Lab #6: Stability Analysis, Frequency Response

INTRODUCTION

In this lab, you and a partner will analyze stability and frequency responses with bode plots. You will learn how to reduce steady-state error and learn how to create a lag compensator.

Please have the following outline for your report:

1. A few sentences of introduction of the topic of the lab.
2. Answers to each problem with concise explanations on your process in solving and outcome.
3. A paragraph concluding the report explaining the goals, what you learned, and any other conclusions.

MATLAB

I)   CRUISE-CONTROL CAR
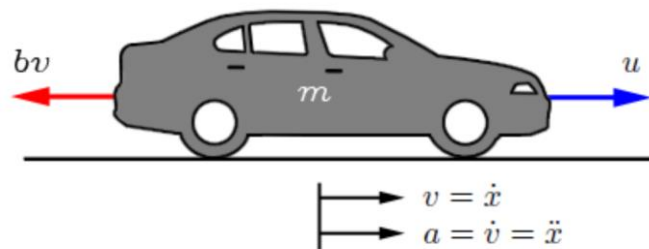
This system has the following properties:

m	Vehicle mass	1000 kg
b	Damping Coefficient	50 N.s/m
r	Reference speed	10 m/s
u	Control Force	500 N

1) Using a proportional controller find the open loop response with a step response and bode plot (if it can be generated) with gain of 1. Note the change in vehicle mass is now 1000 kg. Note that your system looks like this:
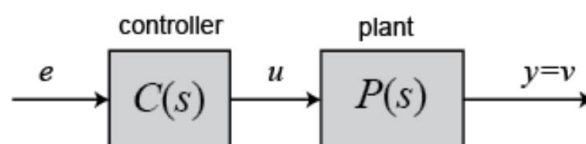
2) What is the current steady-state error of the system? Confirm this from your step response from step 1.
3) Find the gain needed to reduce the steady-state error to less than 2%. Plot the bode diagram of the Proportional controller with the new gain and its closed loop step response.
4) Is this a reasonable step response?
5) Creating a Lag compensator:

   a. The lag compensator adds gain at the low frequencies while keeping the bandwidth frequency at the same place. This is what we need: Larger low frequency gain to reduce the steady-state error and keep the same bandwidth frequency to maintain the desired rise time. The transfer function of the lag controller is:

   $$C_{lag}(s) = \frac{s + z_0}{s + p_0}$$

   b. The pole and zero of a lag compensator needs to be placed close together. Also, steady-state error is will be reduced by a factor z_0/p_0. Let z0=0.1, p0=0.02, and Kp=1000. Plot the bode plot of Kp*Clag*_YourController_. Also plot the closed loop step response. Compare the performance to the previous step response.

SIMULINK

TRAIN SYSTEM: MODELING

We will consider a train consisting of an engine and a car. Assuming that the train only travels in one dimension (along the track), we want to apply control to the train so that it starts and comes to rest smoothly, and so that it can track a constant speed command with minimal error in steady state.

