LAB 2: Dynamic Systems Modeling in Simulink

Course Number	**MENG 3020**
Course Title	**System Modeling & Simulation**
Semester/Year	**Fall 2024**

Lab/Tutorial Report No.	**Lab 2**
Report Title	**Dynamic Systems Modeling in Simulink**
Section No.	0 NA
Group No.	N/A
Submission Date	September 26th 2024
Due Date	September 26th 2024

Student Name	Signature*	Total Mark
Michael McCorkell		41 / 50

* By signing above, you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a ZERO on the work or possibly more severe penalties. https://academic-regulations.humber.ca/2021-2022/17.0-ACADEMIC-MISCONDUCT

LAB 2 Grading Sheet

Student First Name:	**Student Last Name:**	
Part A: Simulink Model of Mass Lifting System		10 **/10**
Part B: Simulink Model of Vehicle Suspension System		15 **/20**
Post Lab Assignment		11.5 **/15**
General Formatting: Clarity, Writing style, Grammar, Spelling, Layout of the report		4.5 **/5**
Total Mark		41 **/50**

LAB 2: Dynamic Systems Modeling in Simulink

Formatting?

OBJECTIVES

- To learn how to simulate non-linear differential equation model in Simulink.
- To compute the time response of a system to any arbitrary input signal.
- To explore modeling of equations of motion in Simulink.

INTRODUCTION

Simulink provides a graphical user interface that uses various types of elements called ***blocks*** to create a simulation of a dynamic system, which is a system that can be modeled with differential equations whose independent variable is time.

For example, one block type is a *multiplier*, another performs a *sum*, and another is an *integrator*. Its graphical interface enables you to position the blocks, resize them, label them, specify block parameters, and interconnect the blocks to describe complicated system for simulation.

Type **`simulink`** in the MATLAB **`Command`** window to start **Simulink**. The Simulink window opens. To create a new model, click on the **`Blank Model`**. A new untiled window opens for you to create the model. Then open the **`Library Browser`** by selecting it in the drop-down menu under the **`View`** menu. To select a block from the **`Library Broser`**, click on the appropriate library category and a list of blocks within that category then appears. Click on the block name or icon, hold the mouse button down, drag the block to the new model window, and release the button. You can access help for that block by right-clicking on its name or icon and selecting **`Help`** from the drop-down menu.

Refer to ***MENG-2000 Tutorial 2: Simulink Interactive Tutorial*** document for more details on start **Simulink**.

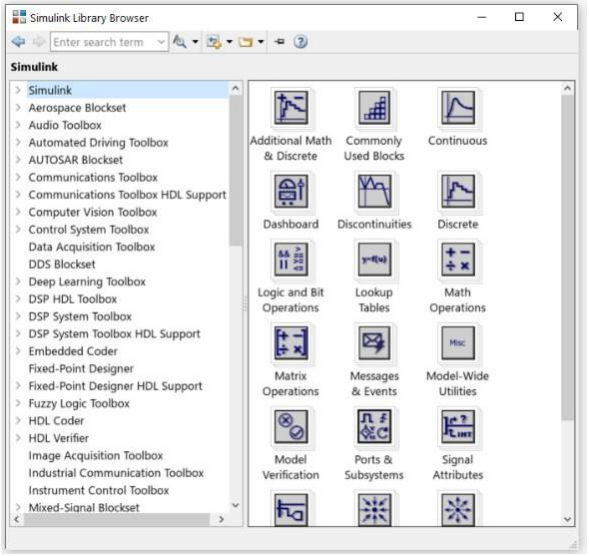


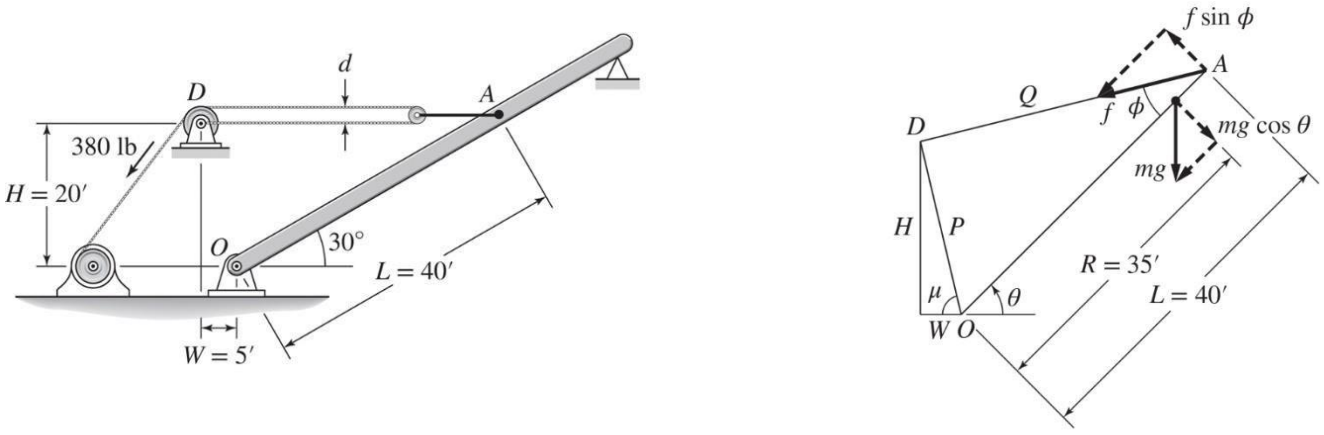
Figure 1: Simulink Library Browser

Simulink model files have the default extension **`.slx`** as of Release 2019a. The **`.slx`** format is a new compressed format that provides smaller file sizes. The **`.mdl`** format is still available as a save option.

You can use the **`File`** menu in the model window to *Open*, *Close*, and *Save* model files. To print the block diagram of the model, select **`Print`** on the **`File`** menu. Use the **`Edit`** menu to copy and paste blocks. You can also use the mouse for these operations. For example, to delete a block, click on it, and press the **`Delete`** key.

Part A: Simulink Model of Mast Lifting System

A mast weighting $500\ lb$ is hinged at its bottom to a fixed support at point O. The mast is $70\ ft$ long and has a uniform mass distribution, so its center of mass is $35\ ft$ from point O. The winch applies a force $f = 380\ lb$ to the cable. The mast is supported initially at angle of $\theta = 30°$, and the cable at A is initially horizontal. Assume that the pulley inertias are negligible and that the pulley diameter d is very small compared to the other dimensions.



(a) (b)

Figure 2: A system for lifting a mast

Part (b) of the figure shows the geometry of the mast at some angle $\theta \geq 30°$ with the diameter d neglected.

From the law of cosine:

$$Q = \sqrt{P^2 + L^2 - 2PL\cos(180° - \mu - \theta)}$$

The equation of motion about the fixed point O is obtained as:

$$J\ddot{\theta} = -mgR\cos\theta + \frac{fLP}{Q}\sin(\theta + \mu)$$

The moment of inertia is:

$$J = \frac{1}{3}m(70)^2 = 25400\ \text{slug} - \text{ft}^2$$

The force f at point A is twice the applied force of $380\ lb$, because of the action of the small pulley. Thus $f = 760\ lb$. With the given values, the equation of motion becomes:

$$25400\,\ddot{\theta} = -17500\cos\theta + \frac{626000}{Q}\sin(\theta + 1.33), \qquad \theta(0) = \frac{\pi}{6}, \quad \dot{\theta}(0) = 0$$

where,

$$Q = \sqrt{2020 + 1650\cos(\theta + 1.33)}$$

The equation of motion cannot be solved in closed form to find $\theta(t)$, so it must be solved numerically using the method to be introduced below.

The goal is to create and run a Simulink model to solve for and plot $\theta(t)$ for $\theta(t) \leq \pi/2$ rad.

1. Start **Simulink** and open a new window.
2. Create the following Simulink model to solve for and plot $\theta(t)$ for $\theta(t) \leq \pi/2$ rad.

