Linear Control Systems

Laboratory Experiment 3: Modeling of Physical Systems using SIMULINK

Objectives: The objective of this exercise is to use graphical user interface diagrams to model the physical systems for the purpose of design and analysis of control systems. We will learn how MATLAB/SIMULINK helps in solving such models.

List of Equipment/Software

Following equipment/software is required:

- MATLAB/SIMULINK

Category Soft-Experiment

Deliverables

A complete lab report including the following:

- Summarized learning outcomes.
- MATLAB scripts, SIMULINK diagrams and their results for all the assignments and exercises should be properly reported.

Overview:

This lab introduces powerful graphical user interface (GUI), **Simulink** of Matlab. This software is used for solving the modeling equations and obtaining the response of a system to different inputs. Both linear and nonlinear differential equations can be solved numerically with high precision and speed, allowing system responses to be calculated and displayed for many input functions. To provide an interface between a system's modeling equations and the digital computer, block diagrams drawn from the system's differential equations are used. A block diagram is an interconnection of blocks representing basic mathematical operations in such a way that the overall diagram is equivalent to the system's mathematical model. The lines interconnecting the blocks represent the variables describing the system behavior. These may be inputs, outputs, state variables, or other related variables. The blocks represent operations or functions that use one or more of these variables to calculate other variables. Block diagrams can represent modeling equations in both input-output and state variable form.

We use MATLAB with its companion package **Simulink**, which provides a graphical user interface (GUI) for building system models and executing the simulation. These models are constructed by drawing block diagrams representing the algebraic and differential equations that describe the system behavior. The operations that we generally use in block diagrams are summation, gain, and integration. Other blocks, including nonlinear elements such as

multiplication, square root, exponential, logarithmic, and other functions, are available. Provisions are also included for supplying input functions, using a signal generator block, constants etc and for displaying results, using a scope block.

An important feature of a numerical simulation is the ease with which parameters can be varied and the results observed directly. MATLAB is used in a supporting role to initialize parameter values and to produce plots of the system response. Also MATLAB is used for multiple runs for varying system parameters. Only a small subset of the functions of MATLAB will be considered during these labs.

SIMULINK

Simulink provides access to an extensive set of blocks that accomplish a wide range of functions useful for the simulation and analysis of dynamic systems. The blocks are grouped into libraries, by general classes of functions.

- Mathematical functions such as summers and gains are in the Math library.
- Integrators are in the Continuous library.
- Constants, common input functions, and clock can all be found in the Sources library.
- Scope, To Workspace blocks can be found in the Sinks library.

Simulink is a graphical interface that allows the user to create programs that are actually run in MATLAB. When these programs run, they create arrays of the variables defined in Simulink that can be made available to MATLAB for analysis and/or plotting. The variables to be used in MATLAB must be identified by Simulink using a "To Workspace" block, which is found in the Sinks library. (When using this block, open its dialog box and specify that the save format should be Matrix, rather than the default, which is called Structure.) The Sinks library also contains a Scope, which allows variables to be displayed as the simulated system responds to an input. This is most useful when studying responses to repetitive inputs.

Simulink uses blocks to write a program. Blocks are arranged in various libraries according to their functions. Properties of the blocks and the values can be changed in the associated dialog boxes. Some of the blocks are given below.

SUM (Math library)

A dialog box obtained by double-clicking on the SUM block performs the configuration of the SUM lock, allowing any number of inputs and the sign of each. The sum block can be represented in two ways in Simulink, by a circle or by a rectangle. Both choices are shown

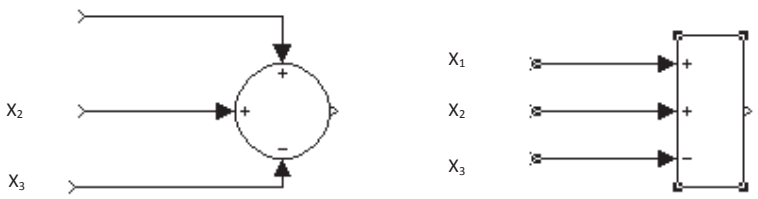


Figure 1: Two Simulink blocks for a summer representing y = x 1+ x2 – x3

GAIN (Math library)

A gain block is shown by a triangular symbol, with the gain expression written inside if it will fit. If not, the symbol - k - is used. The value used in each gain block is established in a dialog box that appears if the user double-clicks on its block.

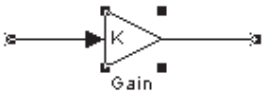


Figure 2: Simulink block for a gain of K.

