University of Waterloo

Faculty of Engineering

Department Of Electrical And Computer Engineering

Hello.

NE 216

Laboratory 5

FALL 2011

Prepared by

Sharda, Vid

UW Student ID Number: 20833192

UW User ID: v2sharda@uwaterloo.ca

Prepared by

Chockalingam Shanmugam, Anish

UW Student ID Number: 20814528

UW User ID: a2chocka @uwaterloo.ca

2A Nanotechnology Engineering

November 15, 2020

Having watched the presentation, we will go through the following exercise:

**5.1-7** **Your Solution**

Enter your implementation of dp45 here. It must contain all comments and documentation as in previous assignments.

function [t\_out, y\_out] = dp45( f, t\_rng, y0, h, eps\_abs )

function [t\_out, y\_out] = dp45( f, t\_rng, y0, h, eps\_abs )

A = [ 0 0 0 0 0 0 0;

1 0 0 0 0 0 0;

1/4 3/4 0 0 0 0 0;

11/9 -14/3 40/9 0 0 0 0;

4843/1458 -3170/243 8056/729 -53/162 0 0 0;

9017/3168 -355/33 46732/5247 49/176 -5103/18656 0 0;

35/384 0 500/1113 125/192 -2187/6784 11/84 0]';

by = [5179/57600 0 7571/16695 393/640 -92097/339200 187/2100 1/40]';

bz = [ 35/384 0 500/1113 125/192 -2187/6784 11/84 0]';

c = [0 1/5 3/10 4/5 8/9 1 1]';

[N, y0\_cols] = size( y0 );

if (y0\_cols ~= 1)

throw(MException('MATLAB:invalid\_argument', ...

'The value is less than 1, change it'));

end

% Initialize t\_out and y\_outlen(

n\_K = 7;

K = zeros( N, n\_K ); %edited

t0 = t\_rng(1);

tf = t\_rng(2);

t\_out(1) = t0;

y\_out = y0;

% Initialize our location to k = 1

k = 1;

while t\_out(k) < tf

% Use Dormand Prince to find two approximations

% y\_tmp and z\_tmp to approximate y(t) at

% t = t\_out(k) + h for the current value of h

for m = 1:n\_K

K(:, m) = f( t\_out(k) + h\*c(m), y\_out(:, k) + h\*c(m)\*K\*A(:, m) ); %EDITED

end

y\_tmp = y\_out(:, k) + h\*K\*by; %EDITED

z\_tmp = y\_out(:, k) + h\*K\*bz; %EDITED

% Calculate the scaling factor 's'

s = ((h\*eps\_abs)/(2\*(tf-t0)\*norm(y\_tmp - z\_tmp))); %EDITED

if s >= 2

% We use z\_tmp to approximate y\_out(k + 1)

% t\_out(k + 1) is the previous t-value plus h

% Increment k and double the value of h for the

% next iteration.

y\_out(:, k+1) = z\_tmp;

t\_out(k+1) = t\_out(k) + h;

h = h\*2;

k = k+1;

elseif s >= 1

% We use z\_tmp to approximate y\_out(k + 1)

% t\_out(k + 1) is the previous t-value plus h

% In this case, h is neither too large or too

% small, so only increment k

y\_out(:, k+1) = z\_tmp;

t\_out(k+1) = t\_out(k) + h;

k = k+1;

else s < 1

% Divide h by two and try again with the smaller

% value of h (just go through the loop again

% without updating t\_out, y\_out, or k)

h = h/2;

end

% We must make one final check before we end the loop:

% if t\_out(k) + h > tf, we must reduce the

% size of h so that t\_out(k) + h == tf

if t\_out(k) + h > tf

h = tf - t\_out(k);

end

end

y\_out

end

**5.8** **Testing your Implementation**

**5.8*a*** Give your output to the function call

format long

[t5a, y5a] = dp45( @f5a, [0, 1], [3/10 1/2]', 0.1, 1e-5 )

t5a =

0 0.100000000000000 0.300000000000000 0.700000000000000 1.000000000000000

y5a =

0.300000000000000 0.301027433242441 0.308934939722046 0.346907125159289 0.395415916716352

0.500000000000000 0.466212959694911 0.405658801111873 0.309729945354145 0.256363104345417

