Assignment

Linear Systems and Control (MMAN 3200, session T1.2020)

MATHEMATICAL MODELING WITH SPEED AND DIRECTION CONTROL OF A DC MOTOR

PRE-REQUISITES

By now the students are expected to be able to derive mathematical models of simple electrical and mechanical systems and build differential equations (DE) and Laplace-Transformation corresponding to those physical systems.

OBJECTIVES

The objective of this online experiment is to show how a permanent-magnet DC motor may be controlled by varying the magnitude and direction of its armature current and recognize its torque/speed characteristics. In addition, this task will help students to understand the concept of PID controller.

LAB INSTRUCTIONS

- This online lab activity comprises of three parts: Pre-lab and Lab Exercises.
- The students should perform and demonstrate each lab task separately for stepwise evaluation (please ensure that tutors have signed each step after ascertaining its functional verification)

LAB REPORT INSTRUCTIONS

All questions should be answered precisely to get maximum credit. Lab report must ensure following items:

- Lab objectives
- MATLAB/Simulink
- Results (graphs/tables) duly commented and discussed
- Conclusion

THEORY AND PRE-LAB

DC motors that are used in feedback-controlled devices are called DC servomotors. Applications of DC servomotors abound, e.g. in robotics, computer disk drives, printers, aircraft flight control systems, machine tools, flexible manufacturing systems, automatic steering control etc. DC motors are classified as *armature-controlled* DC motors and *field-controlled* DC motors.

This experiment will focus on modeling, identification, and position control of an armature-controlled DC servomotor. We will first develop the governing differential equations and the Laplace domain transfer function model of an armature-controlled DC motor. Next, we will tend to the identification of the unknown system parameters that appear in the transfer function model of the DC servomotor. Finally, we will develop and implement a position-plus-velocity feedback controller to ensure that the DC motor angular position response tracks a step command.

The motor, we would be discussing, is a permanent-magnet type and has a single armature winding. Current flow through the armature is controlled by power amplifiers as in figure A so that rotation in both directions is possible by using one, or both inputs.

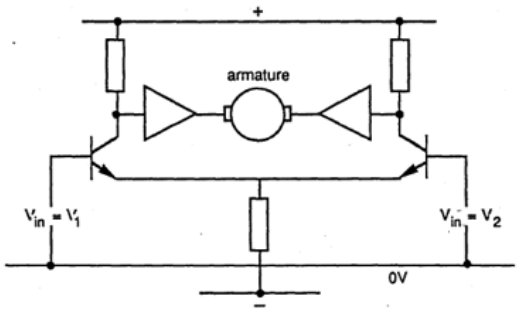


Figure-A: Armature Control of Permanent Magnet DC Motor

MODELLING OF ARMATURE-CONTROLLED DC MOTOR

As the motor accelerates, the armature generates an increasing *back-emf*, V_b, which tends to oppose the driving voltage V_a. The armature current is thus roughly proportional to $V_a - V_b$. If the speed drops (due to loading) V_b reduces, the current increases and so does the motor torque. This tends to oppose the speed drop. That is why this mode of control is called 'armature-control' and gives a speed proportional to V_a.

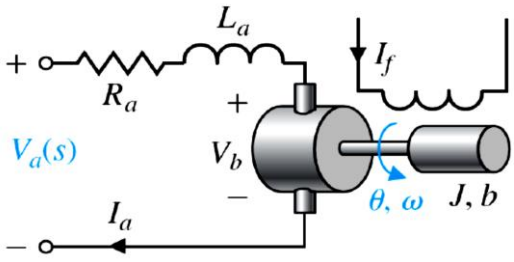


Figure-B: Armature Controlled DC Motor

To derive a mathematical model for the motor under discussion, consider the schematic shown in figure-B. As we know that a motor is an electro-mechanical device, so its mathematic model consists of two differential equations each representing one of its aspects. First, we observe that the torque of a permanent-magnet DC motor is directly proportional to its armature current. Stating mathematically, we say,

$$T_m = K_t \cdot i_a(t) \qquad (1)$$

where K_t is motor torque constant.