The mass of the engine and the car will be represented by M_1 and M_2, respectively. Furthermore, the engine and car are connected via a coupling with stiffness k. In other words, the coupling is modeled as a spring with a spring constant k. The force F represents the force generated between the wheels of the engine and the track, while μ represents the coefficient of rolling friction.
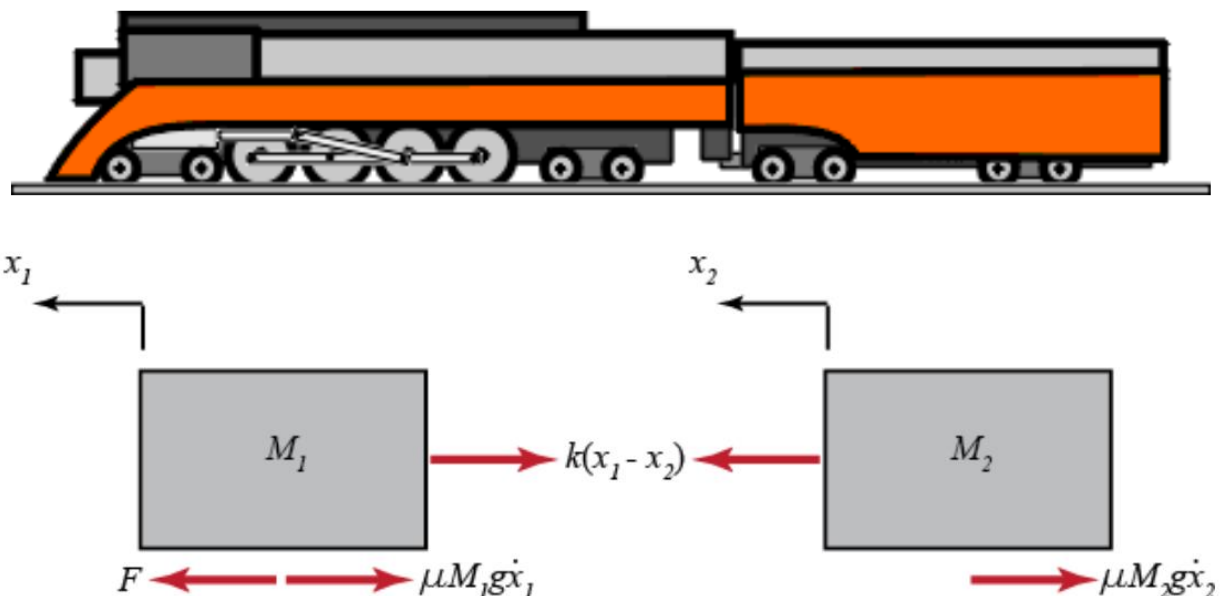
Applying Newton's second law in the horizontal direction based on the above free-body diagrams leads to the following governing equations for the train system.

$$\Sigma F_1 = F - k(x_1 - x_2) - \mu M_1 g \dot{x}_1 = M_1 \ddot{x}_1$$

$$\Sigma F_2 = k(x_1 - x_2) - \mu M_2 g \dot{x}_2 = M_2 \ddot{x}_2$$

1) This set of system equations can now be represented graphically without further manipulation. Specifically, we will construct two copies (one for each mass) of the general expression $\Sigma F = ma$ or $a = (\Sigma F)/m$. First, open Simulink and open a new model window. Then drag two Sum blocks (from the Math Operations library) into your model window and place them approximately as shown in the figure below.

2) The outputs of each of these Sum blocks represents the sum of the forces acting on each mass. Multiplying each output signal by $1/M$ will give us the corresponding acceleration of each mass. Now drag two Gain blocks (from the Math Operations Library) into your model and attach each one with a line from the output of one of the Sum blocks. Label these two signals as "Sum_F1" and "Sum_F2" in order to make your model more clear. This is accomplished by double-clicking in the space above each of the two signal lines and entering the desired label.

3) These Gain blocks should contain $1/M$ for each of the masses. We will define the variables M1 and M2 in the MATLAB workspace, so we can just enter the corresponding variable names in each of the Gain blocks. Double-click on the upper Gain block and enter "1/M1" into the **Gain** field. Similarly, enter "1/M2" in the **Gain** field of the second Gain block.

4) You will notice that the gains did not appear in the image of the Gain blocks, rather the blocks display a value of -K-. This is because the blocks are too small on the screen to show the full variable name inside the triangle. The blocks can be resized so that the actual gain value can be seen. To resize a block, select it by clicking on it once. Small squares will appear at the corners. Drag one of these squares to stretch the block. Your model should appear as below.
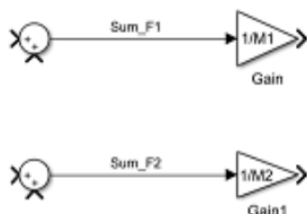

5) The outputs of these gain blocks are the accelerations of each of the masses (the train engine and car). The governing equations we derived above depend on the velocities and displacements of the masses. Since velocity can be determined by integrating acceleration, and position can be determined by integrating velocity, we can generate these signals employing integrator blocks. Drag a total of four Integrator blocks from the Continuous library into your model, two for each of our two accelerations. Connect these blocks and label the signals as shown below. Specifically, the first integrator takes the acceleration of mass 1 ("x1_ddot") as an input and generates the velocity of mass 1 ("x1_dot"). The second integrator then takes this velocity and outputs the displacement of the first mass ("x1"). The same pattern holds for the integrators for the second mass.