**5.8*b*** Give your output to the function call

It appears with the initial-conditions as stated in the predator-prey population model that the period (the time it takes for the system to make one cycle) is a value somewhere on the range [6.6, 7.0]. Using

tf = 6.8;

[t5a, y5a] = dp45( @f5a, [0, tf], [3/10 1/2]', 0.001, 1e-10 );

y5a(:, 1) % The initial value

y5a(:, end) % The last point

norm( y5a(:, end) - y5a(:, 1) )

plot( y5a(1, :), y5a(2, :) )

With the initial value of *h* given and **abs = 10-10, find a value of *t*final such that the norm is less than 0.001. You will receive full marks for a correct answer; however, you are welcome to add additional details in case you’re worried your answer may be incorrect.

tf = 6.6;

[t5a, y5a] = dp45( @f5a, [0, tf], [3/10 1/2]', 0.001, 1e-10 );

while norm( y5a(:, end) - y5a(:, 1) )>0.001

tf=tf+0.001

[t5a, y5a] = dp45( @f5a, [0, tf], [3/10 1/2]', 0.001, 1e-10 );

end

tf = 6.819 6.82 – 6.824

**5.8*c*** Find the output of this function call with format long:

>> format

>> [t5b, y5b] = dp45( @f5b, [0, 0.4], [1 1 1]', 0.1, 1e-1 )

t5b =

0 0.0500 0.1000 0.1500 0.2000 0.2500 0.3000 0.3250 0.3500 0.3750 0.4000

y5b =

1 1.2872 2.1329 3.7368 6.5428 11.0431 16.6861 18.8514 19.5579 18.3191 15.3661

1 2.4008 4.4720 7.9648 13.7324 21.7775 27.1885 25.0992 18.6015 9.5311 1.1051

1 0.9640 1.1141 1.8178 4.1801 11.0164 26.2064 35.9627 44.0030 47.6993 46.7626

Copy and paste your Matlab commands and output here.

t5b =

Columns 1 through 6

0 0.050000000000000 0.100000000000000 0.150000000000000 0.200000000000000 0.250000000000000

Columns 7 through 11

0.300000000000000 0.325000000000000 0.350000000000000 0.375000000000000 0.400000000000000

y5b =

Columns 1 through 6

1.000000000000000 1.287885377620408 2.133327234916710 3.736841520357459 6.542619573305995 11.043088695328429

1.000000000000000 2.399753332773258 4.471164431309435 7.963999973958884 13.731329278118773 21.776122949106735

1.000000000000000 0.963952520226649 1.114021441981560 1.817787222047045 4.180002556723617 11.016454057232497

Columns 7 through 11

16.686151643356162 18.850162923277111 19.555756226686814 18.317279000451165 15.365791420727835

27.183783147801634 25.094579630333790 18.599323172841387 9.533261853386465 1.110819456477330

26.207355525917784 35.960840313471586 43.997955790227536 47.693096331128075 46.758034054934598

**5.8*d*** What is the difference between the two plots:

[t5b, y5b] = dp45( @f5b, [0, 10], [1 1 1]', 0.001, 1e-5 );

plot3( y5b(1,:), y5b(2,:), y5b(3,:) )

and

[t5b, y5b] = dp45( @f5b, [0, 10], [1 1 1]', 0.001, 1e-5 );

