Jared Rivera 19 April 2017 804603106

CEE 103 HW#2 Scripts

%HW2 P1 Bisection method

clear all; clc; clear all;

a=0;

b=2;

i=1;

tol=10^(-4);

while abs(a-b)>tol && i<1000

x(i)=(a+b)/2;

[res]=funca(a);

[res\_new]=funca(x(i));

if sign(res)==sign(res\_new)

a=x(i);

else

b=x(i);

end

i=i+1;

end

%HW2 P1 Newton's method

clear all; clc; clear all;

x(1)=1; %for prob a

%x(1)=1; %for prob b

tol=10^(-6);

for i=1:100

[res,dres]=funca(x(i));

if abs(res)<tol

break

else

dx=-res/dres;

end

x(i+1)=x(i)+dx;

end

%HW2 P1 Secant method

clear all; clc; clear all;

x(1)=0;

x(2)=2;

tol=10^(-6);

for i=1:100

xn=x(i+1);

x0=x(i);

[res]=funca(xn);

dres=(funca(xn)-funca(x0))/(xn-x0);

if abs(res)<tol

break

else

dx=-res/dres;

end

x(i+2)=x(i+1)+dx;

end

%HW2 P3

%Modified Newton's Method

%Includes a quadratic Taylor Expansion rather than a linear one.

clear all; clc; close all;

x(1)=1;

tol=10^(-12);

for i=1:100

[res,dres,d2res]=func3(x(i));

if abs(res)<tol

break

elseif dres^2-2\*res\*d2res>0

dx1=-(dres+sqrt(dres^2-2\*res\*d2res))/d2res;

dx2=-(dres-sqrt(dres^2-2\*res\*d2res))/d2res;

diffx1=dx1+res/dres;

diffx2=dx2+res/dres;

if diffx2>diffx1

dx=dx1;

else

dx=dx2;

end

else

dx=-res/dres;

end

x(i+1)=x(i)+dx;

end

function [ y,dy ] = funca( x )

%Function for problem 1 part a

y=x^3-x^2-x-1;

dy=3\*x^2-2\*x-1;

end

function [ y,dy ] = funcb( x )

%Function for problem 1 part b

y=sin(x)-exp(-x);

dy=cos(x)+exp(-x);

end

function [ y,dy,d2y ] = func3( x )

%Function provided for problem 3

y=exp(x)-x^4+x\*exp(x);

dy=2\*exp(x)-4\*x^3+x\*exp(x);

d2y=3\*exp(x)-12\*x^2+x\*exp(x);

end