BAB 2

PENYELESAIAN PERSAMAAN NON LINEAR

METODE GRAFIK DAN TABULASI

A. Tujuan

a. Memahami Metode Grafik dan Tabulasi
b. Mampu Menentukan nilai akar persamaan dengan Metode Grafik dan Tabulasi
c. Mampu membuat program untuk menentukan nilai akar dengan Metode Grafik dan Tabulasi dengan Matlab

B. Perangkat dan Materi

a. Software Matlab
b. Metode Grafik

C. Dasar Teori

Fungsi f di sini adalah fungsi atau persamaan tak linear. Nilai $x = x_0$ yang memenuhi (1) disebut akar persamaan atau fungsi tersebut. Sehingga x_0 di sini menggambarkan fungsi tersebut memotong sumbu-x di $x = x_0$.

Persamaan atau fungsi f dapat berbentuk sebagai berikut:

a. **Persamaan aljabar atau polinomial**
   $f(x) = p_n(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$
b. **Persamaan transenden**
   Yaitu persamaan yang mengandung fungsi antara lain trigonometri, logaritma, atau eksponen
   Contoh: (i) $e^x + \cos(x) = 0$ (ii) $\ln(x) + \log(x^2) = 0$
c. **Persamaan campuran**
   Contoh: (i) $x^3 \sin(x) + x = 0$ (ii) $x^2 + \log(x) = 0$

Untuk polinomial derajat dua, persamaan dapat diselesaikan dengan rumus akar persamaan kuadrat. Misalkan bentuk persamaan kuadrat adalah: $ax^2 + bx + c = 0$ dapat dicari akar-akarnya **secara analitis** dengan rumus berikut.

$$x_{12} = \frac{-b \pm \sqrt{b^2 + 4ac}}{2a}$$

Untuk polinomial derajat tiga atau empat, rumus-rumus yang ada sangat kompleks dan jarang digunakan. Sedangkan untuk menyelesaikan polinomial dengan derajat yang lebih tinggi atau persamaan tak linear selain polinomial, tidak ada rumus yang dapat digunakan untuk menyelesaikannya. Metode Numerik memberikan cara-cara untuk menyelesaikan bentuk tersebut, yaitu **metode hampiran.** Penyelesaian numerik dilakukan dengan hampiran yang berurutan (**metode iterasi**), sedemikian sehingga setiap hasil adalah lebih teliti dari perkiraan sebelumnya. Dengan melakukan sejumlah proisedur iterasi yang dianggap cukup, akhirnya didapat hasil perkiraan yang mendekati hasil eksak (hasil yang benar) dengan toleransi kesalahan yang diijinkan.Metode iterasi mempunyai keuntungan bahwa umumnya tidak sangat terpengaruh oleh merambatnya error pembulatan.

