IIT BOMBAY

EE324: Control Systems Laboratory

Lab 2

Name: Prakhar Diwan
Roll No.: 180100083

January 23, 2021

Contents

1 Question 1

For my roll number and name, the following is the value of the constants:-

$$a = 83 \text{ (Roll no. - 180100083)}$$
$$b = 16 \text{ (First letter of first name - 'P')}$$

Hence the transfer function is:

$$G(s) = \frac{83}{s + 16}$$

1.1 Part a

The Scilab code used for building the specified LTI system is shown below:-

```
a = 83;
b = 16;
s = poly(0,'s'); // defining the variable s
G = a/(s+b); // defining the transfer function
sys = syslin('c', G); // building linear continuous time system of tranfer function G(s)
```

1.2 Part b

As G(s) is a 1^{st} order system and from it's study we know that,

$$\tau = \frac{1}{b} \text{ (Time constant)}$$
$$t_s = \frac{ln(50)}{b} \text{ (2\% settling time)}$$
$$t_{r,start} = \frac{ln(10/9)}{b} \text{ (start of rise time)}$$
$$t_{r,end} = \frac{ln(10)}{b} \text{ (end of rise time)}$$
$$T_{rise} = t_{r,end} - t_{r,start}$$

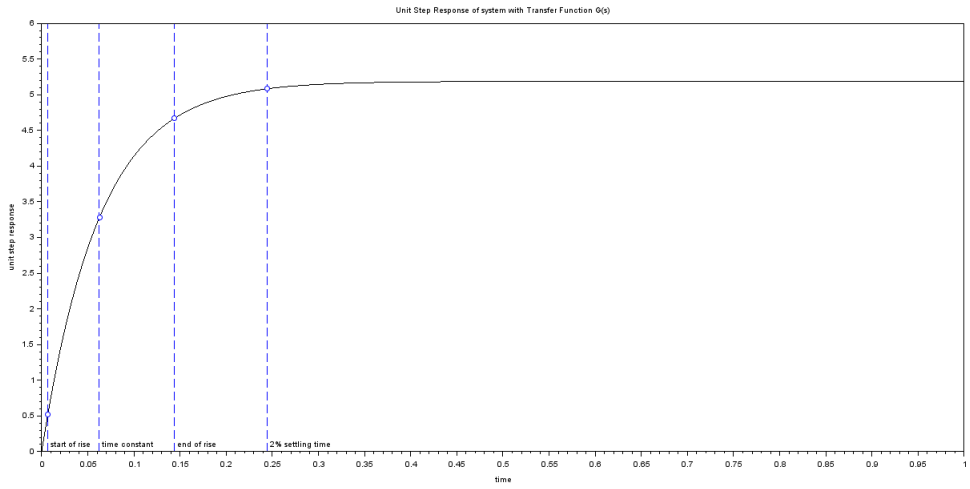

Figure 1: The important points have been marked here. Settling time indicates the '2%' settling time. Rise time start indicates the time when the response reaches '10%' of its steady state value. Rise time end indicates the time when the response reaches '90%' of its steady state value.

Substituting the value of 'b' in the above equations, we have the following results:

$$\begin{aligned}
\tau &= \frac{1}{16} = 0.0625s \\
t_s &= \frac{ln(50)}{16} = 0.245s \\
t_{r,start} &= \frac{ln(10/9)}{16} = 0.00659s \\
t_{r,end} &= \frac{ln(10)}{16} = 0.144s \\
\therefore T_{rise} &= 0.144 - 0.00659 = 0.137s \text{ (rise time)}
\end{aligned}$$

The scilab code has been given in the appendix under section Question 1 and Part b.

1.3 Part c

The rise time is given by the following formula:

$$\begin{aligned}
T_{rise} &= t_{r,end} - t_{r,start} \\
&= \frac{ln(10)}{b} - \frac{ln(10/9)}{b} \\
&= \frac{ln(9)}{b}
\end{aligned}$$

Hence it is independent of 'a'. So we should obtain a straight line parallel to the X-axis.