We also note that the back-emf which is generated in the armature-controlled DC motor is directly proportional to the angular velocity of armature. Thus,

$$V_b = K_b \cdot \omega_a(t) \qquad (2)$$

Here K_b is the motor constant. In SI units, however, the value of $K_b$and K_t is equal.

When the motor rotates and produces a torque as given in equation 1, there emerges a *retarding* torque, because of friction, in response to T_m. If we assume the friction to be viscous friction, as we assumed for the case of mass-spring model in previous labs, we get

$$T_r = b \cdot \omega_a(t) \tag{3}$$

Newton's second law of motion, when translated into angular mechanics, takes the following form.

$$\sum T = J \cdot \alpha(t) \tag{4}$$

Here J is moment of inertia and $\alpha(t)$ is angular acceleration. Putting value, we get

$$T_m - b \cdot \omega(t) = J \cdot \frac{d\omega(t)}{dt} \tag{5}$$

Or,

$$\frac{d\omega(t)}{dt} = \frac{1}{J} (K_t \cdot i_a(t) - b \cdot \omega(t)) \tag{6}$$

Equation (6) is the mechanical equation for an armature-controlled DC motor.

Now we turn to the electrical aspect of this motor. Applying Kirchhoff's Voltage Law to figure 2, we get,

$$V_a = R \cdot i_a(t) + L \cdot \frac{di_a(t)}{dt} + V_b \tag{7}$$

Substituting equation (2) and rearranging the terms gives,

$$\frac{di_a(t)}{dt} = \frac{1}{L} \cdot (V_a - R \cdot i_a(t) - K_b \cdot \omega_a(t)) \tag{8}$$

Equation (8) gives the electrical equation of the motor. Combining these two equations results in a complete model of armature-controlled DC motor.

BUILDING MODEL IN SIMULINK

Using equations (6) and (8) build a Simulink model. Feel free to ask your lab instructor if you stuck anywhere. Your final model should look like the one shown in figure-C.

Once the model is built, save it and run the following commands on MATLAB command window.

```
>> J=3.2284E-6;
>> b=3.5077E-6;
>> Kt=0.0274;
>> Kb = 0.0274;
>> R=4;
>> L=2.75E-6;
```

(note: all these parameters are expressed/assumed in proper engineering units.)

