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

Department of Electrical Engineering and Computer Science

Faculty Member: Ms. Neelma Naz Dated: 04/02/2023

Semester: 6th Section: BEE 12C

EE-371: Linear Control Systems

Lab 7: Performance of Systems

Lab Instructor: Yasir Rizwan

Group Members

Name	Reg. No	Lab Report Marks	Viva Marks	Total
		10 Marks	5 Marks	15 Marks
Danial Ahmad	331388			
Muhammad Umer	345834			
Tariq Umar	334943			

1 Table of Contents

2 Model Verification

2.1 Objectives

The objectives of this lab are:

- Learn how to compute the transient and steady state characteristics of a system in MATLAB.

2.2 Introduction

The purpose of this lab report is to learn how to compute the transient and steady state characteristics of a system in MATLAB. Transient and steady state characteristics are important for analyzing the performance and stability of a system under different inputs and conditions. We will then calculate and plot the transient and steady state characteristics such as rise time, settling time, overshoot, peak time, steady state error, etc.

MATLAB is a powerful tool that can help us simulate and visualize the system's response to various inputs and parameters. In this report, we will use MATLAB to model a second-order system with different damping ratios and natural frequencies, and then apply step inputs to observe the transient and steady state behavior of the system.

2.3 Software

MATLAB is a high-level programming language and numerical computing environment. Developed by MathWorks, it provides an interactive environment for numerical computation, visualization, and programming. MATLAB is widely used in various fields, including engineering, science, and finance, due to its capabilities for matrix and vector operations, implementation of algorithms, and creation of graphical representations of data.

3 Lab Procedure

3.1 Exercise 1

Find the rise time, peak time, peak value, overshoot, settling time and the steady state error for step input of the following systems.

$$\frac{2s+2}{s^2+9s+20}, \qquad \frac{s+1}{s^3+12s^2+47s+60}, \qquad \frac{1}{s+10}$$

```
tf_a = tf([2 2], [1 9 20]);
tf_b = tf([1 1], [1 2 47 60]);
tf_c = tf(1, [1 10]);

step_a = stepinfo(tf_a) %#ok<*NOPTS>
step_b = stepinfo(tf_b)
step_c = stepinfo(tf_c)

                                              Output
step_a =
         RiseTime: 0.0510
    TransientTime: 1.5940
     SettlingTime: 1.5940
      SettlingMin: 0.0917
      SettlingMax: 0.1949
        Overshoot: 94.9219
       Undershoot: 0
             Peak: 0.1949
         PeakTime: 0.2878


step_b =
         RiseTime: 0.1360
    TransientTime: 11.1598
     SettlingTime: 11.6485
      SettlingMin: 0.0025
      SettlingMax: 0.0378
        Overshoot: 126.5680
       Undershoot: 0
             Peak: 0.0378
         PeakTime: 0.4627


step_c =
         RiseTime: 0.2197
    TransientTime: 0.3912
     SettlingTime: 0.3912
      SettlingMin: 0.0905
      SettlingMax: 0.1000
        Overshoot: 0
       Undershoot: 0
             Peak: 0.1000
         PeakTime: 1.0546
```

3.2 Exercise 2

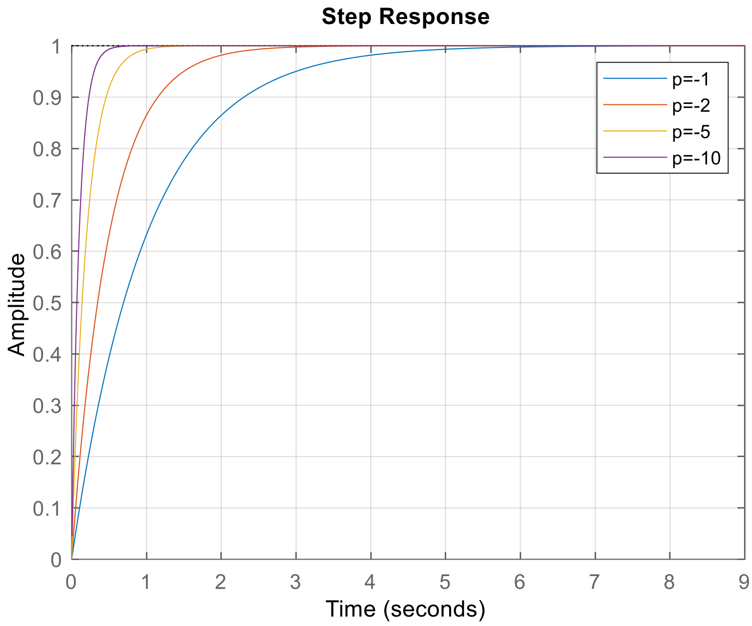
Consider the systems of the following form:

$$\frac{p}{s-p}$$

This system has a pole at p, it has no zeros, and the gain is equal to the negative of the pole i.e., $-p$. Using MATLAB, plot the step response of systems of this form for $p = -1, -2, -5, -10$. Plot all the step responses on a single figure. For each system also find the values of the various performance characteristics (rise time, overshoot, steady state error, etc.). Comment on how the pole of a first order system affects the step response of the system.

```
sys_a = zpk([], -1, 1);
step(ss(sys_a));
sys_b = zpk([], -2, 2);
hold on
step(ss(sys_b));
sys_c = zpk([], -5, 5);
step(ss(sys_c));
sys_d = zpk([], -10, 10);
step(ss(sys_d));
legend('p=-1', 'p=-2', 'p=-5', 'p=-10');
grid
                              Output
step_a =
         RiseTime: 2.1970
    TransientTime: 3.9121
     SettlingTime: 3.9121
      SettlingMin: 0.9045
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 10.5458

step_b =
         RiseTime: 1.0985
    TransientTime: 1.9560
     SettlingTime: 1.9560
      SettlingMin: 0.9045
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 5.2729

step_c =
         RiseTime: 0.4394
    TransientTime: 0.7824
     SettlingTime: 0.7824
      SettlingMin: 0.9045
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 2.1092
step_d =
         RiseTime: 0.2197
```

```
    TransientTime: 0.3912
     SettlingTime: 0.3912
      SettlingMin: 0.9045
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 1.0546
```



The addition of an extra pole decreases the speed of the system's response, with the decrease strength depending on how close the pole is to the jw axis.

3.3 Exercise 3

Now fix the pole to a constant value and let's see the effect of changing the gain 'k'.

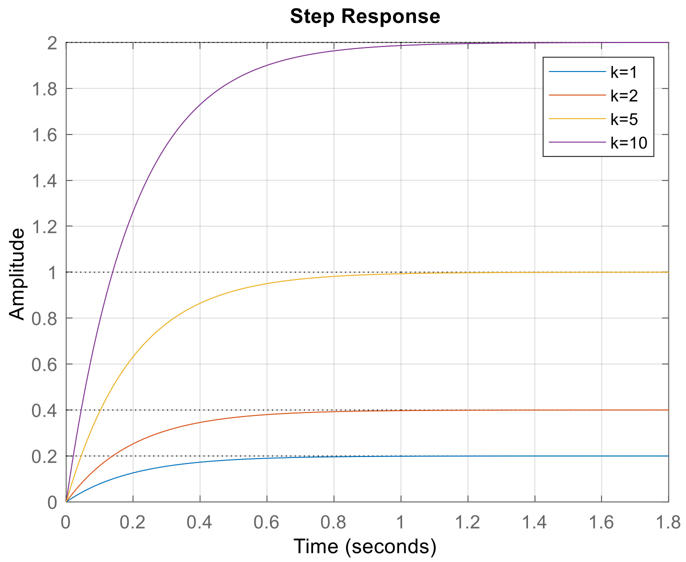
$$\frac{k}{s-p}$$

Using MATLAB, plot the step response of systems of this form for = -5 and $k = 1, 2, 5, 10$. Plot all the step responses on a single figure. For each system also find the values of the various performance characteristics. Comment on the effects of changing the gain.

```
sys_a = zpk([], -5, 1);
step(ss(sys_a));
sys_b = zpk([], -5, 2);
hold on
step(ss(sys_b));
sys_c = zpk([], -5, 5);
step(ss(sys_c));
sys_d = zpk([], -5, 10);
step(ss(sys_d));
legend('k=1', 'k=2', 'k=5', 'k=10');
grid
```

Output

```
step_a =
       RiseTime: 0.4394
    TransientTime: 0.7824
     SettlingTime: 0.7824
      SettlingMin: 0.1809
      SettlingMax: 0.2000
        Overshoot: 0
       Undershoot: 0
             Peak: 0.2000
         PeakTime: 2.1092

step_b =
       RiseTime: 0.4394
    TransientTime: 0.7824
     SettlingTime: 0.7824
      SettlingMin: 0.3618
      SettlingMax: 0.4000
        Overshoot: 0
       Undershoot: 0
             Peak: 0.4000
         PeakTime: 2.1092

step_c =
      RiseTime: 0.4394
    TransientTime: 0.7824
     SettlingTime: 0.7824
      SettlingMin: 0.9045
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 2.1092

step_c =
         RiseTime: 0.4394
    TransientTime: 0.7824
     SettlingTime: 0.7824
      SettlingMin: 1.8090
      SettlingMax: 1.9999
        Overshoot: 0
       Undershoot: 0
             Peak: 1.9999
         PeakTime: 2.1092
```



The change of gain does not contribute to the overall speed of the system's response, but rather, changes the final value at which the system settles.