hold off

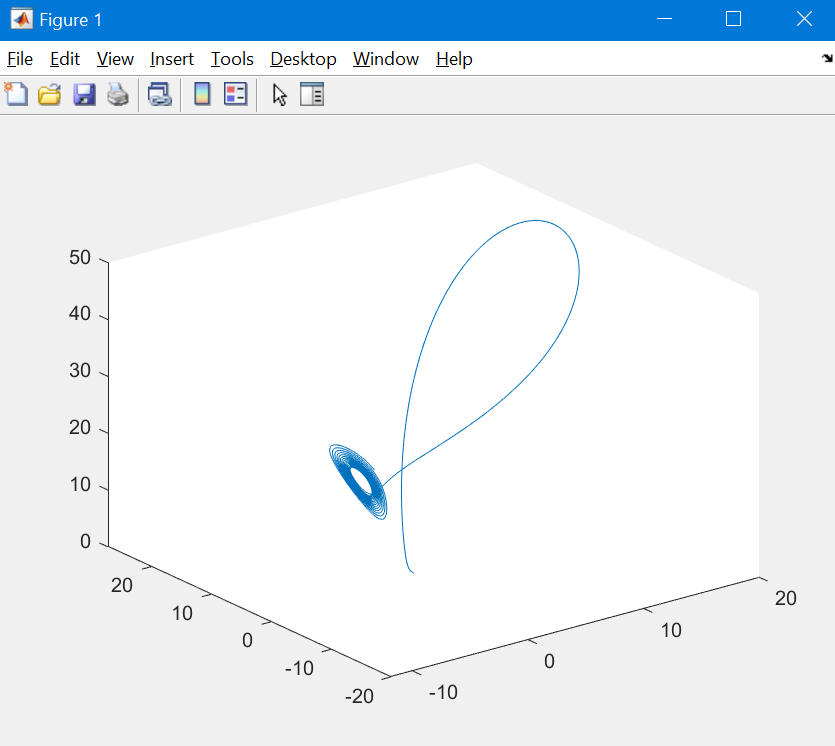
plot( t5b, y5b(1,:), 'r' )

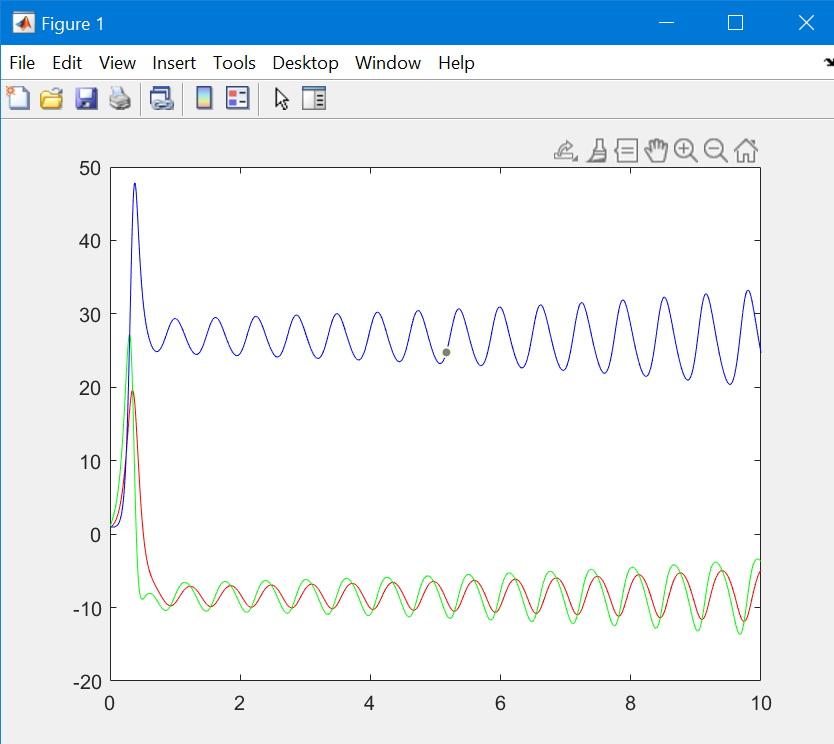
hold on

plot( t5b, y5b(2,:), 'g' )

plot( t5b, y5b(3,:), 'b' )

One is a 3-dimensional plot using the x, y and z. While the third one is 3, 2 dimensional plots. As you can see with the 3D plot, it starts at an initial value, has variability at first but then begins to spiral into the loop. This is what the 2nd graph represents (variability at first) where the first part is the big loop that happens in the 3D graph. As you go further, the 3D image shows the spiral getting bigger, which is what the sine wave in the 2D graph shows, with increasing amplitude. (The values oscillate around different centers with varying amplitude, thereby creating a spiral)





**5.8*e*** Find the output of

[t5c, y5c] = dp45( @f5c, [0, 1], [1.2, 1.3]', 0.1, 1e-4 )

t5c =

0 0.1000 0.3000 0.5000 0.7000 0.9000 1.0000

y5c =

1.2000 1.3011 1.3425 1.2068 0.9480 0.6233 0.4535

1.3000 0.7273 -0.2772 -1.0339 -1.5062 -1.6948 -1.6915

using format long.