a. LOKALISASI AKAR

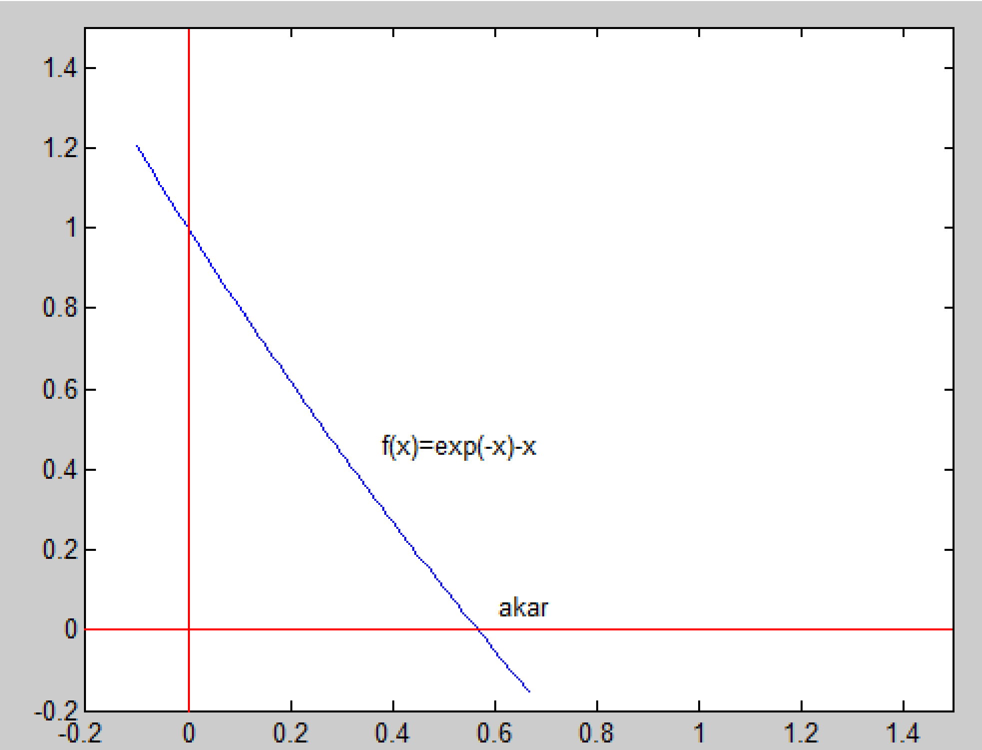
Lokasi akar persamaan tak linear diselidiki untuk memperoleh tebakan awal, yaitu:

(a) Cara grafik

Cara grafik ini dibedakan menjadi dua macam yaitu:

(i) Cara grafik tunggal

Misalkan $f(x) = exp(-x) - x$



Gambar-1

Gambar-1 bisa dibuat dengan Matlab sbb.

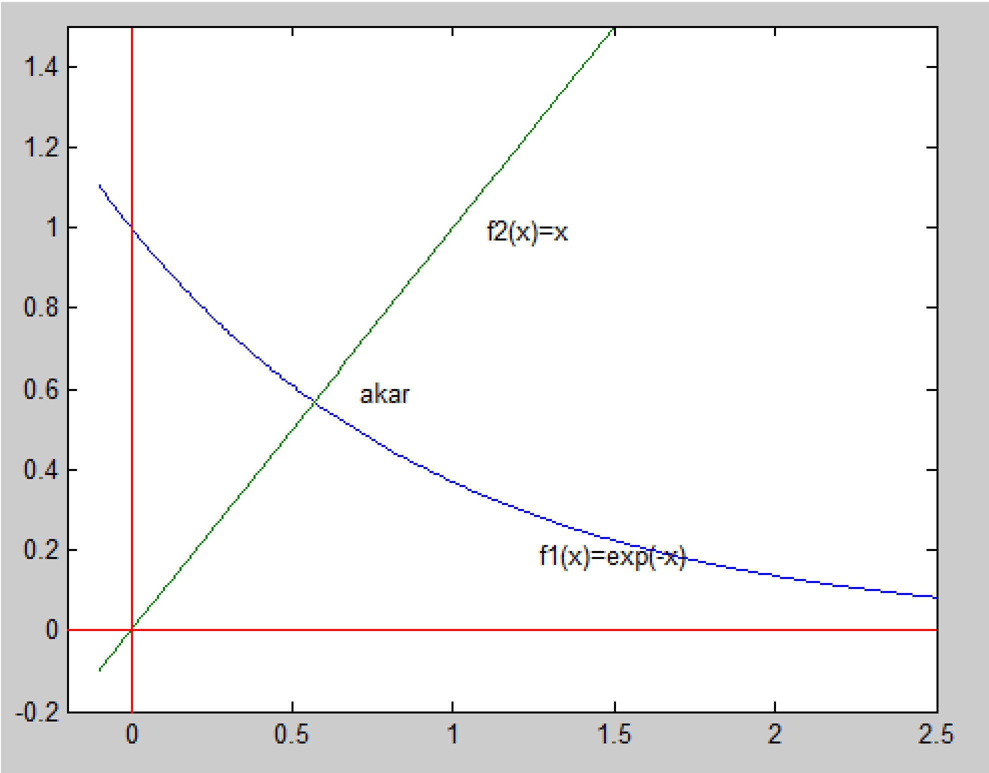
```
x=-0.1:0.01:0.67;

f=exp(-x) - x;

x1=-0.2:0.01:1.5;

y1=0.*x1;

y2=-0.2:0.01:1.5;x2=0.*y2;

plot(x,f,x1,y1,'r',x2,y2,'r');

axis([-0.2 1.5 -0.2 1.5]);

gtext('f(x)=exp(-x)-x');

gtext('akar');
```

