EXERCISE-1(A)

MODELING, SIMULATION OF LTI SYSTEMS AND OBTAINING THEIR RESPONSES FOR DIFFERENT TEST SIGNALS

DATE: REGISTER NUMBER:

PREREQUISITE KNOWLEDGE:

Fundamentals of modelling and simulation procedures, use of standard test signals and their relevance in the response analysis of systems, basics of developing models in MATLAB and LabVIEW environment.

OBJECTIVES:

- To learn the concepts of modeling and simulation of dynamic systems using software platforms.
- To understand the importance of standard test signals in the response analysis of LTI systems.

PRE-LAB READINGS:

Linear Time-Invariant Systems in MATLAB:

Control System Toolbox in MATLAB offers extensive tools to manipulate and analyze linear time-invariant (LTI) models. It supports both continuous- and discrete-time systems. Systems can be single-input/single-output (SISO) or multiple-input/multiple-output (MIMO). You can specify LTI models as:

$$P(s) = \frac{s+2}{s^2+s+10}$$

Note: All LTI models are represented as a ratio of polynomial functions.

Examples of Creating LTI Models

Building LTI models with Control System Toolbox is straightforward. The following sections show simple examples. Note that all LTI models, i.e. TF, ZPK and SS are also MATLAB objects.

Example of Creating Transfer Function Models

You can create transfer function (TF) models by specifying numerator and denominator coefficients. For example,

```
>>num = [1 0];

>>den = [1 2 1];

>>sys = tf(num,den)
```

A useful trick is to create the Laplace variable, s. That way, you can specify polynomials using s as the polynomial variable.

```
>>s=tf('s');
```

```
>>sys= s/(s^2 + 2*s + 1);
```

Example of Creating Zero-Pole-Gain Models

To create zero-pole-gain (ZPK) models, you must specify each of the three components in vector format. For example,

```
>>sys = zpk([0],[-1 -1],[1])
```

Response of Linear systems to different inputs

You can simulate the LTI systems to inputs like impulse, step and other standard inputs and see the plot of the response in the figure window. MATLAB command ‘impulse’ calculates the unit impulse response of the system, ‘step’ calculates the unit step response of the system and ‘lsim’ simulates the (time) response of continuous or discrete linear systems to arbitrary inputs. When invoked without left-hand arguments, all three commands plots the response on the screen. For example:

To obtain an impulse response

```
>> H = tf([2 5 1],[1 2 3]);
```

```
>>impulse(H)
```

To obtain a step response type

```
>>step(H)
```

Time-interval specification:

To contain the response of the system you can also specify the time interval to simulate the system to. For example,

```
>> t = 0:0.01:10;
```

```
>>impulse(H,t)
```

Or

```
>>step(H,t)
```

Simulation to Arbitrary Inputs:

To simulates the (time) response of continuous or discrete linear systems to arbitrary inputs use‘lsim’. When invoked without left-hand arguments, ‘lsim’ plots the response on the screen.

`lsim(sys,u,t)` produces a plot of the time response of the LTI model sys to the input time history ‘t’,’u’. The vector ‘t’ specifies the time samples for the simulation and consists of regularly spaced time samples.

```
T = 0:dt:Tfinal
```

The matrix u must have as many rows as time samples (`length(t)`) and as many columns as system inputs.

Simulate and plot the response of the system

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

to a square wave with period of four seconds. First generate the square wave with gensig. Sample every 0.1 second during 10 seconds:

```
>>[u,t] = gensig('square',4,10,0.1);
```

Then simulate with lsim.

```
>> H = tf([2 5 1],[1 2 3])
>>lsim(H,u,t)
```

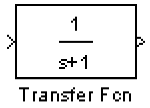
Solving the Differential Equation Using SIMULINK

To create a simulation in Simulink, follow the 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.
- 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.
- 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.
- 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. Select the duration of the simulation to be 10 seconds.

