Math 329 -- Boundary Value Problem Exercises

1) SHOOTING

Consider the equation: y''+2y'+3y=0   y(0)=1,   y(10)=0

Create a Matlab derivative file-function to solve this:

```
function [dvars] = bvpexample(t, vars)
y1=vars(1);
y2=vars(2);
dy1=y2;
dy2=-2*y2-3*y1;
dvars = [dy1; dy2];
```

Use ode45 and shooting to find a solution by guessing the initial velocity.

```
[t,vars]=ode45(@bvpexample, [0 10], [1 GUESS]);
```

Note, you can print the value of y1 at the right boundary with the command

```
vars(end,1)
```

Use ode45 and shooting to find a solution which matches the boundary condition at t=10 to less than 10^-5. Try this by guessing initial values.
How many significant digits did you need for y'(0) to get this accuracy for y(10)?

2) Consider the same equation with the conditions y(0)=1, y(10)=2.
Use shooting again to find a solution which matches the boundary condition at t=10 to less than 10^-5. How many significant digits did you need for y'(0) to get this accuracy for y(10)?

3) FINITE DIFFERENCE METHODS

Solve the same equation as for number 1 above with the methods of finite differences. Look at the example script bvpfd.m from our webpage.
First read it. Is it setting up diagonal matrices like in class?
Next, get a feel for it.
Uncomment the lines %n=5 and %full(LHS)   Then run the script.
Is LHS what you expected for n=5?

Plot the solution: plot(x,y_fd)

Now comment out the full(LHS) line again and repeat with a more realistic resolution, say n=100. It might be convenient to set n from the command line.
So, comment out the line setting n and run something like:

```
n=100; bvpfd; plot(x,y_fd)
```

What is the value of y(10) for this solution?
Plot the shooting and finite diff solutions on the same graph.
Do they agree? Increase the accuracy until they agree visually.

4) VARIABLE COEFFICIENTS

Consider the equation: y''+2(x^2)y'+(2x+1)3y = cos(x*pi/10))   y(0)=1,y(10)=0

Now the diagonals have to depend on x. So, let's create some more helpful sparse matrices such as the following:

```
n=5; x=linspace(0,10,n);  % To check that this is working
xID =sparse(2:n-1,2:n-1,x(2:n-1),n,n);
x2ID=sparse(2:n-1,2:n-1,x(2:n-1).^2,n,n);   %or    x2ID=xID.*xID;
cosx=sparse(2:n-1,2:n-1,cos(x(2:n-1)*pi/10)),n,n);
```

Check that these commands did what you expect.

```
full(xID)
full(x2ID)
full(cosx)
```

Now try combinations: what does 2*xID+1 look like?
These matrices are suppose to help us solve variable coefficient problems.
We multiply the derivative matrices by these.
So now: solve the ODE...

```
LHS= ???    # (something like secderv +2*x2ID*derv + ...)
RHS= ???    # put in the cos(x*pi/10) function as a vector.
            # Then overwrite the boundary values with RHS(1)=... and RHS(n)=...
y_fd= ???   # solve for the solution
```