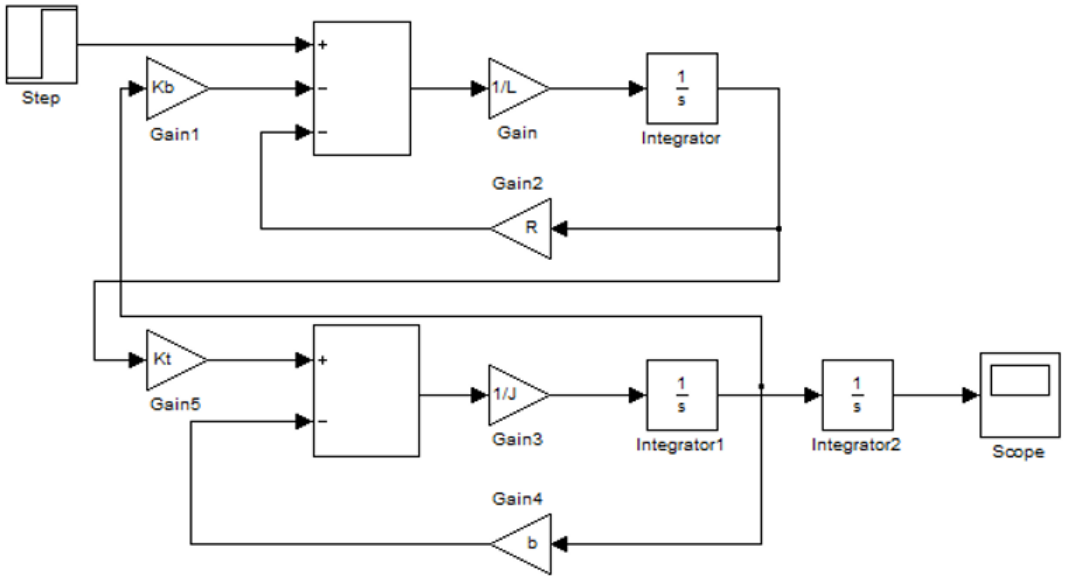


Figure-C: Simulink model of DC Motor

To simulate this system, first an appropriate simulation time must be set. Select Parameters from Simulation menu and enter "0.4" in the Stop Time field. 0.4 seconds is long enough to view the open-loop response. Also, in the Parameters dialog box, it is helpful to change the Solver Options method. Click on the field which currently contains "ode45 (Dormand-Prince)". Select the option "ode15s (stiff/NDF)". Since the time scales in this example are very small, this stiff system integration method is much more efficient than the default integration method. Now run the simulation (Ctrl-t or Start on the Simulation menu). When the simulation is finished, double-click on the scope and hit its auto scale button.

CREATING A FUNCTION BLOCK IN SIMULINK

We can encapsulate the whole Simulink model designed above in a single block to be used as a component in subsequent designs. It enables us to abstract unnecessary details of a subsystem and allows us to work with the block through its interface only. To do so, remove the input(s) and output(s) of a Simulink model and replace them with In and Out blocks. Now select the whole system and hit on Create Subsystem option from Edit menu. Alternatively, you may right click on the selected model and select Create Subsystem from the menu. The complete model will be housed in a single block with input(s) and output(s) going in and coming out of it as shown in figure-D. Block, respectively (these blocks can be found in the Connections block library). This defines the input and output of the system for the extraction process.

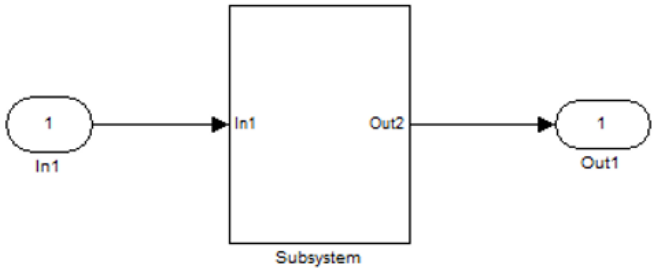


Figure-D: Open-Loop DC Motor System

EXERCISE 1 (OPEN-LOOP ANALYSIS)

Apply step-response in the open-loop DC Motor system and fill the following table:

DC Voltage	**Max $\omega(t)$**	**Max $\theta(t)$**	**DC Voltage**	**Max $\omega(t)$**	**Max $\theta(t)$**
1			-1		
2			-2		
3			-3		
4			-4		

Note: Replace In1 with Step block and Out1 with Scope

EXERCISE 2 (DE TO TRANSFER-FUNCTION CONVERSATION)

Obtain the transfer function $\boldsymbol{\omega(s)/V(s)}$ from the equations (7) and (8). Design a Simulink model and apply unit-step response to the obtained transfer-function $\boldsymbol{\omega(s)/V(s)}$**.**

EXERCISE 3 (CLOSED-LOOP RESPONSE WITH P, PI, AND PID CONTROLLER)

Consider the case in which a unit-step function is applied as input to a close loop system, which has a unity negative feedback (H(s) = 1), with controller $G_c(s)$, and G(s) the open-loop DC Motor model.

