CONTROL SYSTEMS LABORATORY
ECE311
LAB 1: The Magnetic Ball Suspension System: Modelling and Simulation Using Matlab

1 Introduction and Purpose

The purpose of this experiment is to familiarize you with the simulation of a magnetic ball-suspension system using MATLAB. You will also become familiar with modelling a nonlinear system using SIMULINK and performing linearization.

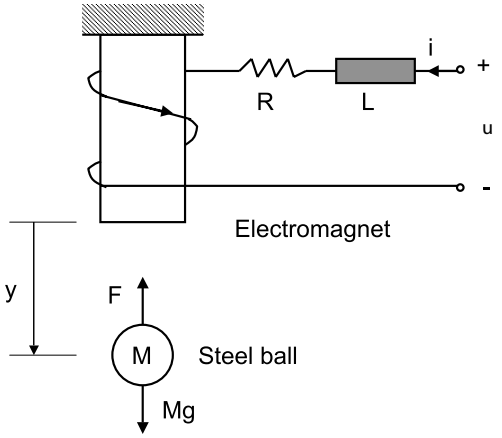

Figure 1: block diagram of a magnetic ball-suspension system

Consider the block diagram of a magnetic ball-suspension system (Figure.1). The position y of the steel ball of mass M can be controlled by adjusting the current i in the electromagnet through the input voltage u. The electromagnet is represented as a series RL circuit with winding resistance R and winding inductance L. The force exerted by the electromagnet on the ball is given as

$$F = \frac{i^2}{y^2}.$$

The following physical parameters will be used throughout this lab:

- $g = 9.8$ m/s^2
- $M = 1$ Kg
- $R = 3$ Ohm

- $L = 1$ H

In this lab we will model and simulate this system.

2 Preparation

In the preparation, you will first find a nonlinear state model of the magnetic ball-suspension system, and then linearize it about an equilibrium point.
Carry out the following steps in your preparation.

1. Take the state vector $x = [\ y \quad \dot{y} \quad i\ ]^T$. Write down the state model for the magnetic ball-suspension system with input the voltage u and output the position y:

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x,u) \end{aligned}$$

   Hint: There are two subsystems to model. First, the steel ball is modelled using Newton's second law. It is subject to two forces, Mg pointing downward and $F = i^2/y^2$ pointing upward. Second, the electromagnet is modelled as an RL circuit with input voltage u and inductor current i. This is the same current appearing in the expression of F above. Write the two equations arising from the two subsystems, and then put them in state space form using the above definition of the state x.

2. Find all the equilibrium conditions of the system parametrized by the position of the steel ball. Specifically, given $y^\star$, find $\dot{y}^\star$, $i^\star$, and $u^\star$ as functions of $y^\star$ such that the pair $(x,u) = ([y^\star \ \ \dot{y}^\star \ \ i^\star]^\top, u^\star)$ is an equilibrium condition of the nonlinear system. You will find two solutions for the equilibrium current $i^\star$. Pick the positive one.

   Linearize the model about the equilibrium condition $(x^\star, u^\star)$, to obtain a system

$$\begin{aligned} \frac{d}{dt}(\delta x) &= A\delta x + B\delta u \\ \delta y &= C\delta x + D\delta u. \end{aligned}$$

   Find A, B, C, D, (the entries of these matrices will be functions of $y^\star$) and define $\delta x, \delta u, \delta y$.

3. Set $y^\star = 1$, and derive the open-loop transfer function $G(s) = \delta Y(s)/\delta U(s)$ from the input δu to the output δy of this linearized system.

4. Find the corresponding impulse-response function $g(t) = \mathcal{L}^{-1}(G(s))$ of this linearized system and plot it.

3 Experiment

In the experiment, MATLAB will be used to build a SIMULINK model of the nonlinear system, and then linearize it to get a linear approximation.

3.1 Building the SIMULINK Model

1. Recall that if you take the state vector $x = [\ y \quad \dot{y} \quad i\ ]^T$, the nonlinear differential equation for x can be written in the following form

$$\dot{x} = f(x,u), y = h(x,u)$$

   which has been obtained in your preparation.

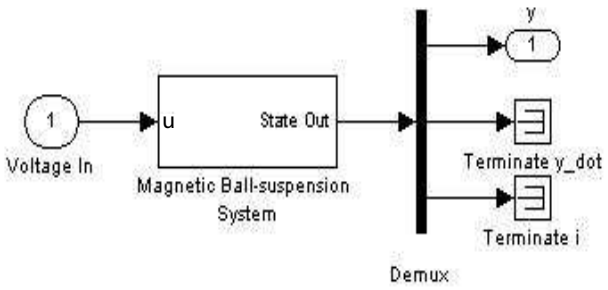

Figure 2: The high-level Simulink model

2. You are now going to implement your state model in Simulink. You will begin by creating the block diagram depicted in Figure 2. Open the Library browser of Simulink and look at the `Commonly Used Blocks`. Using the following blocks:

   `Subsystem Demux In1 Out1 Terminator`

   create a diagram identical to the one in Figure 2.

