National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Department of Electrical Engineering and Computer Science

Faculty Member: Ma'am Neelma Naz **Dated:** 21/09/2022

Semester: 6th **Section:** BEE 12C

EE379: Control Systems

Lab 5 : **State space, response of systems to various inputs, and interconnections of systems**

Lab Instructor: Sir. Yasir Rizwam

Group Members

Student Name	Reg. No.	Lab Report Marks / 10	Viva Marks / 5	Total /15
Muhammad Ahmed Mohsin	333060			
Imran Haider	332569			
Zafar Azhar	340908			

1 Table of Contents

State space, response of systems to various inputs, and interconnections of systems

2 INTRODUCTION

Learn how to create state space models in MATLAB, Simulink and LabVIEW. • Learn how to find time response of a system for various inputs in MATLAB, Simulink and LabVIEW. • Interconnection of systems in MATLAB.

3 STATE SPACE REPRESENTATION IN MATLAB

We have already seen two ways to represent models i.e. by a set of differential equations and by transfer functions. In this handout we will learn a third way of representing models called the state space representation. Learning this representation is vital because it is the basis of all the modern control techniques. It is important to learn all these different types of representations. All of these representations are used in control and modeling literature. Moreover, each representation has its own benefits. For example as you will learn later, it is easy to plot a Bode diagram if you have a transfer function representation. Whereas, for pole placement (a controller design technique), state space representation is more suitable. Nonetheless at this stage you should be aware with these basic system representations. A state space representation is a mathematical model of a physical system as a set first order differential equations that relate the input variables, output variables and state variables. The state variables are the smallest possible subset of variables that can represent the entire state of the system at any given time. One advantage of state space is that unlike transfer functions it can also be used for non-linear and/or time variant systems. Don’t worry if you don’t understand too much about state variables or state space over here. It should be covered in more detail in the theory part of the course.

4 PRE LAB TASK

4.1 CODE

```
%% Pre-Lab
```

```
R = 1;
L = 1e-3;
C = 1e-6;

A = [-R/L -1/L; 1/C 0];
B = [1/L; 0];
C = [-R -1];
D = 1;

rlc_ss = ss(A, B, C, D);
rlc_tf = tf(rlc_ss);
rlc_ss2 = ss(rlc_tf);

display(rlc_tf)
```

4.2 Output

```
rlc_tf =

          s^2
  --------------------
  s^2 + 1000 s + 1e09

Continuous-time transfer function.
```

5 Lab Task 1

Use the transfer functions of motor speed, motor position and pendulum arm angle to find their state space representation.

5.1 Code

```
% Variables
J_eq = 1.843e-6;
m = 0.0270;
r = 0.0826;
K_g = 1;
J_m = 1.80e-4;
n_g = 1;
n_m = 0.69;
L = 0.0955;
```

```
g = 9.9;
K_t = 0.0334;
R = 8.7;
a = J_eq + m * r ^ 2 + n_g * K_g ^ 2 * J_m;
b = m * L * r;
c = (4/3) * m * L ^ 2;
d = m * g * L;
e = 2.7183;
f = (n_m * n_g * K_t * K_g) / R;
% Motor speed
num = [-c 0 f*d 0];
den = [(b^2-a*c) -e*c a*d e*d 0];
speed_tf = tf(num, den);
% Motor position
num = [-c 0 f*d];
den = [(b^2-a*c) -e*c a*d e*d 0];
position_tf = tf(num, den);

% Pendulum arm angle
num = [-b*f 0 0];
den = [(a*c-b^2) e*c -a*d e*d 0];
arm_tf = tf(num, den);

speed_ss = ss(speed_tf);
position_ss = ss(position_tf);
arm_ss = ss(arm_tf);

display(speed_ss)
display(position_ss)
display(arm_ss)
```

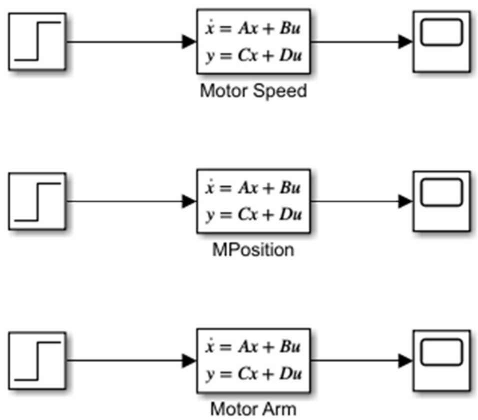
Output:

```
speed_ss =

  A =
              x1         x2         x3         x4
   x1  -1.193e+04      1.951      113.2          0
   x2          64          0          0          0
   x3           0        128          0          0
   x4           0          0          1          0

  B =
       u1
   x1  64
   x2   0
   x3   0
   x4   0

  C =
             x1         x2         x3         x4
   y1     68.56          0  -0.001724          0

  D =
       u1
   y1   0

Continuous-time state-space model.


position_ss =

  A =
              x1         x2         x3         x4
   x1  -1.193e+04      1.951      113.2          0
   x2          64          0          0          0
   x3           0        128          0          0
   x4           0          0          1          0

  B =
       u1
   x1   8
   x2   0
   x3   0
   x4   0
```

