Ministry of Higher Education and Scientific Research
Med Boudiaf University - M'sila
Department: Electrical Engineering
Module: Control Systems
Lab Supervisor: A. ZORIG
3rd Year electrical engineering

Lab Work No. 1

Introduction to MATLAB and Simulink

I. OBJECTIVE

The objective of this lab is to introduce the student to the numerical computation software MATLAB, as well as the associated simulation tool Simulink, and to discover the functions and tools of MATLAB related to the study of control systems.

2. Presentation and Introduction to MATLAB:

MATLAB, short for "**MAT**rix **LAB**oratory", is software designed to provide a high-level numerical computation environment. It is particularly efficient for matrix calculation and has extensive graphical capabilities for, for example, visualizing complex mathematical objects.

Its operation is based on an interpreted programming language that allows for very rapid development. Conversely, for applications requiring higher computational performance, a compiled language, such as C++ or Fortran, is more suitable.

In its "graphical" form, MATLAB has an interface including the MATLAB environment itself, where MATLAB commands can be executed directly, as well as a graphical environment, which may include several windows: list of variables in use, command

history, ..., and various more or less standard menus, "File", ("New", "Open"...), "Configuration", "Help",...

In this introductory paragraph to MATLAB, we will only focus on the core MATLAB environment, the basic commands and syntax of MATLAB instructions.

3. General Commands

Help to request help
Cd changes/displays the current directory
who lists variables in the workspace
clear *var* removes the variable *var* from the workspace
clear all removes all variables
Close all closes all figures

3.1. Common Operations

= assignment of a value to a variable (e.g., a = 2)
+, -, *, / usual operations on variables or numerical values
i complex number: (1i)^2^ = −1
Abs, angle, real, imag usual operations on complex numbers

4. Definition and Operations on Vectors and Matrices

4.1. Definition of matrices and vectors:

[,,;,,,,;,,] manual definition of a matrix (e.g., A = 1,2,3;4,5,61,2,3;4,5,6)
start:step:end definition of a regular vector spanning the interval [start, end] with the step (e.g., A = 1 : 1 : 6); by default, the step is 1 if omitted (e.g., A = 1 : 6)
linspace (start,end,N) definition of a vector spanning the interval [start; end] with N regularly spaced values

*__*The usual operations +, -, , /, ...*__ act indifferently on real numbers, complex numbers, or matrices (provided their dimensions allow it).

- Addition: A+B

-Subtraction: A-B

-Multiplication: A*B and B*A

- Inversion: inv(A)

-Transposition: transpose(A) or A'

Other operations are also available:

. , ./ , .^ , ...* (The usual operators preceded by a dot) operations on matrices performed element-wise (e.g., compute A = 1:61:6 .* 7:127:12)

length length of a vector

size dimension of a matrix

<, <=, >, >=, == comparison of elements of two matrices, element-wise

sum, mean, ... sum, average, ..., of the elements of a vector

sin, cos, exp, log, ... generally, all common functions apply to matrices element-wise (eg. log ([1, 2, 3]) = [log 1, log 2, log 3])

find finds non-zero elements in a matrix

nnz counts the number of non-zero elements in a matrix

4.2. Some Useful Matrices

-identity matrix of dimension n: eye(n)

- matrix of zeros of dimension m × n: zeros(m,n)

-matrix of ones of dimension m × n: ones(m,n)

4.3. Matrix Analysis

-eigenvalues: eig(A)

- rank: rank(A)

- trace: trace(A): the sum of the diagonal elements of the matrix

-determinant: det(A)

4.4. Extracting Elements from a Matrix

M(i,j) element of matrix M located at row i and column j

V(end) last element of vector V

M(5:9,3) elements of matrix M located from row 5 to 9, and on column 3

M(:,j) all rows of matrix M, column j

M(i,:) all columns of matrix M, row i

M(1:5,1:3) elements of matrix M located on rows 1 to 5, and on columns 1 to 3

5. Graphical Functions:

Figure creates a new figure; figure(n) (re)initializes figure number n

plot plots a set of points (e.g. plot ([0:0.1:2*pi],sin([0:0.1:2*pi])))

subplot divides the current figure into several subplots

axis allows manual selection of the scale used

title, xlabel, ylabel, legend: allows adding a general title, axis labels, and a legend to a figure

grid on / off displays a grid on the current plot

hold on /off allows superimposing plots on the same figure

plot3, semilogx, semilogy, loglog, mesh, surf, … other graphical functions (see help...)