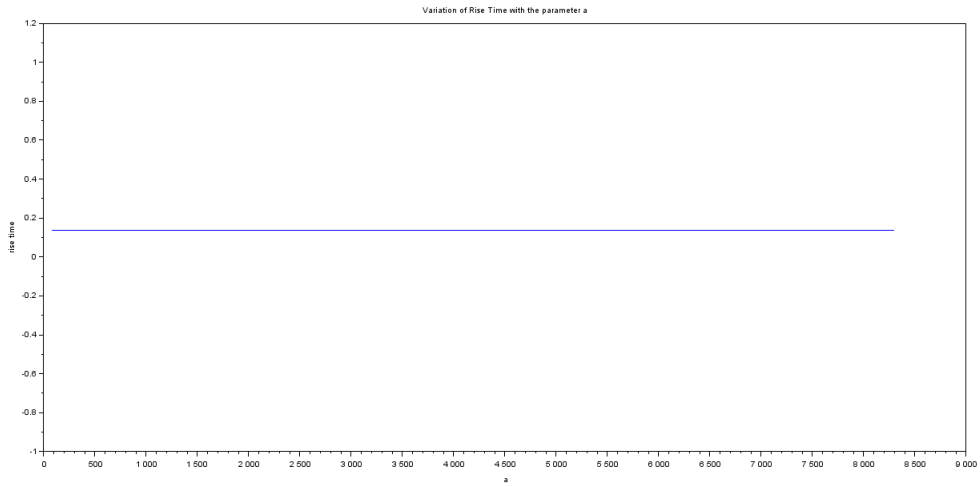

Figure 2: The variation is as expected

The scilab code has been given in the appendix under section Question 1 and Part c.

1.4 Part d

The rise time is given by the following formula:

$$\begin{aligned} T_{rise} &= t_{r,end} - t_{r,start} \\ &= \frac{ln(10)}{b} - \frac{ln(10/9)}{b} \\ &= \frac{ln(9)}{b} \end{aligned}$$

Hence it is inversely proportional to 'b'. So we should obtain a hyperbolic curve.

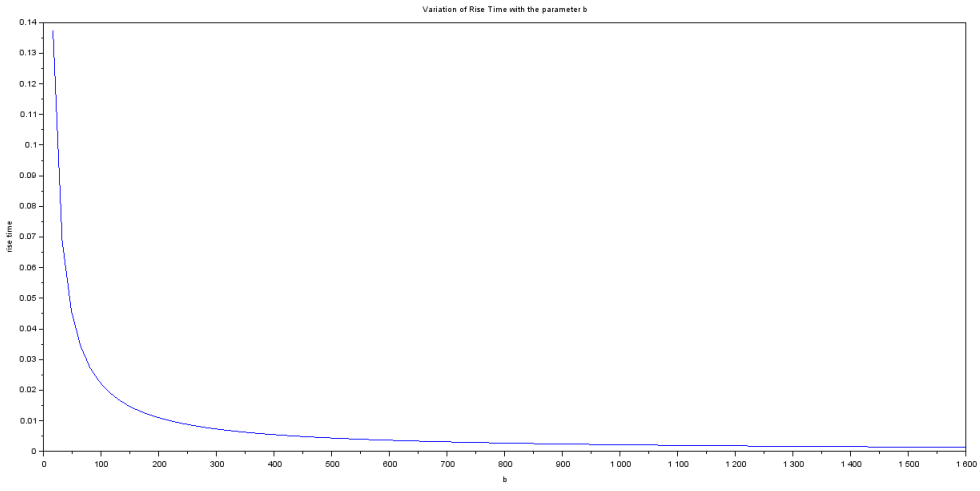

Figure 3: The curve is hyperbolic as expected

The scilab code has been given in the appendix under section Question 1 and Part d.

2 Question 2

If we consider the normalised transfer function of a second order system, $G(s) = \frac{w_n^2}{s^2+2\zeta s w_n+w_n^2}$, where ζ is called the 'damping ratio', then we know that the system will be called underdamped, if $0 < \zeta < 1$. So in this example, I have taken the value of $\zeta = 0.5$, which satisfies the condition. I have also fixed the value of natural frequency, $w_n = 1$.
Hence, the transfer function which I have considered is,

