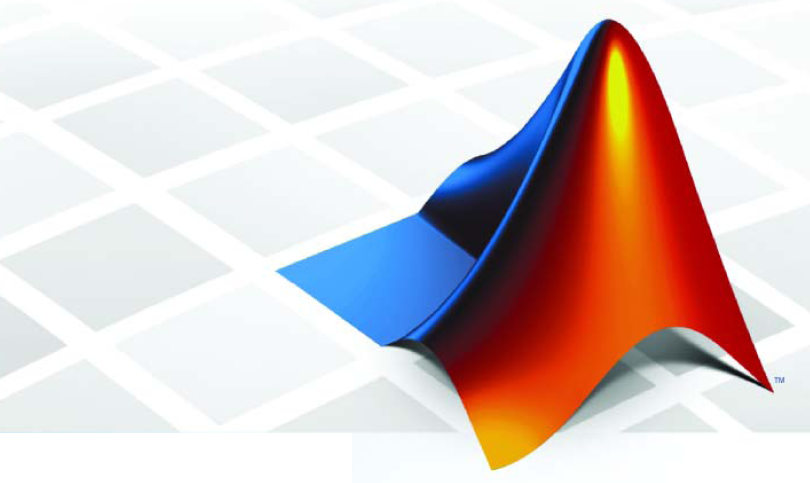
**MATLAB**

**PROGRAMMING**



Prepared by

Masrul huda(parvez)

EQUATION SOLUATION

Solve the equation:

Function [z]= func(x)

z=x.^3+2\*x.^2+log(x)+6\*x-32

end

**Bisection method**

function [root,nitteration]=bisec(x1,x2)

fx1=func(x1);

fx2=func(x2);

i=0;

if (fx1\*fx2)>0

error('reassign arguments according to bisection method')

end

while i>=0

i=i+1;

fx1=func(x1);

fx2=func(x2);

xnew=(x1+x2)/2;

fxnew=func(xnew);

if (fx1\*fxnew)>0

x1=xnew;

else

x2=xnew;

end

if abs(x1-x2)<0.001

break

end

end

root=xnew;

nitteration=i;

end

**Regula falsi method**

==

function[root nitteration]=rf(x1,x2)

fx1=func(x1);

fx2=func(x2);

i=0;

if fx1\*fx2>0

error('reassign arguments according to regulafalsi method')

end

while i>=0

i=i+1;

fx1=func(x1);

fx2=func(x2);

xnew=(x1-(fx1\*(x1-x2))/(fx1-fx2));

fxnew=func(xnew);

if fx1\*fxnew>0

x1=xnew;

else

x2=xnew;

end

if abs(x1-x2)<0.001

break

end

end

root=xnew;

nitteration=i;

end

**Fixed point method**

For this method we have to create a equation, which will find value of x

Then we have to create a function to find x

function [z]=xcalc(x)

Z=(1/6)\*(32-x.^3-2\*x.^2-log(x);

end

function [root nitteration]=fp(x)

i=0;

while i>=0

i=i+1;

xold=x;

xnew=xcalc(x);

x=xnew;

if abs(xold-xnew)<0.001

break

end

end

root=x;

nitteration=i;

end

**Newton raphson method**

For this method we have find out differential of given function

function [z]= dfunc(x)

z=3\*x.^2+4\*x+(1/x)+6;

end

function[root nitteration]=nr(x)

i=0;

while i>=0

i=i+1;

xold=x;

fx=func(x);

dfx=dfunc(x);

xnew=x-(fx/dfx);

x=xnew;

if abs(xold-xnew)<0.001

break

end

root=x;

nitteration=i;

end

**Secant method**

function[root nitteration]=sec(x1,x2)

i=0;

while i>=0

i=i+1;

fx1=func(x1);

fx2=func(x2);

dfx=(fx1-fx2)/(x1-x2);

x3=x2-(fx2/dfx);

x1=x2;

x2=x3;

if abs(x1-x2)<0.001

break

end

end

root=x3;

nitteration=i;

end

Grading

[A,B]=xlsread('Che310','sheet1','A3:C11');

marks=A(:,3);

marks=round((marks/300)\*100);

a=length(marks);

for i=1:a

if marks(i,1) >=80

GRADE(i,1)={'A+'};