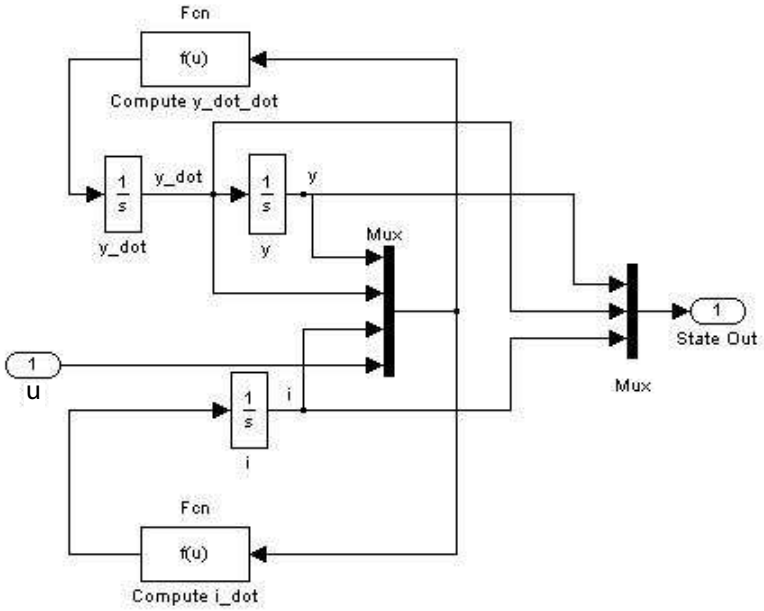

Figure 3: Subsystem Block

3. Next, you need to implement your mathematical model of the magnetic levitation system, which will go inside the `Magnetic Ball-suspension System` block. Double click on this subsystem block. You'll see an input directly connected to an output. Erase the connection between input and output, and replace it with the block diagram in Figure 3. In that diagram, you see three blocks $1/s$. These are integrators, you can find them in the `Continuous` library. There are two integrators in series.

Their input is $\ddot{y}$, and their output is y. The input of these two integrators is computed using a function block $f(\cdot)$, found in the `User-Defined Functions` library. This function takes as input the output of the multiplexer, which is the vector $[y \quad \dot{y} \quad i \quad u]^\top = (x, u)$. Summarizing, the upper part of the block diagram in Figure 3 implements an equation $\ddot{y} = f(x, u)$. you will have to insert the correct function f according to your mathematical model. Now look at the single integrator block in the lower part of the diagram. Its output is i, and its input is di/dt. What feeds this integrator is the output of another function block $f(x, u)$ (this f is different than then one above). Again, you will have to put the appropriate function in this block. Save your complete model in SIMULINK as `magball`.

Have the TA sign here if your model is correct.

3.2 Linearizing the Model in MATLAB

1. For the real magnetic ball-suspension system, we have the following physical parameters:

   - g = 9.8 m/s^2
   - M = 1 Kg
   - R = 3 Ohm
   - L = 1 H

   Substituting in the values of the physical parameters, setting the equilibrium position $y^\star$ to be $y^\star = 1$, and based on the linearized state equation obtained in your preparation, enter the linearized system matrices A and B in the Matlab workspace.

2. Using the Matlab command `eig`, determine the numerical values of the eigenvalues of A. We will see later in this course that if all eigenvalues of A have negative real part, then the linearized system is *stable*. For the linearized maglev system, stability means that for any initial condition, $(y(t), \dot{y}(t), i(t)) \to (y^\star, \dot{y}^\star, i^\star)$ as $t \to \infty$. Vice versa, if A has at least one eigenvalue with positive real part, then the linearized maglev system is *unstable*, meaning that there are initial conditions such that the triple $(y(t), \dot{y}(t), i(t))$ does not converge to the equilibrium $(y^\star, \dot{y}^\star, i^\star)$.

   Look at the eigenvalues you just found: is the linearized system stable or unstable? Discuss with your lab partner the physical intuition behind the result (i.e., why the system is stable or unstable). You'll include a discussion on this in your lab report.

   Using the Matlab command `ss2tf`, use your matrices (A, B, C, D) to find the transfer function of the linearized model. Compare it with the transfer function you found in your preparation and check that they coincide.