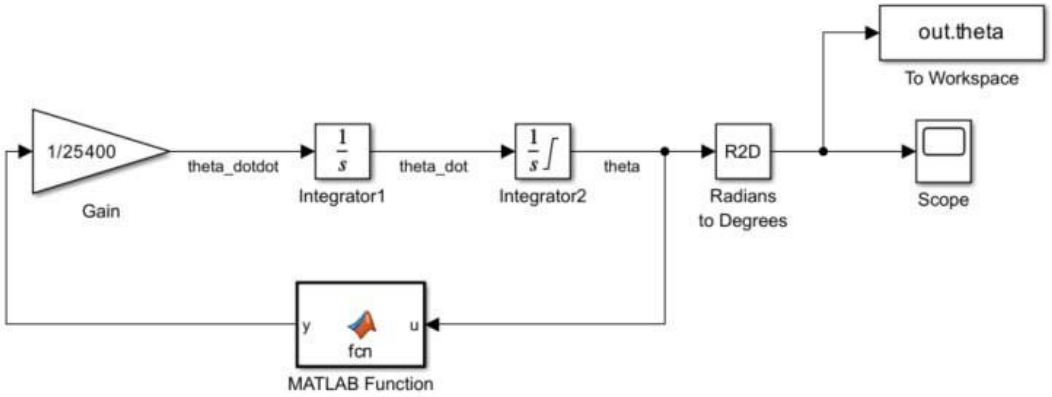


Figure 3: Simulink block diagram model

3. Open the **Simulink Library Browser** window and select the following blocks:
   - The **Gain** block from the **Simulink/Math Operations** category.
   - Change the **Gain** value to `1/25400`.
   - The **Integrator** blocks from the **Simulink/Continuous** category.
   - Set the **Initial condition** of the second integrator block to `30*pi/180` rad, and limit the output theta between **zero** to `90*pi/180` rad.
   - The **Scope** block from the **Simulink/Sink** category.
   - The **To Workspace** block from the **Simulink/Sink** category.
   - Change the output **Variable name** to **theta** and specify the **Save Format** as **Array**.
   - The **R2D** block from **Simulink Extras/Transformations** category.
   - The **MATLAB Function** block from **Simulink/User-Defined Functions** category.
   - Open the **MATLAB Function** block and program the block as below:

```
function y = fcn(u)

y = -17500*cos(u)+626000*sin(1.33+u)/sqrt(2020+1650*cos(1.33+u));
```

4. **Save** the Simulink model file as **Lab2_PartA.slx**.
5. **Run** the simulation for **10 seconds**. Open the **Scope** to check the results. Use **Cursor Measurement** to determine and record the required values in Table 1.

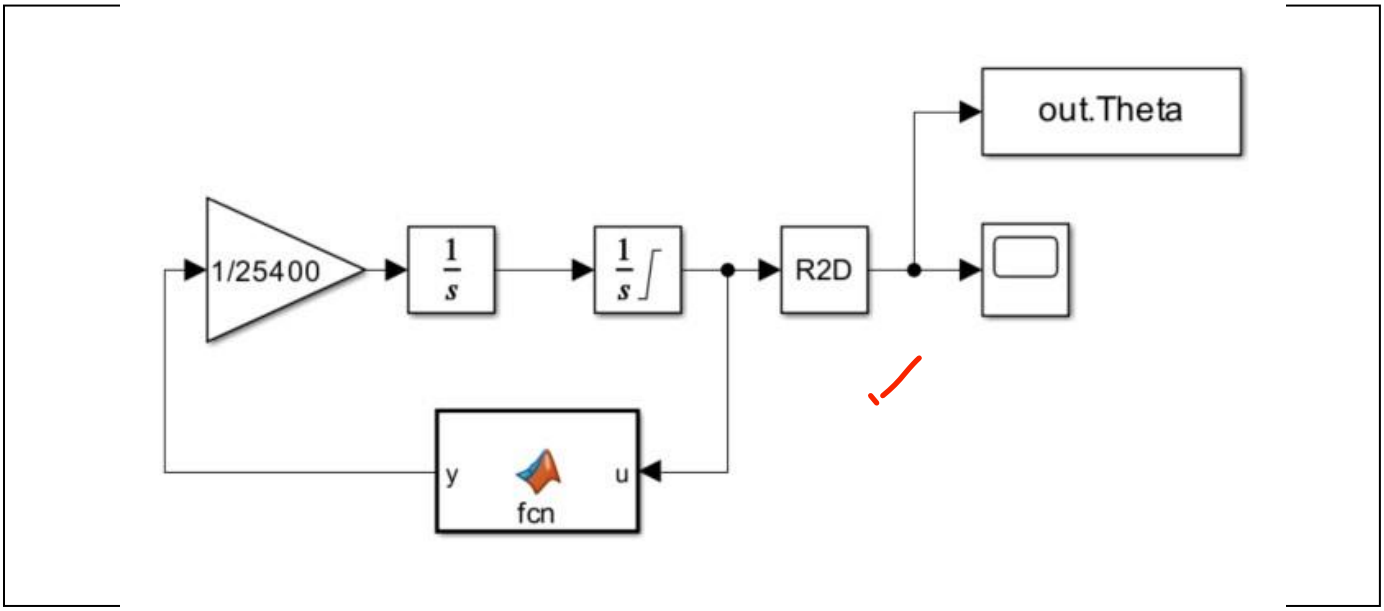
Table 1

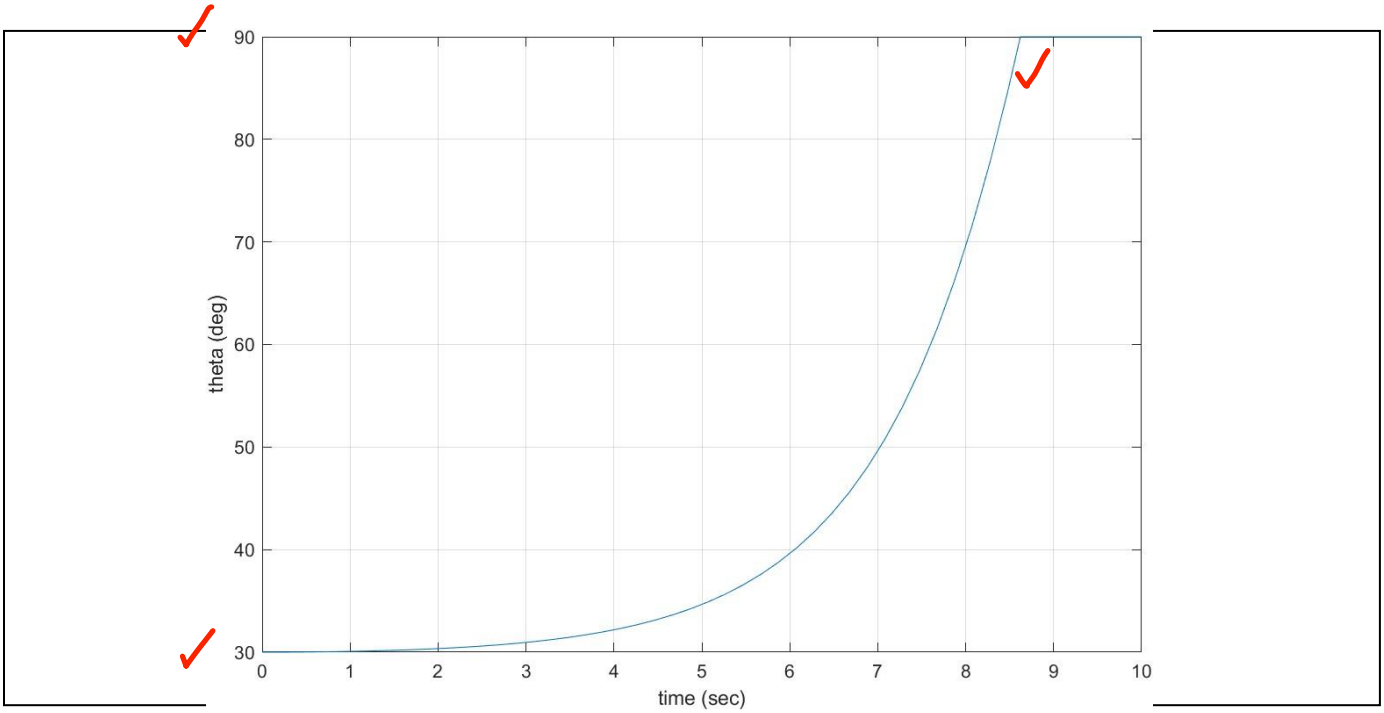
Initial Angle (deg)	Final Angle (deg)	Total time to reach $\theta = 90°$ (sec)
30 °	90°	8.628 sec

6. Plot the graph in MATLAB using the following code:

```
plot(out.tout,out.theta), grid on xlabel('time (sec)'),
ylabel('theta (deg)')
```

7. Provide your Simulink block diagram and the MATLAB plot below:





Part B: Simulink Model of Vehicle Suspension System

The single-mass quarter-car model of a vehicle suspension is shown below. In this simplified model, the masses of the wheel, tire, and axle are neglected, and the *mass* m represents one-fourth of the vehicle mass. The *spring constant* k models the series combination of the elasticity of the tire and the suspension spring. The *damping constant* c models the shock absorber. The equilibrium position of m when $y = 0$ is $x = 0$. The road surface displacement $y(t)$ can be derived from the road surface profile and the car's speed.

