Mech 221: Computer Pre-Lab 5

Hand in the solutions to the two questions in the pre-lab at the *beginning* of the lab.

In the upcoming lab, we will continue to work on solving systems of Differential Equations using MATLAB. We will consider specifically systems in which there are forcing terms that depend discontinuously on time and on the variables. This pre-lab will cover the following topics:

- How to use "if" statements in MATLAB.
- A discussion of how the MATLAB routine `ode45` works.
- Solving differential equations analytically that include discontinuous terms.

if-else blocks in MATLAB

In some computations it may be necessary to carry out one procedure based on whether a condition is satisfied, and perhaps carry out a different procedure if the condition is not satisfied. In general the calling sequence in MATLAB is as follows:

```
if (<condition>)
    <code to executed if <condition> is true>
else
    <code to executed if <condition> is false>
end
```

The conditions have the same syntax as those in `while` loops discussed in pre-lab #3. It should be noted that if you want to do nothing in the event that your condition fails, you have the option to leave out the `else (...)` portion of your code, giving you:

```
if (<condition>)
    <code to executed if <condition> is true>
end
```

Consider the absolute value function as an example. How do we determine the absolute value of a number x? From previous courses, we know that *if* $x \geq 0$, the absolute value of x, $|x|$ equals x or *else* $|x|$ equals $-x$. If we were going get MATLAB to compute the value of $|x|$, we would create the following .m file function:

```
function y = myAbs(x)

if (x >= 0)
    y = x
else
    y = -x
end
```

This returns the same result as the MATLAB built-in function `abs` when x is a scalar.

How `ode45` works

The MATLAB routine, `ode45`, approximates differential equations using a six stage, fourth order accurate Runge-Kutta method. The six stages also allow the computation of a fifth order method. A comparison of these two methods allow an estimation of the error made in a single time step. If this error is too large (larger than a default local error tolerance that can be changed by the user in an option) then the time step is repeated with a smaller time step (which would give a smaller error). This process is repeated until a step is taken with an acceptable error. If the predicted error is much smaller than the tolerance, subsequent step sizes will be taken cautiously to be larger.

Consider solving the problem

$$\ddot{y} + y = 0, \quad y(0) = 1, \dot{y}(0) = 0 \tag{1}$$

as you did in lab #4. The call

```
[t,y] = ode45('vfunc',[0 10], [0 1]');
```

approximated this problem. We can get an idea of how `ode45` works by looking at the variable time steps h that it took. To do this, the following code is used:

```
N = max(size(t))-1;
h = t(2:(N+1))-t(1:N);
plot(t(1:N),h)
```

The computation takes $N = 72$ steps, and the plot generated above is shown in Figure 1 (axes labels were added and the line thickness and fonts were

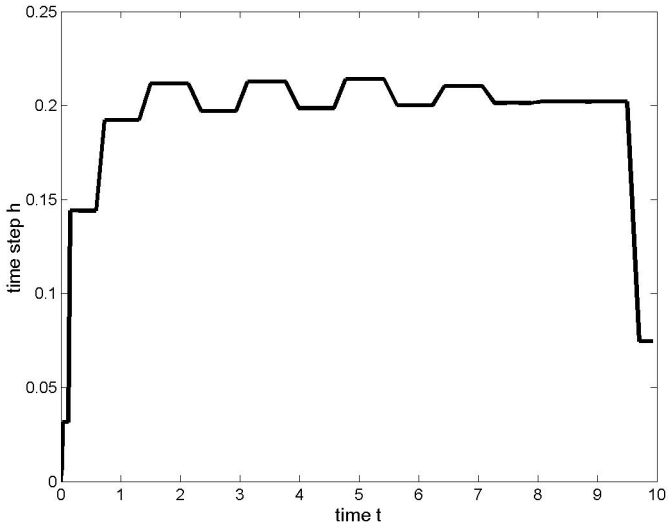

Figure 1: Time step sizes for the unforced oscillator (1) computed using `ode45`.

changed using the figure interface). Notes that the time steps start small (the MATLAB default 0.0001) then increase slowly as MATLAB determines that these step sizes are smaller than needed for the specified default accuracy. The step sizes then plateau at the level that provide the default accuracy. The dip in the size of time step at the end is just to reach the specified time 10 in the `ode45` call. Similar time step plots will be shown and discussed for two other problems below.