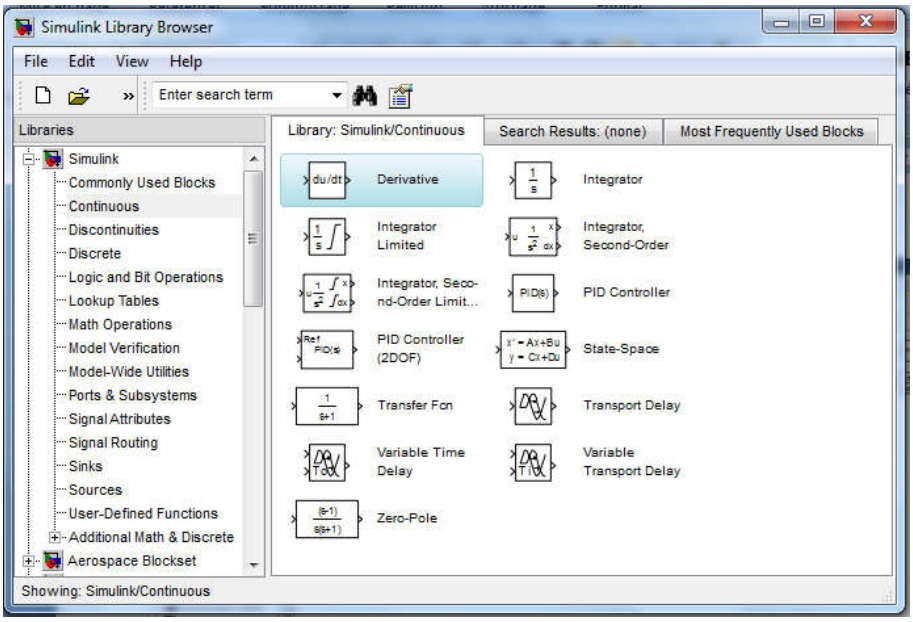
6. Introduction to Simulink:

Simulink is the graphical extension of MATLAB that allows representing mathematical functions and systems as block diagrams and simulating the operation of these systems.

6.1. TO START SIMULINK

In the MATLAB Command Window, type **Simulink**. The Simulink window will open.

The **Simulink** library includes different sections for addressing many aspects of system control. Initially, we will focus on a few sections shown below:



- « **Continuous** »: allows selecting functional blocks modeling linear elements and notably continuous transfer function blocks.
- « **Discrete** »: contains discrete models for building discrete-time systems.
- « **Math Operations** »: allows selecting blocks performing mathematical operations, notably the **Add** block.
- « **Signal Routing** »: allows various data routing on the model lines; notably, the **Mux** block is found here, which allows multiplexing several data into a single line.
- **« Sources »**: signal generators (step, square, sine, etc.).
- **« Sinks »**: which allows finding functional blocks for possible outputs of the model file being simulated. Notably, the functional block « **To Workspace** » is found here, which allows accessing data in the MATLAB workspace.

6.2. Constructing A Simulink Diagram

To begin, in the File menu, choose **New - Model**. An Untitled work window will open.

Open the block libraries by clicking on them (double-click). Drag the blocks needed to build the diagram into the work window. Make connections between the blocks using the mouse. When you (double-)click on a block, a dialog window will open. You can then change the parameters of that block. Once finished, close the dialog window.

Once the diagram is complete, you can save it to a file: in the File menu, choose Save As and give a name (*.mdl) to the file.

Example of Simulink Diagrams:

Reproduce the diagram in Figure 1 to understand how this tool works.

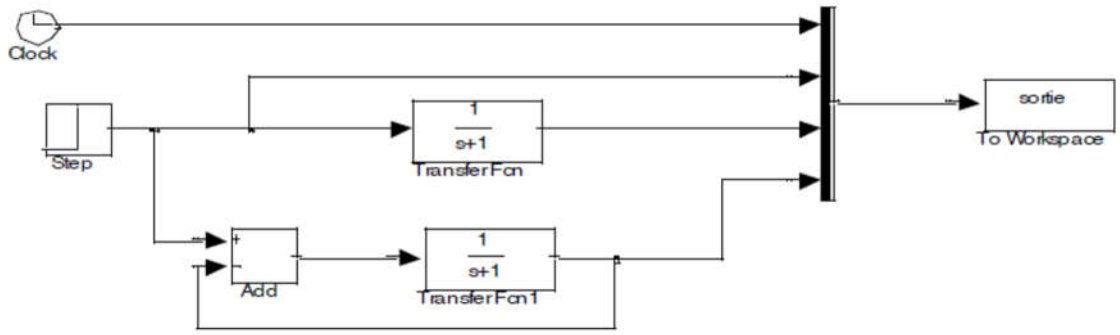


Figure 1: Example Simulink Model.

In the example proposed above, after effectively simulating the model file, you will have in MATLAB a variable named sortie (output), comprising 4 columns:

sortie(:,1): the time instants for which the simulation was performed.

sortie(:,2): the applied step input.

sortie(:,3): the output of the system which has the transfer function 1/(s+1).

sortie(:,4): the output of the closed-loop system, with unit feedback.

The simulation parameters are accessible in the Simulation section of the toolbar.