Dari Gambar-1, terlihat bahwa fungsi $f(x) = exp(-x) – x$ memotong sumbu-x, yaitu $(x_0,0)$. Titik perpotongan tersebut, absisnya (nilai x_0) merupakan **akar** dari $f(x_0) = exp(-x_0) – x_0 = 0$.

(ii) Cara grafik ganda

Misalkan $f(x) = exp(-x) – x$ dan $f_1(x) = exp(x)$, $f_2(x) = x$,

maka $f(x) = f_1(x) - f_2(x)$



Gambar-2

Gambar-2 bisa dibuat dengan Matlab sbb.

```
x=-0.1:0.01:2.5;

f1=exp(-x);

f2=x;

x1=-0.2:0.01:2.5;

y1=0.*x1;

y2=-0.2:0.01:1.5;x2=0.*y2;

plot(x,f1,x,f2,x1,y1,'r',x2,y2,'r');

axis([-0.2 2.5 -0.2 1.5]);

gtext('f1(x)=exp(-x)');

gtext('f2(x)=x');

gtext('akar');
```

Dari Gambar-2, terlihat bahwa fungsi f_1 dan f_2 saling berpotongan, yaitu (x_0, y_0). Titik perpotongan tersebut, absisnya (nilai x_0) merupakan **akar** dari

$f(x_0) = exp(-x_0) – x_0 = 0.$

(b) Cara tabulasi

Nilai-nilai fungsi pada interval yang diminati dihitung dengan membagi interval tersebut menjadi sub interval – sub interval, dan nilai-nilai tersebut ditulis dalam bentuk tabulasi. Jika pada suatu interval nilai fungsi **berubah tanda**, maka pada interval tersebut ada akar.

Misalkan $f(x) = exp(-x) – x$, kemudian dibuat tabulasi dengan bantuan Matlab, yaitu:

```
fprintf(' x f(x) tanda\n');

fprintf('------------------\n');

i=1;beda=0.1;

for x=0:beda:1;

f=exp(-x) - x;

fprintf('%3.1f %6.3f',x,f);

if sign(f)< 0

tanda(i)='-';

fprintf(' %s\n',tanda(i));

else

if sign(f)> 0

tanda(i)='+';

fprintf(' %s\n',tanda(i));

else

tanda(i)='0';

fprintf(' %s\n',tanda(i));

end;

end;

i=i+1;

end;

i=1;
```

```
for x=0:0.1:1;
if tanda(i)=='0'
fprintf('Akarnya adalah = %6.4f\n',x);
else
if i >1
if tanda(i)~= tanda(i-1)
a=x-beda;
b=x;
fprintf('Akar ada di interval [%3.1f, %3.1f]\n', a,b);
end;
end;
end;
i=i+1;
end;
```

maka hasil tabulasinya adalah sbb.

```
x f(x) tanda
------------------
0.0 1.000 +
0.1 0.805 +
0.2 0.619 +
0.3 0.441 +
0.4 0.270 +
0.5 0.107 +
0.6 -0.051 -
0.7 0.203 -
0.8 -0.351 -
```

0.9 -0.493 -

1.0 -0.632 -

Akar ada di interval [0.5, 0.6]

Latihan.
Cari akar persamaan tak linier dengan metode garfik dan tabulasi :

1. $f(x) = 2x^3$
2. $f(x) = (2x+1)^3$

coba dikembangkan dengan mencari sendiri persamaan tak linier yang dapatdiselesaikan dengan metode grafik dan tabulasi.