A Discontinuously Forced Oscillator

There are many examples of mathematical modelling where 'switching' plays an important role. By switching, we mean a discontinuous change of a system parameter or forcing function. For example, the current flowing through a circuit as it is closed, then opened. The Heaviside function, $H(t)$, is an important example of a discontinuous function and is defined as follows:

$$H(t) = \begin{cases} 1 & \text{if } t \geq 0 \\ 0 & \text{if } t < 0 \end{cases} \tag{2}$$

Consider the oscillator equation subject to a Heaviside forcing at time,

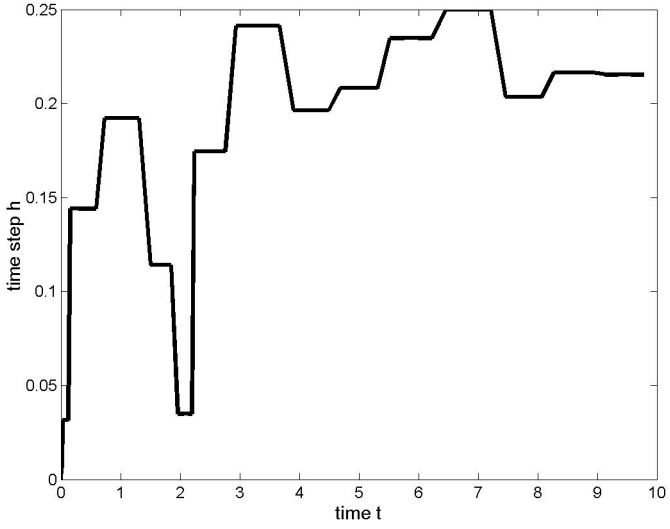

Figure 2: Time step sizes for the discontinuously forced oscillator (3) computed using `ode45`.

$t = 2$.

$$\ddot{y} + y = H(t-2), \quad y(0) = 1, \dot{y}(0) = 0, \quad 0 \le t \le 10 \tag{3}$$

You will solve this problem analytically as a pre-lab question below, and compute solutions numerically with `ode45` in the lab. When you do the computation you will find that it takes $N = 76$ steps, and a plot of time steps generated as described above is shown in Figure 2. Note the same features as in Figure 1 except that much smaller steps are used near $t = 2$. The high order methods used in `ode45` are based on differentiability of the Differential Equation right hand side function. When this function is discontinuous, the method can recognize this and must take smaller time steps to get the desired accuracy.

How would you solve this problem analytically? You have previously encountered problems where you have had to solve Differential Equations with discontinuous terms, but so far they have only been scalar, first order examples. The same procedure is used for higher order problems and systems. Let us review this procedure for this example. First you solve the problem for all $t \le 2$, (i.e. before the Heaviside force is activated). In other words,

the following linear, homogeneous, constant coefficient problem is solved:

$$\ddot{y} + y = 0, \quad y(0) = 1, \dot{y}(0) = 0, \quad 0 \leq t \leq 2 \tag{4}$$

Let $y = y_1(t)$ be the solution for the above problem. Then you take the data $(y_1(2), \dot{y}_1(2))$ which you have just solved for. Use it as the initial condition for the problem when $t \geq 2$,(i.e: when the Heaviside forcing has been activated):

$$\ddot{y} + y = 1, \quad y(2) = y_1(2), \dot{y}(2) = \dot{y}_1(2), \quad 2 \leq t \leq 10 \tag{5}$$

Use the Method of Undetermined Coefficients to solve this problem. Call the solution for this problem $y(t) = y_2(t)$. We now piece together the total solution defining it as follows.

$$y(t) = \begin{cases} y_1(t) & \text{if } 0 \leq t \leq 2 \\ y_2(t) & \text{if } t > 2 \end{cases} \tag{6}$$

You can piece together approximations of these two smooth (no discontinuities) problems using two calls to `ode45` but in this example there is no reason to do so.

