Lab Exercise #3 - Loops

INSTRUCTIONS

Complete the exercises below and upload them to Canvas as a single MATLAB script file using the naming convention "ENGR131_21F_Lab##_abc123.m", replacing abc123 with your Case ID, and ## with the two-digit lab number.

For example, if Dr. Williams were submitting Lab 3 it would be ENGR131_21F_Lab03_mrw8.m

For your script, please perform the following:

1. Separate each question into separate, runnable sections using the "%%" comment notation.
2. You may use the code and notes from class, the textbook, MATLAB's documentation, and anything you find using Google to solve these problems.
3. Use comments as appropriate to indicate your thoughts and how your code works (or is supposed to work). This is 5 points (10%) of your grade.

QUESTIONS

There are 2 questions for this lab.

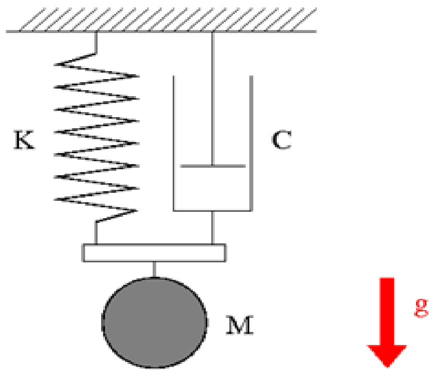
1. PLOTTING REPEATED BEHAVIORS (30 PTS)

The spring-mass-damper system shown in fig. 1a, will behave over time as shown in fig. 1b when released from a particular height above its rest point as governed by the equation in eq. 1.

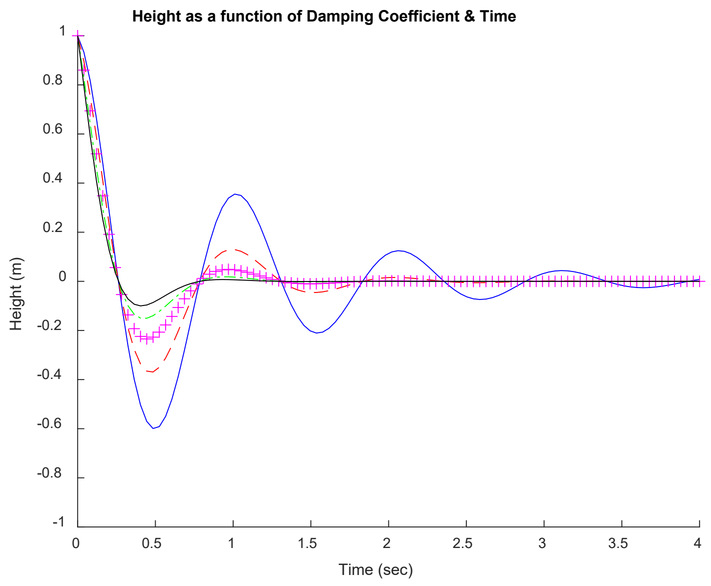
$$H = e^{-\gamma t} \mathrm{a} \cos(\omega t) \qquad \text{(eq. 1)}$$

Where H is the height above rest, γ is the damping coefficient, a is the starting height before release, ω is the frequency of oscillation, and t is the time from release. For this exercise:

- The damping coefficient will be variable and computed from input by the user.
- t will be a vector of values from 0 to 4 seconds
- a = 1
- $\omega = 6$.



(a)



(b)

```
Please enter a starting damping coefficient: -2
You entered -2.00.  Please enter a positive starting damping coefficient.

Please enter a starting damping coefficient: 1
You entered 1.00.  This is satisfactory.

Please enter the number of damping values to plot: 5.6
You entered 6.  This is satisfactory.
```

(c)

Fig. 1. Diagram of system (a) and its height above equilibrium over time using a series of different damping coefficients (b). An example of the prompt to the user for input and error checking to ensure a positive damping is entered as well as correcting the number of damping values to plot to be a whole number.