6) Now, drag two Scopes from the Sinks library into your model and connect them to the outputs of these integrators. Label them "x1" and "x2".
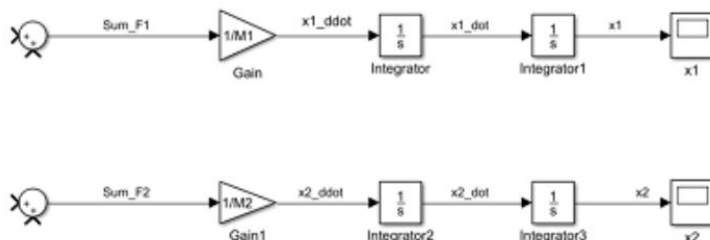

7) Now we are ready to add the forces acting on each mass. First, we need to adjust the inputs on each Sum block to represent the proper number of forces (we will worry about the signs later). Since there is a total of three forces acting on mass 1, double-click on the corresponding Sum block and change the **List of signs**

field to "|+++". The symbol "|" serves as a spacer. There are only 2 forces acting on mass 2, therefore, we can leave that Sum block alone for now.

8) The first force acting on mass 1 is just the input force, F. Drag a Signal Generator block from the Sources library and connect it to the uppermost input of the corresponding Sum block. Label this signal as "F".

9) The next force acting on mass 1 is the rolling resistance force. To generate this force, we can tap off the velocity signal and multiply by an appropriate gain. Drag a Gain block into your model window. Tap off the "x1_dot" signal and connect it to the input of this new Gain block (draw this line in several steps if necessary). Connect the output of the Gain block to the second input of the Sum block. Double-click the Gain block and enter "mu*g*M1" into the **Gain** field. The rolling resistance force, however, acts in the negative direction. Therefore, change the list of signs of the Sum block to "|+-+". Next, resize the Gain block to display the full gain and label the output of the Gain block "Frr1". Your model should now appear as follows.
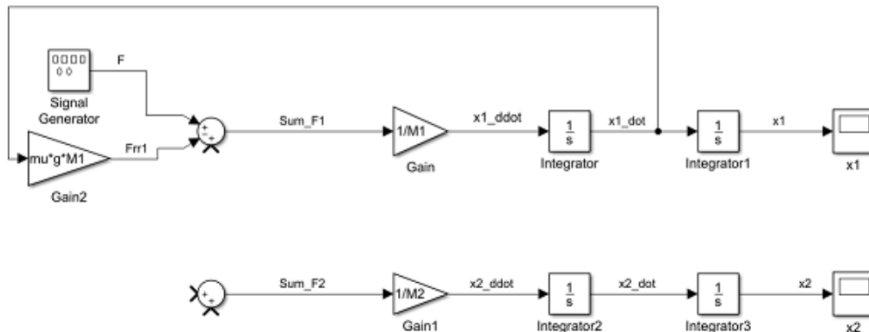
10) The last force acting on mass 1 is the spring force. Therefore, we need to generate a signal $(x_1 - x_2)$ which we can then be multiplied by a gain k to create the force. Drag a Subtraction block (or a Sum block or an Addition block) below the rest of your model. In order to change the direction of this block, right-click on the block and choose **Rotate & Flip > Flip block** from the resulting menu. Alternatively, you can select the block then hit **Ctrl-I**. Now, tap off the "x2" signal and connect it to the negative input of the Subtract block. Also, tap off the "x1" signal and connect it to the positive input. This will cause signal lines to cross. Lines may cross, but they are only actually connected where a small circle appears (such as at a tap point).