Linear Time-Invariant Systems in SIMULINK:

The Transfer Fcn block models a linear system by a transfer function of the Laplace-domain variable s. The block can model single-input single-output (SISO) and single-input multiple output (SIMO) systems.



Conditions for Using This Block

The Transfer Fcn block assumes the following conditions:

- The transfer function has the form

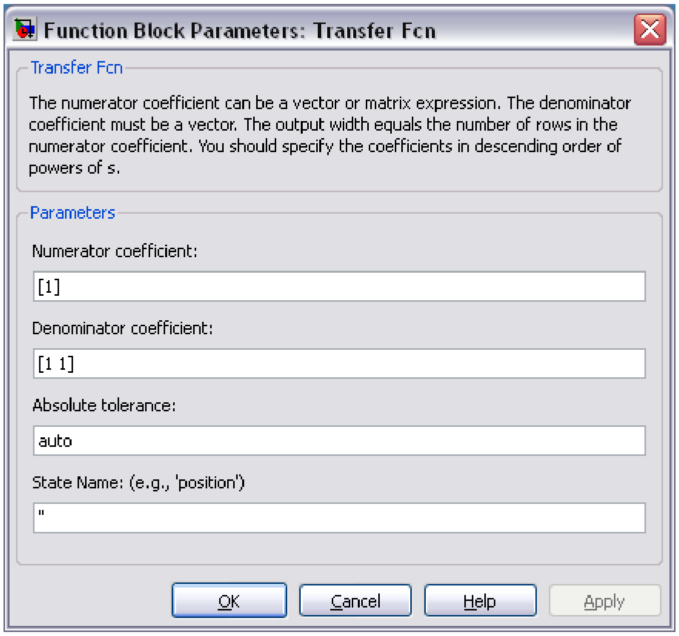
$$H(s) = \frac{y(s)}{u(s)} = \frac{num(s)}{den(s)} = \frac{num(1)s^{nn-1} + num(2)s^{nn-2} + \ldots + num(nn)}{den(1)s^{nd-1} + den(2)s^{nd-2} + \ldots + den(nd)}$$

where u and y are the system input and outputs, respectively, nn and nd are the number of numerator and denominator coefficients, respectively. num(s) and den(s) contain the coefficients of the numerator and denominator in descending powers of s.

- The order of the denominator must be greater than or equal to the order of the numerator.

For a single-output system, the input and output of the block are scalar time-domain signals. To model this system:

- Enter a vector for the numerator coefficients of the transfer function in the Numerator coefficients field.
- Enter a vector for the denominator coefficients of the transfer function in the Denominator coefficients field.



The numerator coefficient can be a vector or matrix expression. The denominator coefficient must be a vector. The output width equals the number of rows in the numerator coefficient. You should specify the coefficients in descending order of powers of s.

Numerator coefficient: [1]

Denominator coefficient: [1 1]

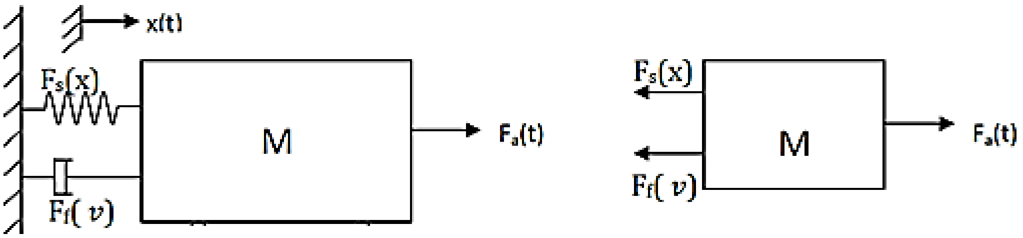
Absolute tolerance: auto

State Name: (e.g., 'position') ''

PRE-LAB TASKS:

TASK-1:

- Derive the transfer function relating displacement X(s) and input force F(s) of the mass-spring-damper system.
  The parameter values are M = 750 kg, K= 0 N/m, and B =30 N.s/m. The input force increases from 0 to 300 N at t = 125 s.