$$G(s) = \frac{1}{s^2 + s + 1}$$

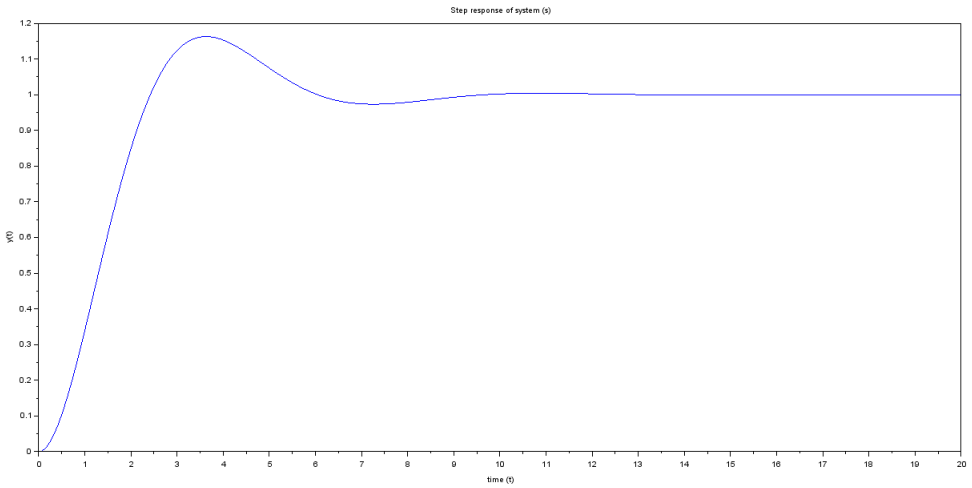

Figure 4: Plot for the unit step response of our second order underdamped system

We can observe the kink near the origin, as well as the oscillations in the output after which it attains a steady state. The code used has been mentioned in Question 2 Part a of appendix.
After this, we observe the variation of unit step response of this second order system with the damping ratio. Upon varying the damping ratio from 0 to 2 in steps of 0.25, we get the following plots:

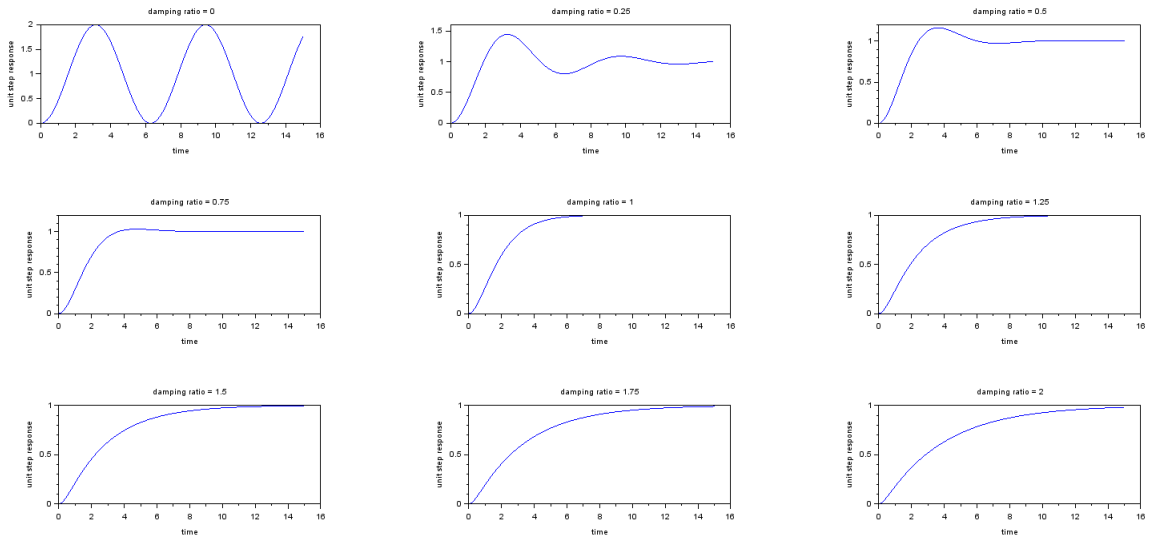

Figure 5: Series of plots showing the variation of the damping ratio

The code used for above plot has been mentioned in Question 2 Part b of appendix. After this I also plotted overlapping plots in a single graph.

