MATH 6601

PROJECT – 3

MUHAMMED EMIN

OZTURK

QUESTION -1

A – Using plotting try to find change of sign intervals for the four solutions



This is graph of f(x) = x4 + 2x3 -14x2  + 2x + 1

Roots of this equation



We can choose 4 interval such that

I1 = [ -6 -4 ]

I2 = [ -1 0 ]

I3 = [ 0 1 ]

I4 = [ 1 3 ]



Question 1-b )



Question 1-c

syms x;

x0=2;

N=1000;

tol= 10^(-10);

f2=@(x) (4\*x^3 + 6\*x^2 - 28\*x + 2);

disp(['Execution of problem 1-c by Newton:']);

[X,err]=Newton(x,sym(f2),x0,tol,N);

Result

Execution of problem 1-c by Newton:

We have converged to the root, r = 1.9531 in 4 steps.

Question 2

Xk+1 = Xk - DF( Xk )-1 F(Xk) ( Newton methods )

DF(Xk) is jacobian matrix of F(Xk)

We can form F(Xk) for this problem such that

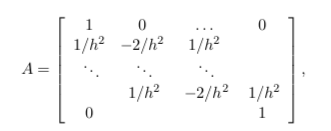
Au = b ;

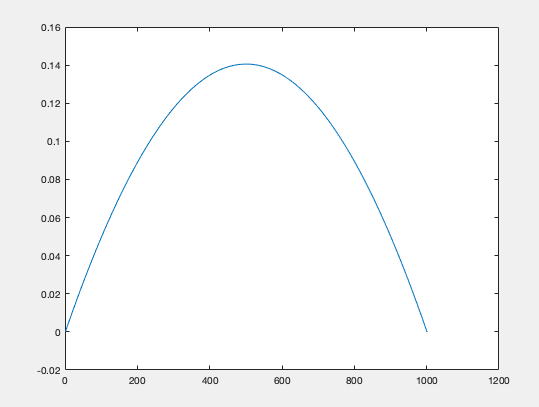
F(U) = Au – b ;

F(Uk) = Auk – bk ; such that b = - lamda \* eU

Uk+1 = Uk – DF( Uk )-1 F( Uk )

Solution Graph





It takes 4 iteration to be converged

Codes for question -2

Main

close all;clc;clear

%set up the mesh

n= 1001 ;

h= 1/ (n-1) ;

%x= [0:h:1] ;

%A = diag(ones(N-1,1)) + diag(ones(N-1,1),2) - 2\*diag(ones(N,1),1);

%A[2:N-1;2:N-1] =diag(ones(N-1,1)) + diag(ones(N-1,1),2) - 2\*diag(ones(N,1),1);

u = zeros(n,1);

tol= 10^(-10);

f=myf(u);

Jacobi=myjacobi(u);

Nmax=10000;

[X,err]=MyNewton(f,Jacobi,u,tol,Nmax);

X

xgraph=linspace(1,1001);

ygraph=X;

figure

plot(ygraph);

Function Myjacobi

function J1=myjacobi(u)

n = length(u);

h= 1/(n-1) ;

J1=zeros(n,n) ;

J1(1,1)=1 ;

for i=2:n-1

J1(i,i-1) = 1/h^2;

J1(i, i) = -2/h^2 + exp(u(i));

J1(i,i +1) =1/h^2;

end

J1(n,n) = 1;

end

Function myf

function [ufun]=myf(u)

n = length(u);

h=1/(n-1) ;

ufun=zeros(n,1) ;

ufun(1)= 0;

for i=2:n-1

ufun(i) = ( u(i-1) -2\*u(i) + u(i+1))/h^2 + exp(u(i));

end

ufun(n) = 0;

end

Function myNewton

function [uout,err]=MyNewton(f,Df,u,tol,N)

% Input Arguments:

% - x, the symbolic variable we are passing the function of

% - f, the symbolic function we have in terms of x

% - x0, the initial guess, starting point of our iteration

% - tol, the desired tolerance up to which we accept our solution

% - N, the max number of iterations reached in case of NOT convergence

% Output Arguments:

% - X is the vector of iterates

% - err is the vector of all errors at each iteration

flag=1; % case of insuccess

n=length(u);

%uout=zeros(n,1);

err=zeros(n,1); % initialize the output to zero

uout=u; % first entry of the array X of iterates is the initial guess

%previousu=zeros(n,1);

jacobiMatrix=Df;

functionF=f;

maxe=0;

for i=1:N

previousu=uout;

uout= uout - inv(jacobiMatrix)\*functionF;

err=abs(uout - previousu);

disp(['it is in ',num2str(i),' steps.']);

maxe = max(err);

if (maxe<tol)

%this is the check of convergence of our iterations. In

%positive case, we change the value of our flag variable to 0

flag=0;

%root=X(i+1);

disp(['We have converged to the root, r = in ',num2str(i),' steps.']);

break;

elseif(i==N)

disp('We have NOT converged. We have reached the maximum number of steps allowed.');

break;

end

jacobiMatrix=myjacobi(uout);

functionF=myf(uout);

end

end