KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
Electrical Engineering Department

EE 380 - Control Engineering **Experiment # 2**

Introduction to SIMULINK and Simulation of a Simple Speed Control System

OBJECTIVES:

1. To become familiar with the Dynamic system simulation software **SIMULINK.**
2. To simulate a simple first-order speed control system using **SIMULINK**.

INTRODUCTION

SIMULINK is a program for simulating dynamic systems. As an extension to MATLAB, SIMULINK adds many features specific to dynamic systems while retaining all of MATLAB's general purpose functionality.

SIMULINK has two phases of use: model definition and model analysis. A typical session starts by either defining a model or recalling a previously defined model, and then proceeds to analysis of that model.

To facilitate model definition, SIMULINK adds a new class of windows called *block diagram* windows. In these windows, models are created and edited principally by mouse driven commands. Part of mastering SIMULINK is to become familiar with the manipulation of model components within these windows.

After you define a model, you can analyze it either by choosing options from the SIMULINK menus or by entering commands in MATLAB's command window.

The progress of a simulation can be viewed while the simulation is running, and the final results can be made available in the MATLAB workspace when a simulation is complete.

TUTORIAL

To master SIMULINK, you must learn how to manipulate blocks and how to build models. You must also become familiar with the types of blocks available. Finally, you must learn how to use the analysis tools provided in SIMULINK. The construction and simulation of a simple model, described in the following section, introduces you to each of these concepts.

SIMULINK uses the metaphor of a block diagram to represent dynamic systems. Defining a system is much like drawing a block diagram. Instead of drawing the individual blocks, blocks are copied from libraries of blocks, either the standard block library supplied with SIMULINK or block libraries you build yourself.

Copyright © Electrical Engineering Department, KFUPM.

The standard block library is organized into several subsystems, grouping blocks according to their behavior. Blocks can be copied from these or any other libraries or models into your model.

Example 1: Generate the sinusoid $4 \sin(2\pi t)$ and display it on the scope.

1. Double-click on the MATLAB icon to display the MATLAB command window.
2. Open the SIMULINK block library by entering the command SIMULINK, or by double-clicking on the SIMULINK icon . This command displays a new window containing icons for the subsystem blocks that make up the standard library.
3. Open the Sources subsystem by double-cliking on the Sources block. This displays another SIMULINK window containing all the blocks that produce output but have no input.
4. Select **New** from the **File** menu to open a new system window.
5. Drag the Signal Generator block from the Sources window to the new, as yet untitled, window. [Notice that you only drag a copy of the block]
6. Open the Signal Generator Block by double-cliking on its icon. This displays a dialog box showing the controls for specifying the waveform, amplitude, and frequency of the signal generated by the block.
7. Select the text box marked **Frequency** by positioning the cursor on it, set the frequency to 1, and the **Units** to Hertz. Select the text box **Amplitude** and set the peak to 4. Click on the **OK** button to close this window.
8. Open the **Sinks library**, and drag a copy of the Scope block to the new system window.
9. Open the **Scope** to see that its window is a graphic representation of an oscilloscope. Use the pointer to select a suitable horizontal range. To set the Y-Axis Limits right-click on an axes and choose **Axes Properties**.
10. The angle bracket (>) pointing out of the **Signal Generator block's** icon represents its output port, and the angle bracket pointing into the **Scope** icon represents its input port. To Connect these two blocks , use the left mouse button to click on either the output or input port of one block, drag to the other block's input or output port to draw a connecting line, and then release the left mouse button. When the blocks are connected, the angle brackets disappear, and a line with an arrowhead shows the direction of the data flow.
11. Pull down the **Simulation** menu and choose **Simulation Parameters .** A dialog box opens showing all the simulation parameters that can be modified. Select the text box for the Maximum Step Size parameter, change the value to 0.01, and click on the **OK** button to close the simulation parameter dialog box.
12. Start the simulation by choosing Start from the **Simulation** menu. The **Signal Generator** outputs the sinusoidal waveform for each time step and the **Scope** shows its input as the trace of a sine wave.
13. The simulation stops when the stop-time in the Control Panel dialog box has been reached, or by choosing **Stop** on the **Simulation** menu.
14. Choose Save from the File menu to display a dialog box in which you can specify the filename and directory for this model.

Copyright © Electrical Engineering Department, KFUPM.

Example 2: Speed-Control System

A speed-control system is shown in basic form in Fig. 1. The system is composed of a dc motor whose shaft speed ω is to be held constant, a power amplifier that controls the motor voltage, a tachometer coupled to the motor shaft, and a circuit that detects the deviation or error between the reference input and the feedback signals. Fig. 2 is a block diagram representation of the system.