METODE BISEKSI

A. Tujuan

a. Memahami Metode Biseksi
b. Mampu Menentukan nilai akar persamaan dengan Metode Biseksi
c. Mampu membuat program untuk menentukan nilai akar dengan Metode Biseksi dengan Matlab

B. Perangkat dan Materi

d. Software Matlab
c. Metode Biseksi

C. Dasar Teori

Metode Bisection (Setengah Interval)

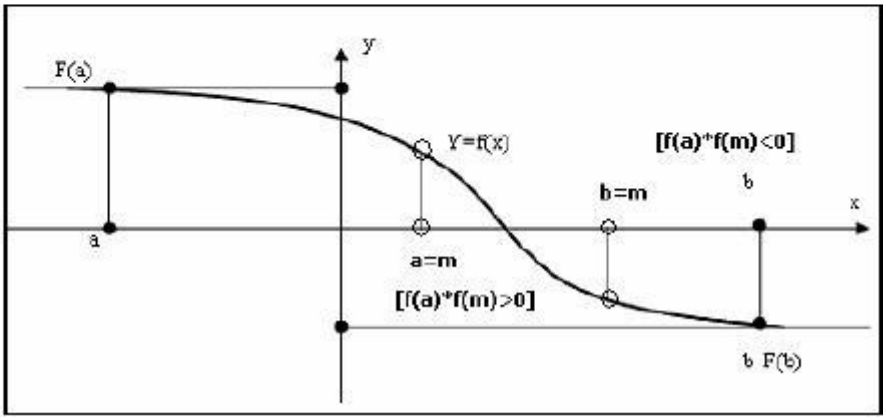
Landasan utama dari metode ini adalah menentukan suatu interval dalam suatu fungsi dimananilai fungsi dari ujung-ujungnya(batas bawah dan batas atas) harus berbeda tanda untuk menunjukkan bahwa fungsi tersebut memotong sumbu horisontal, kemudian interval tersebut dipecah menjadi dua bagian yang sama untuk mendekati titik potong dengan sumbu horisontal.

Di dalam aplikasinya, langkah awal yang dilakukan adalah menetapkan nilai sembarang a dan b sebagai batas bawah dan batas atas interval nilai fungsi yang dicari. Titik a dan b memberikan harga bagi fungsi f(x) untuk x = a dan x = b.

Selanjutnya adalah memeriksa apakah f(a).f(b)<0, jika demikian maka terdapat akar fungsi dalam interval yang ditinjau. Jika tidak, maka nilai a dan b ditetapkan lagi sedemikian rupa sehingga terpenuhi ketentuan perkalian f(a).f(b)<0, yaitu nilai f(a) dan f(b) mempunyai tanda yang berbeda.

Langkah selanjutnya adalah mencari nilai tengah interval a dan b dengan rumus m=(a+b)/2, lalu diperiksa apakah nilai mutlak f(m)< toleransi(misal 10^{-6}).

Jika benar, nilai x = m adalah solusi yang dicari (akar dari persamaan tersebut). Jika tidak terpenuhi, ditetapkan batasan baru dengan mengganti nilai b=m apabila f(a).f(m)<0, dan mengganti a=m bila f(a).f(m)>0 seperti terlihat pada Gambar 3.1 Proses menemukan mbaru dilakukan seperti prosedur yang telah dijelaskan.



Secara sederhana dari langkah-langkah yang dijelaskan di atas dapat disusun suatu algoritma program sebagai berikut:

1. Tentukan fungsi f(x), batas bawah a, batas atas b, toleransi, dan jumlah iterasi maksimum.
2. Hitung f(a) dan f(b).
3. Periksa apakah f(a).f(b)> 0; jika ya, keluar dari progam karena pada interval yang diberikan tidak terdapat akar persamaan.
4. Hitung nilai m = (a+b)/2.
5. Jika nilai mutlak f(m) < toleransi, tuliskan m sebagai hasil perhitungan, dan akhiri program; jika tidak, lanjutkan ke langkah berikutnya.
6. Jika jumlah iterasi > iterasi maksimum, akhiri program.
7. Jika f(a).f(m)<0, maka b=m, jika tidak, a = m.
8. Kembali ke langkah (2).