3.4 Exercise 4

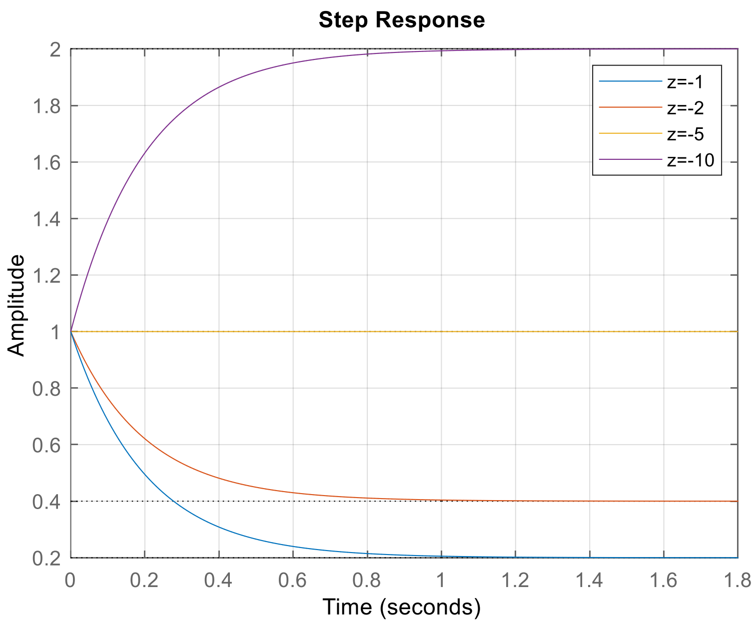
Now we will introduce a zero and see the effect of changing it. Consider the system:

$$\frac{k(s-z)}{s-p}$$

Using MATLAB, plot the step response of systems of this form for $p = -5$, $k = 1$ and $z = -1, -2, -5, -10$. Also have a plot for no zero. Plot all the step responses on a single figure. For each system also find the values of the various performance characteristic. Comment on the effects of changing the zeros.

```
sys_a = zpk(-1, -5, 1);
step(ss(sys_a));
sys_b = zpk(-2, -5, 1);
hold on
step(ss(sys_b));
sys_c = zpk(-5, -5, 1);
step(ss(sys_c));
sys_d = zpk(-10, -5, 1);
step(ss(sys_d));
legend('z=-1', 'z=-2', 'z=-5', 'z=-10');
grid

Output
step_a =
        RiseTime: 0
    TransientTime: 0.7824
     SettlingTime: 1.0597
      SettlingMin: 0.2000
      SettlingMax: 1
        Overshoot: 400.0000
```

```
       Undershoot: 0
             Peak: 1
         PeakTime: 0

step_b =
        RiseTime: 0
    TransientTime: 0.7824
     SettlingTime: 0.8635
      SettlingMin: 0.4000
      SettlingMax: 1
        Overshoot: 150
       Undershoot: 0
             Peak: 1
         PeakTime: 0

step_c =
        RiseTime: 0
    TransientTime: 0
     SettlingTime: 0
      SettlingMin: 1
      SettlingMax: 1
        Overshoot: 0
       Undershoot: 0
             Peak: 1
         PeakTime: 0

step_c =
        RiseTime: 0.3219
    TransientTime: 0.7824
     SettlingTime: 0.6438
      SettlingMin: 1.8005
      SettlingMax: 2.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 2.0000
         PeakTime: 2.1092
```



The addition of an extra zero increases the speed of the system's response, with the increase strength depending on how close the pole is to the *jw* axis.

3.5 Exercise 5

Use the formulas given above to find the values of the pole of a first order system that would give:

- rise times of 0.1, 0.5 and 1
- settling times of 1, 1.5 and 2

```
T_r = [0.1, 0.5, 1];
T_s = [1, 1.5 2];
p_a =- (2.2) ./ T_r;
p_b =- (4) ./ T_s;

                                          Output
T_r =
    0.1000    0.5000    1.0000

T_s =
    1.0000    1.5000    2.0000

p_a =
  -22.0000   -4.4000   -2.2000

p_b =
   -4.0000   -2.6667   -2.0000
```

