikan dari Metode Newton, yaitu nilai turunan f'(x) didekati dengan beda hingga (Δ)



gambar 1. Penentuan nilai turunan fungsi dengan metode Secant.

Dimana,

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

Sehingga dalam persamaan Newton-Rhapson menjadi:

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

Algoritma program untuk metode Secant:

- Tentukan $X0$, $X1$, toleransi, dan jumlah iterasi maksimum.
- Hitung $Xbaru = X1 - f(X1)(X1 - X0)/f(X1) - f(X0)$.
- Jika nilai mutlak $(Xbaru - X1) <$ toleransi, diperoleh tulisan xbaru sebagai hasil perhitungan.
- jika tidak, lanjutkan ke langkah berikutnya.
- Jika jumlah iterasi $>$ iterasi maksimum, akhiri program.
- $X = Xbaru$, dan kembali ke langkah (2).

Contoh 1:

hitung akar persamaan dari :

$f(x) = x^3 - 3x - 20$,

Perkiraan awal

X 1= 6, f(6)=178

X 2= 2, f(2)=-18

iterasi pertama:

x3=2.3673469

f(x3)=-13.83464426

iterasi kedua:

X 1= 2 , f(2)=-18

X2=2.3673469, f(x2)=-13.83464426

x3= 2.3673469--13.83464426 =3.587438053

F(x3)= 15.40697963

Iterasi	X_1	X_2	X_3	$f(x_1)$	$f(x_2)$	$f(x_3)$
1	6	2	2.367346900	178	-18	-13.83464426
2	2	2.367346900	3.587438053	-18	-13.83464426	15.40697963
3	2.367346900	3.587438053	2.944590049	-13.83464426	15.40697963	-3.302376572
4	3.587438053	2.944590049	3.058058742	15.40697963	-3.302376572	-0.576057128
5	2.944590049	3.058058742	3.082034087	-3.302376572	-0.576057128	0.029936467
6	3.058058742	3.082034087	3.080849690	-0.576057128	0.029936467	-0.000248906
7	3.082034087	3.080849690	3.080859456	0.029936467	-0.000248906	-1.06044E-07

Contoh 2
hitung akar persamaan dari :

$y = x^3 + x^2 - 3x - 3$

dengan menggunakan metode secant, disyaratkan bahwa batas kesalahan relatif < 0.01%.

Hasil :

Iterasi	*x* 0	x1	x2	F(x0)	F(x1)	εa(%)
1	1	2	1,571429	-4	3	
2	2	1,571429	1,705411	3	-1,36443	7,856304
3	1,571429	1,705411	1,735136	-1,36443	-0,24775	1,713119
4	1,705411	1,735136	1,731996	-0,24775	0,029255	-0,18126
5	1,735136	1,731996	1,732051	0,029255	-0,00052	0,003137
6	1,731996	1,732051	1,732051	-0,00052	-1E-06	6,34E-06

Keuntungan: cepat konvergen

Kerugian: tidak selalu konvergen (bisa divergen)