Where

$$a = \frac{dv(t)}{dt} = \frac{d^2x(t)}{dt^2} \text{ is the acceleration,}$$

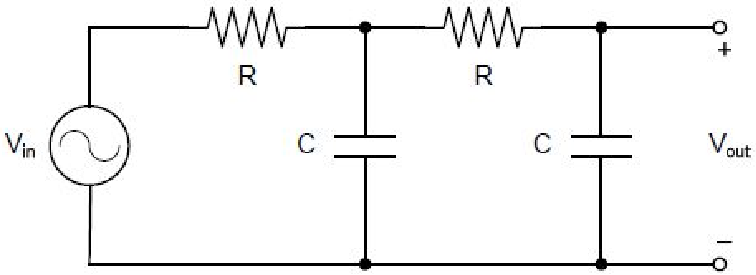
$$v = \frac{dx(t)}{dt} \text{ is the speed,}$$

and

$$x(t) \text{ is the displacement.}$$

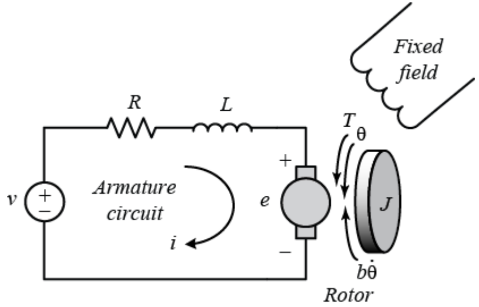
TASK-2:

- Derive the transfer function (V_{out}/V_{in}) of the following circuit with the applied voltage as input and the voltage across the capacitor as output. Apply the constant parameters in the transfer function such that the circuit consists of a 1kΩ resistor and a 160uF capacitor.



TASK-3:

- Derive the transfer function ($\theta(s)/V(s)$) of armature-controlled DC motor. Apply parameter values R=4.38 ohm; L=2.15 mH; Torque constant K_t=1.94 Nm/A ; Back emf constant K_b= 1.43 V/rad/s ; J= $2.2*10^{-4}$ kg.m^2 and Friction torque= 0.4 N.m .



IN-LAB TASKS:

TASK-1:

- **Program the model you derived in pre-lab task-1 using MATLAB commands for realizing the system the Translational mechanical system involving Spring Mass damper and plot its impulse and step responses.**

TASK-2:

- **Program the model you derived in pre-lab task-1 using MATLAB commands for realizing the system the Electrical circuit involving RC elements and plot its impulse and step responses.**

TASK-3:

- **Program the model you derived in pre-lab task-1 using MATLAB commands for realizing the system the Armature controlled DC motor and plot its impulse and step responses.**

TASK-4:

- **Develop the models of the above systems in SIMULINK as well and obtain their impulse and step responses.**

ADDITIONAL TASKS/LEARNING FOR BONUS MARKS:

- **Obtain the responses of the above systems for other test input signals as well and present your interpretations in terms of input-output relationship and system dynamics. Try to relate the basic physics behind the system intuitively.**
- **Change the system parameters in the developed models, explore what happens to the response of the systems for each type of input. Present your interpretations in-terms of input-output relationship against a parameter variations and comment about Linear Time Invariant and Time Variant systems. Try to relate the basic physics behind the system intuitively.**
- **Check if the developed models were reasonably accurate representations. If not, learn what aspects of systems were assumed negligible to arrive at such approximate and simplified versions. Analyze whether those assumptions are always valid or not. Present deeper understanding about the trade off among model accuracy, order of the system, simulation time/complexity and system needs.**
- **Develop the model of a field-controlled DC motor and simulate its response for various test signals. You may also develop model and simulate the response of any other complex systems you have encountered in system dynamics course.**

RESULTS & INFERENCES:

Evaluation Component	Maximum Marks	Marks Obtained
Pre-lab Tasks	10	
In-Lab Tasks	20	
Post-lab Tasks	10	
Bonus Tasks	10	
Signature of Faculty with Date		

(This page must be the last page of the exercise)