3.6 Exercise 6

Find the damping ratio and the natural frequency of the following systems:

$$\frac{5}{s^2 - 4s + 5}, \quad \frac{2}{s^2 - 2s + 2}, \quad \frac{5}{s^2 - 2s + 5}$$

```
tf_a = tf(5, [1 -4 5]);
tf_b = tf(2, [1 -2 2]);
tf_c = tf(5, [1 -2 5]);

disp("damp_a: "); damp(tf_a)
disp(newline + "damp_b: "); damp(tf_b)
disp(newline + "damp_c: "); damp(tf_c)

                                          Output
damp_a:
         Pole               Damping       Frequency      Time Constant
                                         (rad/seconds)     (seconds)
  2.00e+00 + 1.00e+00i    -8.94e-01      2.24e+00        -5.00e-01
  2.00e+00 - 1.00e+00i    -8.94e-01      2.24e+00        -5.00e-01

damp_b:
         Pole               Damping       Frequency      Time Constant
                                         (rad/seconds)     (seconds)
  1.00e+00 + 1.00e+00i    -7.07e-01      1.41e+00        -1.00e+00
  1.00e+00 - 1.00e+00i    -7.07e-01      1.41e+00        -1.00e+00
```

```
damp_c:
         Pole                  Damping        Frequency       Time Constant
                                            (rad/seconds)       (seconds)
  1.00e+00 + 2.00e+00i     -4.47e-01        2.24e+00         -1.00e+00
  1.00e+00 - 2.00e+00i     -4.47e-01        2.24e+00         -1.00e+00
```

3.7 Exercise 7 & 8

Write a MATLAB function that takes the damping ratio and natural frequency as arguments and returns a transfer function of the form given in equation (2). Call this function my_second_order_tf.

```
function tf_ret = my_second_order_tf(zeta, w_n)
    tf_ret = tf(w_n ^ 2, [1, 2 * zeta * w_n, w_n ^ 2]);
end
```

3.8 Exercise 9

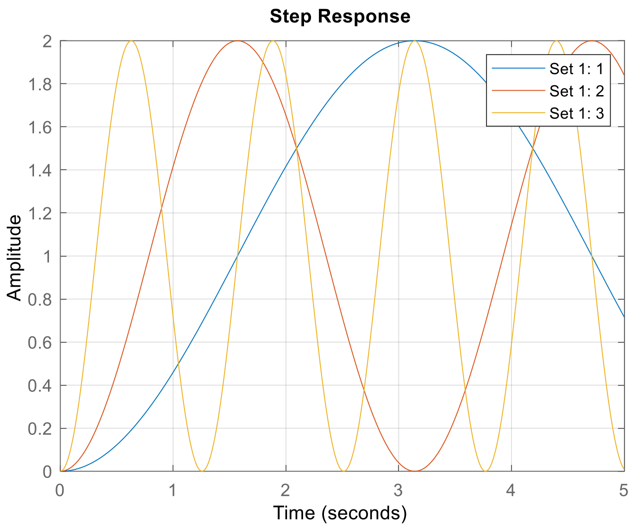
Using the function that you have just created, my_second_order_tf, make transfer functions for the following sets of damping ratios and natural frequencies:

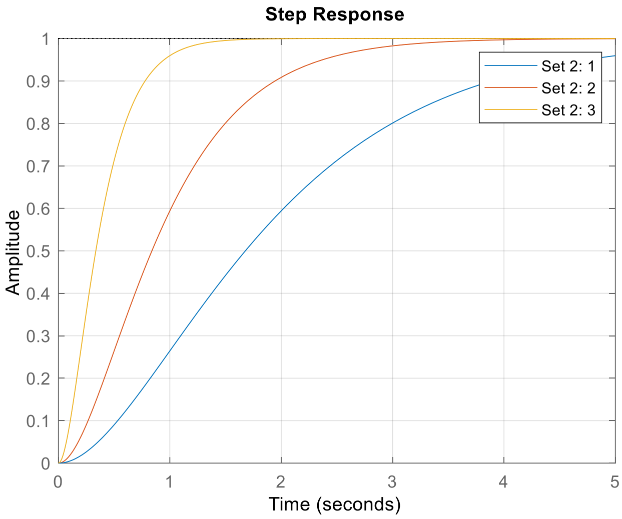
Set 1: $\zeta = 0,\ \omega_n = 1{,}2{,}5$ (See the note given below)
Set 2: $\zeta = 1,\ \omega_n = 1{,}2{,}5$
Set 3: $\zeta = 0{,}0.5{,}1{,}2,\ \omega_n = 1$