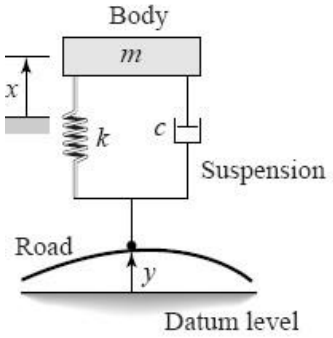
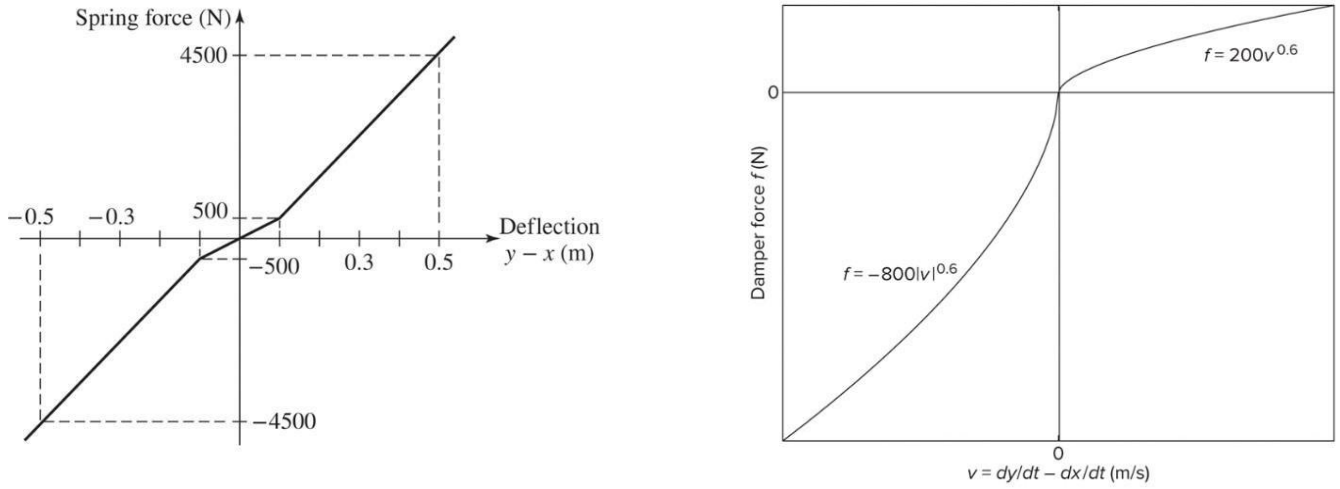


Figure 4: A quarter-car model with a single mass

Assume that the spring and damper elements have the **nonlinear** models as shown below,



(a) (b)

Figure 5: (a) Hardening spring model, (b) Degressive damper model.

The bump is represented by the trapezoidal function $y(t)$ shown below. This function corresponds approximately to a vehicle traveling at $30\ mi/h$ over a $0.2\ m$ high road surface elevation $48\ m$ long.

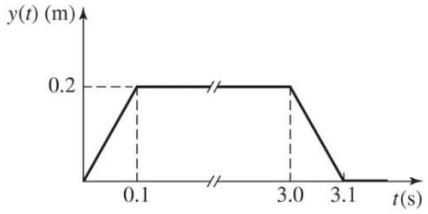
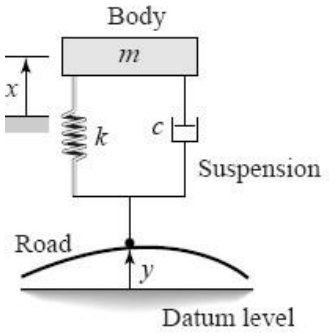


Figure 6: Road bump profile



The system model from Newton's law is:

$$f_s + f_d = m\ddot{x}$$

where,

m is one-fourth of the vehicle mass ($m = 400\ kg$),

f_s is the nonlinear spring function shown in Figure 5a,

f_d is the is the nonlinear damper function shown in Figure 5b,

x is the displacement of the car mass,

y is the displacement of road bump profile shown in Figure 6,

The corresponding simulation diagram is shown as:

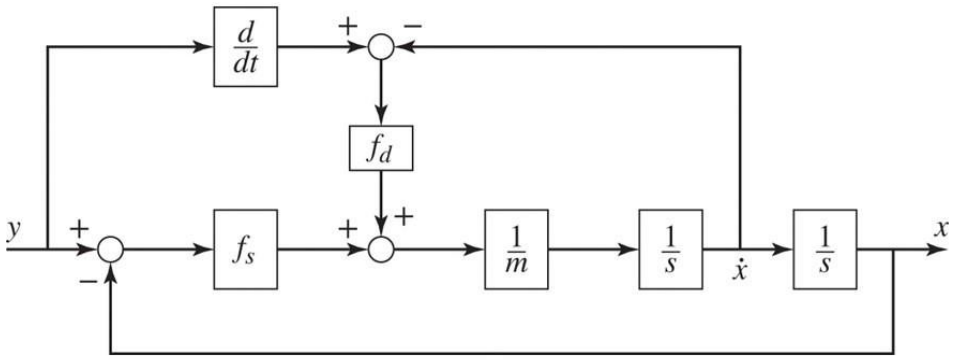


Figure 7: Block diagram model

This diagram shows we need to compute $\dot{y}$. Because Simulink uses numerical and not analytical methods, it computes derivatives only approximately, using the **Derivative** block. We must keep this in mind when using rapidly changing or discontinuous inputs.