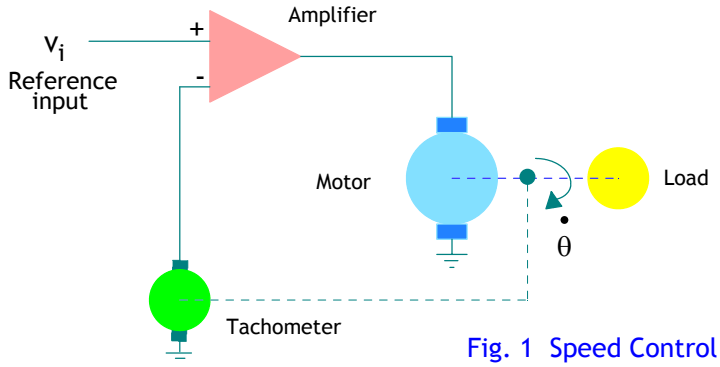


Fig. 1 Speed Control System

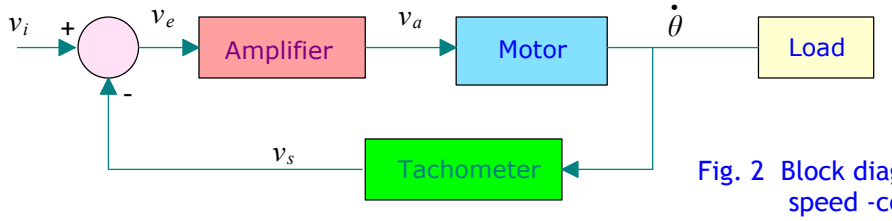


Fig. 2 Block diagram of speed -control system

In analyzing the speed-control system, we will first determine the transfer function of the motor as shown in Fig. 3 :

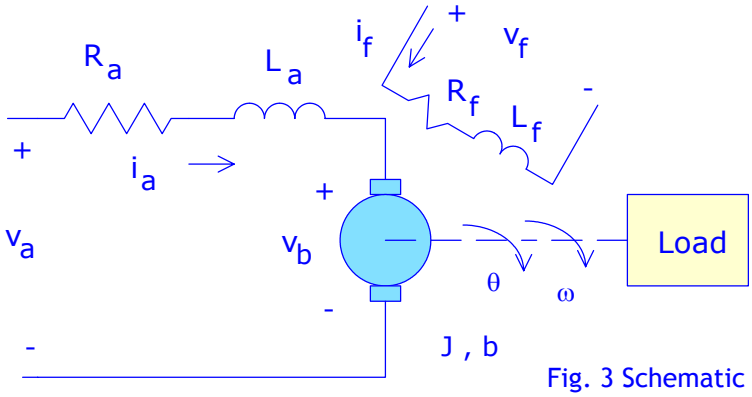


Fig. 3 Schematic diagram of a dc-motor

The torque developed by the motor is given by:

$$T_m = K_1 \phi(t)\ i_a(t) \qquad (1)$$

Copyright © Electrical Engineering Department, KFUPM.

The air-gap flux of the motor is proportional to the field current (neglecting saturation), so that :

$$T_m = K_1 K_f \, i_f(t) \, i_a(t) \qquad (2)$$

If the d.c. motor is armature- controlled, $i_f(t)$ is constant and :

$$T_m(s) = K_m I_a(s) \qquad (3)$$

Also, we have :

$$V_a(s) = V_b(s) + (R_a + sL_a) I_a(s) \quad ; \quad V_b(s) = K_b \Omega(s) \quad \text{(back e.m.f.)} \qquad (4)$$

The load torque for rotating inertia is given by:

$$T_l = Js^2\Theta(s) + bs\Theta(s) \qquad (5)$$

Also, we have

$$T_m(s) = T_l(s) + T_d(s) \quad ; \qquad (6)$$

[$T_d(s)$ is the disturbance torque and is often negligible.]