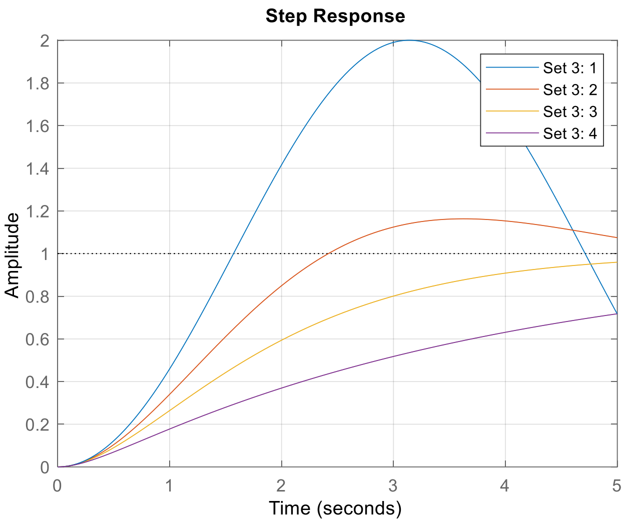
For each set of values plot the step responses on a single figure. For each system also find the values of the various performance characteristic. Comment on the effects of changing the natural frequency and the damping ratio. Classify each of the above systems as undamped, underdamped, critically damped or overdamped.

```
% Set 1
tf_a = my_second_order_tf(0, 1);
tf_b = my_second_order_tf(0, 2);
tf_c = my_second_order_tf(0, 5);
% Set 2
tf_d = my_second_order_tf(1, 1);
tf_e = my_second_order_tf(1, 2);
tf_f = my_second_order_tf(1, 5);
% Set 3
tf_g = my_second_order_tf(0, 1);
tf_h = my_second_order_tf(0.5, 1)
tf_i = my_second_order_tf(1, 1);
tf_j = my_second_order_tf(2, 1);

figure
t = 0:0.01:5;
hold on
step(tf_a, t); step(tf_b, t); step(tf_c, t);
legend('Set 1: 1', 'Set 1: 2', 'Set 1: 3')
grid
figure
hold on
step(tf_d, t); step(tf_e, t); step(tf_f, t);
legend('Set 2: 1', 'Set 2: 2', 'Set 2: 3')
grid
figure
```

```
hold on
step(tf_g, t); step(tf_h, t);
step(tf_i, t); step(tf_j, t);
legend('Set 3: 1', 'Set 3: 2', 'Set 3: 3', 'Set 3: 4')
grid
```







Set 1: Undamped Systems; The speed of the system's response increases directly proportional to the change in natural frequency.
Set 2: Critically Damped Systems; The speed of the system's response increases directly proportional to the change in natural frequency.
Set 3: As natural frequency remains constant, there is no change in system's response speed, however, damping ratios:

$$\zeta = \begin{cases} 0 \in \ Undamped \\ < 1 \in \ Underdamped \\ = 1 \in Crtically\ Damped \\ > 1 \in Overdamped \end{cases}$$

3.9 Exercise 10

Using the formulas given above, find the values of damping ratio and natural frequency that result in $\%OS = 10$ and $T_s = 1$.

```
syms T_s T_p p_OS zeta w_n;
eq1 = T_s == 4/(zeta*w_n);
eq2 = T_p == pi/(w_n*sqrt(1-zeta^2));
eq3 = p_OS == 100*exp(-(zeta*pi/sqrt(1-zeta^2)));
eq4 = zeta == -log(p_OS/100)/(sqrt(pi^2+(log(p_OS/100))^2));

eq4 = subs(eq4, p_OS, 10);
zeta_ans = double(solve(eq4, zeta));

eq1 = subs(eq1, {T_s, zeta}, [1, zeta_ans]);
w_n_ans = double(solve(eq1, w_n));

                                        Output

zeta_ans =
    0.5912

w_n_ans =
    6.7664
```

3.10 Exercise 11

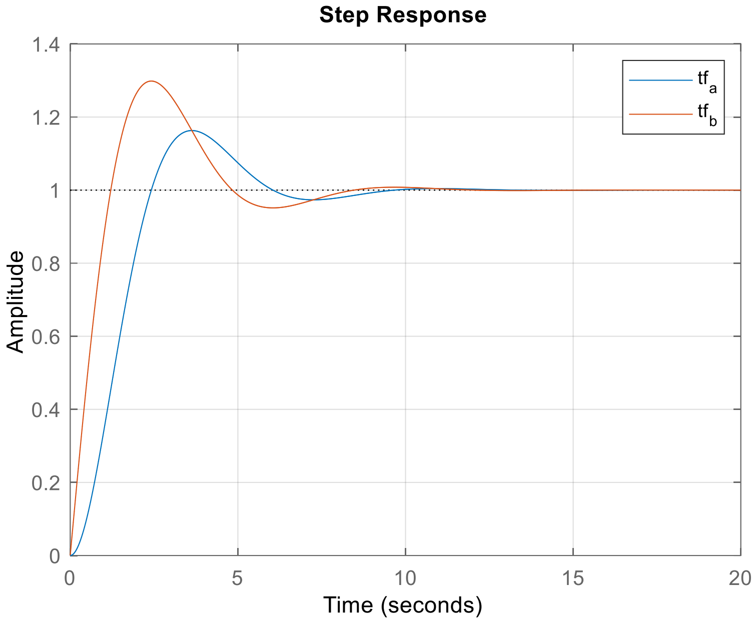
For the systems given below, find the natural frequency, the damping ratio, and transient characteristics. Also plot their step responses on a single graph:

$$\frac{1}{s^2+s+1}, \qquad \frac{s+1}{s^2+s+1}$$

Comment on your observations.

```
tf_a = tf(1, [1 1 1]);
tf_b = tf([1 1], [1 1 1]);

figure
t = 0:0.01:20;
hold on
step(tf_a, t)
step(tf_b, t)
grid
legend('tf_a', 'tf_b')
```

The addition of extra zero to the 2nd order system causes the speed of the step response to be faster than the original system.




```
damp_a:
         Pole                 Damping       Frequency       Time Constant
                                          (rad/seconds)       (seconds)

 -5.00e-01 + 8.66e-01i       5.00e-01       1.00e+00          2.00e+00
 -5.00e-01 - 8.66e-01i       5.00e-01       1.00e+00          2.00e+00

damp_b:
         Pole                 Damping       Frequency       Time Constant
                                          (rad/seconds)       (seconds)

 -5.00e-01 + 8.66e-01i       5.00e-01       1.00e+00          2.00e+00
 -5.00e-01 - 8.66e-01i       5.00e-01       1.00e+00          2.00e+00

step_a:
        RiseTime: 1.6390
    TransientTime: 8.0759
     SettlingTime: 8.0759
      SettlingMin: 0.9315
      SettlingMax: 1.1629
        Overshoot: 16.2929
       Undershoot: 0
             Peak: 1.1629
         PeakTime: 3.5920


step_b:
         RiseTime: 0.9409
    TransientTime: 7.5054
     SettlingTime: 7.5054
      SettlingMin: 0.9403
      SettlingMax: 1.2984
        Overshoot: 29.8352
       Undershoot: 0
             Peak: 1.2984
         PeakTime: 2.3947
```

3.11 Exercise 12

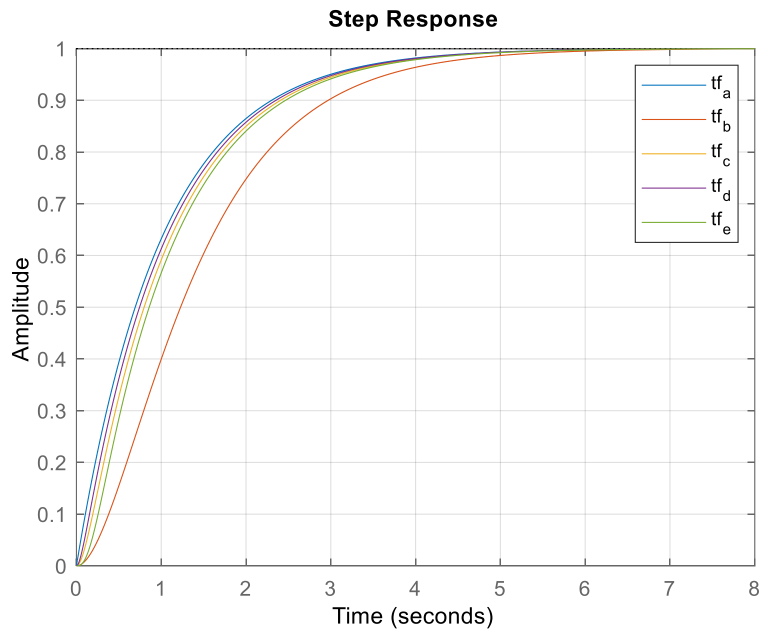
For the systems given below, find the natural frequency, the damping ratio, and transient characteristics. Also plot their step responses on a single graph.

$$\frac{1}{(s+1)}, \frac{2}{(s+1)(s+2)}, \frac{10}{(s+1)(s+10)}, \frac{20}{(s+1)(s+20)}, \frac{125}{(s+1)(s+10+5i)(s+10-5i)}$$