Dalam fortran

```
IMPLICIT NONE
INTEGER, PARAMETER :: dpr=KIND(0.D0)
REAL(dpr), INTENT(IN) :: small
REAL(dpr), INTENT(OUT) :: x0,fx0
REAL(dpr) :: left,right,x1,x2,x3,fleft,fright,fx1,fx2,fx3,error


DO

 WRITE(*,*)
 WRITE(*,*) 'Masukkan batas kiri, kanan.'
 READ (*,*) left,right
 WRITE(*,*)

 fleft =problem(left)
 fright=problem(right)

 IF (fleft*fright > 0.0_dpr) THEN
  WRITE(*,*) 'Akar fungsi tidak berada dalam batas. Masukkan nilai batas
lain.'
 ELSE
  WRITE(*,*) 'Bagus! Akar fungsi berada dalam batas.'
  EXIT
 END IF

END DO

IF (fleft == 0.0_dpr) THEN
 x0 = left
 fx0=fleft
 RETURN
END IF

IF (fright == 0.0_dpr) THEN
 x0 = right
 fx0=fright
 RETURN
END IF
```

```
x1 =(left*fright-right*fleft)/(fright-fleft)
fx1=problem(x1)

IF (fleft*fx1 <= 0.0_dpr) THEN
 right = x1
 fright=fx1
ELSE
 left = x1
 fleft=fx1
END IF

x2 =(left*fright-right*fleft)/(fright-fleft)
fx2=problem(x2)

DO

 x3 =x2-fx2*(x2-x1)/(fx2-fx1)
 fx3=problem(x3)

 error=ABS(x2/x3-1.0_dpr)

 IF (error <= small) THEN
  x0 = x3
  fx0=fx3
  EXIT
 ELSE
  x1 = x2
  fx1=fx2
  x2 = x3
  fx2=fx3
 END IF

END DO

END
```

```
C Program: Solusi Persamaan Aljabar Non-Linier Tunggal (PANLT)
C dengan Metode 'SECANT'
C VARIAN: Program sederhana/Non-Subroutine
C Kondisi proses dinyatakan dalam variabel 'flag'
C flag = 0; berarti sistem masih dalam proses iterasi
C flag = 1; berarti proses telah mencapai konvergensi
C flag = 2; berarti jumlah iterasi maksimum telah terlampaui
C -----------------------------------------------------------
implicit none
REAL*8 eps,f,x,x0,x1
INTEGER flag,iter,maxiter
WRITE(*,'(A,$)') 'Harga-harga awal x0, x1 : '
READ(*,*) x0,x1
WRITE(*,'(A,$)') 'Jumlah iterasi maksimum : '
READ(*,*) maxiter
WRITE(*,'(A,$)') 'Epsilon/kriteria proses : '
READ(*,*) eps
iter = 0
flag = 0
DO WHILE(flag .EQ. 0)
   x = x1 - f(x1)*(x1 - x0)/(f(x1) - f(x0))
      IF (ABS(x - x1) .LE. eps) THEN
         flag = 1
      ELSEIF (iter .GT. maxiter) THEN
         flag = 2
      ELSE
            iter = iter + 1
            x0 = x1
            x1 = x
      ENDIF
ENDDO
WRITE(*,*) 'x0 = ',x0
WRITE(*,*) 'x1 = ',x1
WRITE(*,*) 'x = ',x
WRITE(*,*) 'f(x) = ',f(x)
WRITE(*,*) 'Flag = ',flag
WRITE(*,*) 'Jumlah iterasi = ',iter
STOP
END
FUNCTION f(x)
REAL*8 f,x
f = x - exp(1.0D0/x)
RETURN
END
```

METODE TERBUKA

AKAR GANDA

Akar ganda berpadanan dengan suatu titik dimana fungsi menyinggung sumbu x.

Misalnya, akar ganda–dua dihasilkan dari persamaan

$$f(x) = x^3 - 5x^2 + 7x - 3$$

$$f(x) = (x - 3)(x - 1)(x - 1)$$

$$\downarrow$$

$$x = 1$$

Akar ganda

- Akar ganda dua
- Akar ganda tiga
- Akar ganda empat
- Dan seterusnya

Penyelesaian akar ganda

- Ralston dan Rabinowitz (1978)

$$x_{i+1} = x_i - m \frac{f(x_i)}{f'(x_i)}$$

Kelemahan:
multiplisitas akar harus diketahui

Dimana m adalah bilangan multiplisitas akar

Misalnya : akar tunggal, m = 1

akar ganda dua, m = 2

akar ganda tiga, m = 3, dst

Penyelesaian akar ganda

- Ralston dan Rabinowitz mendefinisikan suatu fungsi baru yaitu:

$$u(x) = \frac{f(x)}{f'(x)}$$

yaitu untuk mengembangkan suatu bentuk alternatif dari metode Newton-Rapshon menjadi:

$$x_{i+1} = x_i - \frac{u(x_i)}{u'(x_i)}$$

Penyelesaian akar ganda

- Persamaan tersebut dideferensialkan untuk memberikan:

$$u'(x) = \frac{f''(x)f'(x) - f(x)f''(x)}{\left[f'(x)\right]^2}$$

dan setelah disubtitusikan ke persamaan semula menjadi:

Penyelesaian akar ganda

Metode Newton-Rapshon yang dimodifikasi untuk akar ganda

$$x_{i+1} = x_i - \frac{f(x_i)\,f'(x)}{\left[f'(x_i)\right]^2 - f(x_i)\,f''(x_i)}$$

Tugas minggu depan

- Buatlah program untuk mencari akar persamaan $f(x)=x^4-8,6x^3-35,51x^2+464x-998,46$ dengan menggunakan metode secant. Gunakan tebakan awal $x_{i-1}=7$ dan $x_i=9$, toleransi kesalahan 0.001%
- Lakukan perubahan dengan metode Newton Raphson dengan tebakan awal $x_i=7$
- Amati hasil program dengan 2 metode yang berbeda tersebut, tuliskan analisa anda dalam laporan