GPA(i,1)=4;

elseif marks(i,1) >=75

GPA(i,1)=3.75;

GRADE(i,1)={'A'};

elseif marks(i,1) >=70

GPA(i,1)=3.5;

GRADE(i,1)={'A-'};

elseif marks(i,1) >=65

GPA(i,1)=3.25;

GRADE(i,1)={'B+'};

elseif marks(i,1) >=60

GPA(i,1)=3;

GRADE(i,1)={'B'};

elseif marks(i,1) >=55

GPA(i,1)=2.75;

GRADE(i,1)={'B-'};

elseif marks (i,1)>=50

GPA(i,1)=2.50;

GRADE(i,1)={'C+'};

elseif marks(i,1) >=45

GPA(i,1)=2.25;

GRADE(i,1)={'C'};

elseif marks(i,1)>=40

GPA(i,1)=2;

GRADE(i,1)={'D'};

else

GPA(i,1)=0;

GRADE(i,1)={'F'};

end

end

xlswrite('che310',GPA,'sheet1','D4:D11')

xlswrite('che310',GRADE,'sheet1','E4:E11')

xlswrite('che310',{'GPA'},'sheet1','D3')

xlswrite('che310',{'GRADE'},'sheet1','E3')

**Ordinary differential equation solve**

function [z]=dfx(x,y)

z=x-2\*y;

end

**Euler method**

function[u]=elr(xf,x0,y0,h)

x=x0:h:xf;

n=length(x);

for i=1:n-1

ynew=y0+dfx(x0,y0)\*h;

y0=ynew;

x0=x0+h;

end

u= ynew;

end

**Modified euler method**

function[u]=meu(xf,x0,y0,h)

x=x0:h:xf;

n=length(x);

for i=1:n-1

dfx1=dfx(x0,y0);

ynew=y0+dfx1\*h;

xnew=x0+h;

dfx2=dfx(xnew,ynew);

dfxac=(dfx1+dfx2)/2;

yac=y0+dfxac\*h;

y0=yac;

x0=xnew;

end

u=yac;

end

**runge katta method**

function[u]=rkm(xf,x0,y0,h)

x=x0:h:xf;

n=length(x);

for i=1:n-1

k1=dfx(x0,y0);

k2=dfx(x0+0.5\*h,y0+.5\*k1\*h);

k3=dfx(x0+0.5\*h,y0+.5\*k2\*h);

k4=dfx(x0+h,y0+k3\*h);

y=y0+(1/6)\*(k1+2\*k2+2\*k3+k4)\*h;

y0=y;

x0=x0+h;

end

u=y;

end

**Numerical intregration**

function[z]=fx(x)

z=10\*(sin(x)+exp(x));

end

**Trapezoidal rule**

Formula for trapezoidal method

function[intregration]= trapi(x0,xn,n)

h=(xn-x0)/n;

x=(x0+h):h:xn;

a=0;

for i=1:(n-1)

a=a+2\*fx(x(i));

end

intregration=(.5\*h)\*(fx(x0)+a+fx(xn));

**Simpson 1/3**

function[intregration ]= simone(x0,xn,n)

check=n/2;

if fix(check)~=check

error('divison number must be multiple of 2')

end

h=(xn-x0)/n;

x=(x0+h):h:xn;

a=0;

for i=1:2:(n-1)

a=a+4\*fx(x(i));

end

for i=2:2:(n-2)

a=a+2\*fx(x(i));

end

intregration=(h/3)\*(fx(x0)+a+fx(xn));

**Simpson 3/8**

function[intregration ]= simthree(x0,xn,n)

check=n/3;

if fix(check)~=check

error('divison number must be multiple of 3')

end

h=(xn-x0)/n;

x=(x0+h):h:xn;

a=0;

for i=1:3:(n-1)

a=a+3\*fx(x(i));

end

for i=2:3:(n-1)

a=a+3\*fx(x(i));

end

for i=3:3:(n-3)

a=a+2\*fx(x(i));

end

intregration=((3\*h)/8)\*(fx(x0)+a+fx(xn));