INTEGRATOR (Continuous library)

The block for an integrator as shown below looks unusual. The quantity 1/s comes from the Laplace transform expression for integration. When double-clicked on the symbol for an integrator, a dialog box appears allowing the initial condition for that integrator to be specified. It may be implicit, and not shown on the block, as in Figure (a). Alternatively, a second input to the block can be displayed to supply the initial condition explicitly, as in part (b) of Figure 3. Initial conditions may be specific numerical values, literal variables, or algebraic expressions.

Figure3: Two forms of the Simulink block for an integrator.

(a) Implicit initial condition. (b) Explicit initial condition.

CONSTANTS (Source library)

Constants are created by the Constant block, which closely resembles Figure 4. Double-clicking on the symbol opens a dialog box to establish the constant's value. It can be a number or an algebraic expression using constants whose values are defined in the workspace and are therefore known to MATLAB.

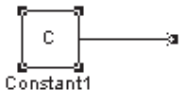


Figure 4: A constant block

STEP (Source library)

A Simulink block is provided for a Step input, a signal that changes (usually from zero) to a specified new, constant level at a specified time. These levels and time can be specified through the dialog box, obtained by double-clicking on the Step block.

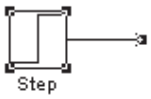


Figure 5: A step block

SIGNAL GENERATOR (Source library)

One source of repetitive signals in Simulink is called the Signal Generator. Double-clicking on the Signal Generator block opens a dialog box, where a sine wave, a square wave, a ramp (sawtooth), or a random waveform can be chosen. In addition, the amplitude and frequency of the signal may be specified. The signals produced have a mean value of zero. The repetition frequency can be given in Hertz (Hz), which is the same as cycles per second, or in radians/second.

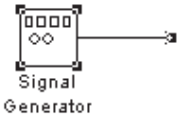


Figure 6: A signal generator block

SCOPE (Sinks library)

The system response can be examined graphically, as the simulation runs, using the Scope block in the sinks library. This name is derived from the electronic instrument, oscilloscope, which performs a similar function with electronic signals. Any of the variables in a Simulink diagram can be connected to the Scope block, and when the simulation is started, that variable is displayed. It is possible to include several Scope blocks. Also it is possible to display several signals in the same scope block using a MTJX block in the signals & systems library. The Scope normally chooses its scales automatically to best display the data.

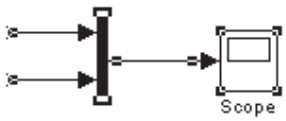


Figure 7: A scope block with MUX block

Two additional blocks will be needed if we wish to use MATLAB to plot the responses versus time. These are the Clock and the To Workspace blocks.

CLOCK (Sources library)

The clock produces the variable "time" that is associated with the integrators as MATLAB calculates a numerical (digital) solution to a model of a continuous system. The result is a string of sample values of each of the output variables. These samples are not necessarily at uniform time increments, so it is necessary to have the variable "time" that contains the time corresponding to each sample point. Then MATLAB can make plots versus "time." The clock output could be given any arbitrary name; we use "t" in most of the cases.

Figure 8: A clock block

To Workspace (Sinks library)

The To Workspace block is used to return the results of a simulation to the MATLAB workspace, where they can be analyzed and/or plotted. Any variable in a Simulink diagram can be connected to a ToWorkspace block. In our exercises, all of the state variables and the input variables are usually returned to the workspace. In addition, the result of any output equation that may be simulated would usually be sent to the workspace. In the block parameters drop down window, change the save format to 'array'.

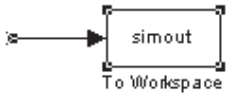


Figure 9: A To Workspace block

In the Simulink diagram, the appearance of a block can be changed by changing the foreground or background colours, or by drop shadow or other options available in the format drop down menu. The available options can be reached in the Simulink window by highlighting the block, then clicking the right mouse button. The Show Drop Shadow option is on the format drop-down menu.

Simulink provides scores of other blocks with different functions.