Contoh:

Persamaan, $f(x) = x^x - 5$. cari akarnya!

Langkah 1:

Nilai awal: batas bawah a=2, batas atas b=3, sehingga:

f(2)=22-5= -1
f(3)=33-5=22
m=(2+3)/2=2.5
f(2.5)=(2.5)2.5-5=|4.8821| > 0.000001

Langkah 2:

karena f(a).f(m)= (-1)(4.8821) < 0, maka b=m, sehingga:

a=2 _ f(2)=22-5= -1
b=2.5 _ f(2.5)=(2.5)2.5-5=4.8821
m=(2+2.5)/2=2.25
f(2.25)=(2.25)2.25-5=|1.2003|> 0.000001

Langkah 3:

karena f(a).f(m)= (-1)(1.2003) < 0, maka b=m, sehingga:

a=2 _ f(2)=22-5= -1
b=2.25 _ f(2.25)=(2.25)2.25-5=1.2003
m=(2+2.25)/2=2.125
f(2.125)=(2.125)2.125-5=|-0.0382|> 0.000001

Langkah 4:

karena f(a).f(m)= (-1)(-0.0382) > 0, maka a=m, sehingga:

a=2.125 _ f(2.125)=22.125-5= -0.0382

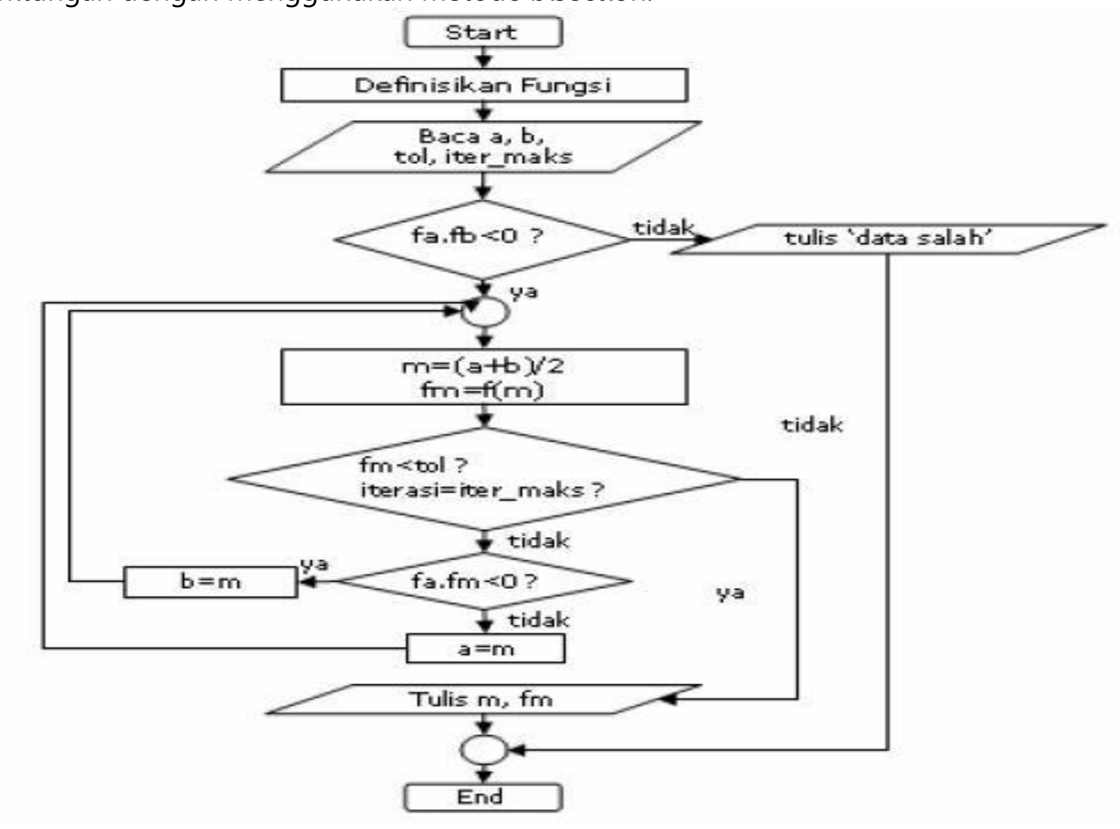
b=2.25 _ f(2.25)=(2.25)2.25-5=1.2003
m=(2.125+2.25)/2=2.1875
f(2.1875)=(2.1875)2.1875-5=|0.5416|> 0.000001