   Find the poles and zeros of $G(s)$.

   Have the TA check your work and sign here.

3. Now you will learn how to use Simulink to automatically linearize nonlinear models, and you will compare the result to your own computations. Using the Matlab command `linmod`, you will linearize your system at the equilibrium point $(x^\star, u^\star)$. First you need to understand the ordering of states in Simulink. For that, issue the command

   ```
   >> [sizes, x0, states]=magball
   ```

   Look at the `states` array. The ordering of the strings in this array coincides with the ordering of states used by Simulink. The string names are the labels you have used under each integrator block $(1/s)$. Now use a command of the form `[A,B,C,D]=linmod('magball',xstar,ustar)` to generate the (A, B, C, D) matrices of the linearization at the equilibrium condition $(x^\star, u^\star)$ found in part 2 of your prep, setting $y^\star = 1$. Check that the matrices (A, B, C, D), other than a possible re-ordering of the states, are the same as those obtained earlier in Step 1.

   Have the TA check your work and sign here.

4. Use the MATLAB command `impulse` to obtain the impulse response and check its appearance vs the result in your preparation.

5. Use the following two ways to obtain the step response (that is, the plot of the output signal when the input fed to the system is the unit step).

   (a) Use the MATLAB command **step**.

   (b) Modify the Fcn blocks in Fig.3 to get the linearized model (as in Fig.4). The two equations in Fcn blocks are corresponding to the A and B matrices you have obtained in Step 3. Save the new SIMULINK model as `magball_linear`. Then use Step block as the input source and place a Scope block to display the position y during the simulation. Set the simulation stop

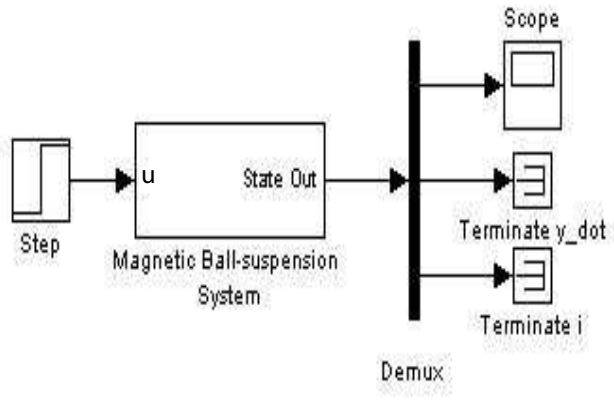

Figure 4: Linearized SIMULINK Model

   time to be 2(sec) and run the simulation.

   Check that the result is the same as the one obtained earlier in part (a).

   Have the TA check your work and sign here.

4 Report

Please submit this document with all TA signatures stapled together with your lab report. For the lab report, please follow the format instructions for Lab 1 provided on the course website.

Appendix

The following is a list of MATLAB commands which may be useful for completing this lab. You may also refer to the MATLAB Tutorial Handout on the course website (under the folder "general handouts") or the reference library on the Mathworks website for more MATLAB functions and the SIMULINK usage.

- **eig** - Determines the eigenvalues of a matrix
- **help** - Access to help on MATLAB commands. e.g. **help plot**
- **impulse** - Simulates an impulse response for a LTI system. e.g. **impulse(sys)**
- **legend** - Creates a legend for a plot. e.g. **legend('Reference Step','Position')**

- **linmod** - Extract the continuous- or discrete-time linear state-space model of a system around an operating point. e.g. **linmod('sys', x0,u0)**, where $x0$ and $u0$ are the state and the input vectors. If specified, they set the operating point at which the linear model is to be extracted.
- **load** - Loads mat files into the workspace.
- **plot** - Creates a plot on a figure. e.g. **plot(time, output,'b')**
- **sim** - Runs a SIMULINK model file. e.g. **sim('***sim_position_pv_ip*02**',10)**
- **ss** - Creates a state-space system. e.g. **sysss = ss(A,B,C,D)**
- **step** - Simulates a step response for a LTI system. e.g. **step(sys)**
- **tf** - Creates a transfer function. e.g. **systf = tf([1 -2],[1 7 2])**
- **tf2ss** - Converts a transfer function to a state-space system. e.g. **sysss=tf2ss([1],[1 -4 7])**
- **title** - Creates a title for a plot. e.g. **title('Step Response for Linear System')**
- **ss2tf** - Converts a state-space system to a transfer function. e.g. **systf=ss2tf(A,B,C,D)**
- **xlabel/ylabel** - Creates a label for the axis of a plot. e.g. **xlabel('Time (seconds)')**