You are encouraged to **browse the Simulink libraries and consult the online Help facility provided with MATLAB.**

GENERAL INSTRUCTIONS FOR WRITING A SIMULINK PROGRAM

To create a simulation in Simulink, follow the steps:

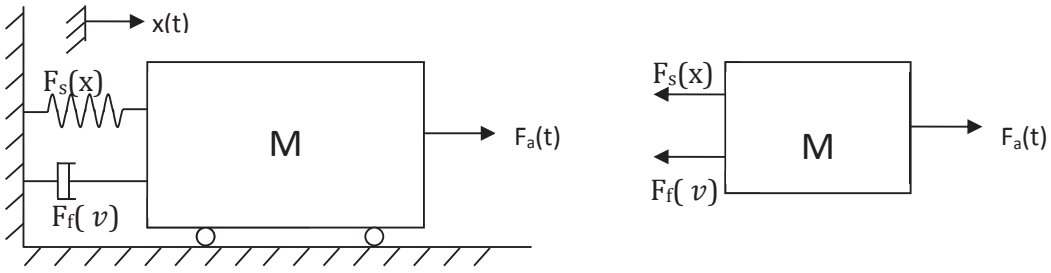
- Start MATLAB.
- Start Simulink.

- Open the libraries that contain the blocks you will need. These usually will include the Sources, Sinks, Math and Continuous libraries, and possibly others.
- Open a new Simulink window.
- Drag the needed blocks from their library folders to that window. The Math library, for example, contains the Gain and Sum blocks.
- Arrange these blocks in an orderly way corresponding to the equations to be solved.
- Interconnect the blocks by dragging the cursor from the output of one block to the input of another block. Interconnecting branches can be made by right-clicking on an existing branch.
- Double-click on any block having parameters that must be established, and set these parameters. For example, the gain of all Gain blocks must be set. The number and signs of the inputs to a Sum block must be established. The parameters of any source blocks should also be set in this way.
- It is necessary to specify a stop time for the solution. This is done by clicking on the Simulation > Parameters entry on the Simulink toolbar.

At the Simulation > Parameters entry, several parameters can be selected in this dialog box, but the default values of all of them should be adequate for almost all of the exercises. If the response before time zero is needed, it can be obtained by setting the Start time to a negative value. It may be necessary in some problems to reduce the maximum integration step size used by the numerical algorithm. If the plots of the results of a simulation appear "choppy" or composed of straight-line segments when they should be smooth, reducing the max step size permitted can solve this problem.

Mass-Spring System Model

Consider the Mass-Spring system used in the previous exercise as shown in the figure. Where $F_s(x)$ is the spring force, $F_f(\dot{x})$ is the friction coefficient, x(t) is the displacement and $F_a(t)$ is the applied force:



The differential equation for the above Mass-Spring system can then be written as follows

$$M\frac{d^2x(t)}{dt^2} + B\frac{dx(t)}{dt} + Kx(t) = F_a(t) \qquad (1)$$

For Non-linear such case, (1) becomes

$$M\frac{d^2x(t)}{dt^2} + B\frac{dx(t)}{dt} + Kx^r(t) = F_a(t) \qquad (2)$$

Exercise 1: Modeling of a second order system

Construct a Simulink diagram to calculate the response of the Mass-Spring system. The input force increases from 0 to 8 N at t = 1 s. The parameter values are M = 2 kg, K= 16 N/m, and B =4 N.s/m.

Steps:

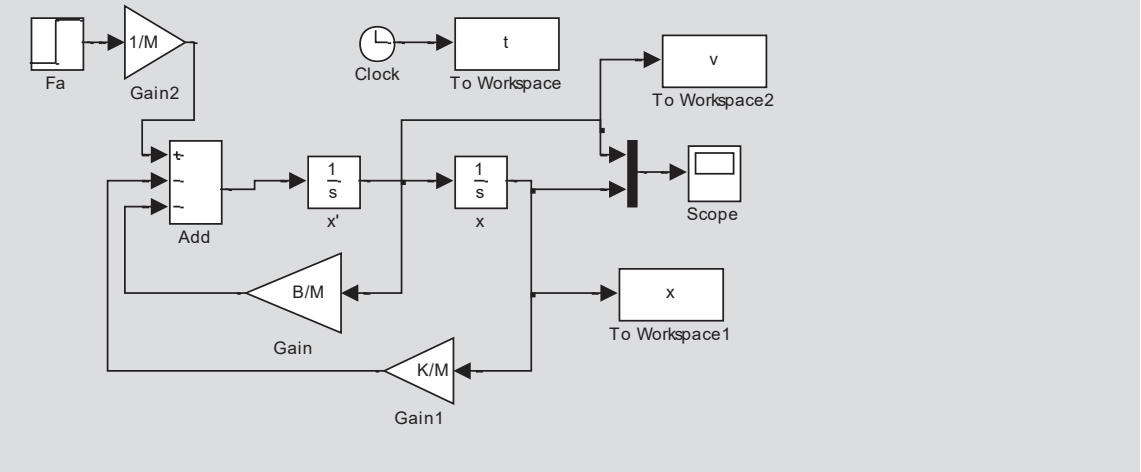
- Draw the free body diagram.
- Write the modeling equation from the free body diagram
- Solve the equations for the highest derivative of the output.
- Draw a block diagram to represent this equation.
- Draw the corresponding Simulink diagram.
- Use Step block to provide the input fa(t).
- In the Step block, set the initial and final values and the time at which the step occurs.
- Use the "To Workspace" blocks for t, fa(t), x, and v in order to allow MATLAB to plot the desired responses. Set the save format to array in block parameters.
- Select the duration of the simulation to be 10 seconds from the Simulation > Parameters entry on the toolbar

