Homework 7

AE\_5830 Dr. Hosder

Matt Pahayo

2021

Text, letter

Description automatically generated

**Results**

Q1:

**Table 1 for Question 1**



Chart

Description automatically generated

The Simpson 1/3 method has a larger slope than the Trapezoidal method. The error thus decreases more as the mesh size is decreased.

Q2:

Text

Description automatically generated with medium confidence



Q3:

Text, letter

Description automatically generated



Q4:

Graphical user interface, text, application

Description automatically generated

Text, letter

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated



Text

Description automatically generatedChart, line chart

Description automatically generated

Chart, line chart

Description automatically generated



**Appendix A – code**

% main.m

clc

clear all

close all

format longg

%=========================================================================

%q1

%=========================================================================

xstart = 1;

xend = 2;

xstart = 0;

xend = pi;

Trap = zeros(20,1);

Simp = Trap;

meshSize = Simp;

I = pi\*ones(20,1);

i = 1;

for intervals = 2:2:40

[range,xpoints] = getRange(xstart,xend,intervals);

h = xpoints(2)-xpoints(1);

meshSize(i,1) = h;

Trap(i,1) = Trapz(h,intervals,range);

Simp(i,1) = Simp13(h,intervals,range);

i = i + 1;

end

errTrap = abs(I-Trap);

errSimp = abs(I-Simp);

hold on

set(gca, 'XScale', 'log', 'YScale', 'log');

loglog(meshSize,errTrap)

loglog(meshSize,errSimp)

legend('Trapezoidal','Simpson 1/3','location','northwest')

title('Error vs.Mesh size')

ylabel('error')

xlabel('mesh size')

grid on

hold off

%=========================================================================

%q2

%=========================================================================

% Iquad3 = gaussQuad(@f,0,pi,3);

% Iquad4 = gaussQuad(@f,0,pi,4);

%=========================================================================

%q3

%=========================================================================

I = 0.0158434;

syms y % start with x first

[range,xpoints]=getRange2(@f2,0,.5,2);

h = xpoints(2)-xpoints(1);

g(y) = Simp13(@f2,h,2,range);

[range,xpoints]=getRange(g,0,.5,2);

h = xpoints(2)-xpoints(1);

Isimp = double(Simp13(g,h,2,range))

g(y) = gaussQuad(@f2,0,.5,3);

Iquad = double(gaussQuad(g,0,.5,3))

errSimp = abs(Isimp - I)

errQuad = abs(Iquad - I)

%=========================================================================

%q4

%=========================================================================

syms w

f(w) = 1/5\*w\*exp(3\*w)-1/25\*exp(3\*w)+1/25\*exp(-2\*w)

xi = 0;

yi = 0;

h = .1;

steps = 11;

[xh,yh] = Heun(xi,yi,h,steps);

[xe,ye] = Euler(xi,yi,h,steps);

[xme,yme] = mEuler(f,xi,yi,h,steps);

[xRK,yRK] = RK4(xi,yi,h,steps);

[xmid,ymid] = midpoint(xi,yi,h,steps);

xlin = 0:.1:1

exact = transpose(f(xlin))

figure(1)

hold on

fplot(f)

plot(xh,yh)

plot(xe,ye)

plot(xme,yme)

plot(xmid,ymid)

ylim([yi 5])

xlim([xi 1])

legend('exact','Heun','Euler','modified Euler','midpoint')

grid on

title('ODE approximations')

xlabel('t')

ylabel('f(t)')

hold off

figure(2)

hold on

plot(xh,yh-exact)

plot(xe,ye-exact)

plot(xme,yme-exact)

plot(xmid,ymid-exact)

ylim([yi 5])

xlim([0 1])

legend('Heun','Euler','modified Euler','midpoint')

grid on

title('ODE error')

xlabel('t')

ylabel('error')

hold off

figure(3)

hold on

fplot(f)

plot(xRK,yRK)

ylim([yi 4])

xlim([xi 1])

legend('exact','RK4')

grid on

title('ODE approximations')

xlabel('t')

ylabel('f(t)')

hold off

figure(4)

hold on

plot(xRK,yRK-exact)

legend('RK4')

grid on

title('ODE error')

xlabel('t')

ylabel('error')

hold off

%=========================================================================

%q5

%=========================================================================

syms w

xi = 1;

yi = 1;

h = .1;

steps = 11;

xlin = 1:.1:2

f(w) = w/(1+log(w))