Comment on your observations.

```
tf_a = zpk([], -1, 1);
tf_b = zpk([], [-1 -2], 2);
tf_c = zpk([], [-1 -10], 10);
tf_d = zpk([], [-1 -20], 20);
tf_e = zpk([], [-1 -10-5*1i -10+5*1i], 125);

figure
t = 0:0.01:8;
hold on
step(tf_a, t)
step(tf_b, t)
step(tf_c, t)
step(tf_d, t)
step(tf_e, t)
grid
legend('tf_a', 'tf_b', 'tf_c', 'tf_d', 'tf_e')
```

The addition of extra pole to the 1st order system causes the speed of the step response to be slower than the original system.



```
damp_a:
    Pole         Damping        Frequency      Time Constant
                              (rad/seconds)      (seconds)
 -1.00e+00      1.00e+00        1.00e+00         1.00e+00

damp_b:
    Pole         Damping        Frequency      Time Constant
                              (rad/seconds)      (seconds)
 -1.00e+00      1.00e+00        1.00e+00         1.00e+00
 -2.00e+00      1.00e+00        2.00e+00         5.00e-01

damp_c:
    Pole         Damping        Frequency      Time Constant
                              (rad/seconds)      (seconds)
 -1.00e+00      1.00e+00        1.00e+00         1.00e+00
 -1.00e+01      1.00e+00        1.00e+01         1.00e-01

damp_d:
    Pole         Damping        Frequency      Time Constant
                              (rad/seconds)      (seconds)
 -1.00e+00      1.00e+00        1.00e+00         1.00e+00
 -2.00e+01      1.00e+00        2.00e+01         5.00e-02

damp_e:
          Pole                 Damping        Frequency      Time Constant
                                            (rad/seconds)      (seconds)
 -1.00e+00                    1.00e+00        1.00e+00         1.00e+00
 -1.00e+01 - 5.00e+00i        8.94e-01        1.12e+01         1.00e-01
 -1.00e+01 + 5.00e+00i        8.94e-01        1.12e+01         1.00e-01

step_a:
        RiseTime: 2.1970
    TransientTime: 3.9121
     SettlingTime: 3.9121
      SettlingMin: 0.9045
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 10.5458

step_b:
         RiseTime: 2.5901
    TransientTime: 4.6002
     SettlingTime: 4.6002
      SettlingMin: 0.9023
      SettlingMax: 0.9992
        Overshoot: 0
       Undershoot: 0
             Peak: 0.9992
         PeakTime: 7.7827

step_c:
         RiseTime: 2.2150
    TransientTime: 4.0174
     SettlingTime: 4.0174
      SettlingMin: 0.9005
      SettlingMax: 0.9993
        Overshoot: 0
       Undershoot: 0
             Peak: 0.9993
         PeakTime: 7.3591
```

```
step_d:
         RiseTime: 2.2000
    TransientTime: 3.9634
     SettlingTime: 3.9634
      SettlingMin: 0.9040
      SettlingMax: 0.9993
        Overshoot: 0
       Undershoot: 0
             Peak: 0.9993
         PeakTime: 7.3222

step_e:
         RiseTime: 2.2117
    TransientTime: 4.0769
     SettlingTime: 4.0769
      SettlingMin: 0.9001
      SettlingMax: 0.9994
        Overshoot: 0
       Undershoot: 0
             Peak: 0.9994
         PeakTime: 7.5433
```

3.12 Exercise 13

For the systems given below, find the natural frequency, the damping ratio, and transient characteristics. Also plot their step responses on a single graph:

$$\frac{5}{(s+1+2i)(s+1-2i)}$$

$$\frac{5}{(s+1)(s+1+2i)(s+1-2i)}$$

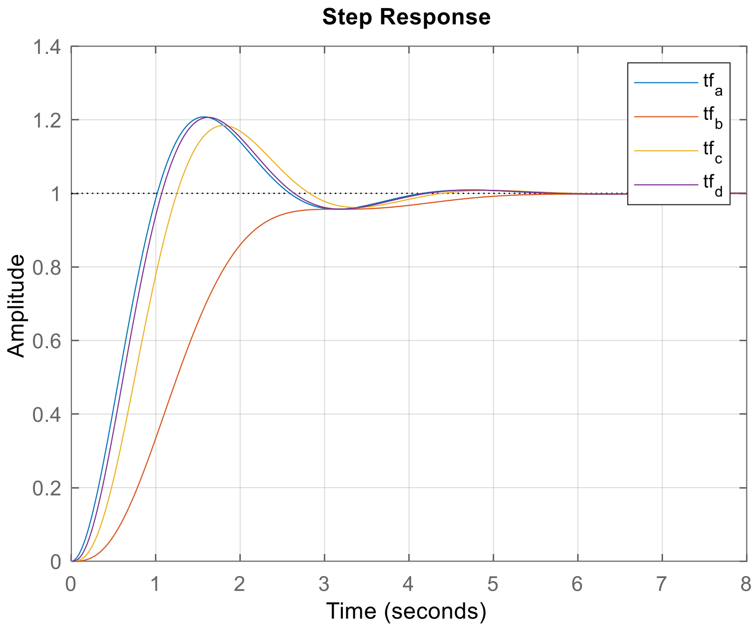
$$\frac{25}{(s+5)(s+1+2i)(s+1-2i)}$$

$$\frac{100}{(s+20)(s+1+2i)(s+1-2i)}$$

Comment on your observations.

```
tf_a = zpk([], [-1-2*1i -1+2*1i], 5);
tf_b = zpk([], [-1 -1-2*1i -1+2*1i], 5);
tf_c = zpk([], [-5 -1-2*1i -1+2*1i], 25);
tf_d = zpk([], [-20 -1-2*1i -1+2*1i], 100);

figure
t = 0:0.01:8;
hold on
step(tf_a, t)
step(tf_b, t)
step(tf_c, t)
step(tf_d, t)
grid
legend('tf_a', 'tf_b', 'tf_c', 'tf_d')
```

The addition of extra pole to the 2nd order system causes the speed of the step response to be slower than the original system.




```
damp_a:
        Pole                 Damping       Frequency      Time Constant
                                          (rad/seconds)    (seconds)
 -1.00e+00 - 2.00e+00i      4.47e-01       2.24e+00        1.00e+00
 -1.00e+00 + 2.00e+00i      4.47e-01       2.24e+00        1.00e+00

damp_b:
        Pole                 Damping       Frequency      Time Constant
                                          (rad/seconds)    (seconds)
 -1.00e+00                  1.00e+00       1.00e+00        1.00e+00
 -1.00e+00 - 2.00e+00i      4.47e-01       2.24e+00        1.00e+00
 -1.00e+00 + 2.00e+00i      4.47e-01       2.24e+00        1.00e+00

damp_c:
        Pole                 Damping       Frequency      Time Constant
                                          (rad/seconds)    (seconds)
 -1.00e+00 - 2.00e+00i      4.47e-01       2.24e+00        1.00e+00
 -1.00e+00 + 2.00e+00i      4.47e-01       2.24e+00        1.00e+00
 -5.00e+00                  1.00e+00       5.00e+00        2.00e-01

damp_d:
        Pole                 Damping       Frequency      Time Constant
                                          (rad/seconds)    (seconds)
 -1.00e+00 - 2.00e+00i      4.47e-01       2.24e+00        1.00e+00
 -1.00e+00 + 2.00e+00i      4.47e-01       2.24e+00        1.00e+00
 -2.00e+01                  1.00e+00       2.00e+01        5.00e-02

step_a:
        RiseTime: 0.6903
   TransientTime: 3.7352
    SettlingTime: 3.7352
     SettlingMin: 0.9149
     SettlingMax: 1.2079
```

```
        Overshoot: 20.7866
       Undershoot: 0
             Peak: 1.2079
         PeakTime: 1.5658

step_b:
         RiseTime: 1.5889
    TransientTime: 4.4520
     SettlingTime: 4.4520
      SettlingMin: 0.9076
      SettlingMax: 0.9993
        Overshoot: 0
       Undershoot: 0
             Peak: 0.9993
         PeakTime: 7.7827

step_c:
         RiseTime: 0.7733
    TransientTime: 3.9210
     SettlingTime: 3.9210
      SettlingMin: 0.9145
      SettlingMax: 1.1843
        Overshoot: 18.4293
       Undershoot: 0
             Peak: 1.1843
         PeakTime: 1.8052

step_d:
         RiseTime: 0.6967
    TransientTime: 3.7853
     SettlingTime: 3.7853
      SettlingMin: 0.9094
      SettlingMax: 1.2064
        Overshoot: 20.6433
       Undershoot: 0
             Peak: 1.2064
         PeakTime: 1.6118
```

4 Conclusion

The purpose of this lab report was to learn how to compute the transient and steady state characteristics of a system in MATLAB. Transient and steady state characteristics are important for analyzing the performance and stability of a system under different inputs and conditions. MATLAB is a powerful tool for simulating and visualizing the behavior of systems using numerical methods and graphical tools. In this report, we used MATLAB to model a first-order system and a second-order system with different parameters and inputs. We then calculated and plotted the transient and steady state characteristics such as rise time, settling time, overshoot, peak time, steady state error, etc. We also compared the results with the theoretical values obtained from analytical solutions.