Given below is a file that will set up the MATLAB workspace by establishing the values of the parameters needed for the Simulink simulation of the given model.

M-file for parameter values

```
% This file is named exl_parameter.m.
% Everything after a % sign on a line is a comment that
% is ignored by M This file establishes the
% parameter values for exl_model.mdl.
%
M2;             %kg
K= 16;          %N/m
B=4;            % Ns/m
```

Simulink block diagram



Plotting the outputs in MATLAB:

The file to create the plots of the output is given below. Create the file and save it by the name given below.

M-file to produce the plot

```
% This file is named exl_plot.m.
% It makes a plot of the data produced by exl_model.mdl.
plot(t,x); grid              % Plots x for the case with B=4.
xlabel('Time (s)');
ylabel ('Displacement (m) ')
```

A semicolon in a physical line ends the logical line, and anything after it is treated as if it were on a new physical line. A semicolon at the end of a line that generates output to the command window suppresses the printing of that output.

Program Execution:

Follow the following steps to execute these files:

- Enter the command exl_parameter in the command window. This will load the parameter values of the model.
- Open the Simulink model exl_model.mdl and start the simulation by clicking on the toolbar entry Simulation> Start.
- Enter the command exl_plot in the command window to make the plot.

Making Subplots in MATLAB:

When two or more variables are being studied simultaneously, it is frequently desirable to plot them one above the other on separate axes, as can be done for displacement and velocity in. This is accomplished with the subplot command. The following M-file uses this command to produce both plots of displacement and velocity.

M-file to make subplots

```
% This file is named exl_plot2.m.
% It makes both plots, displacement and velocity.
% Execute exlparameter.m first.
subplot(2,l,1);
plot(t,x); grid % Plots x for the case with B=4. xlabel ('Time (s) ') ;
ylabel ('Displacement (m) '); subplot(2,1,2);
plot(t,v); grid % Plots v for the case with B=4. xlabel('Time (s)');
ylabel('Velocity (m per s)');
```

Exercise 2: Simulation with system parameter variation

The effect of changing B is to alter the amount of overshoot or undershoot. These are related to a term called the damping ratio. Simulate and compare the results of the variations in B in exercise 1. Take values of B = 4, 8, 12, 25 N-s/m.

Steps:

Perform the following steps. Use the same input force as in Exercise 1.

- Begin the simulation with B = 4 N-s/m, but with the input applied at t = 0
- Plot the result.
- Rerun it with B = 8 N.s/m.
- Hold the first plot active, by the command hold on
- Reissue the plot command plot(t,x), the second plot will superimpose on the first.
- Repeat for B = 12 N-s/m and for B = 25 N-s/m
- Release the plot by the command hold off
- Show your result.

Running SIMULINK from MATLAB command prompt

If a complex plot is desired, in which several runs are needed with different parameters, this can using the command called "sim". "sim" command will run the Simulink model file from the Matlab command prompt. For multiple runs with several plot it can be accomplished by executing ex1_model (to load parameters) followed by given M-file. Entering the command ex1_plots in the command window results in multiple runs with varying values if B and will plot the results.

M-file to use "sim" function and produce multiple runs and their plots

```
% This file is named ex2_plots.m.
% It plots the data produced by exl_model.mdl for
% several values of B. Execute exl_parameter.m first.
sim('exl_model')      % Has the same effect as clicking on
                      % Start on the toolbar.
plot(t,x)             % Plots the initial run with B=4
hold on               % Plots later results on the same axes % as the first.
B = 8;                % New value of B; other parameter values % stay the same.
sim('exl_model')      % Rerun the simulation with new B value.
plot(t,x)             % Plots new x on original axes.
B 12; sim('exl_model');plot(t,x)
B = 25; sim('exl_model' ) ;plot(t,x)
hold off
```