t5c =

Columns 1 through 6

0 0.100000000000000 0.300000000000000 0.500000000000000 0.700000000000000 0.900000000000000

Column 7

1.000000000000000

y5c =

Columns 1 through 6

1.200000000000000 1.301059241000000 1.342464154558350 1.206804155010630 0.947961994454632 0.623317214258966

1.300000000000000 0.727349702500000 -0.277169176269166 -1.033861230884215 -1.506170320184978 -1.694788803293874

Column 7

0.453498304302876

-1.691483687029024

**5.8*f*** For the RLC circuit shown in the slides, use the example where the input voltage is cut after 5 s. Replace the plot in Figure 1 with the output of this plot.

[t5d, y5d] = dp45( @f5d, [0, 10], [0, 0]', 0.1, 1e-8 );

plot( t5d, y5d(1,:), 'r' );

hold on  
plot( t5d, y5d(2,:), 'b' );

title( 'v2sharda and a2chocka' ) % or title( 'uwuserid' )

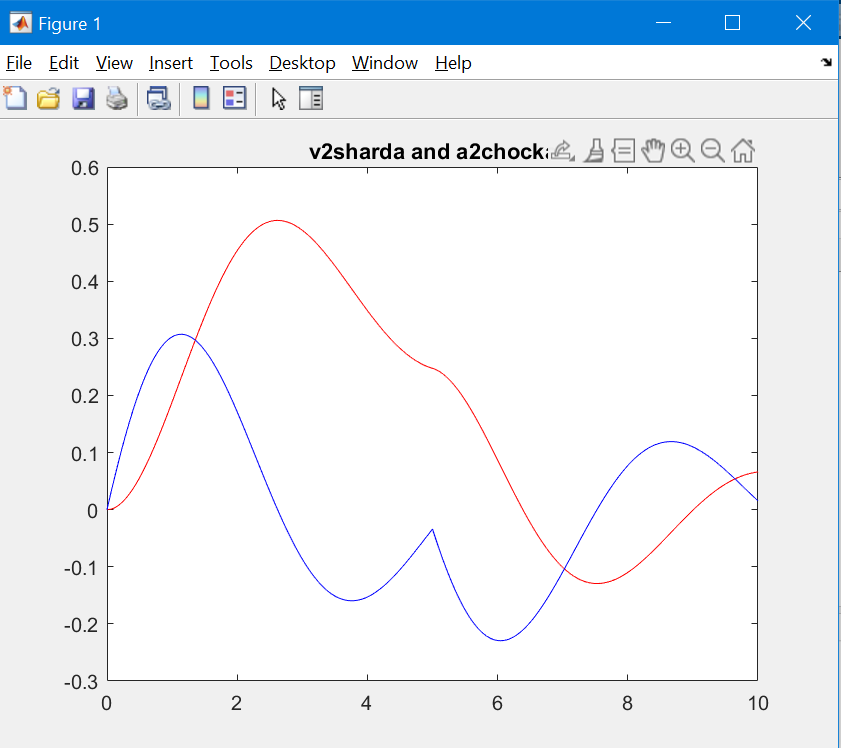


Figure 1. The response of a system to the input voltage .

The discontinuity in the input voltage causes the second function to have a cusp. Explain physically what is happening at this point with respect to the charge and current at this point.

This is because as the Dormand Prince function gets closer to the discontinuity, the values of h become smaller and smaller due to s. This causes the points around the discontinuity to be refined. When the s value gets close enough and makes the jump to the other side, it leaves a peak value in the middle as the cusp. As the values move away from the discontinuity, the points revert to having larger distances in between.

**5.8*g*** **Micro/Nano Actuators – Current Sensors**

You have been asked by your employer to analyses the behavior of a MEMS actuator. The micro actuator consists of a cantilever beam with a magnet attached to its end, as shown in Figure 2 and Figure 3.



Figure 2. A micro actuator with a cantilever beam with a magnet.[[1]](#footnote-1)



Figure 3. Figure 2 from above.[[2]](#footnote-2)

This micro actuator is placed near a current carrying wire with an alternative current passing through it. Therefore, there will be a magnetic field around the wire which will induce a force on our micro actuator causing an oscillatory vertical motion.