8. Start **Simulink** and open a new window.
9. Create the following Simulink model to solve for and plot $x(t)$

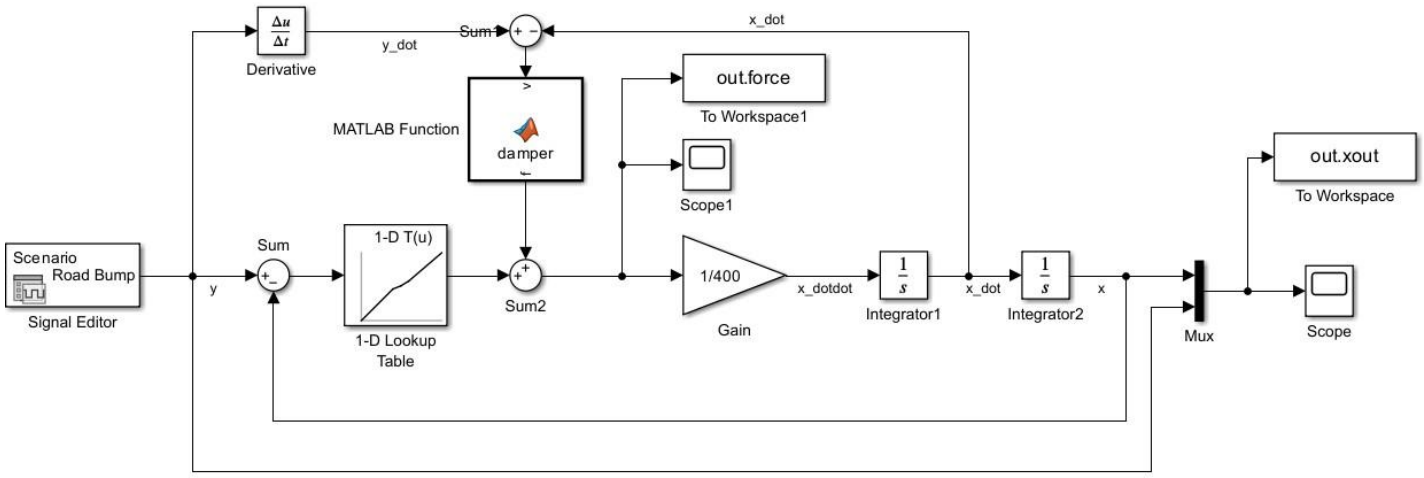


Figure 8: Simulink block diagram model

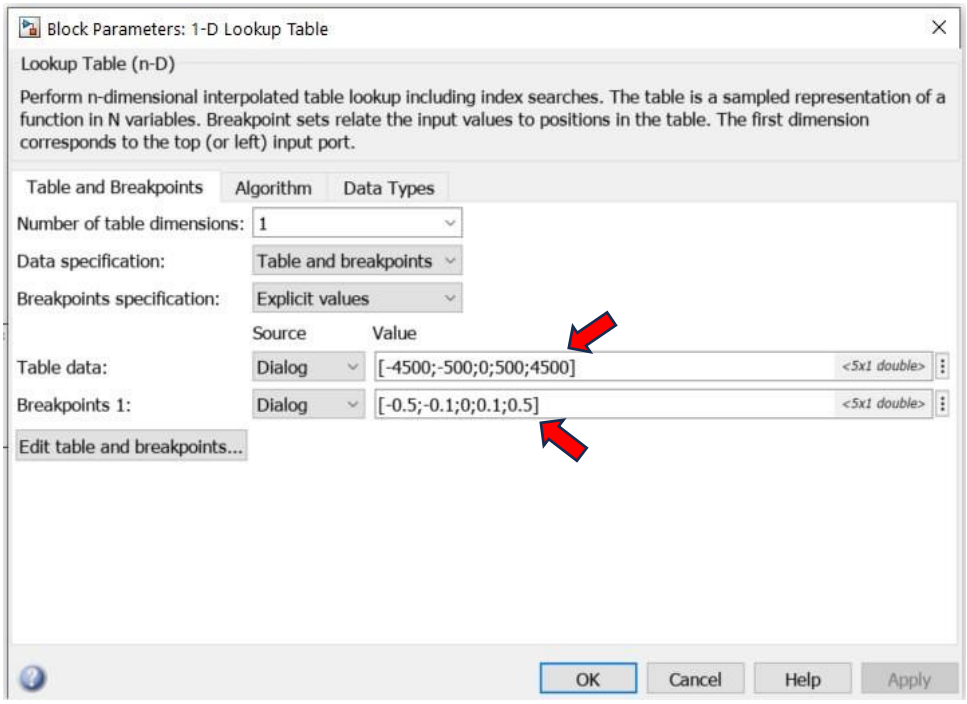
10. Open the **Simulink Library Browser** window and select the following blocks:

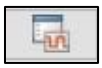
    - The **Gain** block from the **Simulink**/**Math Operations** category.
    - Change the **Gain** value to `1/400`.
    - The **Integrator** blocks from the **Simulink**/**Continuous** category.
    - The **Derivative** block from the **Continuous** category.
    - The **Mux** block from the **Signal Routing** category.
    - The **Sum** blocks from the **Math Operations** category.
    - Set the signs as `|+-, ++|` and `+-`. Rotate the blocks if required.
    - Two **Scope** block from the **Simulink**/**Sink** category.

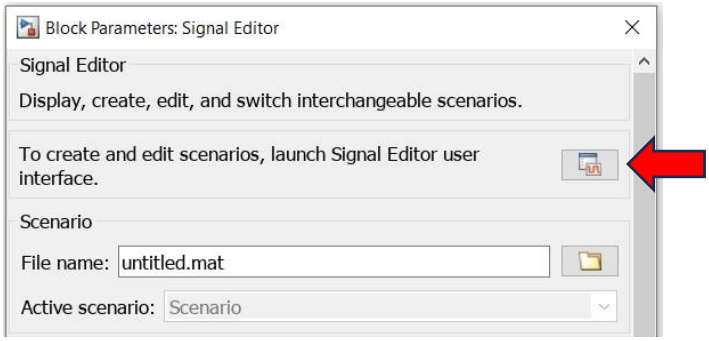
- Two **To Workspace** blocks from the **Simulink/Sink** category.
- Change the output **Variable name** of **To Workspace** block to **xout** and specify the **Save Format** as **Array**.
- Change the output **Variable name** of **To Workspace1** block to **force** and specify the **Save Format** as **Array**.
- The **MATLAB Function** block from **User-Defined Functions** category.
- Open the **MATLAB Function** block and program the block as below:

```
function f = damper(v)
 if v <=
0
      f = -800*(abs(v)).^(0.6);
else
      f = 200*v.^(0.6);
end
```

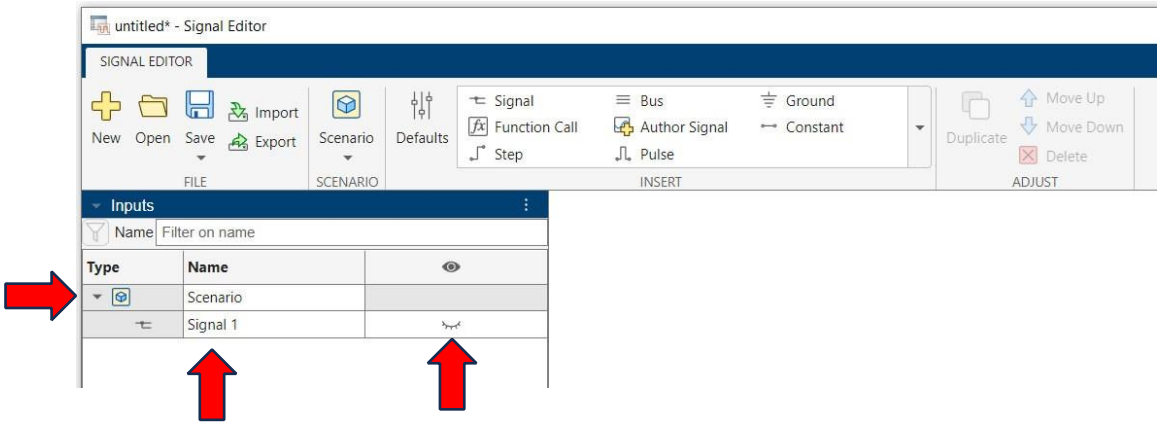
- The **1-D Lookup Table** block from **Lookup Tables** category.
- Enter `[-0.5 -0.1 0 0.1 0.5]` for the **Table data** values and `[-4500 -500 0 500 4500]` for the **Breakpoints 1** values. Use default settings for the remaining parameters.



