LAPORAN PEMROGRAMAN AIDED

PENYELESAIAN PERSAMAAN NON-LINEAR



LATIHAN 4

Georgius Bagas Wicaksono

4210161025

PROGRAM STUDI TEKNOLOGI GAME

DEPARTEMEN TEKNOLOGI MULTIMEDIA KREATIF

POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

2018

**Praktikum I Metode Tabel**

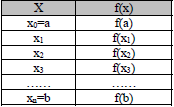
DASAR TEORI

Penyelesaian persamaan non-linear adalah penentuan akar-akar persamaan non linear dimana akar sebuah persamaan f(x) = 0 adalah nilai-nilai x yang menyebabkan nilai f(x) sama dengan nol. Persamaan f(x) adalah titik potong antara kurva f(x) dan sumbu x.

Theorema 1.1.

Suatu range *x=[a,b]* mempunyai akar bila *f(a) dan f(b)* berlawanan tanda atau memenuhi *f(a).f(b)<0*

Secara sederhana, untuk menyelesaikan persamaan non linier dapat dilakukan dengan menggunakan metode table atau pembagian area.Dimana untuk x = [*a*,*b*] atau x di antara *a* dan *b* dibagi sebanyak N bagian dan pada masing-masing bagian dihitung nilai *f(x)* sehingga diperoleh tabel :



Dari tabel ini, bila ditemukan *f(xk)=0* atau mendekati nol maka dikatakan bahwa *xk* adalah penyelesaian persamaan *f(xk)=0*.Bila tidak ada *f(xk)* yang sama dengan nol, maka dicari nilai *f(xk)* dan *f(xk+1)* yang berlawanan tanda, bila tidak ditemukan maka dikatakan tidak mempunyai akar untuk x = [*a*,*b*], dan bila ditemukan maka ada 2 pendapat untuk menentukan akar persamaan, yaitu :

1. Akar persamaan ditentukan oleh nilai mana yang lebih dekat, bila *|f(xk)|* ≤ *|f(xk+1)|* maka akarnya *xk*, dan bila *|f(xk+1)|<|f(xk)|* maka akarnya xk+1.

2. Akarnya perlu di cari lagi, dengan range x = [*xk* , *xk*+1]

.

Algoritma Metode Tabel :

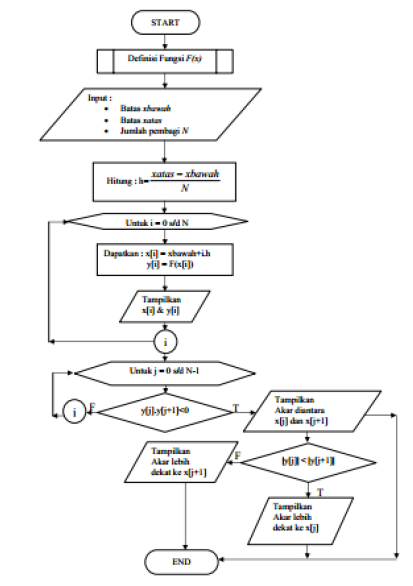
1. Definisikan fungsi f(x)
2. Tentukan range untuk x yang berupa batas bawah xbawah dan batas atas Xatas.
3. Tentukan jumlah pembagian N
4. Hitung step pembagi h



1. Untuk i = 0 s/d N dicari k dimana

* Bila f(Xk+1) = 0 maka Xk adalah penyelesaian
* Bila f(Xk+1) < 0 maka :
* Bila |f(Xk+1) maka Xk adalah penyelesaian
* Bila tidak Xk+1 adalah penyelesaian atau dapat dikatakan penyelesaian berada di antara Xk dan Xk+1

Flowchart



Listing Program

#include<stdio.h>

#include<math.h>

#define maks 10000

float a, b, n;

int i, j;

double h, x[maks], y[maks];

void input();

void tabel();

double f();

void akar();

main()

{

char jawab;

do{

input();

tabel();

akar();

fflush(stdin);

printf("\nMau menghitung lagi ? ");

scanf("%c", &jawab);

} while(jawab=='y' || jawab=='Y');

}

void input()

{

printf("\nMasukkan batas bawah : ");

scanf("%f", &a);

printf("Masukkan batas atas : ");

scanf("%f", &b);

printf("Masukkan jumlah pembagi n : ");

scanf("%f", &n);

h=(b-a)/n;

}

double f(double x)

{

return exp(-x)-x;

}

void tabel()