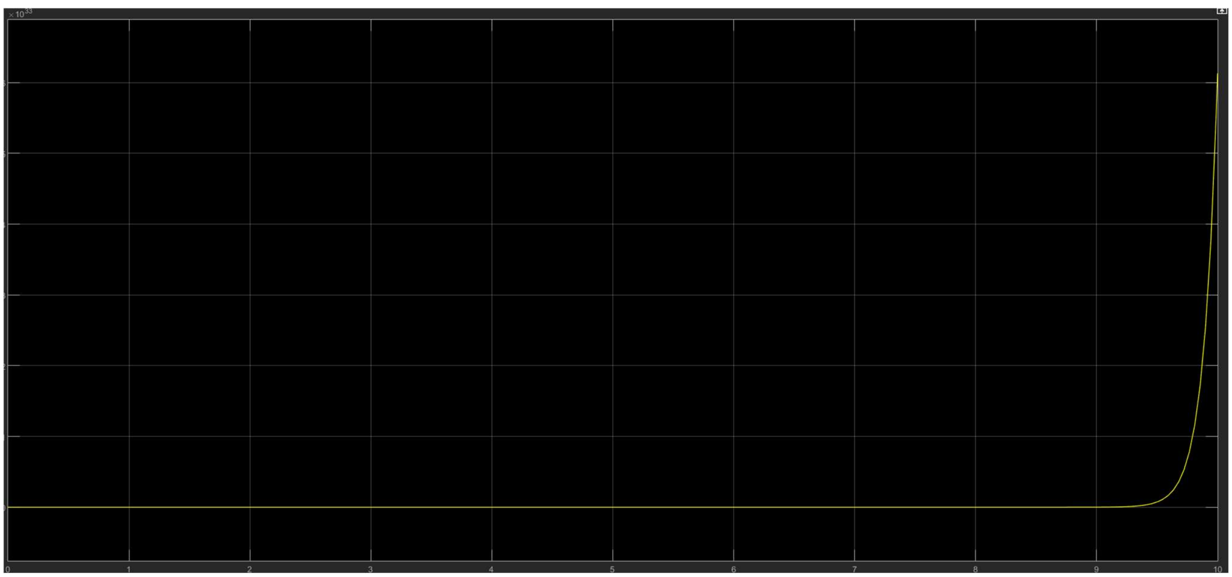
```
  C =
             x1        x2        x3        x4
   y1         0      8.57         0  -0.01379

  D =
       u1
   y1   0

Continuous-time state-space model.


arm_ss =

  A =
                 x1          x2          x3          x4
   x1  -1.193e+04       1.951      -113.2           0
   x2          64           0           0           0
   x3           0         128           0           0
   x4           0           0           1           0

  B =
         u1
   x1  0.25
   x2     0
   x3     0
   x4     0

  C =
            x1        x2        x3        x4
   y1        0   -0.4713         0         0

  D =
       u1
   y1   0

Continuous-time state-space model.
```

