Lab 3

Exploration of Second-Order ODE Models

MATLAB routines like `ode45` only provide numerical solutions to IVPs involving first-order ODEs. Fortunately, you can typically convert a higher-order ODE into something that only involves first derivatives by introducing new variables. In this laboratory session you will

1. Learn how to reduce a higher-order IVP to a first-order system.
2. Learn how to implement `ode45.m` for the resulting systems.
3. Analyze solutions using different graphing techniques.

Reducing a Higher-Order ODE

Let's start with an example. Consider the IVP

$$y'' + 4y' + 3y = \cos t \quad \text{with} \quad y(0) = -1, \quad y'(0) = 0.$$

To reduce the order of the ODE, we introduce the function

$$v(t) = y'(t).$$

As a result, $v' = y''$ so that the ODE can be written as $v' + 4v + 3y = \cos t$. This equation involves only first-order derivatives, but at the expense of having two unknown functions y and v instead of one. The initial conditions must also be transformed into initial conditions for y and v, so we get the system

$$\begin{aligned} y' &= v \\ v' &= \cos t - 4v - 3y \end{aligned} \quad \text{with} \quad \begin{aligned} y(0) &= -1 \\ v(0) &= 0 \end{aligned}$$

We are now ready to implement the IVP in MATLAB.

MATLAB Implementation

The following code shows how to solve the IVP above for $0 \le t \le 12$ using the MATLAB routine `ode45`.

```
function LAB3ex1
t0 = 0; tf = 12;                         % initial and final times
y0 = -1; v0 = 0;                         % initial conditions
[t,Y] = ode45(@f,[t0,tf],[y0,v0]);
y = Y(:,1); v = Y(:,2);                  % extract y and v from output matrix Y
figure(1); plot(t,y,'b+-',t,v,'ro-'); % time series for y and v
legend('y(t)','v(t)=y''(t)'); grid on % note the use of '' for '
xlabel('t');
figure(2); plot(y,v);                    % phase plot
xlabel('y'); ylabel('v=y'''); grid on
%-----------------------------------
function dYdt = f(t,Y)
y = Y(1); v = Y(2);
dYdt = [ v ; cos(t)-4*v-3*y ];
```

Notice how the initial conditions are specified in the calling sequence of `ode45`, in the form of an array `[y0,v0]` (`[y0;v0]` is also valid). The routine `ode45` returns a vector `t` containing the times in the interval [0,12] at which the solution was computed, and a matrix `Y` with two columns of the same length as `t`, one for each of `y` and `v` (in that order). Note how these approximations for y and v are retrieved from `Y`.

The time series for both y and v are plotted in the MATLAB figure window 1, and the phase plot showing v vs y is shown in the figure window 2.

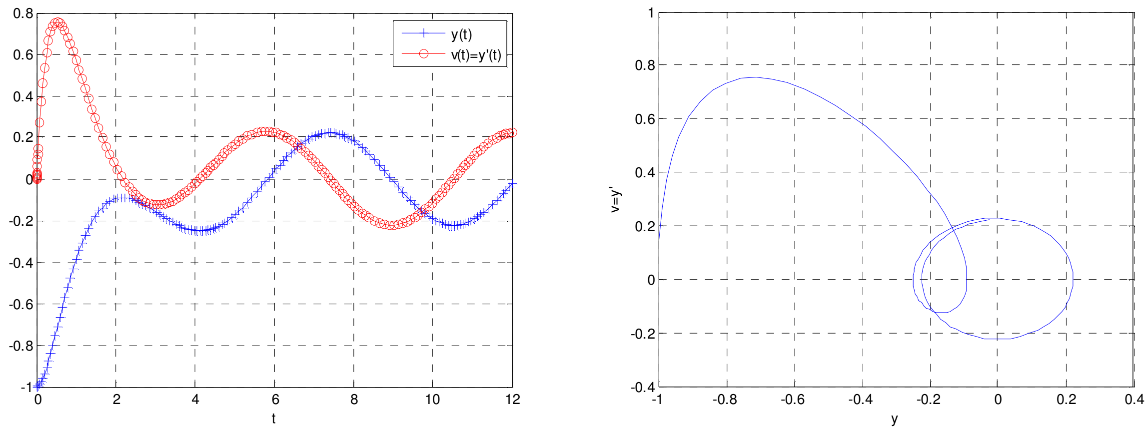


Figure 1. Time series $y = y(t)$ and $v = v(t) = y'(t)$ (left) and phase plot $v = y'$ vs. y (right).