11) Now, we can multiply this difference by the spring constant to generate the spring force. Drag a Gain block into your model to the left of the Subtraction block. Change the value of the Gain block to "k" and connect the output of the Subtract block to its input. Then connect the output of the Gain block to the third input of the Sum block for mass 1 and label the signal "Fs". Since the spring force acts on mass 1 in the negative direction, it is necessary to again change the list of signs of the Sum block to "|+--". Your model should appear as follows.
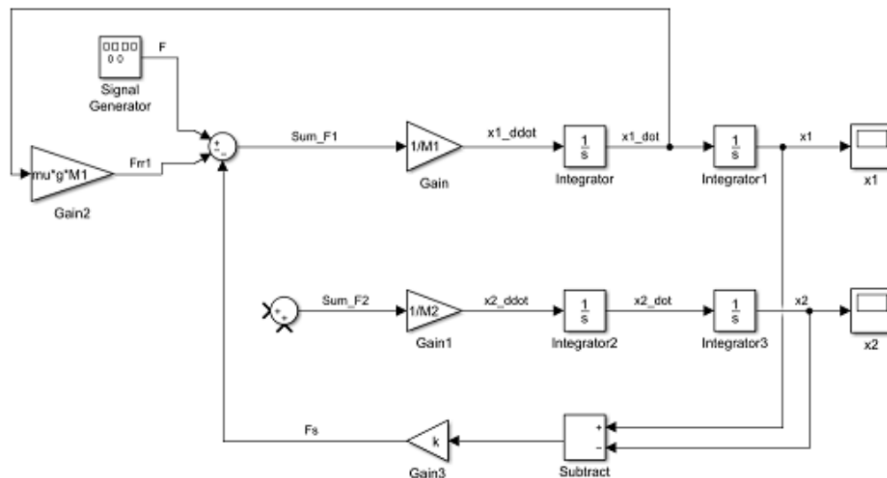
12) We can now apply forces to mass 2. For the first force, we will use the same spring force we just generated, except that it is applied to mass 2 in the positive direction. Simply tap off the spring force signal "Fs" and connect it to the first input of the Sum block for mass 2.

13) The last force applied to mass 2 is its rolling resistance force. This force is generated in an analogous manner to the rolling resistance force applied to mass 1. Tap off the signal "x2_dot" and multiply it by a Gain block with value "mu*g*M2". Then connect the output of the Gain block to the second input of the corresponding Sum block and label the signal "Frr2". Changing the second input of the Sum block to be negative will lead to the following model.

14) Add another Scope block to your model from the Sinks library. Tap a line from the "x1_dot" signal and connect it to the Scope block. Label this scope as "x1_dot" and your model should appear as in the following.
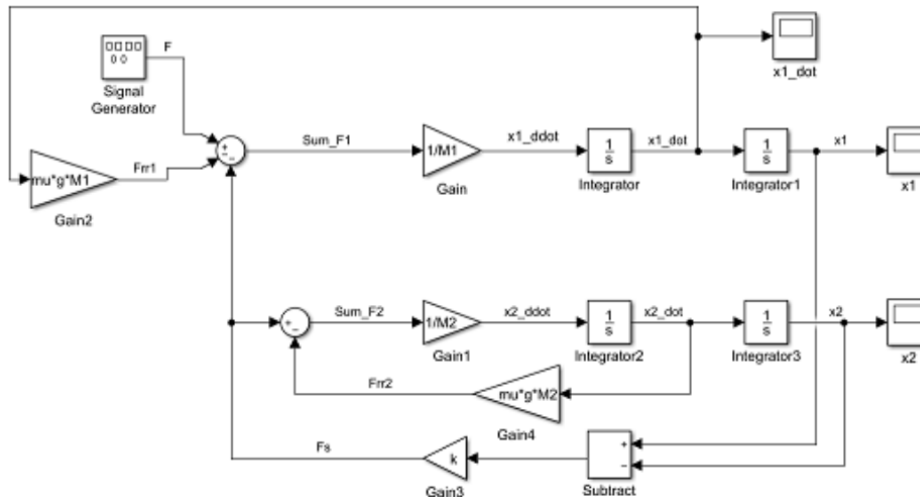
15) Create a new m-file and run it with the following commands: M1=1, M2=0.5, k=1, F=1, mu=0.02, and g=9.8. Simulink will recognize these MATLAB variables for use in the model.

16) Now, we need to give an appropriate input to the engine. Double-click on the Signal Generator block (outputs "F"). Select `square` from the **Wave form** drop-down menu and set the **Frequency** field to equal "0.001". You may leave the **Units** as the default `Hertz`. Also enter "-1" into the **Amplitude** field (positive amplitude steps negative before stepping positive).

17) The last step before running the simulation is to select an appropriate simulation time. To view one cycle of the 0.001 Hz square wave, we should simulate the model for 1000 seconds. Select **Model Configuration Parameters** from the **Simulation** menu at the top of the model window and change the **Stop Time** field to "1000". Close the dialog box.

18) Now, run the simulation and open the "x1_dot" scope to examine the velocity output. Take a screenshot and include in your report and also include a screenshot of the final Simulink model.