6 SIMULINK SPACE

6.1 SIMULINK:



6.2 OUTPUT

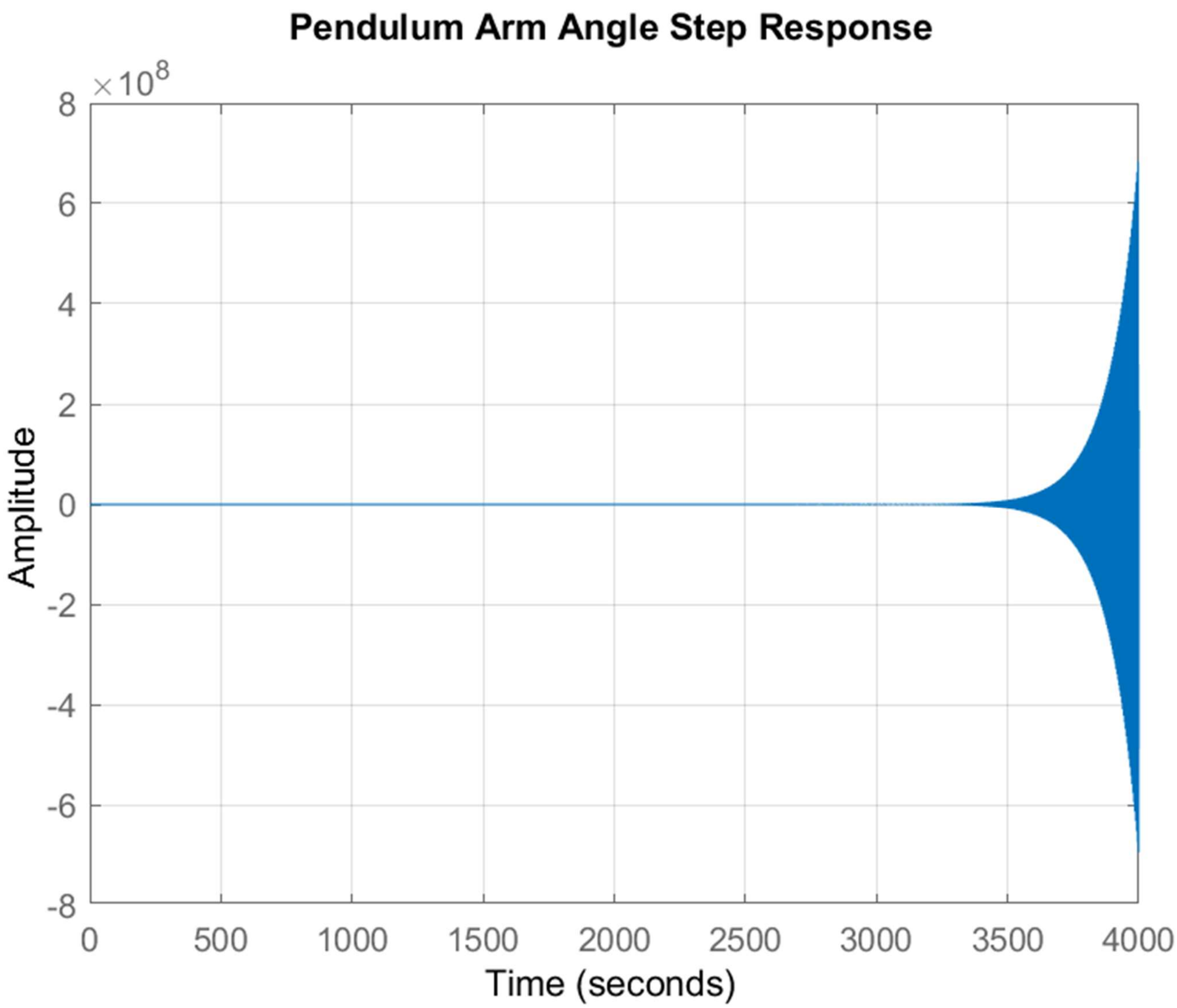


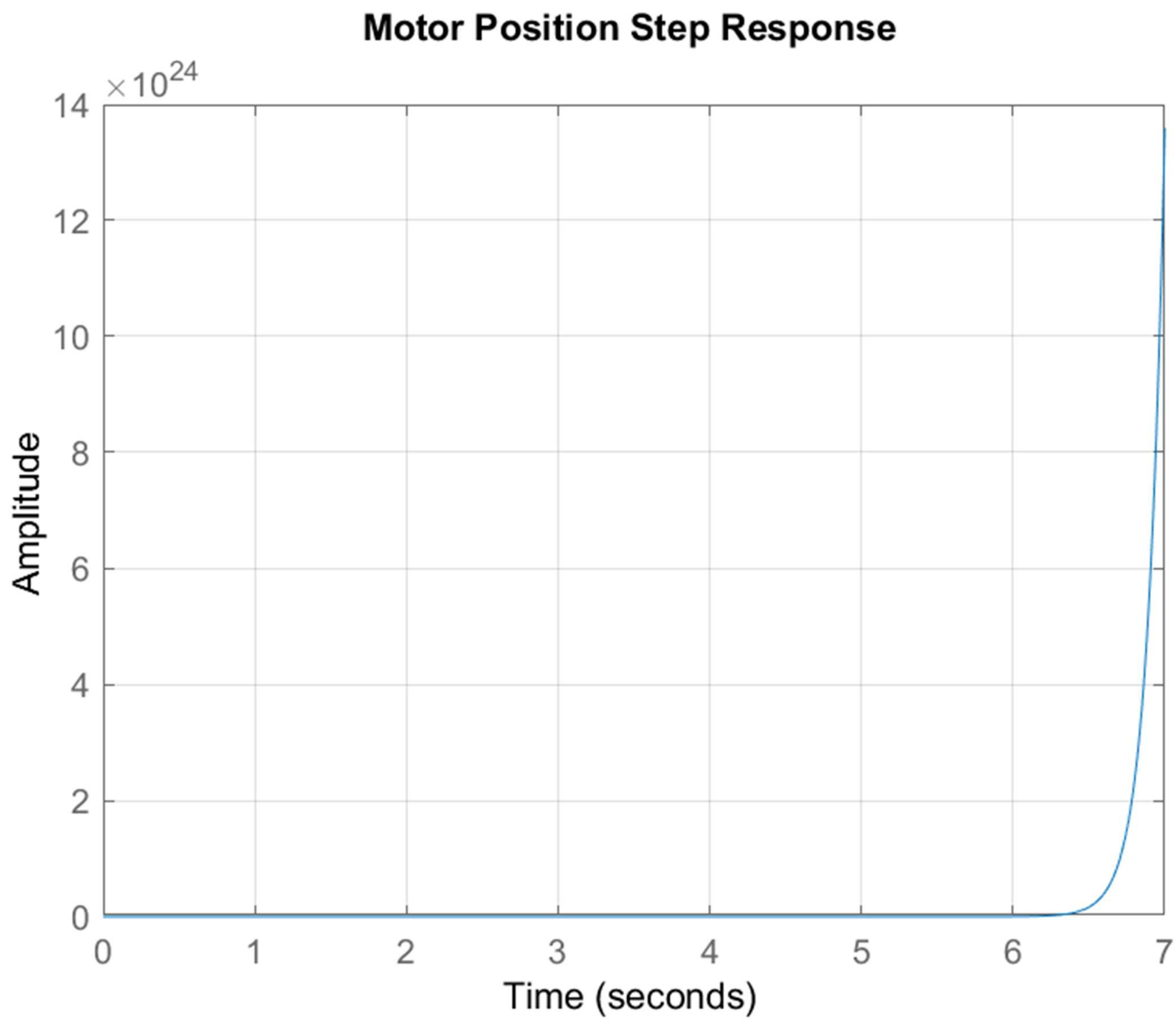
6.3 CODE

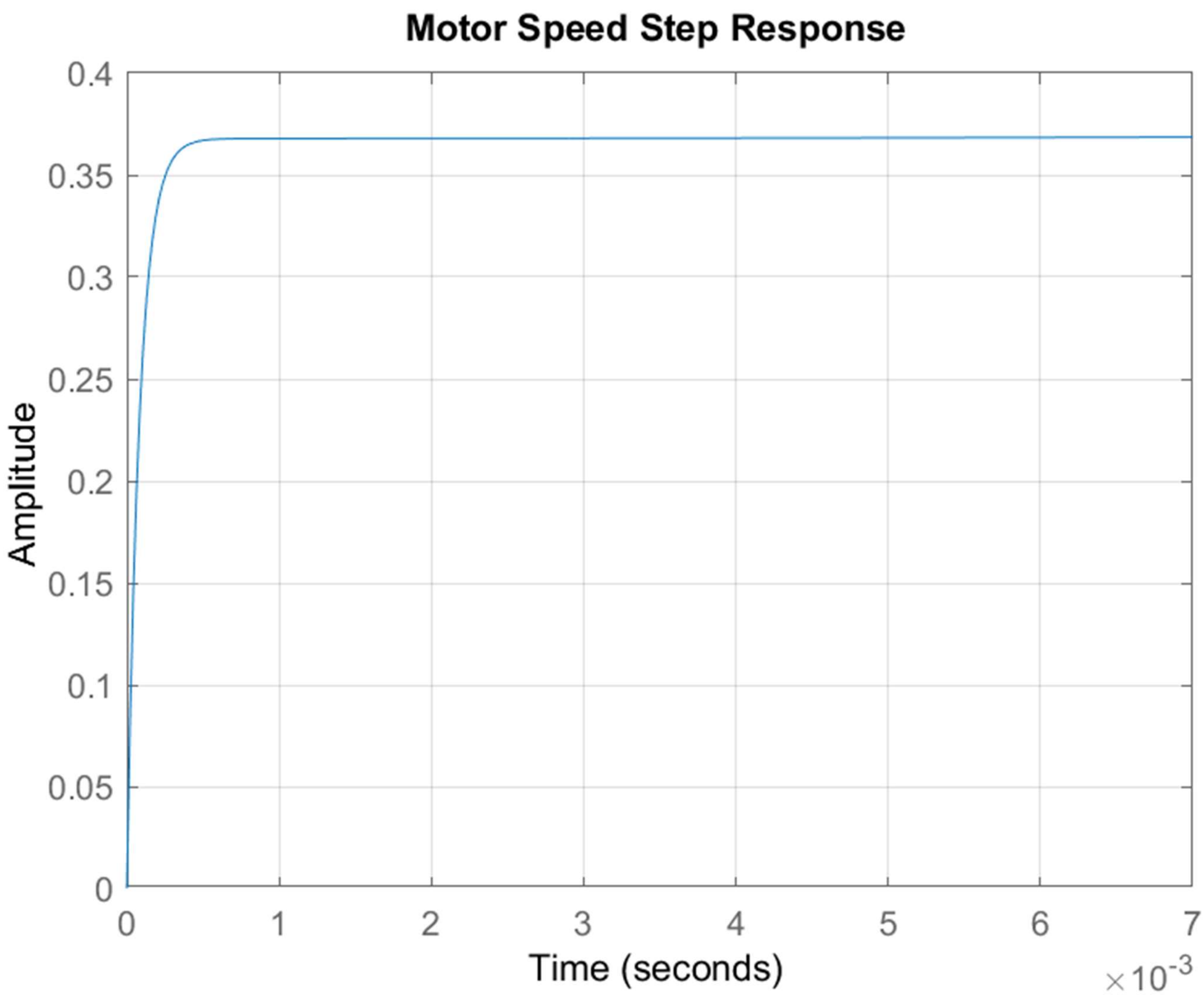
```
speed_ss = ss(speed_tf);
position_ss = ss(position_tf);
arm_ss = ss(arm_tf);
```

```
figure
step(speed_tf)
grid
title('Motor Speed Step Response')

figure
step(position_tf)
grid
title('Motor Position Step Response')

figure
step(arm_tf)
grid
title('Pendulum Arm Angle Step Response')
```

6.4 Screenshot:




Motor Position Step Response
×10^24
14
12
10
8
6
4
2
0
Amplitude
0
1
2
3
4
5
6
7
Time (seconds)

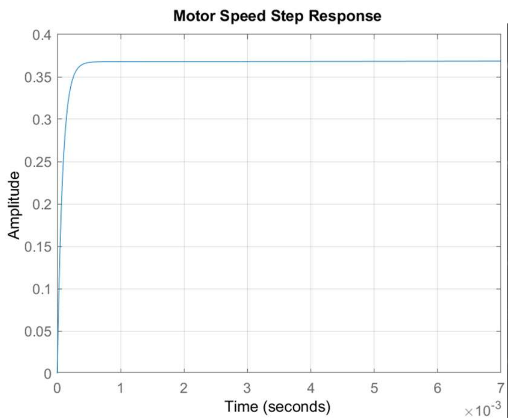


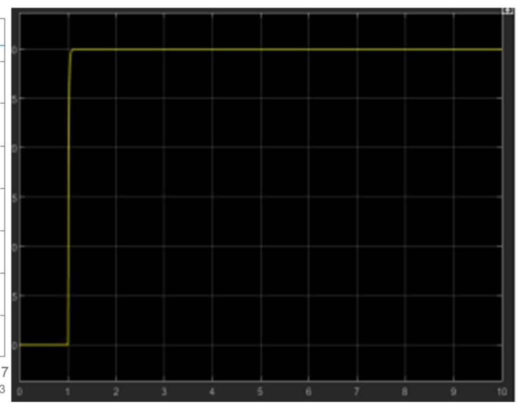
6.5 Comments

The motor increases its speed as the input step voltage is applied, eventually reaching a stable state, which is typical behavior for a motor. The motor's position response exhibits a phenomenon known as "ringing," where it initially surpasses the target value before eventually settling down. The pendulum arm angle displays a sinusoidal response, which is caused by the gravitational force acting on the pendulum. This force causes the pendulum to accelerate towards its equilibrium position, overshoot it, and then oscillate around it due to momentum

7 COMPARISON

Compare the unit step response for motor speed obtained from MATLAB, LabVIEW, and Simulink. Are they similar? Are they expected to be similar?