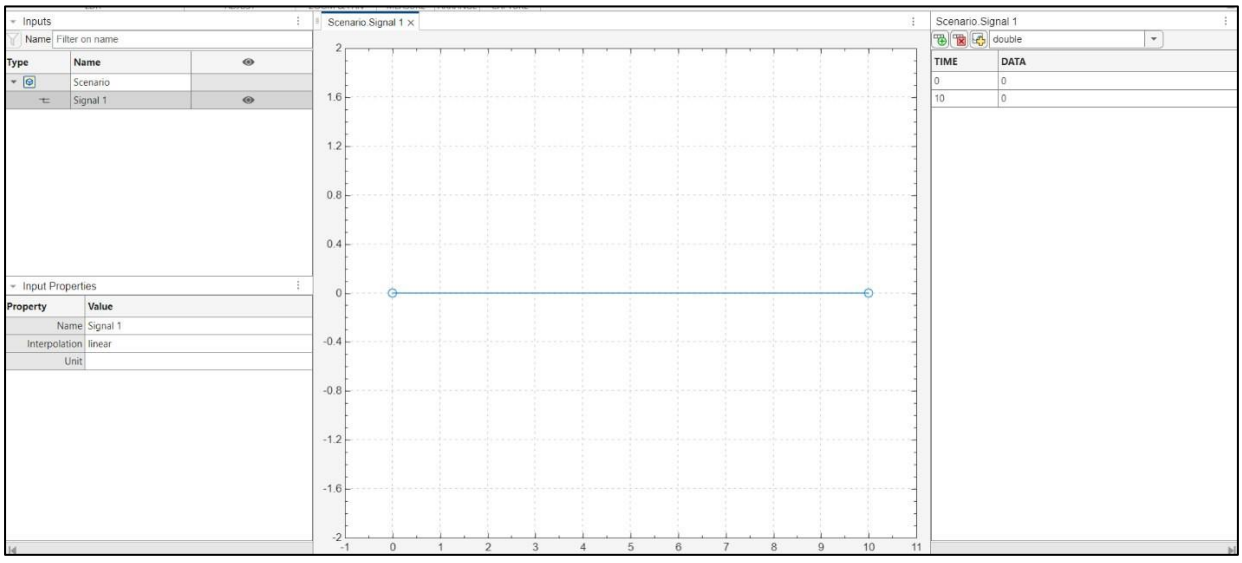
- The **Signal Editor** block from **Source** category.
- Open the **Signal Editor** block to define the input function profile based on the Figure 6 as below:
- Click on  icon to open the **Signal Editor** window.



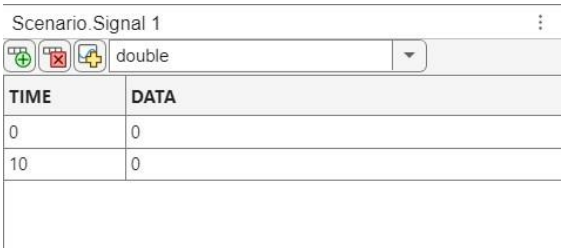
- Click on the small drop-down icon beside the **Scenario** to see the signal present in it, which is named **Signal 1** by default.



- Double-click on the  icon in the right side of the **Signal 1** to reveal what is in this signal.

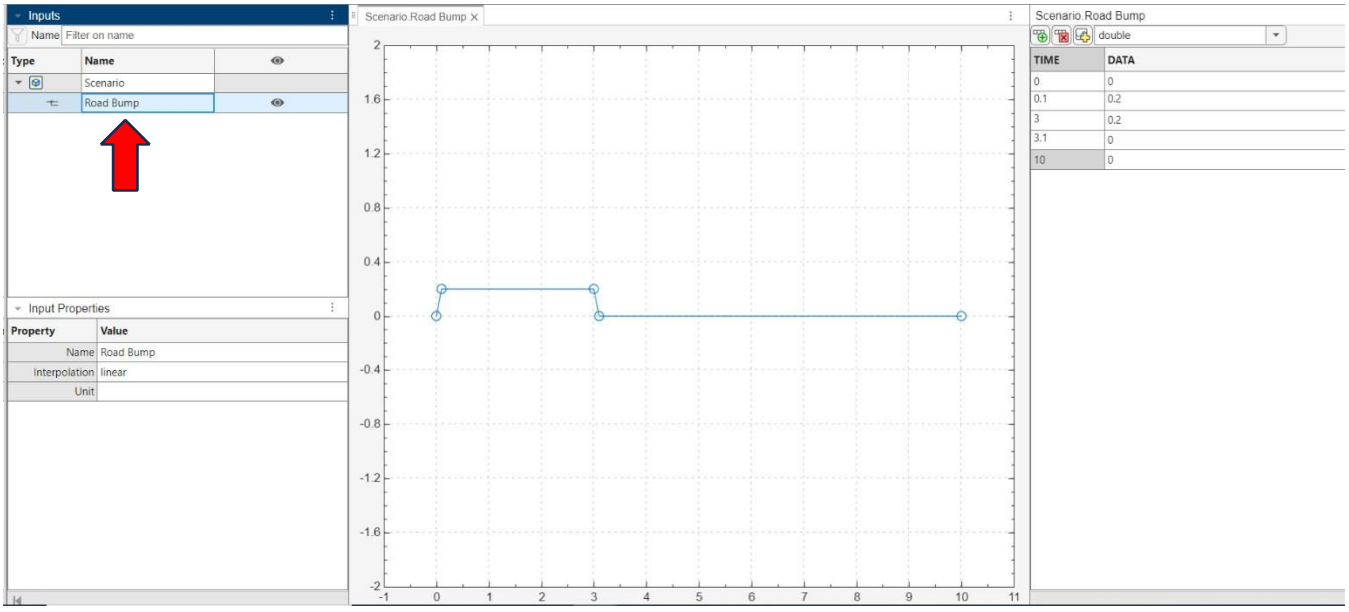


- The signal is blank at the first time. You can add the data for your signal profile by editing the data table in the right side.

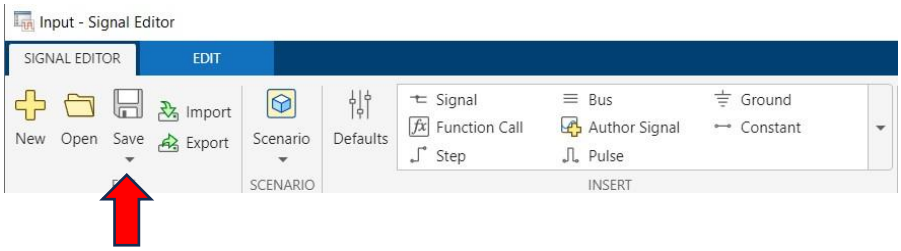


- Click on  icon to add more rows to the data table.
- Enter the **Time** values as `[0 0.1 3 3.1 10]` and **Data** values as `[0 0.2 0.2 0 0]` based on road bump profile from Figure 6.

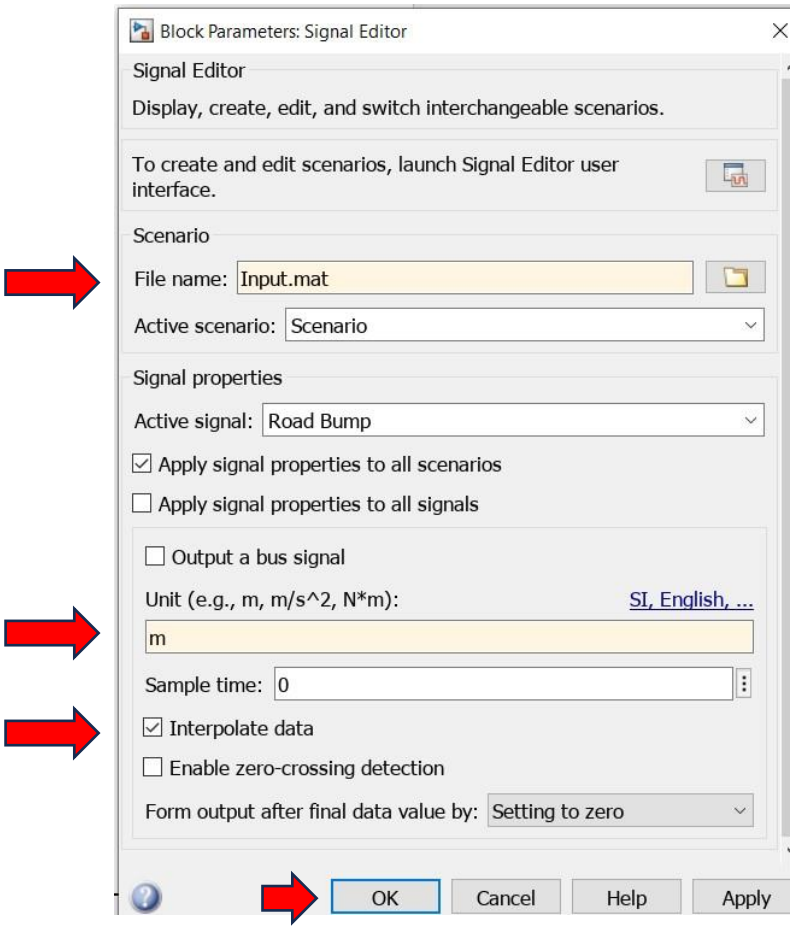
- The road bump profile will be created. Click on the signal name and rename the signal as **Road_Bump**.