Langkah 5:
karena f(a).f(m)= (-0.0382)(0.5416) < 0, maka b=m,sehingga:
a=2.125 _ f(2.125)=22.125-5= -0.0382
b=2.1875 _ f(2.1875)=(2.1875)2.1875-5=0.5416
m=(2.125+2.1875)/2=2.1563
f(2.1563)=(2.1563)2.1563-5=|0.2430|> 0.000001

Seterusnya sampai didapatkan f(m)<toleransi yang disyaratkan. Langkah-langkah tersebut disusun

Langkah	a	f(a)	m	f(m)	b	F(b)
1	2	-1			3	22
			2.5	4.8821		
2	2	-1			2.5	4.8821
			2.25	1.2003		
3	2	-1			2.25	1.2003
			2.125	-0.0382		
4	2.125	-0.0382			2.25	1.2003
			2.1875	0.5416		
5	2.125	-0.0382			2.1875	0.5416
			2.1563	0.2430		

Dengan toleransi sebesar 10-6, pada langkah ke 23 iterasi berhenti dengan hasil
x = 2.12937256693840. Contoh di atas bisa memberikan gambaran bahwa metode bisection cukup mudah untuk dipergunakan. Di bawah ini gambar diagram alir untuk proses perhitungan dengan menggunakan metode bisection.



Implementasi dengan MATLAB:

```
function m= TengahInterval(f,a,b,n)

% f=fungsi,a=nilai awal,b=nilai akhir,n=jumlah iterasi
format long % format angka yang dipakai 15 digit di belakang koma
fa = f(a);
fb = f(b);
if fa*fb > 0.0 % jika nilai f(a) dan f(b) sama tanda
      error('pesan kesalahan:sama tanda')
end
for i=1:n
      m=(a+b)/2;
      y=f(m);
      disp([m y]) % menampilkan m dan f(m) ke layar
      if abs(y) <= 0.000001 % toleransi dipenuhi (akar persamaan 
ditemukan)
      break % menghentikan iterasi
      end
      if fa*y < 0
            b=m;
      else
            a=m;
      end
end
```

Cara menggunakan program di atas untuk menyelesaikan persoalan $f(x)=x^x-5$ dalam MATLAB, di command window ketikkan perintah:

>>f=inline('x^x-5') % mendefinisikan fungsi $f(x)=x^x-5$
>>x=TengahInterval(f,-1,3,15) % memanggil file tengahint.m a=-1,b=3, iterasi=15
dari dua perintah di atas MATLAB akan memberikan jawaban:
m y
1 -4
2 -1
2.50000000000000 4.88211768802618
2.25000000000000 1.20027091141992
2.12500000000000 -0.03821735673994
2.18750000000000 0.54161544380854
2.15625000000000 0.24250328354650
2.14062500000000 0.09992110005946
2.13281250000000 0.03030574169645
2.12890625000000 -0.00409119118124
2.13085937500000 0.01307328653945
2.12988281250000 0.00448256839976
2.12939453125000 0.00019357102259
2.12915039062500 -0.00194933919712
2.12927246093750 -0.00087801640156

x =
2.12927246093750

Metode Newton Raphson

A. Tujuan

d. Memahami Metode Newton Raphson
e. Mampu Menentukan nilai akar persamaan dengan Metode Newton Raphson
f. Mampu membuat program untuk menentukan nilai akar dengan Metode Newton Raphson dengan Matlab