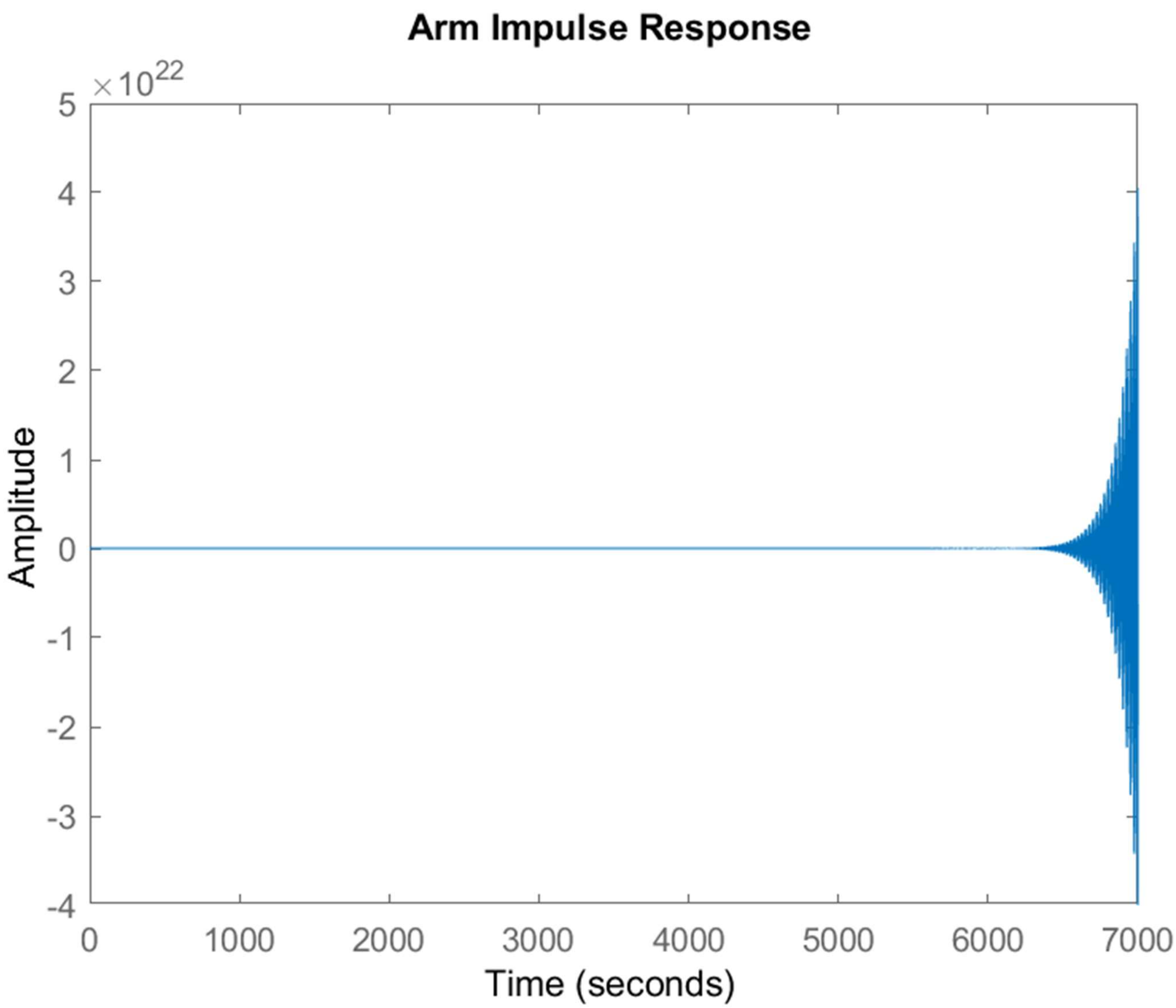


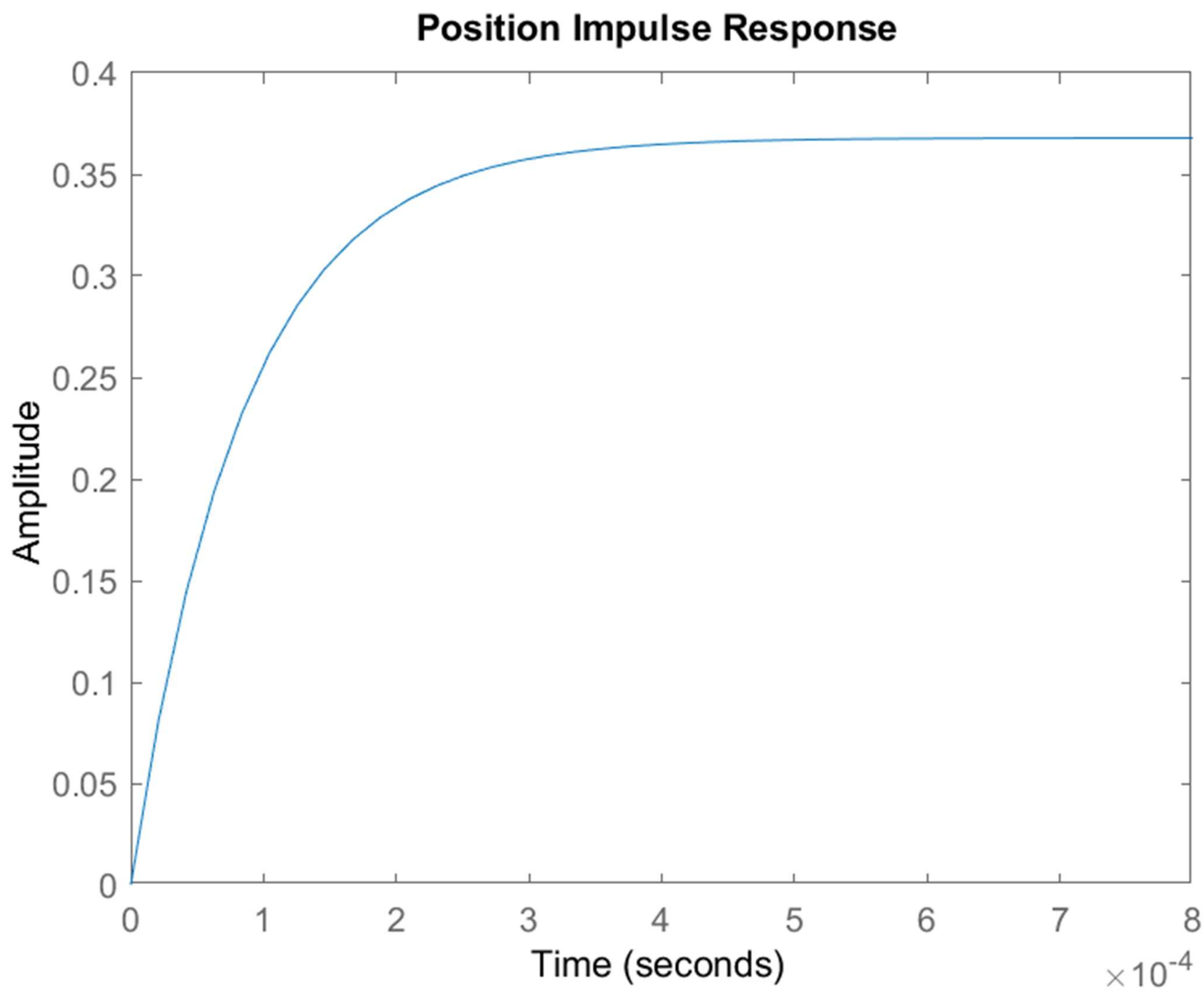


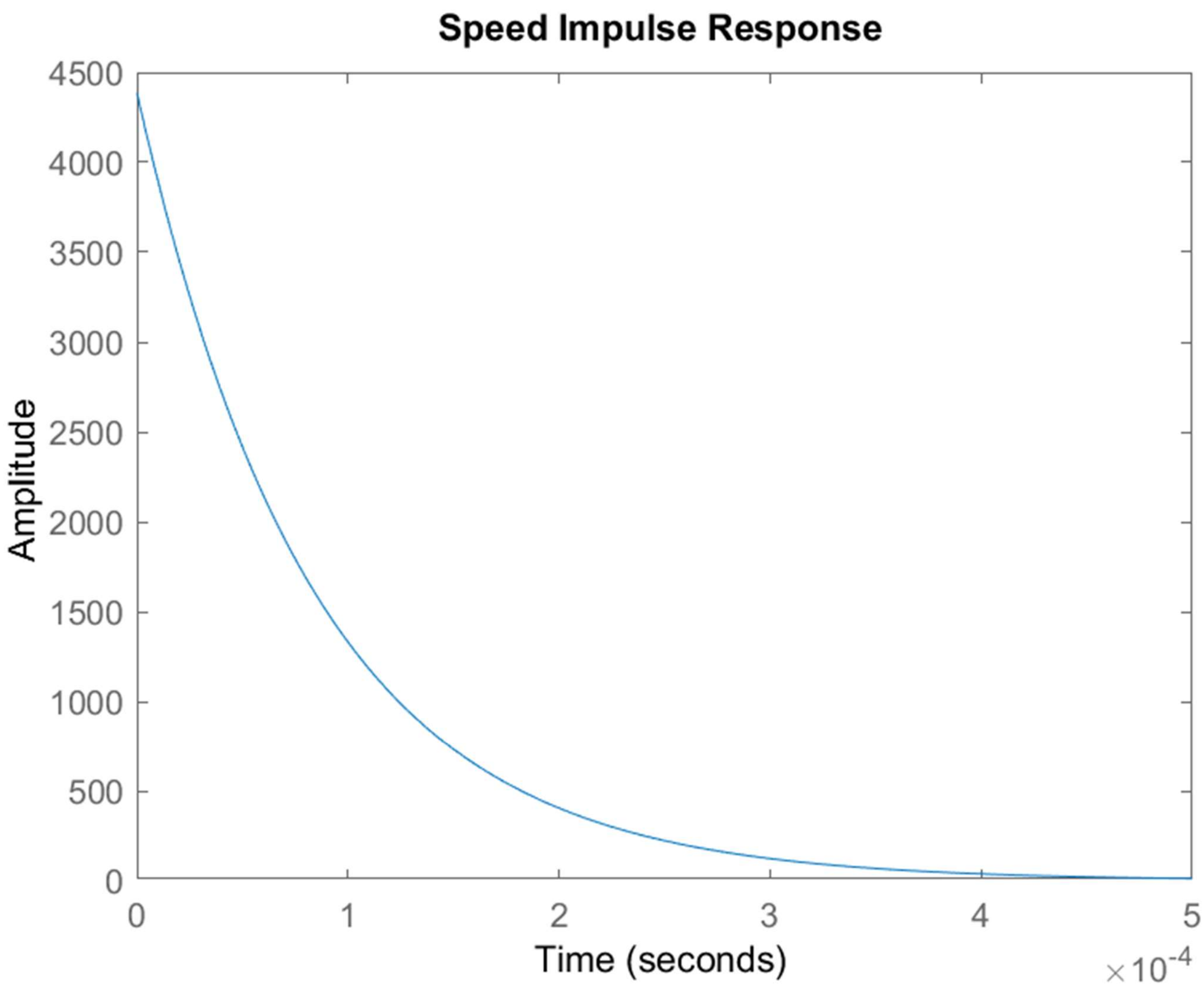
Although we can create our own custom input in both Simulink and LabVIEW, the MATLAB function step() adjusts to the design being tested and internally maintains a variable step time and steady state. This can cause changes in the settling time and response amplitude. Despite this, the response obtained from Simulink and LabVIEW is comparable.