- **Save** the signal file as **Input.mat** in the same folder you will save the Simulink file.



- The signal file name will be appeared in the **Block Parameters Dialogue** window.



- Set the signal data unit as **m** (meter) in the **Unit** box.
- Check marks the **Interpolate data** box.
- Click **OK** to close the dialogue window.

11. **Save** the Simulink model file as **Lab2_PartB.slx**.
12. **Run** the simulation for **10 seconds**. Open the **Scope** block to check the results. Use **Cursor Measurement** to determine and record the required values in Table 2.

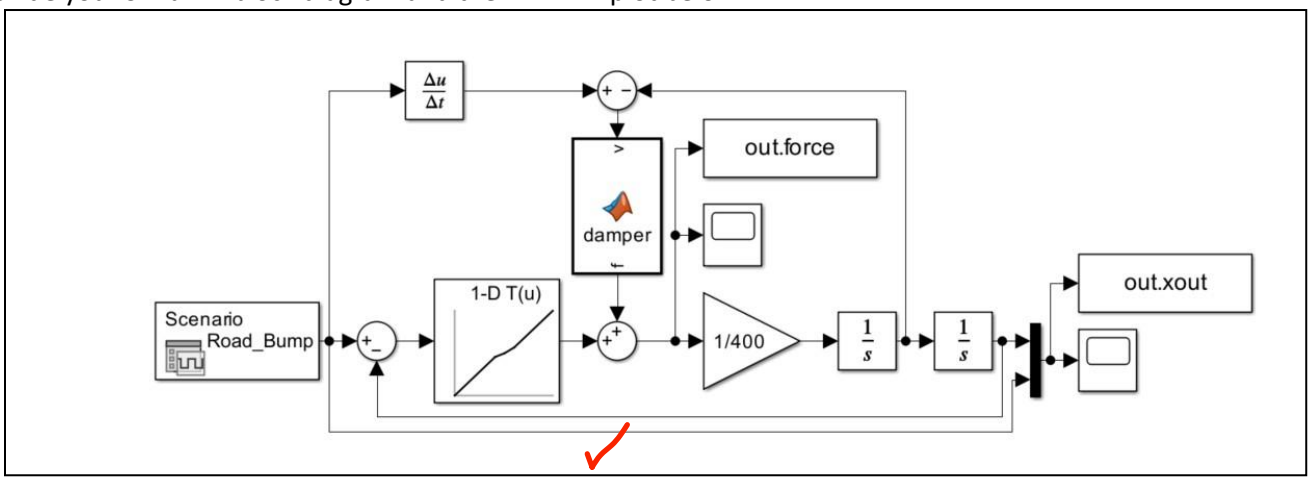
Table 2

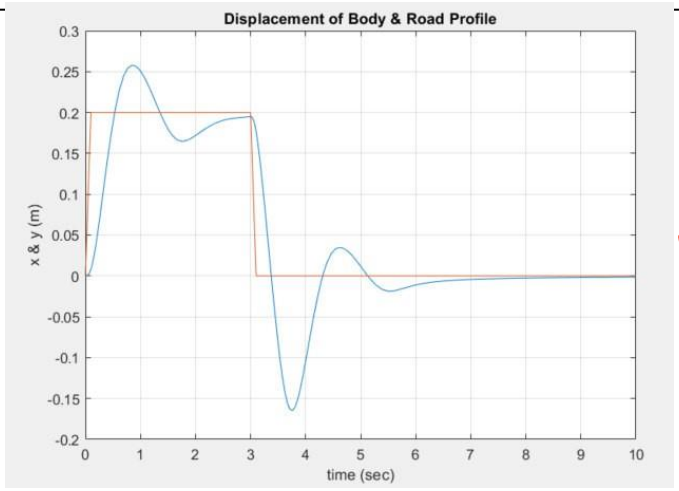
Maximum Overshoot (m)	Maximum Undershoot (m)
0.0578 m	0.1641 m

13. Plot the road bump profile and the displacement of body in MATLAB using the following code:

```
plot(out.tout,out.xout), grid on
xlabel('time (sec)'), ylabel('x & y (m)')
title('Displacement of Body & Road Profile')
```

14. Provide your Simulink block diagram and the MATLAB plot below:

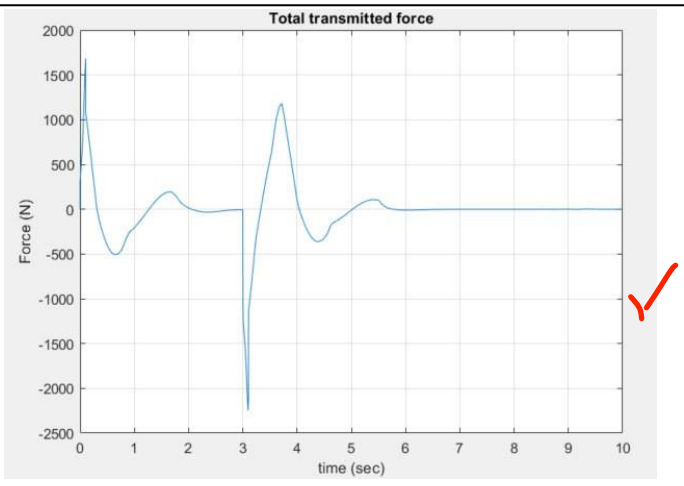




15. Open the **Scope 1** block to check the results, which displays the overall spring and damper forces transmitted to the chassis.
16. Plot the force graph in MATLAB using the following code:

```
plot(out.tout,out.force), grid on
xlabel('time (sec)'), ylabel('Force (N)')
title('Total transmitted force')
```

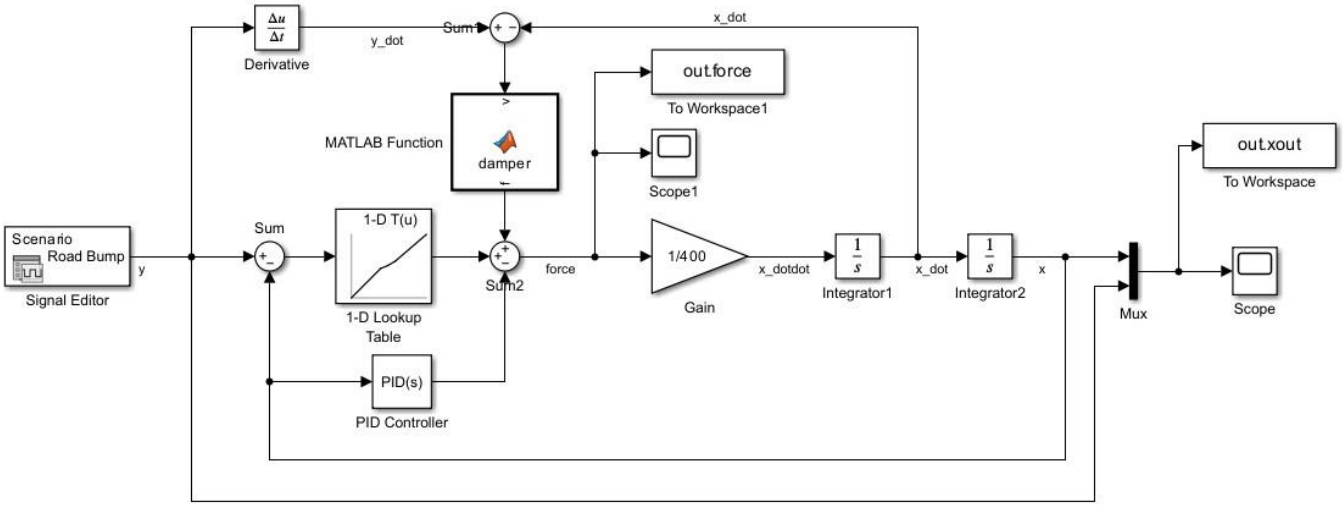
17. Provide the force plot below:



18. Oscillation of the body mass can be decreased by adding a PID controller to the system. The controller takes the displacement of body $x(t)$ as the input and generate a control signal to suppress the large force applied by the suspension system.

