2.670 ME Tools

Solving O.D.E.s with Matlab

This is a set of pages you need to step through to finish the last matlab assignment.

2.670
ME Tools

Solving ODEs with Matlab

K. Otto
2.670 ME Tools

ODEs

- We need to solve equations of the form

$$\dot{T} = f(T(t), t)$$
$$T(0) = T_0$$

- and higher order and/or coupled equations such as

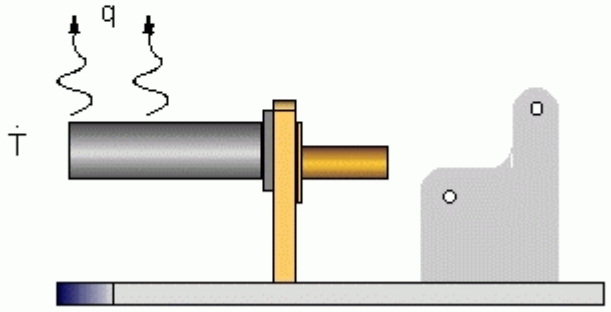
$$\ddot{x}_1 = \frac{k}{m_1}(x_1 - x_2) + \frac{c}{m_1}\dot{x}_1 \qquad x_1(0) = X \quad x_2(0) = 0 \quad \dot{x}_1(0) = 0 \quad \dot{x}_2(0) = V$$

$$\ddot{x}_2 = \frac{k}{m_2}(x_2 - x_1)$$

2.670
ME Tools

Cooling of the Engine

- As an example. lets consider the engine cooling as a lumped mass



Consider other Cooling Modes

- Convective Cooling of a lumped mass

$$q = hA(T(t) - T_\infty)$$

- Radiative Cooling of a lumped mass

$$q = \varepsilon\sigma A(T^4(t) - T_\infty^4)$$

Temperature ODE

Convection Cooling $\Rightarrow q_{conv} = hA(T(t) - T_\infty)$

Radiative Cooling $\Rightarrow q_{rad} = \varepsilon\sigma A(T^4(t) - T_\infty^4)$

Lumped Mass $\Rightarrow -q_{lost} = mc_p \frac{dT}{dt}$

$$\frac{dT}{dt} = \frac{-hA}{mc_p}\left(T(t) - T_\infty\right) - \frac{\varepsilon\sigma A}{mc_p}\left(T^4(t) - T_\infty^4\right)$$

There is no closed form solution to this equation.

Solving ODEs with Matlab

Matlab provides a numerical differential equation solver.

```
%   ODE23 and ODE45 are functions for the numerical solution of
%   ordinary differential equations.  They employ automatic step
%   size Runge-Kutta-Fehlberg integration methods.  ODE23 uses a
%   simple 2nd and 3rd order pair of formulas for medium accuracy
%   and ODE45 uses a 4th and 5th order pair for higher accuracy.
```

m-file of the Derivative

Create an m-file of the derivative as a function

```
function dTemp = Tempdot(t, T)

h=10; A=0.005; m=0.1; cp=400;
Tinf=273+18;
emiss=0.95; boltz=5.67e-8;

dTemp = -h*A/m/cp*(T+273-Tinf)-
  emiss*boltz*A/m/cp*
  ((T+273)^4-(Tinf)^4);
```

This uses temperatures in centigrade, and converts internally to Kelvin.

Where do the constants come from?

Calling the Solver

Call the o.d.e. solver `ode45`. The call must specify the derivative function, the start time, the end time, and the initial conditions.

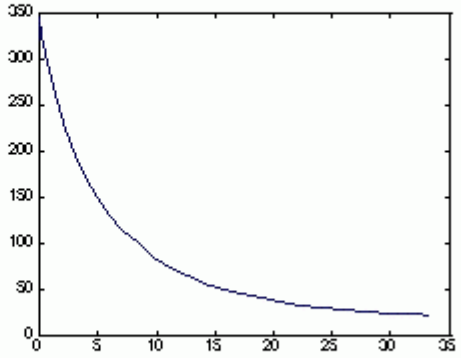
```
>> [time, Temp] = ode45('Tempdot',[0,2000],345)
```

Output time vector
Output Temperature vector
Start time
End time
Initial Temperature condition
Function that defines the derivative

The size of the returned vectors are determined by `ode45`, and depend upon how rapid the function changes.

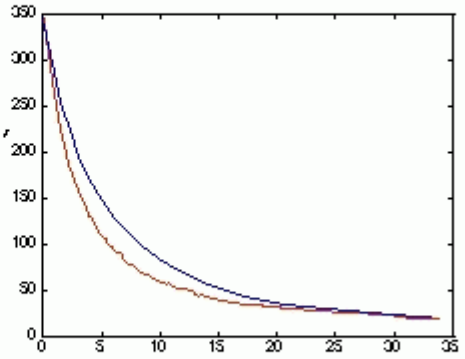
Plotting the Solution

```
>> plot(time/60,
        Temp)
```



Comparison with Data

```
>> load('temp.dat')
>> plot(temp(:,1),
     (temp(:,2)-32)*5/9,
     time/60,Temp)
```



Note the conversion of the temp data from Fahrenheit.

Do you believe this model?

Higher Order Derivatives

Consider

$$m\ddot{x} + c\dot{x} + kx = f\sin(\omega t)$$

How do you solve this equation with a second order derivative?

Recall matlab can only solve first order equations.

Converting to a First Order Vector System

- **Numerical solvers must have higher order problems expressed as**
  - first order and
  - vectorized equations

Let $y = \dot{x}$

Then
$$\dot{y} = \frac{c}{m} y + \frac{k}{m} x - \frac{f}{m} \sin(\omega t)$$
$$\dot{x} = y$$

These 2 Problems are Equivalent

$$\dot{y} = \frac{c}{m} y + \frac{k}{m} x - \frac{f}{m} \sin(\omega t)$$

These are the same, since $y = \dot{x}$

$$\ddot{x} = \frac{c}{m} \dot{x} + \frac{k}{m} x - \frac{f}{m} \sin(\omega t)$$

First Order Vector Equations

- So to solve second and higher order O.D.E.'s such as

$$m\ddot{x} + c\dot{x} + kx = f\sin(\omega t)$$

- convert to first order vector equations

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \end{Bmatrix} = \begin{bmatrix} -k/m & -c/m \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} x \\ y \end{Bmatrix} + \begin{Bmatrix} f\sin(\omega t) \\ 0 \end{Bmatrix}$$

- and solve as a first order system

$\dot{\vec{x}} = A\vec{x} + B$ with initial conditions $\vec{x}(0)$

m-file of the Derivative

```
function delx=diffx(t,xy)

x=xy(1); y=xy(2);
c=0.1;  m=1; k=3;
w=3; f=0.5;
ydot=-c/m*y-k/m*x+f/m*sin(w*t);
xdot=xy(2);
delx=[xdot, ydot];
```

local variables x and y are equal to the x and y passed in as $(x,y) =$ `xy = [xy(1) xy(2)]`

$\dot{y} = \ddot{x}$ as a function of (x,y)

$\dot{x}$ as a function of (x,y) (Understand why?)

return the function value, the derivative (the vector $\dot{\vec{x}}$).

Solving and Plotting

```
>> [time, pos] =
   ode45('diffx',
   0, 150, [2 0])
>> plot(time,
        pos(:,1))
```

pos has both $[\vec{x}, \vec{y} = \dot{\vec{x}}]$
So select the first column.