{

printf(" x\t\t f(x)\n");

for(i=0;i<n;i++)

{

x[i]=a+(i+1)\*h;

y[i]=f(x[i]);

printf(" %.3f\t\t%f\n", x[i], y[i]);

}

printf("\n");

}

void akar()

{

for(j=0;j<n;j++)

{

if(y[j]\*y[j+1]<0){

printf("Akar diantara %f dan %f\n", y[j], y[j+1]);

break;

}

}

if(fabs(y[j])<fabs(y[j+1]))

{

printf("Akar yang lebih dekat = %f\n", x[j]);

printf("Error = %f\n", fabs(y[j]));

}

else

{

printf("Akar yang lebih dekat = %f\n", x[j+1]);

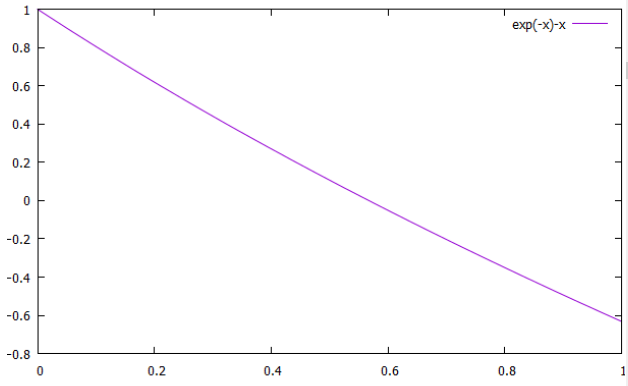
printf("Error = %f\n", fabs(y[j+1]));

}

}

Pengamatan awal

1. Gambar kurva fungsi F(x) = e-x-x dalam bentuk GNUPLOT

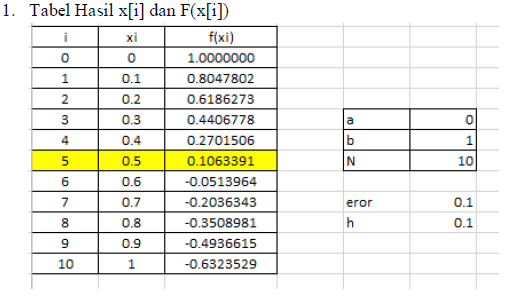


1. Perkiraan batas bawah dan batas atas

* Batas bawah = 0
* Batas atas = 1

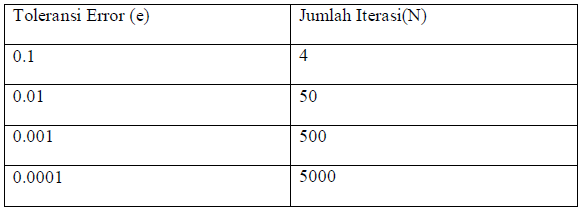
HASIL PERCOBAAN

1. Tabel Hasil x[i] dan F(x[i])



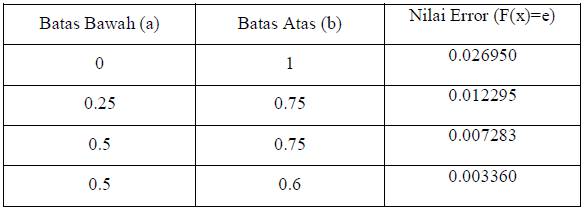
1. Pengamatan terhadap parameter
2. Toleransi error terhadap jumlah iterasi

* Tabel



Dari percobaan diatas, dapat diketahui bahwa semakin banyak jumlah iterasi (N) maka besaran toleransi akan semakin mendekat kepada 0. Dengan jumlah iterasi yang besar, nilai pembagi area (h) akan semakin kecil sehingga toleransi eror semakin mendekati angka 0.

1. Pengubahan nilai batas bawah dan batas atas dengan 20 iterasi



Dari percobaan tersebut dapat diketahui semakin sempit range batas atas dan bawah maka toleransi error akan mendekati angka 0.

KESIMPULAN

Setiap variabel yang dimasukkan ke percobaan mempengaruhi nilai toleransi error, baik jumlah iterasi, nilai batas atas, maupun batas bawah. Semakin banyak jumlah interasi maka jumlah toleransi eror akan semakin mendekati angka 0. Dan semakin kecil range antara batas atas dengan batas bawah, maka toleransi error akan semakin mendekati angka 0.

