MODUL I SOLUSI PERSAMAAN NONLINEAR

Tujuan :

1. Dapat menghitung akar persamaan nonlinear dengan metode Biseksi, metode Newton Raphson dan metode Secant
2. Mencari besarnya kesalahan dari suatu perhitungan akar persamaan nonlinear dengan metode Biseksi, metode Newton Raphson, dan metode Secant

Petunjuk Praktikum :

1. Lengkapi penggal program di bawah ini serta cetak keluarannya.
2. Buatlah laporan praktikum. Adapun isi laporan meliputi :
    a. Program dan cetak keluarannya
    b. Pembahasan hasil/keluaran

Pendahuluan

Pencarian akar(penyelesaian) suatu persamaan non-linear, $y = f(x)$, adalah mencari suatu harga x^* , yang apabila disubstitusikan ke dalam persamaan itu, akan memberikan harga fungsi nol. Secara matematika

$$f(x^*) = 0$$

Sebagai contoh, akar dari persamaan $f(x) = x - e^{1/x}$ adalah 1.763223 karena, apabila harga $x^* = 1.763223$ disubstitusikan ke dalam persamaan itu, harga fungsi itu menjadi nol.

Metode-metode numeric yang banyak digunakan untuk menyelesaikan persamaan non-linear adalah metode biseksi, metode regula falsi, secant, newton –Raphson dan titik tetap. Tiga metode akan dibicarakan di sini adalah metode biseksi , metode Newton-Raphson dan metode Secant.

A. Metode Biseksi

Dalam metode Biseksi, interval yang mengandung akar dibagi menjadi dua secara berurutan hinggga ukuran interval mengecil dan akhirnya mencapai harga toleransi kesalahan yang diinginkan. Dalam interval [a,b] terdapat sebuah akar (yang akan dicari), apabila dipenuhi :

$$f(a) * f(b) \leq 0$$

Algoritma :

Masukan :

Batas kiri dan kanan interval, a dan b

Toleransi tol, Maksimum iterasi $maxit$

Fungsi, dinyatakan sebagai$fx(x)$

Keluaran :

Akar pendekatan, x_m

Proses :

1. $x_l = a, x_r = b$
2. Jika $fx(x_l) = 0$ maka x_l merupakan akar
3. Jika $fx(x_r) = 0$ maka x_r merupakan akar
4. Jika tanda $(fx(x_l))$ = tanda $(fx(x_r))$ maka bukan akar
5. n = $\frac{\ln(\frac{\Delta x}{\varepsilon})}{\ln 2}$
6. For i in range(n) :
7. $x_m = 0.5 * (x_l + x_r),\ fm = fx(x_m)$
8. $delta = |x_m - x_l|$
9. Jika $|fm| > |fx(x_r)|$ and $|fm| > |fx(x_r)|$ maka bukan akar
10. Jika $fm = 0$ maka x_m merupakan akar
11. Jika tanda $(fx(x_r))$ $\neq$ tanda $(fx(x_m))$ maka

    $x_l = x_m,\ fx(x_l) = fx(x_m)$

    Jika tidak

    $x_r = x_m\ , fx(x_r) = fx(x_m)$
12. Akar pendekatan $x_m = (x_l + x_r)/2$
13. Selesai

Tugas01 :

Diberikan fungsi $f(x) = e^x + x^2 - 3x - 2 = 0$terdapat sebuah akar riil dalam selang [-1.0, 1.0]. Carilah akar tersebut dengan metode Biseksi dengan toleransi kesalahan 1e-5.

Format Luarannya:

Pencarian Akar dari $f(x) = e^x + x^2 - 3x - 2 = 0$

Dengan Metode Biseksi

iter	x_l	x_r	x_m	$delta$	$f(x_l)$	$f(x_m)$	$f(x_l) * f(x_m)$

Akar pendekatannya : ……..

Dengan Toleransi : …….

B. Metode NEWTON-RAPHSON

Metode Newton Raphson, dalam mencari akar suatu fungsi nonlinear $y = f(x)$memerlukan evaluasi harga fungsi dan turunannya pada sembarang titik x yang merupakan harga awal tebakan akar fungsi tersebut. Metode ini didasarkan atas perluasan deret Taylor di sekitar suatu titik. Karenanya, apabila harga awal tebakan jauh dari akar sebenarnya, konvergensi akan lambat atau mungkin tidak dicapai sama sekali.

Algoritma :

Masukan : Fungsi, dinyatakan sebagai $fx(x)$

Turunan fungsi, dinyatakan sebagai $dfx(x)$

Harga Tebakan awal x_c

Toleransi eps, maksimum iterasi $maxit$

Keluaran : Akar pendekatan, x_{c+1}

Proses :

1. iter=0
2. Hitung $fx(x_c)$ dan $dfx(x_c)$
3. iter=iter+1
4. $x_{c+1} = x_c - \frac{fx(x_c)}{dfx(x_c)}$
5. $delta = |x_{c+1} - x_c|$
6. Jika$delta \leq eps$& iter>maxit maka Akar pendekatan = x_{c+1}, Selesai
7. $x_c = x_{c+1}$, kembali ke langkah 2

Tugas02:

Diberikan fungsi $f(x) = e^x + x^2 - 3x - 2 = 0$mempunyai akar riildalam selang [-1.0, 1.0]. Carilah akar tersebut dengan metode Newton Raphson dengan toleransi kesalahan 1e-5.