Question 1

- Find $y_1(t)$,$y_2(t)$ and thus find $y(t)$ on $0 \leq t \leq 10$ for the switching oscillator described above. You will use your answer to check your MATLAB code in the lab.
- Rewrite (3) as a first order vector differential equation. You will need to remember this for the lab.
- *Hand in the two part solution for $y(t)$ found above and the first order system version of (3).*

An Oscillator with Friction

Consider an oscillator subject to friction. We consider here not damping (which is often modelled as a force proportional to velocity) but surface friction, which is a constant force acting in the direction opposite to the motion or holding an object at rest unless that force is exceeded. In this discussion we don't consider the difference between static friction and dynamic friction.

A scaled equation that governs an oscillator subject to friction is given by:

$$\ddot{y} + y = F(y, \dot{y}) \quad y(0) = 1, \dot{y}(0) = 0 \tag{7}$$

The frictional force is scaled to $\beta = 0.3$. If $\dot{y} > 0$ then $F = -\beta$; if $\dot{y} < 0$ then $F = \beta$ (the friction force opposes the motion if the object is in motion). If $\dot{y} = 0$ (the mass is not moving) there are three cases to consider:

1. if $|y| \leq \beta$ then $F = y$ (the spring force $-y$ cannot overcome the frictional force and the mass stays at rest).
2. if $y > \beta$ then $F = \beta$.
3. if $y < -\beta$ then $F = -\beta$.

You will solve this problem analytically as a pre-lab question below, and compute solutions numerically with `ode45` in the lab. When you do the computation you will find that it takes $N = 155936$ (!) steps, and a plot of time steps generated as described above is shown in Figure 3. Time steps get shorter at the discontinuity in F when the mass changes direction. The time steps are extremely short ($h \approx 0.0001$) when the mass stops moving. Explaining this behaviour is part of a question on the lab.

To solve this problem analytically, we would do the following. Given the initial conditions evaluate the behaviour of the oscillator. Is it moving? Will it begin to move if it is not already moving? (that is, will it overcome friction?) In the case when it will move and is not already moving, you solve the oscillator equation as though the object were already moving. In other words, If $\dot{y} = 0$, then evaluate the differential equation to see if the object will accelerate, and in which direction. If it will accelerate, then solve the differential equation as though $\dot{y} \neq 0$. Get the solution until the next time when $\dot{y} = 0$, then re-evaluate. If it will not accelerate, then you know the oscillator is stationary for all time following.

Question 2: A Oscillator with Friction

- Solve (7) with the frictional force with $\beta = 0.3$. Be sure to identify the final position of the mass (when friction holds it in place). You will use your answer to check your MATLAB code in the lab.
- *Hand in the solution above, identifying clearly the final position of the mass.*

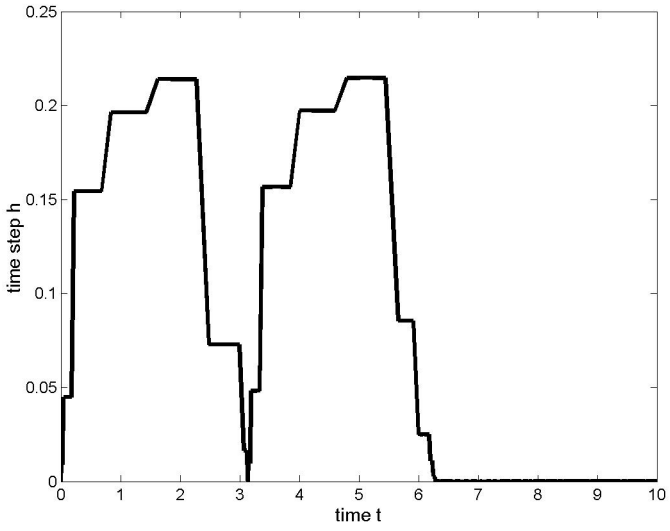

Figure 3: Time step sizes for the oscillator with surface friction (7) computed using `ode45`.

Coming up in Lab #5

Completing the pre-lab will prepare you for the lab, in which the following will be covered:

- Using an "if" statement within a function.
- Solving a DE that has a non-continuous time-dependence.
- Analyzing the numerical solution to the oscillator with friction example.