**Praktikum II Metode Biseksi**

Dasar Teori

Ide awal metode ini adalah metode tabel, dimana area dibagi menjadi N bagian.Hanya saja metode biseksi ini membagi range menjadi 2 bagian, dari dua bagian ini dipilih bagian mana yang mengandung dan bagian yang tidak mengandung akar dibuang.Hal ini dilakukan berulang-ulang hingga diperoleh akar persamaan.

Untuk menggunakan metode biseksi, terlebih dahulu ditentukan batas bawah (a) dan batas atas (b).Kemudian dihitung nilai tengah :

Dari nilai x ini perlu dilakukan pengecekan keberadaan akar. Secara matematik, suatu range terdapat akar persamaan bila f(a) dan f(b) berlawanan tanda atau dituliskan :

f(a) . f(b) < 0

Setelah diketahui dibagian mana terdapat akar, maka batas bawah dan batas atas di perbaharui sesuai dengan range dari bagian yang mempunyai akar.

Algoritma



Flowchart



Listing Program

#include<stdio.h>

#include<math.h>

float a, b, n, e, xr;

int iterasi=0, kondisi=1;

void input();

void biseksi();

double f();

main()

{

char jawab;

do{

input();

biseksi();

fflush(stdin);

printf("\nMau menghitung lagi ? ");

scanf("%c", &jawab);

} while(jawab=='y' || jawab=='Y');

}

void input()

{

printf("\nMasukkan batas bawah : ");

scanf("%f", &a);

printf("Masukkan batas atas : ");

scanf("%f", &b);

printf("Masukkan jumlah iterasi maksimal : ");

scanf("%f", &n);

printf("Masukkan error : ");

scanf("%f", &e);

}

double f(double x)

{

return exp(-x)-x;

}

void biseksi()

{

printf(" a\t \t b\t \t x\t\t f(x)\t\t f(a)\n");

if(f(a)\*f(b)<0)

{

while(kondisi==1)

{

iterasi+=1;

xr=(a+b)/2;

printf(" %f\t %f \t %f \t %lg\t %lg\n", a, b, xr, f(xr), f(a));

if(fabs(f(xr))<e || iterasi>n)

kondisi=0;

else if(f(a)\*f(xr)<0){

b=xr;

}

else{

a=xr;

}

}

printf("\n\nNilai xr [Akar]\t= %lg\n", xr);

printf("f(xr) [Error]\t= %lg\n", f(xr));

}

else{

printf("Tidak ada akar");

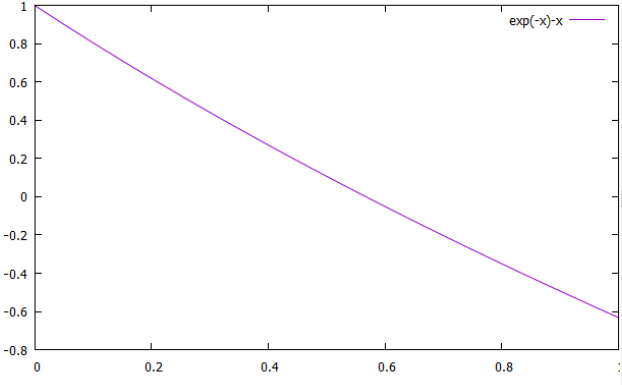
}

printf("\n");

}

PENGAMATAN AWAL

1. Gambar kurva fungsi F(x) = e-x-x dalam bentuk GNUPLOT



1. Perkiraan batas bawah dan batas atas

* Batas bawah = 0
* Batas atas = 1

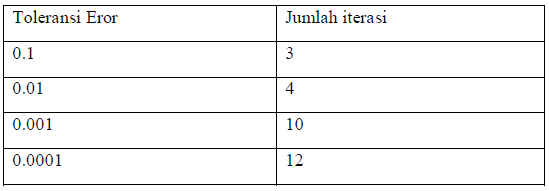
HASIL PERCOBAAN

1. Tabel Hasil Iterasi a, b, xr, f(xr)



1. Pengamatan terhadap parameter
2. Toleransi Error (e) terhadap jumlah iterasi (N)

* Tabel

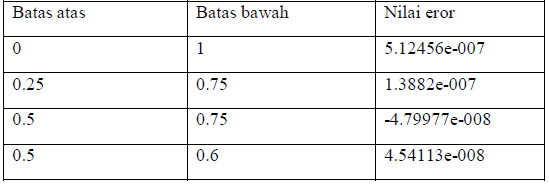


* Analisa

Semakin banyak jumlah iterasi (N) maka toleransi error akan semakin mendekati angka 0.