For this Lab, you will be building on the work of Lab 2, using the same height calculation function, but repeating the calculation and plotting the behavior using a number of different damping coefficients on the same plot.

Recreate the plot in fig. 1b by following the steps below:

a. Create a new script and copy your CalcPosition function from Lab 2 over to this new script (remember, functions stored inside the script file are not available to other scripts).
b. Prompt the user to enter a starting damping coefficient and echo the value back to the user. Use a **while-loop** with an **if-else selection** to check that the input is positive and repeat the prompt until a positive value is entered. Provide feedback about the value entered based on its value similar to as seen in Fig. 1c (3 pts).
    a. To check your work and recreate Fig 1b, use a starting damping coefficient of 1.

c. Repeat step 1b, this time asking the user to enter the number of damping coefficients that should be plotted. Correct any decimal entries to be whole numbers (we're not interested in portions of plot) and check that the number of plots is greater than 0 (because, otherwise, what's the point?) (3 pts).

d. Create a vector of evenly spaced damping coefficients that starts with the starting damping entered by the user and ends with a value that is the starting damping X the number of plots and contains as many values as the number of plots that were entered. Ex: If the user entered a starting damping of 1 and 3 plots, the vector should be [1,2,3]. (2 pts)

e. Create a Time vector that ranges from 0 to 4 and contains 100 evenly spaced values (1 pts)

f. Calculate and plot the height of the system over time for each of the damping coefficients that you calculated

    a. Use a **for-loop** that runs from 1 to the length of the damping coefficient vector created in 1d (6 pts)

    b. Compute the height for each damping coefficient by calling the CalcPosition and passing in your Time vector and the damping coefficient for that iteration (2 pts).
    Ex: DampingCoeff(1,i)

    c. Use a **switch statement** based on the for-loop iterator variable to assign a character array to the color and linetype code you wish to plot for that iteration. This switch statement should have 5 distinct options and a catchall option for anything over 5. (5 pts)
    Ex: switch i
    ```
        case 1
            LineColor = 'b-';
        ….
      end
    ```

    d. Set figure 1 as the active figure (the first time you use this, it will create the figure, the subsequent times, it will "grab" the figure and make it active for the following plot commands.

    e. Set the hold on toggle (so you don't overwrite your plots with each iteration)

    f. Plot the system behavior using the plot command using the character array you set in the switch statement to specify the color and line type codes (3 pts).

g. Add appropriate labels and a title (3 pts)

h. Set the Height limits of the plot to -1 to 1 (1 pts)

i. Turn off the hold toggle (1 pt)

Hints:

- The instructions are in a particular format that should be somewhat familiar. Use this to help you organize the order of your commands.
- Do not be afraid of the character array. Variable = 5 and Variable = 'b-' are the same thing, just with different values and variable types
    - Similarly, you can use a character array anyplace you would normally write the text it contains.

2. Using loops to compute complex outcomes (15 pts)

While loops can be used to compute unknown values be rapidly repeating calculations until the solution is found.

For this question, perform the following:

a. Use a while-loop to compute the damping coefficient required such that, for all other parameters being the same as in Q1, the overshoot past equilibrium (height = 0) is 25% or less of the starting height (10 pts)
    a. Start with a value you know will overshoot equilibrium by a fair amount (you can use your code from Q1 to determine this).
    b. Each time through the loop
        i. Call the CalcPosition function
        ii. Increment the damping coefficient by 0.1
        iii. Increment a counter

b. When complete:
    a. Report the number of iterations and the final computed damping coefficient (2 pts).
    b. Plot the behavior of the system using this damping coefficient in a new figure to demonstrate the results (3 pts).

Hints:

- Be sure to clear the workspace and the command window for each run of the script (helps with error checking)
- Look at the starting height. Your target is 15% of that below equilibrium.
- Don't forget to initialize both your starting damping coefficient as well as your counter.

Revision	Description	Date
A	Original Document	9/10/2021