1. Enter and run this program in MATLAB and use the output to answer the following questions.

   a. For what approximate value(s) of t does y reach a local maximum in the window $0 \le t \le 12$? Check by reading the `Y` matrix. Where are these points on the phase plane? Explain.

   b. What seems to be the long term behavior of y? Check your conjecture by extending the time span to $t = 100$. Explain how your answer is justified by each of the output plots.

   c. Modify the initial conditions. Does the long term behavior of the solution change? If yes, in what way?

   d. Describe the physical characteristics of a spring-mass system modeled by this ODE.

Nonlinear Problems

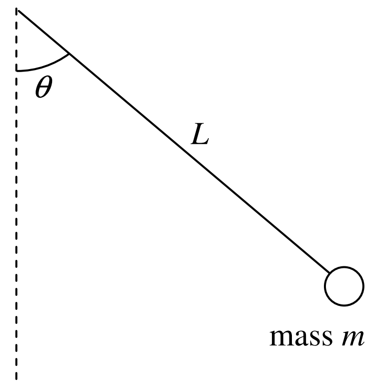
Nonlinear problems do not present any additional difficulty from an implementation point of view (they may present new numerical challenges for integration routines like `ode45`). As an example consider the modified problem

$$y'' + 4y^2 y' + 3y = \cos t \quad \text{with} \quad y(0) = -1, \quad y'(0) = 0.$$

This ODE is very similar to the original one except for the y^2 factor on the left-hand side. Because of this factor, the ODE is nonlinear, however there is little change to the implementation in MATLAB. The top part of `LAB3ex1` is not modified while the ODE definition becomes

```
function dYdt = f(t,Y)
y = Y(1); v = Y(2);
dYdt = [ v ; (cos(t)-4*y^2*v-3*y) ];
```

Pendulums



Previously we have used the approximation $\sin\theta \approx \theta$ for small angles θ to obtain a linear ODE that models the behavior of a pendulum. Now suppose we want to obtain a model that is valid for larger swings of the pendulum.

2. Suppose the pendulum arm is L meters long with a mass m attached at the end. Let θ be the angle formed between the pendulum arm and a vertical line extending downward from the point of rotation.

   a. Show that the total energy of the pendulum at any point is given by

   $$E = \tfrac{1}{2} mL^2 \left(\theta'\right)^2 + mg\left(L - L\cos\theta\right).$$

   b. Use the law of conservation of energy to derive the nonlinear ODE

   $$\theta'' + \frac{g}{L}\sin\theta = 0.$$

   c. Reduce the order of this ODE by introducing the function $\omega(t) = \theta'(t)$ and convert the initial conditions for a pendulum starting at a 45° angle and released at a rate of 0.5 radians/sec to initial conditions for your resulting system.

   d. Use your system and initial conditions with a pendulum of length $L = 0.5$ meters to modify the following program and obtain a numeric solution. What is the period of oscillation? What is the maximum angle the pendulum makes with the vertical? Does the value of m affect the results? If so, how? If not, why not?

```
function LAB3ex2
theta0 = ??; omega0 = ??;                % fill-in ICs (using radians)
[t,Y]=ode45(@f,[0,20],[theta0,omega0]);
theta = Y(:,1); omega = Y(:,2);
figure(1); plot(t,theta,'b+-',t,omega,'ro-');
legend('??','??'); grid on               % fill in appropriate legend
xlabel('??');                            % fill in appropriate axis
figure(2); plot(theta,omega);
xlabel('??'); ylabel('??'); grid on      % fill in appropriate axes
%------------------------------------------------
function dYdt = f(t,Y)
g = 9.81;                                % gravitational accel. [m/s^2]
L = 0.5;                                 % pendulum length [m]
theta = Y(1); omega = Y(2);
dYdt = [ ?? ; ?? ];                      % fill-in ODE
```

   e. Change the initial condition on the rate of change of θ to 8.17 radians/sec, 8.175 radians/sec, and 8.18 radians/sec. What does the result imply about the behavior of the pendulum in each case? Does this make sense? Why or why not?

f. Use the following MATLAB options and redo the previous question. What do these results imply about the motion of the pendulum? What has changed in the numerical procedure?

```
options = odeset('Reltol',1e-12,'AbsTol',1e-12);
[t,Y]=ode45(@f,[0,20],[theta0,omega0],options);
```

g. Examine the phase plot for an initial rate of 2 radians/sec. What does this plot say about the behavior of the pendulum? Explain.

h. Modify the program to solve the linear version of the ODE (from assuming that $\sin\theta \approx \theta$. Try different initial conditions. Explain what is unrealistic about the results from this model.

