Using the basic ODE solver in Matlab

1. Example: Estimate the value of $y(2)$, the solution to

$$y'' + \sin(t)y = 0, \quad y(-1) = 1, y'(-1) = 0$$

SOLUTION: We will actually do more and plot the solution as well. First we need an M-file for the derivatives. Convert the second order D.E. into a system of first order, by defining $u(t) = y(t)$, $v(t) = y'(t)$. Then:

$$\begin{aligned} u' &= v \\ v' &= -\sin(t)u \end{aligned}$$

Now I'll need a Matlab file for the derivatives:

```
function dy=Sample01(t,y)
% dy=Sample01(t,y) - A Sample DE for our ODE solver
dy=zeros(size(y));
dy(1)=y(2);
dy(2)=-sin(t)*y(1);
```

In Matlab, we type the commands:

```
tspan=linspace(-1,2);
%If we don't want all this output, just do: tspan=[-1 2];
[tout,yout]=ode45(@Sample01,tspan,[1;0]);
plot(tout,yout);
```

The desire result is $yout(end, 1) \approx 0.6954$

2. Example: Solve the Van der Pol Equation:

$$u'' - (1 - u^2) + u = 0$$

with various initial conditions. Plot the resulting solution in the parameterized form $(u(t), u'(t))$.

SOLUTION: In Matlab, we will need to write the differential equation as a first order D.E. Matlab specifies that the function input (t, y) and output y'. We do this by introducing a "dummy" variable: Let $x = u$, $y = u'$. This gives a system of first order D.E.'s:

$$\begin{aligned} x' &= y \\ y' &= -x + (1 - x^2)y \end{aligned}$$

I'll write the following function and save it in my directory as `vdp01.m`:

```
function dy=vdp01(t,y)
%Function:  dy=vdp(t,y) is the Van der Pol equation.
%  Although t is never used, we still include it for the
%  ODE solver.
dy=zeros(size(y));
dy(1)=y(2);
dy(2)=-y(1)+(1-y(1)^2)*y(2);
```

I'll write the following as a script, but I could just type the commands in the command window:

```
tspan=linspace(0,40,200);  %I'll have Matlab return the value
                           %my solution at these time values.
%I'll use the following four initial conditions
%(could set these randomly)
IC=[2.5, 2.5; 2.5, -2.5; -2.1, 0.5; 0.5, 0.3];
%Now solve the DE four times
for j=1:4
  [Tout,Yout{j}]=ode45(@vdp01,tspan,IC(j,:)');
end
%Now plot the solutions:
figure(1)
for j=1:4
  plot(Yout{j}(:,1),Yout{j}(:,1));
  if j==1; hold on; end;
end
hold off
```

3. Example: Generate a solution to the Lorenz Equations (these are the "famous" values of the parameters):

$$\begin{aligned} x' &= 10(y-x) \\ y' &= 28x - y - xz \\ z' &= -(8/3)\,z + xy \end{aligned}$$

I'll once again write a function that takes in (t, y) and outputs the derivative:

```
function dy=lorenz01(t,y)
% dy=lorenz01(t,y) are the Lorenz Equations
dy=zeros(size(y));
dy(1)=10*(y(2)-y(1));
dy(2)=28*y(1)-y(2)-y(1)*y(3);
dy(3)=-(8/3)*y(3)+y(1)*y(2);
```

And I'll use a script driver:

```
%% Script for the Lorenz Equations
tspan=linspace(0,75,4000); %Get lots of data!
IC=[5;5;5];
[Tout,Yout]=ode45(@lorenz01,tspan,IC);
plot3(Yout(:,1),Y(:,2),Y(:,3));
```