Previous simulations were done to calculate the force imposed on the tip of the cantilever: *F*(*t*) = 5 cos(4*t*). Given our system, a simplified differential equation for motion can be written as



where

*m* is the equivalent mass,

*d* is the modal damping,

*c* is the mechanical stiffness, and

*F* is the force.

Based on the coefficients, we can write down the equation of motion:

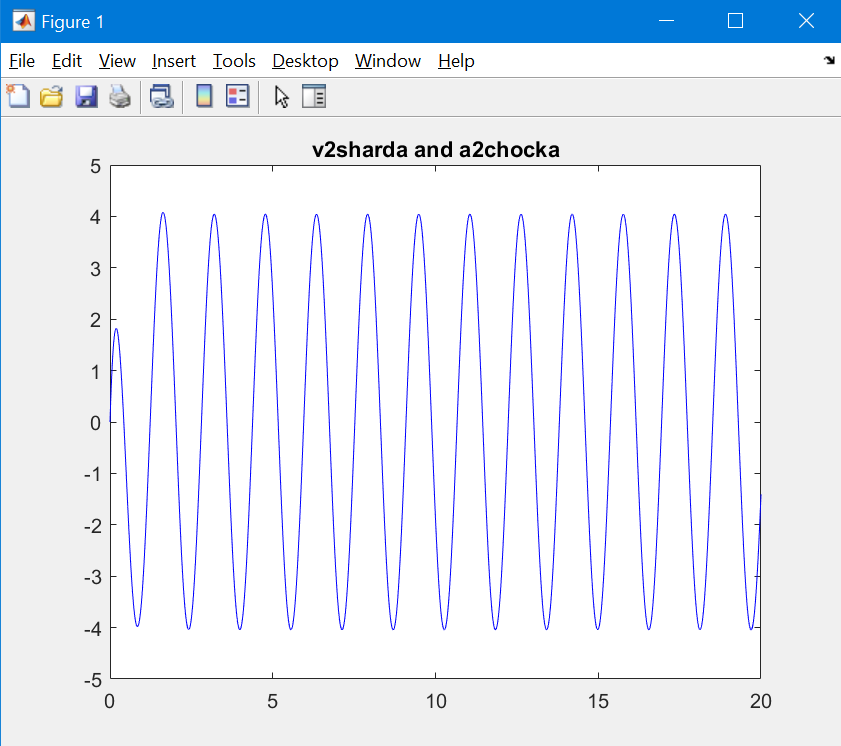


with



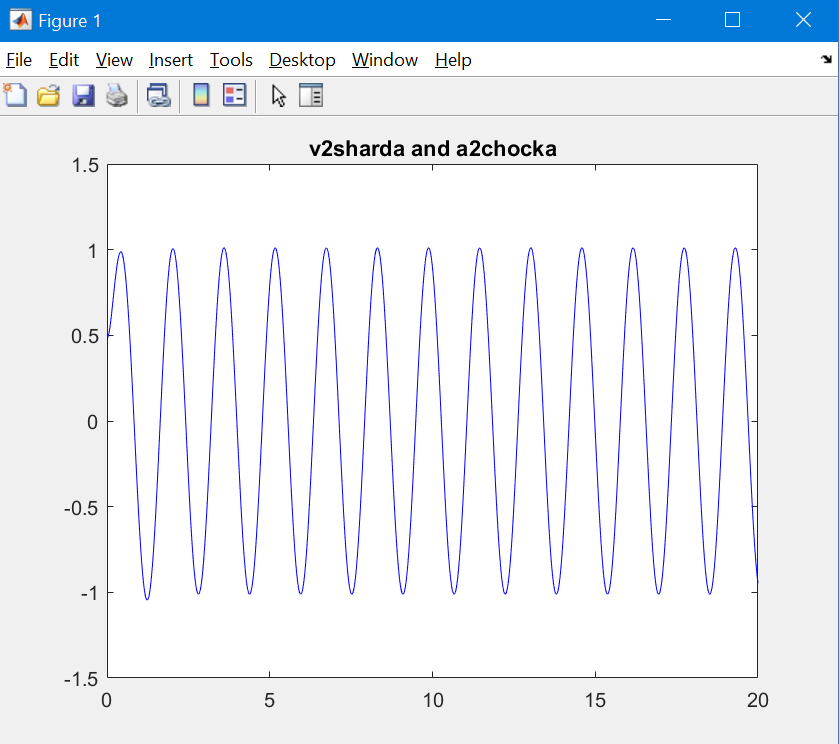
where 20*nnnnnn* is one of your uWaterloo student ID numbers.