Forced Oscillations and Resonance

3. Modify the program `LAB3ex1` and enter parameters to answer each of the following questions. *Print out and label both the time series and phase plane plots to hand in.*

   a. Create an undamped spring-mass system with natural angular frequency of the spring equal to the angular frequency of the external force. This is a system in “resonance.” Explain what you see in both plots in terms of the spring-mass system.

   b. Modify the previous system so that the angular frequency of the external force is slightly different. You should see “beats,” that is fast oscillations that repeatedly grow and decrease in amplitude over longer periods of time. You may need to make your plot cover more or less time to see what is happening. Find a spring-mass system for which the rapid oscillations occur at 12 cycles per minute and whose amplitude increase and decrease at a rate of one beat per minute. Write the mass and spring constants and the angular frequency of the external force on your printouts.

   c. Add a small damping force to your models of each of the situations in Parts a and b. Explain what happens by interpreting your plots.

4. Consider Example 5 on p. 218 of the text. Note especially how the variables are defined by Figure 3.6.7.

   a. Explain why x is the appropriate variable to differentiate (so that x'' shows up in the differential equation) and why a different quantity, $x - y$, is multiplied by the spring constant.

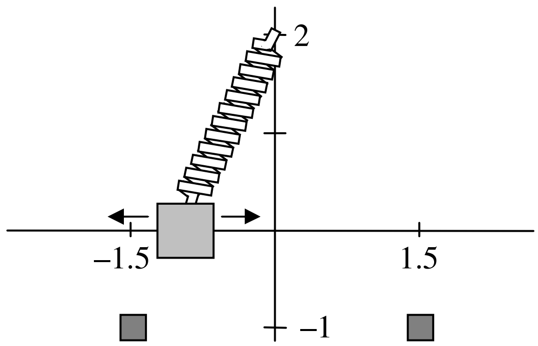
   b. Use MATLAB to numerically solve the differential equation using the given constants and the velocity that results in resonance. Make separate plots for velocities that are slightly larger and smaller than the resonant frequency. Explain what the difference means for the car in question.

   c. Explain why resonance occurs in terms of the road, suspension, and the car.

   d. Suppose the shocks are now connected. Give the appropriate differential equation for the car driving down the same road. Then use MATLAB to numerically solve this equation and graph the result. Explain the difference from the situation with no shock absorber.

Another Nonlinear Model

5. Consider the motion of a 0.2 kg mass that can slide freely along the x-axis. The mass is attached to a spring that has its other end attached to the point $(0, 2)$ on the y-axis, where coordinates are measured in meters. The spring is designed to collapse on itself so its rest length is effectively zero. In addition, the mass is made of iron and is attracted to two identical strong magnets, one located at the point $(-1.5, -1)$ and the other at $(1.5, -1)$. See the figure below.



Recall that the acceleration $x''(t)$ is determined by Newton's second law:

$$\sum \text{forces} = ma$$

Also, each force may be broken into x and y components, so here we will only be interested in the x-components. Recall that Hooke's law says that the force due to the spring is proportional to the distance it is stretched beyond its rest length. Finally, each magnet attracts the mass with a force proportional to the inverse square of the distance between the mass and magnet. Use $k = 0.1$ for the constant of proportionality for the spring, and use $c = 0.2$ for the constant of proportionality for the inverse square law with the magnets.

a. Write the differential equation governing the motion of the mass along the x-axis. Show your work. (Note, you will use the Pythagorean theorem several times.)

b. Use the numerical solver `ode45` in MATLAB with initial conditions

$$x(0) = -1$$
$$x'(0) = 0$$

Print out time series and phase plane graphs of the results. (Hint: You will need to keep careful track of where the forces are positive and negative. You will find the MATLAB function `sign(a)` useful. It returns -1 if $a < 0$, 0 if $a = 0$, and 1 if $a > 0$.)

c. Explain why there are cusps in both graphs. Give one explanation based purely on the physical situation and give another explanation based on the algebraic form of the ODE.

d. If you were to use a much longer time period (say 3 minutes) MATLAB would seem to hang up or crash. If you let it go long enough, it will finish. Explain why `ode45` slows down so much for this last portion of the approximation. (Hint: think in terms of the adaptive step size and what types of things can cause problems. Then reflect back on your answer to Part c.)

e. Find initial conditions that result in a different long-term behavior. Print out your graph and explain what is happening.