ME2-HCPT Final Test Tuesday

Name:	CID number:

This is an open book exercise. You can **reuse/adapt/amend** any of your previous scripts, as long as you submit them.

Duration: 1h 20min

In your H drive create a folder *H:\ME2CPT\Final*.

Comment, sensibly, all your scripts **[5]**

Write at least a main script for each task, and name the files taskA.m, taskB.m, taskC.m. If you need or wish, you can then subdivide the tasks and write any associated functions, at your convenience.

Task A [17]

The files *Zener.txt* and V*Source.txt* contains two sets, (x_Z, y_Z) and (x_S, y_S) respectively, of data points.

1. Read in the files and plot the two sets of data, *y* vs *x*, on the same graph.

The text file contains a heading of characters, these have to be dismissed. Hence, we load the file from second line, first column, i.e. 1,0

```
% read in the two files
% and extract the two data sets
% (xa,ya)
Z = dlmread('Zener.txt',',',1,0);
xz = Z(:,1);
yz = Z(:,2);
%z(xb,yb)
S = dlmread('VSource.txt',',',1,0);
xs = S(:,1);
ys = S(:,2);

% plot the data
plot(xz,yz,'r.')
hold on
plot(xs,ys,'g.')
grid on
```

2. Fit the two sets of points with polynomials: the first, (x_Z, y_Z), with a polynomial of order 6, and the second, (x_S, y_S), with a polynomial of order 2.

Note the fitting with order 6 and 2

```
% fit the data with a polynomial
cz = polyfit(xz,yz,6);
cs = polyfit(xs,ys,2);
```

3. Evaluate the two fitting polynomials in the interval [-8 : 0.01 : 4].

Poly need to be evaluated in the given range and not at the original x_Z, x_S.

```
% evaluate at given range [-8:0.01:4]
x = [-8:0.01:4];
pz = polyval(cz,x);
ps = polyval(cs,x);
```

4. The two fitted polynomials intersect each other at the intersecting point (x_0, y_0). Find the value of x_0 by using the bisection method, with a tolerance of 0.01.

The solution requested is the root of the equation pz – ps = 0.

The easiest and fastest way of finding this root is considering that the difference of two poly is still a poly, hence we can work with the coefficients only:

```
c = cz - [0 0 0 0 cs];
```

The solution is found either invoking the function *root* in Matlab for polynomial c, but not requested in this exercise, or use the bisection function written in Tutorial 3.

In there we defined the mathematical function f(x) = 0 in a separate function. The mathematical function in our case is the polynomial with coefficients c.

Hence, we modify in mybisection

```
ff = myfunc(a) * myfunc(xm);
```

as:

```
ff = polyval(coeff,a) * polyval(coeff,xm)
```

and we call the modified mybisection by passing the coeffcients c:

```
[sol] = mybisection(-8,4,0.01,c)
```

Task B [18]

1. Solve numerically the ordinary differential equation:

$$\frac{d}{dt}\left(e^t \frac{dy}{dt}\right) = 3y^2 cos(t^2)$$

with the initial conditions y(0) = 2 and $\left.\frac{dy}{dt}\right|_{t=0} = \frac{\pi}{4}$.

Compute and plot *y(t)* vs *t* in the range *t* = [0 : 0.01 : 2].

THIS IS AN INITIAL VALUE PROBLEM

Rewrite the eqn as:

$$e^t \frac{dy}{dt} e^t \frac{d^2y}{dt^2} = 3y^2 cos(t^2)$$

And

$$\frac{dy}{dt} + \frac{d^2y}{dt^2} = e^{-t} 3y^2 cos(t^2)$$

This is a second order ODE, as for the pendulum in T6 Task 3/4.
Let's define a new variable:

$$\frac{dy}{dt} = w$$

Then, we can rewrite:

$$\begin{cases} \dfrac{dy}{dt} = w \\ \dfrac{dw}{dt} = -w + e^{-t} 3y^2 cos(t^2) \end{cases}$$

With this format, such as:

$$\begin{cases} \dfrac{dx_1}{dt} = F_1(t, x_1, x_2) \\ \dfrac{dx_2}{dt} = F_2(t, x_1, x_2) \end{cases}$$

We can reuse myeulertwo, as it is, and only insert the function F1 and F2 in myfunc1 and myfunc2.

So, very little coding:

```
function f = myfunc1(t,x1,x2)
f = x2;

function f = myfunc2(t,x1,x2)
f = -x2 + 3*exp(-t)*x1^2*cos(t^2);
```

Just pay attention in the main code when setting the initial conditions:

```
% assign initial conditions
b(1) = 2;
b(2) = pi/4;

% reuse previous function myeulertwo
[t, X] = myeulertwo(b,t0,tend,dt);
```

2. Save the values of *t* and *x* in the file *mysol.txt*, in two columns format.

Pay attention in plotting the correct variable, in this case y, i.e. x1.

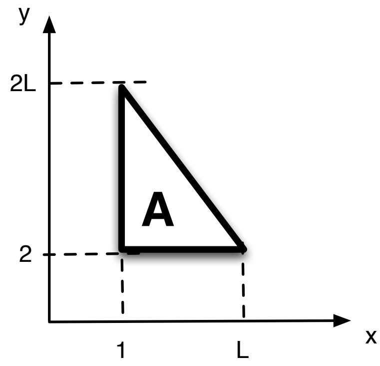
```
plot(t,X(1,:))
grid on
```

Task C [20]

1. Solve numerically, by using the trapezoidal scheme (with $dx = dy = 0.1$), the integral:

$$S = \iint_A (x + y^2) dA$$

over the domain A described in the figure:



2. Repeat the calculation for values of L = [4 : 1 : 50] and plot the values of the integral *S* vs L.

We start looping for every L, in the given range:

```
L = [4:1:50];
% for every value of L, compute the integral
for j = 1 : length(L)
```

Within each iteration, we do compute the double integral.

We set the current x domain, as a function of L

```
x = [1:dx:L(j)]
```

Then, for every point of x we set the y domain

```
    for i = 1 : length(x)
        % for this x set the y domain of integration
        ytop = -2*x(i) + 2*L(j) + 2;
        y = [2:dy:ytop];
```

We evaluate the integrand along the y domain, for the given x:

```
        z = x(i) + y.^2;
```

and we compute the integral along y

```
        G(i) = mytrapz(y,z);
```

Finally, once all the integrals along the y domains have been computed,
We compute the final integral along x.

```
S(j) = mytrapz(x,G);
```

Note that we store the results in an array, as a function of L.

Finally plot

```
      plot(L,S,'o')
```

Upload ALL your scripts and results on Blackboard.

Useful Matlab functions for this test:	
dlmread	- reads multiple lines of numbers from a file
dlmwrite	- writes numerical data to a file
polyfit	- finds coefficients of polynomial to required degree
polyval	- evaluates a polynomial at specified points