Lecture 6 for Math 398 Section 952: Differential Equations

Thomas Shores
Department of Math/Stat
University of Nebraska
Fall 2002

Most of the functions that we study in this lesson can be found in one category of the help files. Start by issuing a "helpwin" command and clicking on "matlab/funfun".

Differential Equations

Many tools are available here. We will use just a few. In this lesson we'll focus on some general purpose numerical method for solving odes. In the next lesson, we'll examine some problems that require more specialized methods.

Phase planes for ordinary differential equations.

Consider the system that results from the pendulum problem whose governing differential equation is $\theta(t)'' + \sin\theta(t) = 0$. Make the change of variables $x_1 = \theta$, $x_2 = x_1'$ to obtain the system

$$\begin{aligned} x_1' &= x_2 \\ x_2' &= -\sin x_1. \end{aligned}$$

Now edit the function file fcn.m to obtain a function with return value `[x(2), -sin(x(1)]`. Let's find three solutions to the system and plot them. Start up Matlab and execute these commands:

```
   > tspan = [0 10]; % time interval of integration
   > yzero1 = [1; 1]; yzero2 = [-5; 2]; yzero3 = [5; -2]; % initial
conditions
   > [t1, y1] = ode45(@fcn, tspan, yzero1); % solutions
   > [t2, y2] = ode45(@fcn, tspan, yzero2);
   > [t3, y3] = ode45(@fcn, tspan, yzero3);
   > plot(y1(:,1), y1(:,2), y2(:,1), y2(:,2), y3(:,1), y3(:,2)); %
phase plane plot
   > hold on
```

This is helpful, but we would have a better understanding if we could visualize the "direction flow" of the system. This is accomplished by a direction field, which we construct next by using a simple function I wrote:

   >

Stiff differential equations.

Matlab has no problem with higher dimensional systems as well. Consider the following problem, which has three positive parameters, a, b and c, in it (the Robertson chemical model for a reaction between three chemicals). Here the prime means derivative with respect to time:

$$\begin{aligned} x_1' &= -ax_1 + bx_2x_3 \\ x_2' &= ax_1 - bx_2x_3 - cx_2^2 \\ x_3' &= cx_2^2 \end{aligned}$$

First edit fcn.m to match the right hand side of this system.We will accomodate these parameters by declaring them to be global, although we could use Matlab ode solvers and pass these extra parameters to the function fcn (see the documentation). Inside the function, declare a, b, c to be global variables. Then execute the following:

```
  > global a
  > global b
  > global c
  > a = 0.04
  > b = 1e4
  > c = 1e7
  > tspan = [0 3]
  > yzero = [1;0;0]
  > [t, y] = ode45(@fcn, tspan, yzero);
  > subplot(121);plot(t,y(:,2))
  > [t, y] = ode15s(@fcn, tspan, yzero);
  > subplot(122);plot(t,y(:,2))
```

Clearly, ode45 has a problem with this function, which ode15s does not. This is an example of a "stiff ode" (note the sudden change) and ode15s is designed to handle such problems.

As one more example, we consider the Rössler system

$$\begin{aligned} x_1' &= -x_2 - x_3 \\ x_2' &= x_1 + ax_2 \\ x_3' &= b + x_3\,(x_1 - c) \end{aligned}$$

Edit a function file fcnR.m to match this by saving the fcn.m file as fcnR.m. Here are some interesting graphs:

```
  > a = 0.2, b = 0.2, c = 2.5;
  > tspan = [0 100], yzero = [1;1;1]
  > options = odeset('AbsTol', 1e-7, 'RelTol', 1e-4); % some of the
ode parameters
  > [t,y] = ode45(@fcnR, tspan, yzero, options);
```

```
   > subplot(221),plot3(y(:,1),y(:, 2), y(:, 3)), title(’c = 2.5’),
grid
   > subplot(223), plot(y(:,1),y(:,2)), title(’c = 2.5’)
   > xlabel y_1(t), ylabel y_2(t)
   c = 5
   > [t,y] = ode45(@fcnR, tspan, yzero, options);
   > subplot(222),plot3(y(:,1),y(:, 2), y(:, 3)), title(’c = 5’), grid
   > subplot(224), plot(y(:,1),y(:,2)), title(’c = 5’)
   > xlabel y_1(t), ylabel y_2(t)
```

End of Lesson