NOTE: *Design of PID controller is out of scope of this course. It is covered in the Control Systems course*.

- Modify the Simulink model diagram by adding the **PID Controller** block from **Simulink/Continuous** category.



19. Open the PID Controller block parameters window and set the controller values as,

    - **Proportional** (P): 10, **Integral** (I): 0, **Derivative** (D): 1000, **Filter coefficient** (N): 80

20. **Run** the simulation for **10 seconds**. Open the **Scope** block to check the results. Use **Cursor Measurement** to determine and record the required values in Table 3.

Table 3

Maximum Overshoot (m)	Maximum Undershoot (m)
0.0063 m ✗ [-1]	0.05072 m

Briefly explain how the PID controller enhances the oscillation of the body mass.

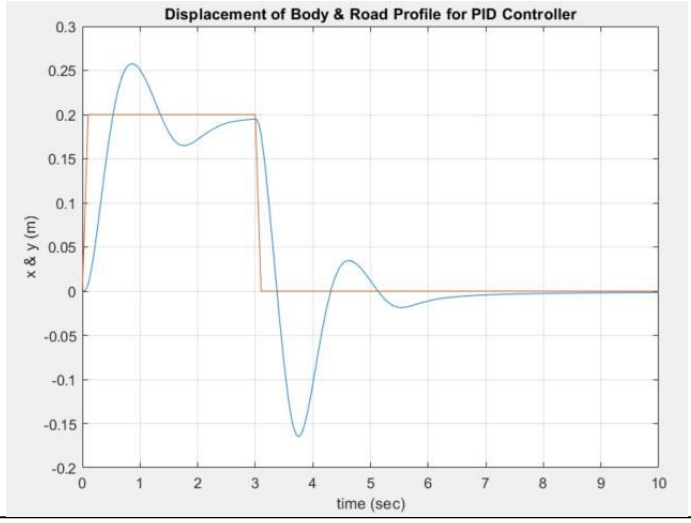
It reduces the overshoot by controlling it with the PID by adding an external force ✓

21. Plot the road bump profile and the displacement of body in MATLAB using the following code:

```
plot(out.tout,out.xout), grid on
xlabel('time (sec)'), ylabel('x & y (m)')
title('Displacement of Body & Road Profile for PID Controller')
```

22. Provide the MATLAB plot below:

This graph is identical with the previous step without PID controller.



The graph is not correct. [-4]

Post Lab Assignment

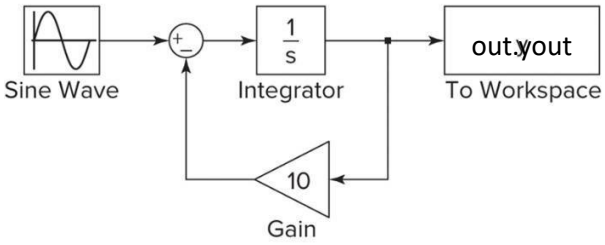
1) Develop a Simulink model to solve the following differential equation model:

$$\dot{y}(t) = -10y(t) + f(t), \quad y(0) = 1$$

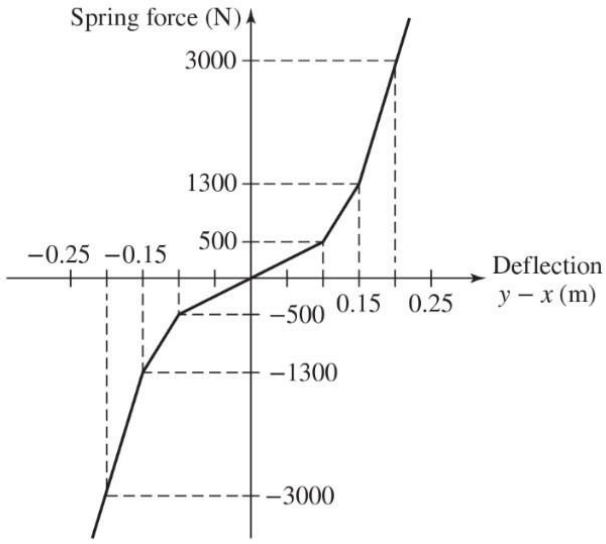
where $f(t) = 2 \sin 4t$, for $0 \leq t \leq 3$.

Run the simulation for **3 seconds** and plot the results in **MATLAB**. Provide the Simulink model, required code to plot the graph and the results. Add the required titles and labels to the graph.

Hint: You can use the following model to develop the simulation.



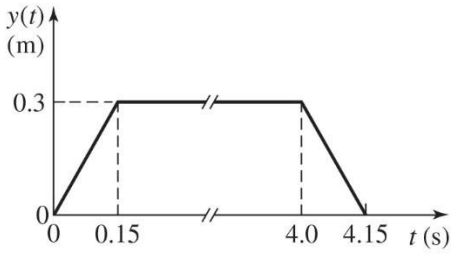
2) Consider the Simulink suspension model developed in Part B Figure 8. Modify the simulation using the following spring and damper relation and the given road bump profile as the input.



Spring Model

$$f_d = \begin{cases} -500|v|^{1.2} & v \le 0 \\ 50v^{1.2} & v > 0 \end{cases}$$

Damper Model



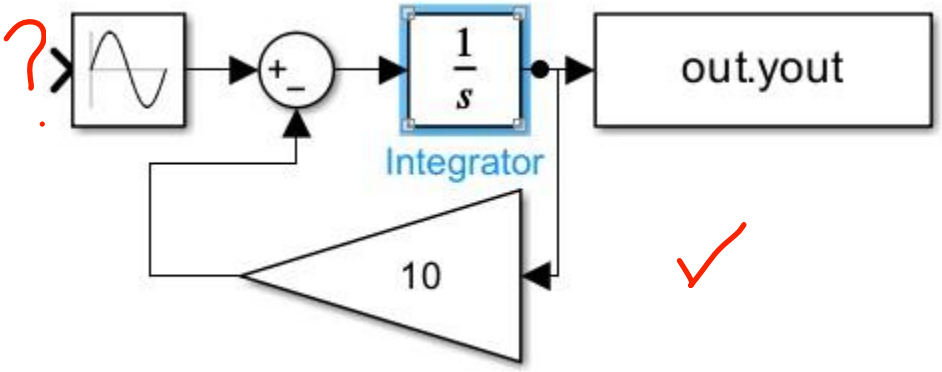
Road Bump Profile

Run the simulation for **15 seconds** and plot the responses in MATLAB. Provide the Simulink model, the required code to plot the graphs and the displacement graph and the force graph. Add the required titles and labels to the graphs.