Format Luarannya:

Pencarian Akar Fungsi $f(x) = e^x + x^2 - 3x - 2 = 0$

Dengan Metode Newton Raphson

iter	x_c	$f(x_c)$	$delta$

Akar pendekatannya : ……..

Dengan Toleransi : …….

A. Penggal Program Biseksi:

```
‘’’

{*****************************************************************}

{          Program untuk Menghitung Akar Persamaan Nonlinear                    }

{                    dari fungsi : f(x) = x^2 -5                                 }

{                   dengan Metode  Biseksi                                       }

{                   Dibuat oleh :                                                }

{                            Nama          :                                     }

{                            NIM           :                                     }

{                            Prog.Studi    :                                     }

{*****************************************************************}

’’’

## module error
''' err(string).
   Prints ’string’ and terminates program
'''
import sys
def err(string):
   print(string)
   input('Press return to exit')
   sys.exit()

## module bisection
```

```
''' root = bisection(f,x1,x2,switch=0,tol=1.0e-9).
   Mencari akar dari f(x) = 0 dengan bisection.
   Akar harus dalam interval (x1,x2).
   Atur  switch = 1 returns root = None if
   f(x) increases upon bisection.
'''
import math
import error
from numpy import sign
def bisection(f,x1,x2,switch=1,tol=1.0e-9):
   f1 = f(x1)
   if f1 == 0.0: return x1
   f2 = f(x2)
   if f2 == 0.0: return x2
   if sign(f1) == sign(f2):
      error.err('Root is not bracketed')
   n = int(math.ceil(math.log(abs(x2 - x1)/tol)/math.log(2.0)))

   for i in range(n):
      x3 = 0.5*(x1 + x2); f3 = f(x3)
      if (switch == 1) and (abs(f3) > abs(f1)) \
                  and (abs(f3) > abs(f2)): return None
      if f3 == 0.0: return x3
      if sign(f2)!= sign(f3): x1 = x3; f1 = f3
      else: x2 = x3; f2 = f3
   return (x1 + x2)/2.0
```

selanjutnya :

```
##  Menggunakan metode Biseksi untuk mencari akar
from bisection import *
def f(x): return x**3 - 10.0*x**2 + 5.0
x = bisection(f, 0.0, 1.0, tol = 1.0e-4)
print('x =', '{:6.4f}'.format(x))
input("Press return to exit")
```

B. Penggal Program Newton Raphson:

```
{*************************************************************}
{          Program untuk Menghitung Akar Persamaan Nonlinear                    }
{                    dari fungsi : f(x) = x^2 -5                                }
{                   dengan Metode  Newton Rapshon                               }
{                   Dibuat oleh :                                               }
{                            Nama            :                                  }
{                            NIM             :                                  }
{                            Prog.Studi      :                                  }
{*************************************************************}
## module newtonRaphson
''' root = newtonRaphson(f,df,a,b,tol=1.0e-9).
   Mencari akmemanggil fungssi f(x) dan turunannya.
'''

def newtonRaphson(f,df,a,b,tol=1.0e-9):
```

```
   import error
   from numpy import sign

   fa = f(a)
   if fa == 0.0: return a
   fb = f(b)
   if fb == 0.0: return b
   if sign(fa) == sign(fb): error.err('Root is not bracketed')
   x = 0.5*(a + b)
   for i in range(30):
      fx = f(x)
      if fx == 0.0: return x

      if sign(fa) != sign(fx): b = x
      else: a = x

      # Mencoba langkah Newton Raphson
      dfx = df(x)
      # If division by zero, push x out of bounds
      try: dx = -fx/dfx
      except ZeroDivisionError: dx = b - a
      x = x + dx
      # If the result is outside the brackets, use bisection
      if (b - x)*(x - a) < 0.0:
         dx = 0.5*(b - a)
         x = a + dx
      # Check for convergence
      if abs(dx) < tol*max(abs(b),1.0): return x
   print('Terlalu banyak iterasi dalam Newton-Raphson')
```

Eksekusi program Newton Raphson :

```
#!/usr/bin/python
## Contoh kasus

def f(x): return x**4 - 6.4*x**3 + 6.45*x**2 + 20.538*x - 31.752
def df(x): return 4.0*x**3 - 19.2*x**2 + 12.9*x + 20.538

def newtonRaphson(x,tol=1.0e-9):
   for i in range(30):
      dx = -f(x)/df(x)
      x = x + dx
      if abs(dx) < tol: return x,i
   print(Terlalu banyak iterasi\n')

root,numIter = newtonRaphson(2.0)
print ('Root =',root)
print ('Number of iterations =',numIter)
```