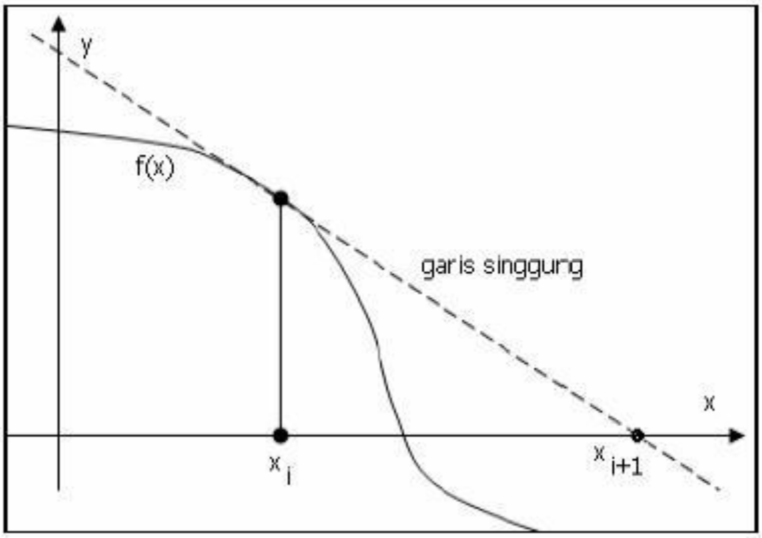
B. Perangkat dan Materi

d. Software Matlab
e. Metode Newton Raphson

C. Dasar Teori

Metode Newton Raphson

Metode yang lebih baik dalam memilih g'(x) adalah dengan membuat garis singgung dari f(x) untuk nilai x yang dipilih, dan dengan menggunakan besaran x dari perpotongan garis singgung terhadap absis sehingga diperoleh nilai xbaru. Metode ini diperlihatkan pada gambar berikut.



Garis singgung f(xi) memotong di x i+1.

Dari diagram di atas terlihat garis singgung terhadap f(x) adalah:

$$f'(x_i) = \frac{f(x_i) - 0}{x_i - x_{i+1}} \quad \text{atau} \quad f'(x_i) = \frac{f(x_i)}{x_i - x_{i+1}}$$

Sehingga

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \qquad \text{dimana i = 0, 1,2,....}$$

Metode ini dikenal dengan Metode Newton-Raphson dan merupakan salah satu cara yang paling dikenal dalam metode penyelesaian fungsi f(x)=0. Keuntungan cara ini adalah sifat konvergensi kuadratik dalam proses iterasi.

Contoh:
Carilah akar dari fungsi $f(x) = x^3 - 3x - 20$ maka $f'(x) = 3x^2 - 3$

Dengan demikian rumus untuk menentukan akarnya adalah :

$x_{i+1} = x_i - \frac{(x^3 - 3x - 20)}{(x^2 - 3)}$ Perkiraan Awal $x_0 = 5$

Langkah 1:

$$f(5) = 5^3 - 3(5) - 20 = 90$$
$$f'(5) = 3(5)^2 - 3 = 72$$
$$x_1 = 5 - \frac{90}{72} = 3.75$$

Langkah 2:

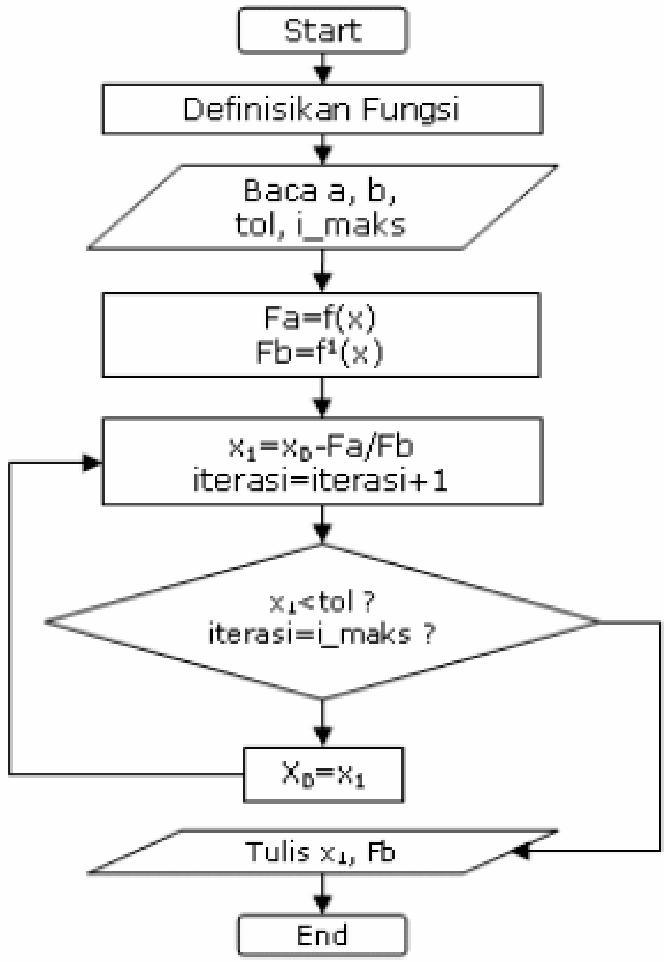
$$f(3.75) = 3.75^3 - 3(3.75) - 20 = 21.84844$$
$$f'(3.75) = 3(3.75)^2 - 3 = 39.1875$$
$$x_2 = 3.75 - \frac{21.4844}{39.1875} = 3.201754$$

Dan seterusnya

Algoritma program untuk metode Newton-Raphson

1. Tentukan fungsi, x0, toleransi, dan jumlah iterasi maksimum.
2. Hitung xbaru = x – f'(x0)/f(x0).
3. Jika nilai mutlak fxbaru < toleransi, diperoleh xbaru sebagai hasil perhitungan;
4. jika tidak, lanjutkan ke langkah berikutnya.
5. Jika jumlah iterasi > iterasi maksimum, akhiri program.
6. x = xbaru, dan kembali ke langkah (2).

Flow Chart Metode Newton Raphson



Implementasi dengan MATLAB

```
function x = MetodeNewton(f,x0,n,tol)
int i;
f0=inline(char(f));   % menyelesaikan persoalan f(x) = 0
                        dengan Metode Newton
g=inline(char(diff(f))); % dengan g sebagai fungsi
                           turunannya.

x = x0;
i=0; % perkiraan awal x dengan nilai x0
fa=f0(x);
while abs(fa) > tol % lakukan sampai toleransi tercapai
     fa=f0(x);
     fb=g(x);
     if fa == 0 or i=n
          return % program berhenti jika f(x) = 0
     end
     x = x - fa./fb; % rumus Newton
     disp([i x fa]) % fa = f(x);
     i=i+1;
end
```

Apabila di run dengan Command Window :

```
f ='x^3-3*x-20'
>> f0=inline(char(f))
f0 =
        Inline function:
    f0(x) = x^3-3*x-20
>> g=inline(char(diff(f)))
g =
    Inline function:h
    g(x) = 3*x^2-3
>> x=MetodeNewton(f,-2,20,0.000001)
         0    0.4444  -22.0000
    1.0000   -8.3806  -21.2455
    2.0000   -5.5715 -583.4706
    3.0000   -3.6161 -176.2331
    4.0000   -2.0583  -56.4347
    5.0000    0.2637  -22.5450
    6.0000   -7.1781  -20.7728
    7.0000   -4.7482 -368.3180
    8.0000   -3.0029 -112.8032
    9.0000   -1.4202  -38.0705
   10.0000    4.6785  -18.6037
   11.0000    3.5875   68.3674
   12.0000    3.1548   15.4078
   13.0000    3.0828    1.9339
   14.0000    3.0809    0.0487
   15.0000    3.0809    0.0000
   16.0000    3.0809    0.0000
x =

    3.0809
```