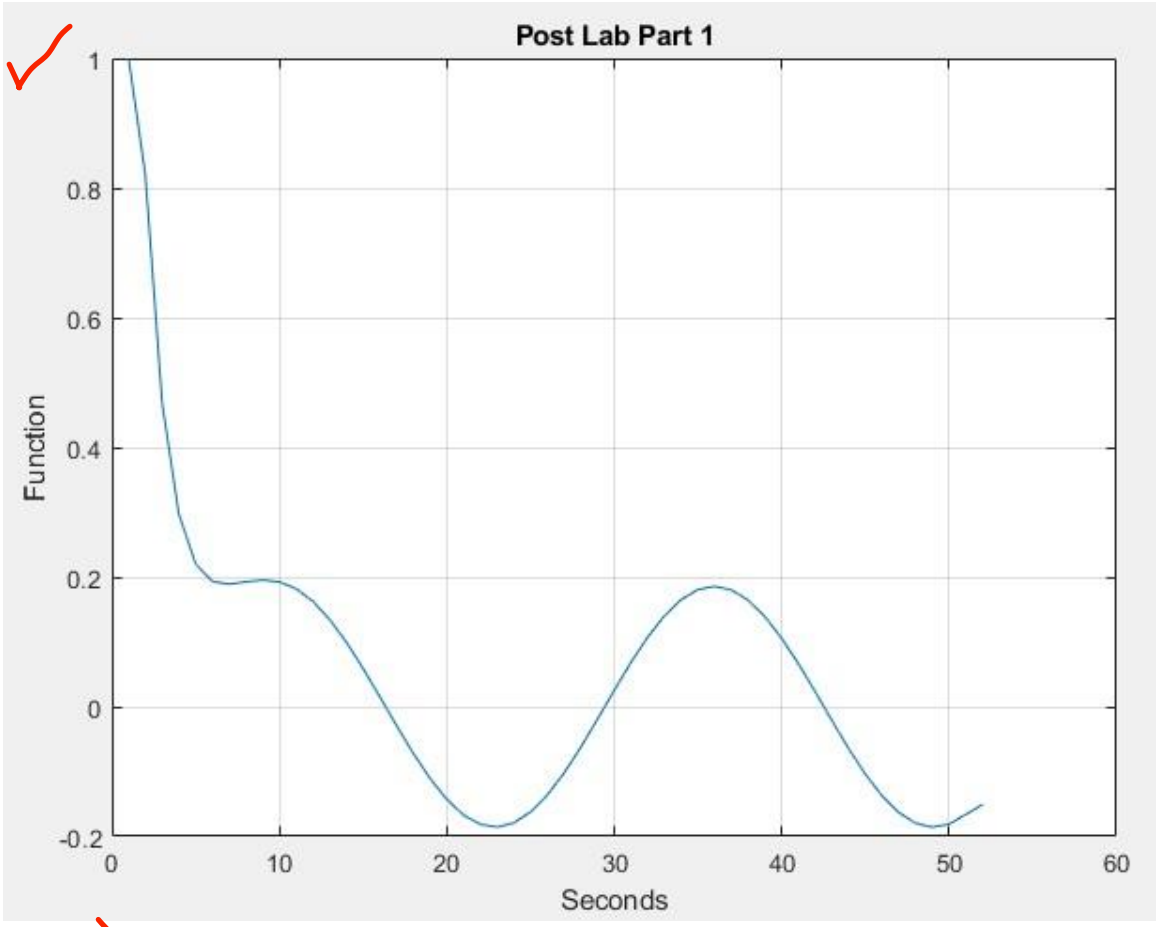
Evaluate the overshoot and undershoot of the mass displacement. Provide the results.

Post Lab Assignment

a)

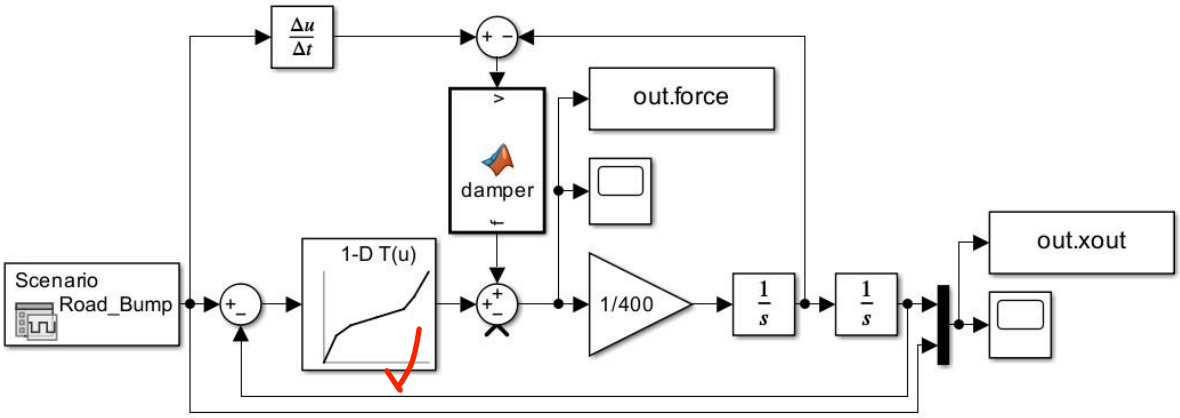


plot(out.yout), grid on xlabel('Seconds'), ylabel('Function') title('Post Lab Part 1')

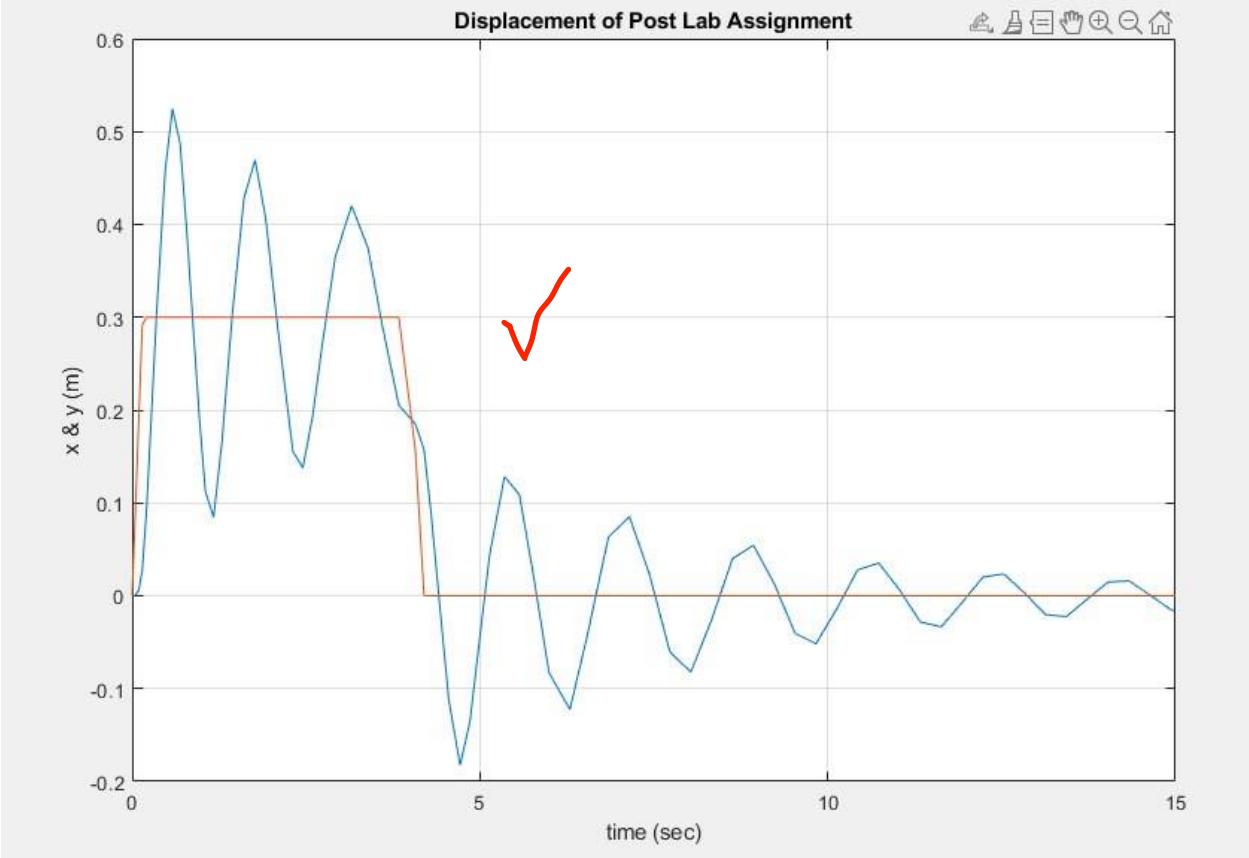


The time range is not correct. [-0.5]

b)



plot(out.tout,out.xout), grid on xlabel('time (sec)'), ylabel('x & y (m)') title('Displacement of Post Lab Assignment')



Overshoot (m): 0.2246 m

Undershoot (m): 0.1814 m

Force graph is missing. [-2]
The code to plot the force graph is missing. [-1]