[x2,y2] = AB2(xi,yi,h,steps);

[x3,y3] = AB3(xi,yi,h,steps);

exact = transpose(f(xlin))

figure(1)

hold on

fplot(f)

plot(x2,y2)

plot(x3,y3)

ylim([yi 1.25])

xlim([xi 2])

legend('exact','AB2','AB3')

grid on

title('AB approximations')

xlabel('t')

ylabel('f(t)')

hold off

figure(2)

hold on

plot(x2,y2-exact)

plot(x3,y3-exact)

ylim([-.005 .005])

xlim([xi 2])

legend('AB2','AB3')

grid on

title('AB error')

xlabel('t')

ylabel('error')

hold off

function I = Trapz(h,intervals,range)

% Trapezoidal integration

sum = 0;

for i = 1:intervals

sum = range(i)+range(i+1)+sum;

end

I = h\*sum/2;

end

function I = Simp13(h,intervals,range)

% Simpson 1/3 integration

sum = 0;

for i = 1:2:intervals

sum = range(i) + 4\*range(i+1)+range(i+2)+sum;

end

I = h\*sum/3;

end

function I = gaussQuad(f,a,b,n)

%myFun - Description

%

% Syntax: output = myFun(input)

%

% Long description

switch n

case 2

c = ones(2,1);

t = [-.57735;.57735];

case 3

c = [.555556;.888889;.555556];

t = [-.774597;0;.774597];

case 4

c = [.347855;.652145;.652145;.347855];

t = [-.861136;-.339981;.339981;.861136];

otherwise

disp('nope')

end

x = zeros(n,1);

for i = 1:n

x(i,1) = (b+a)/2-(b-a)/2\*t(i);

end

I = 0;

for i = 1:n

I = (b-a)/2\*c(i)\*f(x(i)) + I;

end

function [range,xpoints] = getRange(xstart,xend,intervals)

%UNTITLED3 Summary of this function goes here

% Detailed explanation goes here

xpoints = transpose(xstart:(xend-xstart)/(intervals):xend);

range = f(xpoints);

end

function [range,xpoints] = getRange2(f,xstart,xend,intervals)

% range needs to be a sym use this rather than getRange

xpoints = transpose(xstart:(xend-xstart)/(intervals):xend);

range = sym(zeros(intervals+1,1));

for i = 1:intervals+1

range(i,1) = f(xpoints(i));

end

end

function fun = f2(x)

syms y

% fun = log(x+2\*y);

% fun = x\*y^2

% fun = x.\*sin(x);

fun = x\*y\*(exp(y-x));

end

function fun = f(x)

% fun = log(x);

fun = x.\*sin(x);

end

function I = gaussQuad(f,a,b,n)

%myFun - Description

%

% Syntax: output = myFun(input)

%

% Long description

switch n

case 2

c = ones(2,1);

t = [-.57735;.57735];

case 3

c = [.555556;.888889;.555556];

t = [-.774597;0;.774597];

case 4

c = [.347855;.652145;.652145;.347855];

t = [-.861136;-.339981;.339981;.861136];

otherwise

disp('nope')

end

x = zeros(n,1);

for i = 1:n

x(i,1) = (b+a)/2-(b-a)/2\*t(i);

end

I = 0;

for i = 1:n

I = (b-a)/2\*c(i)\*f(x(i)) + I;

end

function dxdy = Deriv(x,y)

dxdy = x\*exp(3\*x)-2\*y;