1. Pengubahan nilai batas bawah(a) dan batas atas terhadap 20 iterasi

* Tabel



* Analisa

Semakin sempit range antara batas bawah dan batas atas maka toleransi error akan semakin mendekati 0.

KESIMPULAN

Setiap variabel yang dimasukkan akan mempengaruhi nilai toleransi. Semakin kecil toleransi error yang ingin dicapai maka akan membutuhkan lebih banyak iterasi. Iterasi yang dibutuhkan lebih sedikit jika dibandingkan dengan metode tabel.

**Praktikum III**

**Metode Regula Falsi**

DASAR TEORI

Metode regula falsi adalah metode pencarian akar persamaan dengan memanfaatkan kemiringan dan selisih tinggi dari dua titik batas range. Seperti halnya metode biseksi, metode ini bekerja secara iterasi dengan melakukan update range.Titik pendekatan yang digunakan oleh metode regula-falsi adalah :



Dengan kata lain, titik pendekatan x adalah nilai rata-rata range berdasarkan F(x).

ALGORITMA



FLOWCHART



Listing Program

#include<stdio.h>

#include<math.h>

float a, b, n, e, xr;

int iterasi=0, kondisi=1;

void input();

void regulafalsi();

double f(double);

main()

{

char jawab;

do{

input();

regulafalsi();

fflush(stdin);

printf("\nMau menghitung lagi ? ");

scanf("%c", &jawab);

} while(jawab=='y' || jawab=='Y');

}

void input()

{

printf("\nMasukkan batas bawah : ");

scanf("%f", &a);

printf("Masukkan batas atas : ");

scanf("%f", &b);

printf("Masukkan jumlah iterasi maksimal : ");

scanf("%f", &n);

printf("Masukkan error : ");

scanf("%f", &e);

}

double f(double x)

{

return exp(-x)-x;

}

void regulafalsi()

{

printf("\ni\t a\t b\t\tx\t f(x)\t\tf(a)\n");

if(f(a)\*f(b)<0)

{

while(kondisi==1)

{

iterasi+=1;

xr=(f(b)\*a-f(a)\*b)/(f(b)-f(a));

printf("%d %f %f %f %f %f\n", iterasi, a, b, xr, f(xr), f(a));

if(fabs(f(xr))<e || iterasi>n)

kondisi=0;

else if(f(a)\*f(xr)<0){

b=xr;

}

else{

a=xr;

}

}

printf("\n\nNilai xr [Akar]\t= %lg\n", xr);

printf("f(xr) [Error]\t= %lg\n", f(xr));

}

else{

printf("Tidak ada akar");

}

printf("\n");

}

PENDAHULUAN

1. Gambar kurva fungsi f(x) = e-x-x dalam bentuk GNUPLOT



1. Perkiraan batas bawah dan batas atas

* Batas bawah = 0
* Batas atas = 1

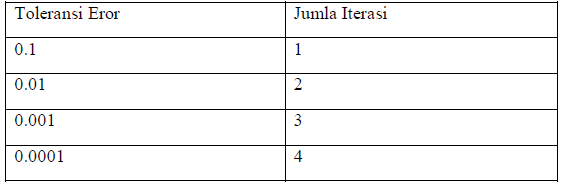
HASIL PERCOBAAN

1. Tabel iterasi a, b, xr, f(xr)



1. Penamaan terhadap parameter
2. Toleransi error terhadap jumlah iterasi

* Tabel

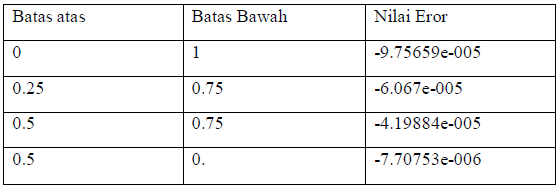


* Analisa

Semakin banyak iterasi maka toleransi error semakin mendekati 0

1. Oengubahan nilai batas atas dan batas bawah terhadap 20 iterasi

* Tabel



6

* Analisa

Semakin sempit range maka toleransi error semakin mendekati angka 0.

KESIMPULAN

Nilai error berbanding terbalik dengan jumlah iterasi dan berbanding lurus dengan jarak antara batas atas dan batas bawah. Semakin banyak iterasi, semakin kecil nilai error, dan semakin kecil jarak antara batas atas dan batas bawah dengan jumlah iterasi maka semakin kecil pula nilai error