Using equations (3) to (6), we obtain

$$\frac{\Omega(s)}{V_a(s)} = \frac{K_m}{(R_a + sL_a)(Js + b) + K_b K_m}$$

However for many motors, the armature time constant $\frac{L_a}{R_a}$ is negligible and therefore:

$$\frac{\Omega(s)}{V_a(s)} = \frac{K_m}{R_a b + K_b K_m} \; \frac{1}{(1 + s\frac{JR_a}{R_a b + K_b K_m})} = \frac{K_M}{1 + s\tau_m} \qquad (7)$$

or in the time domain

$$\tau_m \, \dot{\omega}(t) + \omega(t) = K_M \, v_a(t) \qquad (8)$$

for the purpose of the simulation, the following numerical values may be used:

K_A (amplifier gain)	K_M	τ_m	K_t (tachometer constant)
15 V/V	10 rad/sec/V	2.5 sec.	12 mV /rad/sec

referring to Fig. 2, the equations representing the block diagram of the speed-control system are:

Copyright © Electrical Engineering Department, KFUPM.

$$\dot{\omega}(t) + \frac{1}{\tau_m}\omega(t) = \frac{K_m K_A}{\tau_m} v_e(t)$$

$$v_i(t) - v_s(t) = v_e(t) \qquad (9)$$

$$v_s(t) = K_t \omega(t)$$

Note: The tachometer works essentially as a voltage generator, with the output voltage v_s proportional to the magnitude of the angular velocity ω of the input shaft.

Procedure

1. Draw a block diagram for the system.
2. For the open-loop system ($K_t = 0$), obtain a plot of $\omega(t)$. Assume $v_i(t) =$ unit step
3. Calculate the value of τ_m from your graph and compare it with the theoretical value given.
4. How would your graph change if the inertia of the motor shaft J is doubled?
5. For the closed-loop case ($K_t = 12$ mV/sec), obtain a plot of $\omega(t)$.
6. Calculate the system time constant from your graph and compare it with the theoretical value.
7. How would your graph change if the inertia of the motor shaft J is doubled?
8. How would your graph change if the tachometer constant K_t is doubled?

Notes on Simulation

A simulation can be started from either the command line or the **Simulation** menu. All of the methods use the same arguments and menu parameters. :

Simulation from the Menu

1. A simulation can be run by selecting **Start** from the **Simulation** menu. Set the simulation parameters in the Control Panel dialog box by selecting **Simulation Parameters** from the **Simulation** menu.
2. The Control Panel dialog box has fields in which you can enter numbers or any legal MATLAB expression, for example, the variables "Start time", Stop time", "Min step size", and " Max step size", which can be defined in the MATLAB workspace.
3. The return variables [t, x, y] are used to put the time, state, and output trajectories into the MATLAB workspace. The start and stop times for the simulation are set in the variables tstart, and tfinal. The integration parameters minstep,maxstep, and final control the relative local error, minimum step size, and maximum step size of the simulation.

Simulation from the Command Line

1. To configure a simulation with identical parameters as those described by the Control Panel dialog box, use the command
   [T,X,Y] = SIM('model',TIMESPAN,OPTIONS,UT)

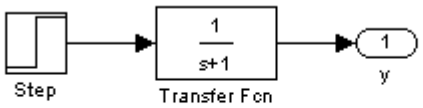
Copyright © Electrical Engineering Department, KFUPM.

where model is the name of the name of a block diagram model. TIMESPAN is one of : Tfinal, [TStart TFinal], or [TStart OutputTimes TFinal].
OutputTimes are time points which will be returned in T, but in general T will include additional time points.
OPTIONS : Optional simulation parameters. This is a structure created with SIMSET using name value pairs.
UT : Optional extern input.
Only the first parameter is required. All defaults will be taken from the block diagram, including unspecified options. Any optional arguments specified will override the settings in the block diagram.

2. All of the integration algorithms have identical calling syntax so that a different method can be selected by simply changing the function name. [euler, rk23, rk45, sim, adams, gear].

SIMULATION STEPS

3. Open the SIMULINK block library by entering the command **simulink**
4. Open the **Continuous** subsystem by double-cliking on the Continuous block.
5. Select **New** from the **File** menu to open a new system window.
6. Build the block diagram of the system by dragging appropriate boxes.
7. Open the Sources subsystem by double-cliking on the Sources block. Drag the step function block .
8. Start the simulation.
9. Output Trajectories from SIMULINK can be plotted using Scope blocks, or Return variables and the MATLAB commands. For example:



Where the block labeled y is an output port taken from the **Sinks** list of blocks. Naming the system tfout ,
[t, x, y] = sim('tfout' , 2)
produces time histories, which can be plotted with plot(t, y)

Report

1. Obtain printouts of the SIMULINK block diagram. of the speed-control system.

2. Attach plots of all time responses for all cases considered. Comments on all your results.

Copyright © Electrical Engineering Department, KFUPM.