**Weddle method**

function[intregration ]= weddle(x0,xn,n)

check=n/6;

if fix(check)~=check

error('divison number must be multiple of 6')

end

h=(xn-x0)/n;

x=(x0+h):h:xn;

a=0;

for i=1:6:(n-5)

a=a+5\*fx(x(i));

end

for i=2:6:(n-4)

a=a+fx(x(i));

end

for i=3:6:(n-3)

a=a+6\*fx(x(i));

end

for i=4:6:(n-2)

a=a+fx(x(i));

end

for i=5:6:(n-1)

a=a+5\*fx(x(i));

end

for i=6:6:(n-6)

a=a+2\*fx(x(i));

end

intregration=((3\*h)/10)\*(fx(x0)+a+fx(xn));

**Simultaneous equation solve**

* **Forward elimination**
* **Backward elimination**
* **Gauss zordan elimination**

**Forward elimination**

a=input('co-efficient matrix');

b=input('constant matrix');

n=length(a);

for i=1:n-1

for j=i+1:n

m=a(j,i)/a(i,i);

a(j,:)=a(j,:)-m\*a(i,:)

b(j)=b(j)-m\*b(i);

end

end

x=zeros(n,1);

x(n)=b(n)/a(n,n);

for k=(n-1):(-1):1

x(k)=(b(k)-a(k,k+1:n)\*x(k+1:n))/a(k,k);

end

disp(x)

**backward elimination**

a=input('coefficient matrix');

b=input('constant matrix');

n=length(a);

for i=n:-1:2

for j=i-1:-1:1;

m=a(j,i)/a(i,i);

a(j,:)=a(j,:)-a(i,:)\*m;

b(j)=b(j)-b(i)\*m;

end

end

x=zeros(n,1);

x(1)=b(1)/a(1,1);

for k=2:n

x(k)=(b(k)-a(k,1:k-1)\*x(1:k-1))/a(k,k);

end

disp(x)

**Gauss zordan elimination**

a=input('co-efficient matrix');

b=input('constant matrix');

n=length(a);

for i=1:n-1

for j=i+1:n

m=a(j,i)/a(i,i);

a(j,:)=a(j,:)-m\*a(i,:)

b(j)=b(j)-m\*b(i)

end

end

for i=n:-1:2

for j=i-1:-1:1;

m=a(j,i)/a(i,i);

a(j,:)=a(j,:)-a(i,:)\*m;

b(j)=b(j)-b(i)\*m;

end

end

for k=1:n

x(k)=b(k)/a(k,k);

end

disp(x)

**Curve fitting**

* Linear curve fitting
* Polynomial curve fitting

Linear curve fitting

x=input('enter value of x')

y=input('enter value of y')

sxx=sum(x.\*x);

syy=sum(y.\*y);

sxy=sum(x.\*y);

sx=sum(x);

sy=sum(y);

n=length(x);

intercept=(sxx\*sy-sxy\*sx)/(n\*sxx-sx.^2)

slope=(n\*sxy-sx\*sy)/(n\*sxx-sx.^2)

ymean=mean(y);

ssy=0;

sse=0;

for i=1:n

ssy=ssy+(y(i)-ymean).^2;

end

for i=1:n

sse=sse+(y(i)-slope\*x(i)-intercept).^2;

end

Rsqure=(ssy-sse)/ssy

x=x';

y=y';

A=zeros(n,2);

A(:,1)=x;

A(:,2)=1;

coefficient=inv(A'\*A)\*A'\*y

**Polynomial curve fitting**

x=input('enter value of x')

y=input('enter value of y')

p=input('order of polynomial')

x=x';

y=y';

n=length(x);

A=zeros(n,p+1);

j=0;

for power=p:-1:0

j=j+1;

A(:,j)=x.^power;

end

coefficient=inv(A'\*A)\*A'\*y

*Author:*

Md. Masrul Huda (Parvez)

Student no: 0702039

“This tutorial might have some mistakes due to Unconsciousness or typing error”