% dxdy = y/x-(y/x)^2;

end

function [x,y] = Euler(xi,yi,h,steps)

x = zeros(steps,1);

y = x;

dydx = x;

x(1,1) = xi;

y(1,1) =yi;

for i = 2:steps

x(i,1) = x(i-1) + h;

dydx(i,1) = Deriv(x(i-1),y(i-1));

y(i,1) = y(i-1)+h\*dydx(i,1);

end

end

function [x,y] = mEuler(f,xi,yi,h,steps)

x = zeros(steps,1);

y = x;

x(1,1) = xi;

y(1,1) = yi;

for i = 2:steps

x(i,1) = x(i-1) + h;

y(i) = y(i-1) + h/2\*(Deriv(x(i-1)+h,y(i-1)+h\*Deriv(x(i-1),y(i-1)))+Deriv(x(i-1),y(i-1)));

end

end

function [x,y] = Heun(xi,yi,h,steps)

x = zeros(steps,1);

y = x;

dydx = x;

x(1,1) = xi;

y(1,1) =yi;

j = 1;

for i = 2:steps+1

x(i,1) = x(i-1) + h;

dydx(i,1) = Deriv(x(i-1),y(i-1));

w(j,1) = y(i-1)+h\*dydx(i,1);

y(i) = w(j);

while true

j = j + 1;

w(j,1) = w(i-1)+(Deriv(x(i-1),w(i-1))+Deriv(x(i),w(j-1)))/2\*h;

if abs(w(j)-w(j-1))<10^-3

y(i+1) = w(j);

break

end

end

end

end

function [x,y] = RK4(xi,yi,h,steps)

x = zeros(steps,1);

y = x;

x(1,1) = xi ;

y(1,1) = yi ;

for i = 2:steps

x(i,1) = x(i-1) + h;

k1 = h\*Deriv(x(i-1),y(i-1));

k2 = h\*Deriv(x(i-1)+h/2,y(i-1)+k1/2);

k3 = h\*Deriv(x(i-1)+h/2,y(i-1)+k2/2);

k4 = h\*Deriv(x(i-1)+h,y(i-1)+k3);

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

end

end

function [x,y] = midpoint(xi,yi,h,steps)

x = zeros(steps,1);

y = x;

x(1,1) = xi;

y(1,1) = yi;

for i = 2:steps;

x(i,1) = x(i-1) + h;

y(i) = y(i-1) + h\*Deriv(x(i-1)+h/2,y(i-1)+h/2\*Deriv(x(i-1),y(i-1)));

end

end

function [x,y] = AB2(xi,yi,h,steps)

% midpoint

x = zeros(steps,1);

y = x;

x(1,1) = xi;

y(1,1) = yi;

for i = 2:2

x(i,1) = x(i-1) + h;

y(i) = y(i-1) + h\*Deriv(x(i-1)+h/2,y(i-1)+h/2\*Deriv(x(i-1),y(i-1)));

end

%AB2

for i = 3:steps

x(i,1) = x(i-1) + h;

y(i) = y(i-1) + h\*(3/2\*Deriv(x(i-1),y(i-1))-1/2\*Deriv(x(i-2),y(i-2)));

end

function [x,y] = AB3(xi,yi,h,steps)

%Rk4

x = zeros(steps,1);

y = x;

x(1,1) = xi ;

y(1,1) = yi ;

for i = 2:3

x(i,1) = x(i-1) + h;

k1 = h\*Deriv(x(i-1),y(i-1));

k2 = h\*Deriv(x(i-1)+h/2,y(i-1)+k1/2);

k3 = h\*Deriv(x(i-1)+h/2,y(i-1)+k2/2);

k4 = h\*Deriv(x(i-1)+h,y(i-1)+k3);

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

end

%AB3

for i = 4:steps

x(i,1) = x(i-1) + h;

y(i) = y(i-1) + h/12\*(23\*Deriv(x(i-1),y(i-1))-16\*Deriv(x(i-2),y(i-2))+5\*Deriv(x(i-3),y(i-3)));

end

end