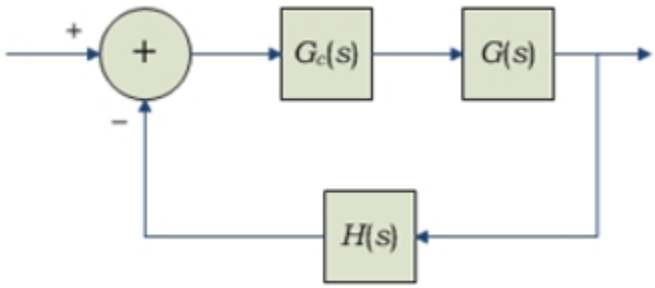


Figure-D: Closed-loop DC Motor System

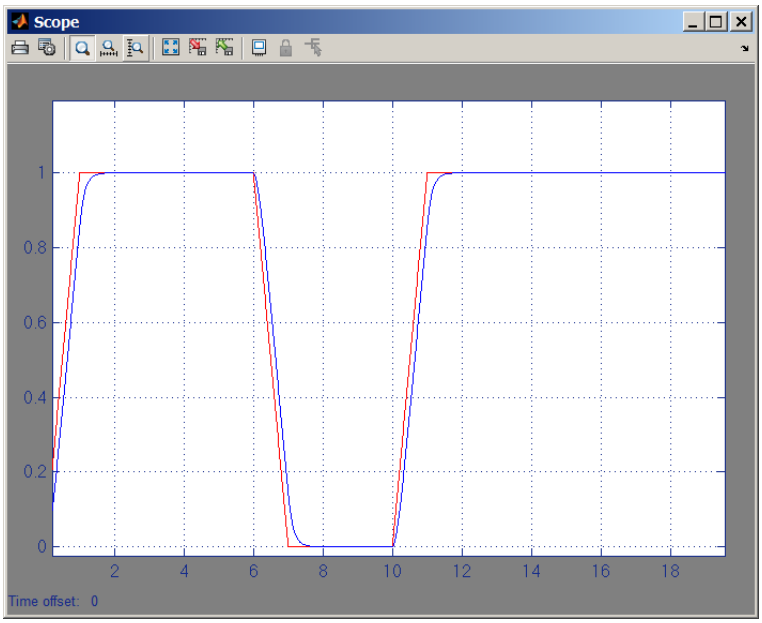
Perform the following Tasks:

1. Design Proportional Controller ($G_c(s)= K_p$). Use appropriate value of K_p and comment on the transient and steady-state response of the system.
2. Design Proportional plus Integral (PI) Controller ($Gc(s) = K_p + 1/s\ K_i$). Use appropriate values of K_p and K_i and comment on the transient and steady-state response of the system.
3. Design Proportional plus Integral (PI) Controller ($Gc(s) = K_p + 1/s\ K_i + s\ K_d$). Use appropriate value of K_p, K_i and K_d and comment on the transient and steady-state response of the system.

Hint: PID Controller block in Simulink

4. Easy task. Try your controllers using less aggressive inputs, such as sequences of ramps. Propose one sequence and present its associated Closed Loop response. An example of such as input can be seen in the

following figure, in which the red curve is the input being applied to certain system, whose output resulted in the blue curve. In this case, the transitions (using linear ramps) were given a duration of 1 second. A Simulink block useful for such type of signal generations is the "repeating sequence" source; you may use it for the mentioned purpose.



Note: This project replaces the original lab project which was based on using a real platform in the MTRN lab. (the "monorail"). However, due to restrictions in accessing the MTRN lab (due to the current epidemic), a replacement project, fully based on simulations (and feasible to be solved at home, individually), has been included. This new project requires the student to apply concepts which have been acquired during weeks 1 to 7 (concepts about state space, discussed on week 9, can be exploited, but those are not requested to be applied for solving the modelling component of this project.)

Note#2: This project is intended to be solved individually; it is not a group assessment.

The deadline for the submission of this project is Monday 27/April (week 11), 10:00AM. Submission site, in Moodle, will be open from week 9, for students who prefer a early submission.

Details about the submission of files will be released via Moodle, by the end of week 8.