HW 6 Solutions

Text

Description automatically generated

Text, letter

Description automatically generated

Text, letter

Description automatically generated

clear all; close all; clc;

a = 0;

b = 60;

h = 0.2;

N = (b-a)/h;

L = 25000;

k = 0.00003;

t(1) = 0;

w(1) = 250;

for i = 1:N

w(i+1) = w(i) + h\*(k\*w(i)\*(L-w(i)));

t(i+1) = t(i) + h;

end

figure

set(0,'DefaultAxesFontSize',12)

set(0,'DefaultAxesFontWeight','bold')

set(gcf,'Color','white')

plot(t,w,'LineWidth',2)

grid on

title('Solution of I(t) Using Euler Method','FontWeight','bold')

xlabel('t','FontWeight','bold')

ylabel('I(t)','FontWeight','bold')

legend('h = 0.2')

b. modifying code in b or just running several times

clear all; close all; clc;

for j = 0:3

a = 0;

b = 60;

h = 0.2\*2^(-j);

N = (b-a)/h;

L = 25000;

k = 0.00003;

if h == 0.2

t1(1) = 0;

w1(1) = 250;

for i = 1:N

w1(i+1) = w1(i) + h\*(k\*w1(i)\*(L-w1(i)));

t1(i+1) = t1(i) + h;

end

elseif h == 0.1

t2(1) = 0;

w2(1) = 250;

for i = 1:N

w2(i+1) = w2(i) + h\*(k\*w2(i)\*(L-w2(i)));

t2(i+1) = t2(i) + h;

end

elseif h == 0.05

t3(1) = 0;

w3(1) = 250;

for i = 1:N

w3(i+1) = w3(i) + h\*(k\*w3(i)\*(L-w3(i)));

t3(i+1) = t3(i) + h;

end

else

t4(1) = 0;

w4(1) = 250;

for i = 1:N

w4(i+1) = w4(i) + h\*(k\*w4(i)\*(L-w4(i)));

t4(i+1) = t4(i) + h;

end

end

end

figure

set(0,'DefaultAxesFontSize',12)

set(0,'DefaultAxesFontWeight','bold')

set(gcf,'Color','white')

plot(t1,w1,'b-',t2,w2,'b:',t3,w3,'r-',t4,w4,'r:','LineWidth',2)

grid on

title('Solution of I(t) Using Euler Method','FontWeight','bold')

xlabel('t','FontWeight','bold')

ylabel('I(t)','FontWeight','bold')

legend('h = 0.2','h = 0.1','h = 0.05','h = 0.025')

Zoomed out on left, zoomed in on right

Chart, line chart

Description automatically generated Chart, line chart

Description automatically generated

Table

Description automatically generated

Text

Description automatically generated

Other options for c:

* Since Simpson’s rule is obtained by finding a 3-point Lagrange interpolating polynomial over smaller intervals to represent the integrand, the composite Simpson’s rule is effectively rewriting the integrand as a Lagrange interpolating polynomial and integrand. Substituting into previous code from last week, you could sub the function evaluations with calls to the solution vector for infected I
* Any function could be fit to the data
  + Matlab built in linear interpolation or cubic splines etc., depending on the functions used, it could possibly be directly integrated by Matlab
  + If you see the solution looks like a sigmoidal function, you could use your continuous least squares to determine parameters to fit a sigmoidal curve. OR you could use built in Matlab. Once the function is fit, again, integrate.
  + You could do a lagrange interpolating polynomial over the interval 0 to 60, but it would be advised to do this not taking all of the solution values (smaller degree polynomial)