8 IMPULSE RESPONSE IN MATLAB

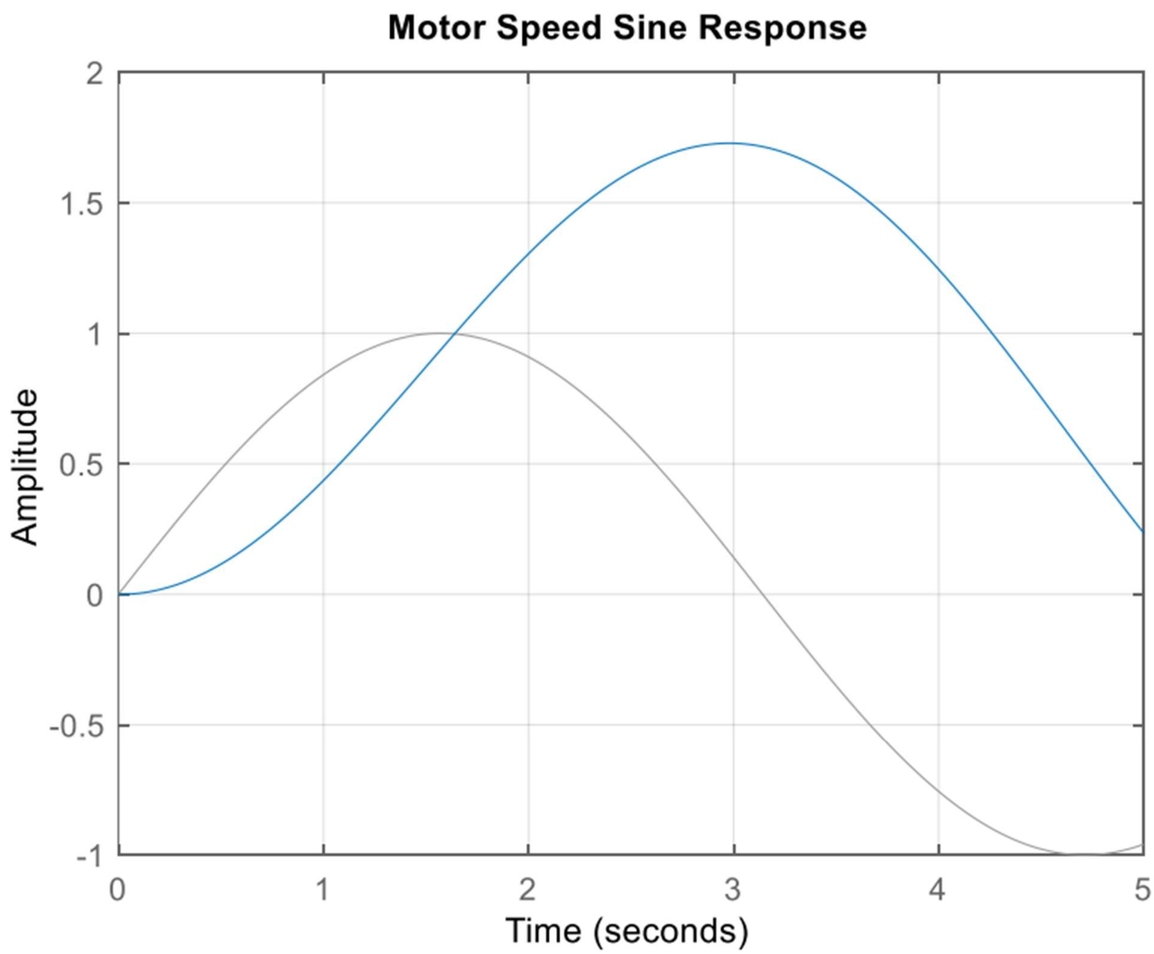
Impulse response are given as:

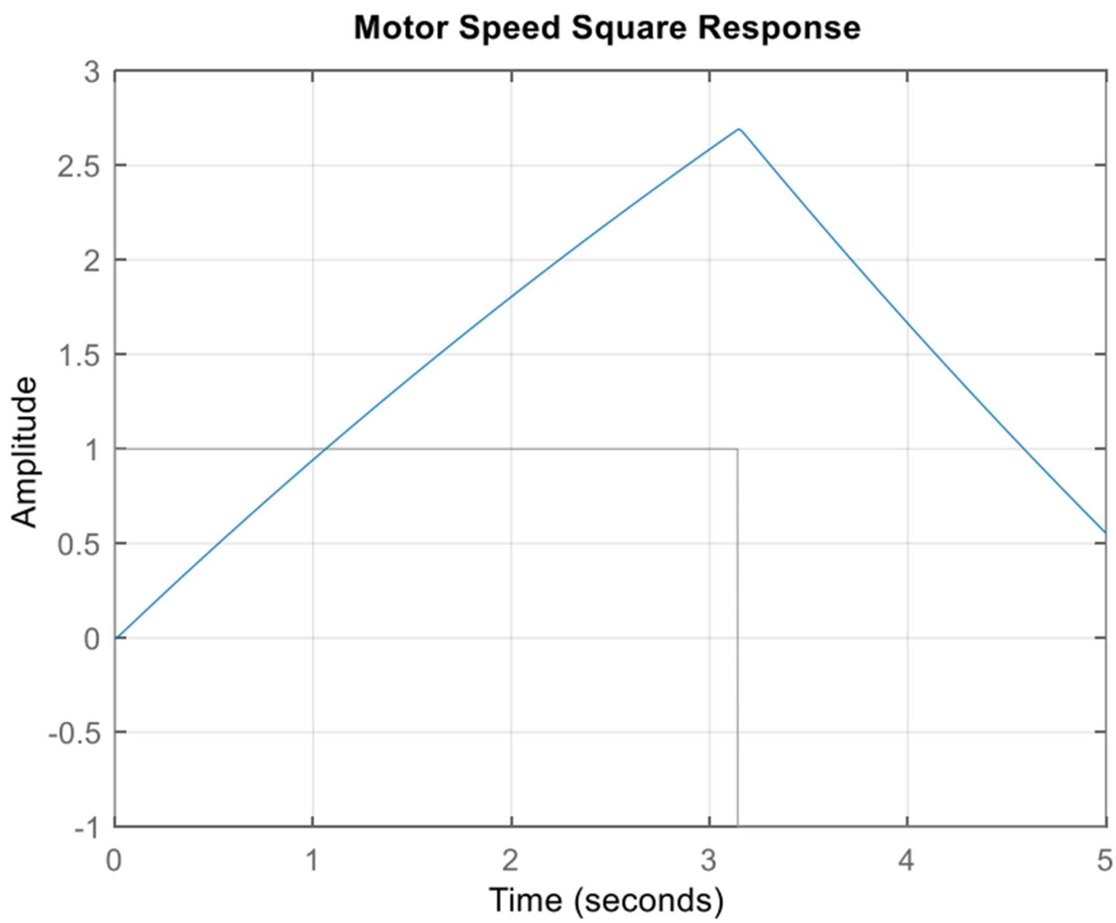

Arm Impulse Response
$\times 10^{22}$
5
4
3
2
1
0
-1
-2
-3
-4
Amplitude
0
1000
2000
3000
4000
5000
6000
7000
Time (seconds)

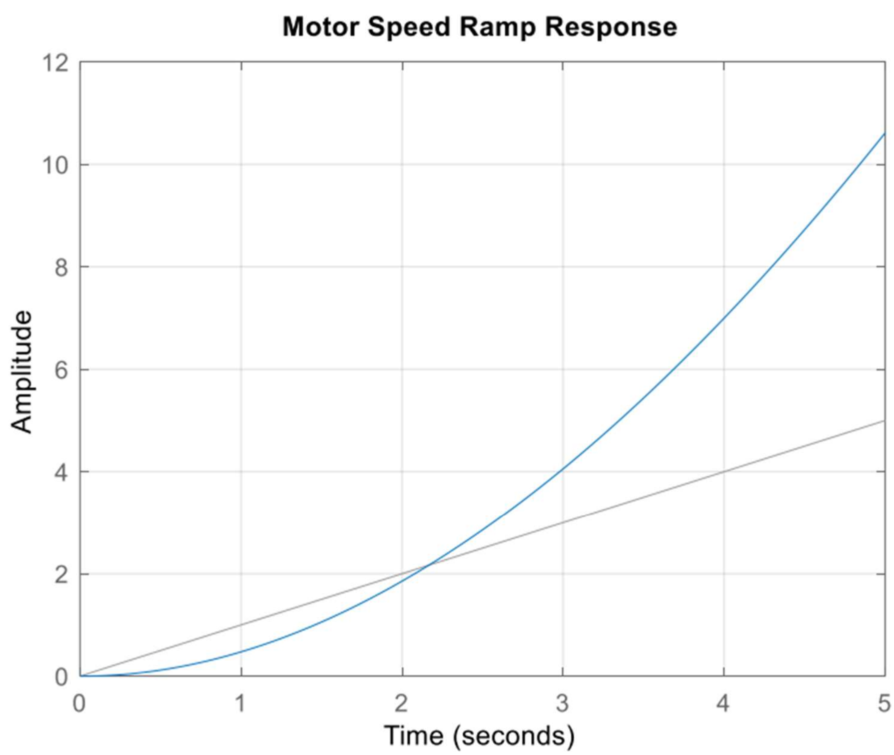

Position Impulse Response
Amplitude
Time (seconds)
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
1
2
3
4
5
6
7
8
×10^-4



9 Response of Arbitrary Inputs


Motor Speed Sine Response
Amplitude
Time (seconds)
2
1.5
1
0.5
0
-0.5
-1
0
1
2
3
4
5


Motor Speed Square Response
Amplitude
Time (seconds)
3
2.5
2
1.5
1
0.5
0
-0.5
-1
0
1
2
3
4
5



10 TIME RESPONSE USING INVERSE LAPLACE:

10.1 CODE:

```
syms s;
speed_tf_syms=
step_input = 1 / s;
impulse_input = 1;
ramp_input = 1 / s ^ 2;

ilaplace(step_input * speed
ilaplace(impulse_input * speed)
ilaplace(ramp_input * speed)
```

10.2 Output:

```
>> step_input =
(10 * exp(-100 * t)) / 999 - (10000 * exp(-t / 10)) / 999 + 10
impulse_input =
(1000 * exp(-t / 10)) / 999 - (1000 * exp(-100 * t)) / 999
ramp_input =
10 * t + (100000 * exp(-t / 10)) / 999 - exp(-100 * t) / 9990 -1001/10
```

11 Inter Connection of MATLAB systems

11.1 Code

```
K = 5;
sys = feedback(speed_tf, K);
display(sys)
```

11.2 Output:

```
sys =

                    0.0003283 s^3 - 6.762e-05 s
  ---------------------------------------------------------
  7.483e-08 s^4 + 0.002534 s^3 - 9.344e-06 s^2 - 0.06973 s


Continuous-time transfer function.
```

12 CONCLUSION:

To sum up, this lab offered a comprehensive education on state space modeling and analysis of systems using MATLAB, Simulink, and LabVIEW. The primary focus of the lab was on developing state space models, conducting simulations, and assessing the response of various inputs. Moreover, the lab explored the interconnection of systems within MATLAB. During the lab, we acquired knowledge of how to create state space models utilizing MATLAB, Simulink, and LabVIEW. We also gained a thorough understanding of the process involved in determining the time response of a system for different inputs. By examining the response of the system, we were able to assess its stability and performance.