Solvers

MATLAB has several solvers.

- **ode45**
  - This is the first choice.
- **ode23**
- Stiff solvers
  - **ode15s**

**ode45**

- Uses variable step size.
- Specify error tolerance instead of step size.
- MATLAB chooses the step size at each step to achieve the limit on the error.
- The default tolerance is good enough for this course.

Syntax

```matlab
[t,y] = ode45(derfile,[t0,tf],y0);
plot(t,y)
```
Solving Systems

- Example:
  \[ x' = v \]
  \[ v' = -9.8 - 0.04v|v| \]

- Change to vector notation. (Use MATLAB vector notation)
  - \( u(1) = x \)
  - \( u(2) = v \)

Derivative m-file \texttt{ball.m}

```matlab
function upr = ball(t,u)
x = u(1);
v = u(2);
xpr = v;
vpr = -9.8 - 0.04*v*abs(v);
upr = [xpr; vpr];
```

Derivative m-file \texttt{ballshort.m}

```matlab
function upr = ballshort(t,u)
upr = zeros(2,1);
upr(1) = u(2);
upr(2) = -9.8 - 0.04*u(2)*abs(u(2));
```
Computing and Plotting Solutions to Systems

- \([t,u] = \text{ode45}('ball', [0,3],[0;50]);\)
- \(\text{plot}(t,u)\) – plots all of the components versus \(t\).
- \(\text{plot}(t,u(:,1))\) – first component versus \(t\).
- \(\text{plot}(u(:,1),u(:,2))\) – second component versus the first. This is a phase plane plot.
- \(\text{plot3}(u(:,1),u(:,2),t)\) – 3-D plot.

Solving Higher Order Equations

- Reduce to a first order system and solve the system.
- Example: The motion of a pendulum is modeled by
  \[ \theta'' = -\frac{g}{L} \sin \theta - D \theta'. \]
- Introduce \(\omega = \theta'\). Notice
  \[ \omega' = -\frac{g}{L} \sin \theta - D \omega. \]

Equivalent First Order System

- \(\theta' = \omega\)
- \(\omega' = -\frac{g}{L} \sin \theta - D \omega\)
- Change to vector notation. (Use MATLAB vector notation)
  - \(u(1) = \theta\)
  - \(u(2) = \omega\)
function upr = pend(t,u)
L= 1;
global D
th = u(1);
om = u(2);
thpr = om;
ompr = -(9.8/L)*sin(th) - D*om;
upr = [thpr; ompr];