Your employer is asking you to approximate the solution on [0, 20] and you should use an initial *h* = 0.01 and **abs = 10–8. You should provide the plot of *t* vs. *x*(*t*) and make sure your uWaterloo User ID(s) are in the title.



>> plot( t5a, y5a(2,:), 'b' );

>> title( 'v2sharda and a2chocka' ) % or title( 'uwuserid' )



>> plot( t5a, y5a(1,:), 'b' );

title( 'v2sharda and a2chocka' ) % or title( 'uwuserid' )

**5.8*h*** Suppose you have a system of two 2nd-order initial value problems defined by


Assuming you defined

,

how would you define

function dw = f5f( t, w )

a1 = 0.3;

a2 = -0.1;

a3 = 0.5;

a4 = 0.8;

b = 1.0;

c1 = 0.2;

c2 = -0.6;

c3 = 0.4;

c4 = 0.7;

d = 0.9;

dw = dw = [w(2)

1-0.3\*w(1)+0.1\*w(2)-0.5\*w(3)-0.8\*w(4)

w(4)

0.9-0.2\*w(1)+0.6\*w(2)-0.4\*w(3)-0.7\*w(4)]; %with constants already inserted

end

Plot the output on the range [0, 40] using *h* = 0.01 and **abs = 10-8 together with the initial conditions given above. Replace the plot in Figure 4 with the plot of *u*(*t*) in blue and *v*(*t*) in red and make sure your UW User ID(s) are in the title.

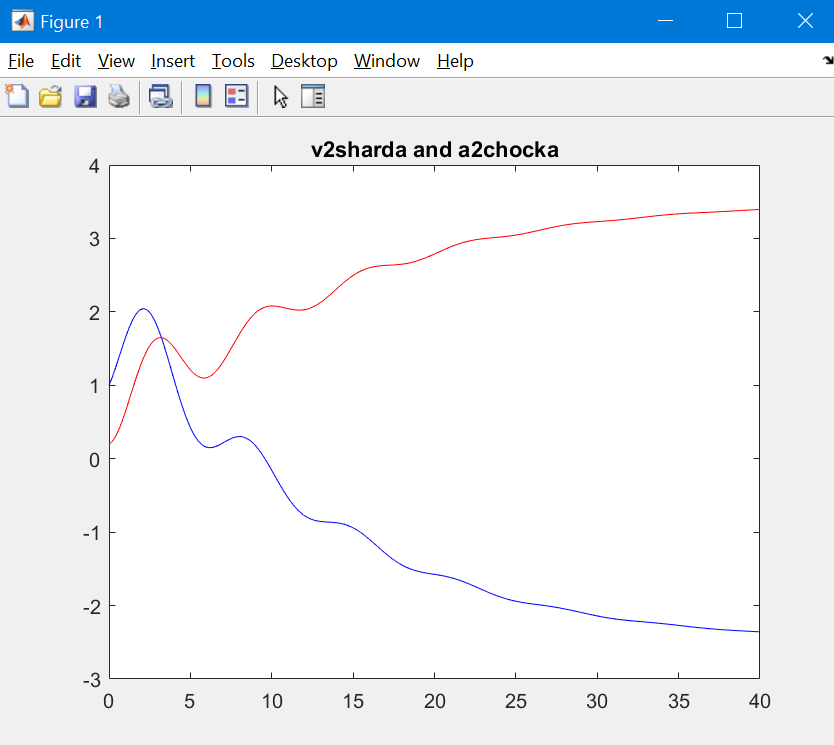


Figure . The coupled functions *u*(*t*) and *v*(*t*).

**5.9** Did you remember to copy your entire functions dp45 into Question 5.7? That is, all comments and all code?

**Yes**

1. *A MEMS AC current sensor for residential and commercial electricity end-use monitoring*

   E S Leland, P K Wright and R M White. [↑](#footnote-ref-1)
2. *ibid*. [↑](#footnote-ref-2)