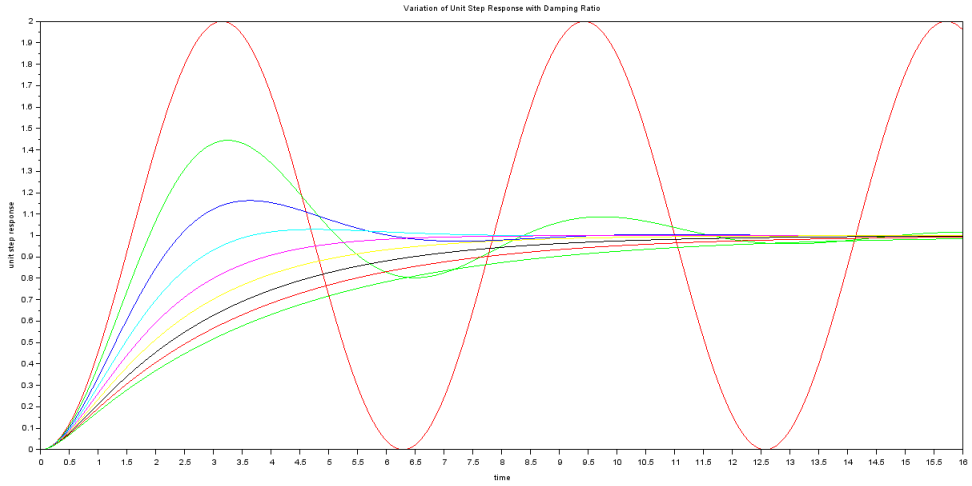

Figure 6: All the plots in the same plot

The code used for above plot has been mentioned in Question 2 Part c of appendix.

From the above two figures, we can see that as the damping ratio increases, the response becomes less oscillatory and more monotonous. For $\zeta = 0$, the response is purely sinusoidal, hence a steady state is not defined for it. Hence only the peak time can be defined for it, which is the **least**. As the **damping ratio increases till it becomes 1**, **percentage overshoot decreases**, **rise time increases**, **2% settling time decreases** & **peak time increases**. As it goes beyond 1, the response becomes monotonically increasing. **Percentage overshoot is 0**, **rise time increases**, **settling time increases** & **peak time is infinite** for all responses.

3 Question 3

We can consider the following system functions:

$$G(s) = \frac{1}{s+1} \text{ (first order)}$$
$$H(s) = \frac{1}{s^2+4s+1} \text{ (second order)}$$

Hence we can see that both the systems have 0 gain. The first order system has a pole at $s = -1$ & the second order system has poles at $s = -3.732$ & $s = -0.268$

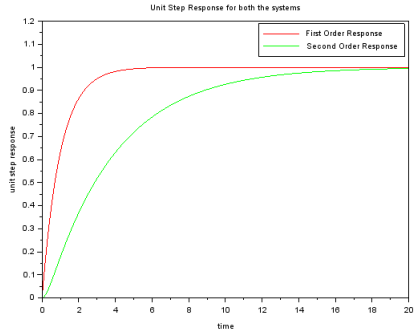

Figure 7: Unit step response for both the systems.

We can observe the following differences:

1. The second order unit step response has a kink at the origin which is not present in the first order unit step response. This is due to the fact that the derivative at the origin is 0 for the second order response, which is not the case for the first order response.
2. The second main difference is that can be observed in the above plot is that the first order response reaches the steady state earlier than the second order response. Its settling time is lower than that of the second order response.

The code has been attached under Question 3 Part a in the appendix.
Suppose the pole $s = -1$ of the second order transfer function was repeated, then we would have the new transfer function as,

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

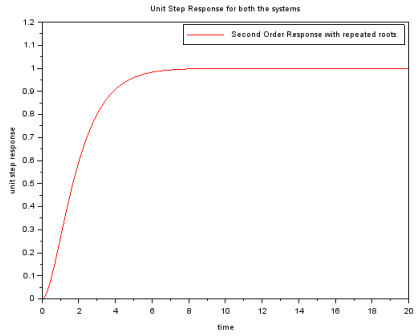

Figure 8: Second order response of system having repeated poles

As we can see from the above figure, on repeating a pole, the gain has increased. The settling time has also increased. However the monotonic nature has been preserved.
The code has been attached under Question 3 Part b in the appendix.

4 Question 4

4.1 Part a

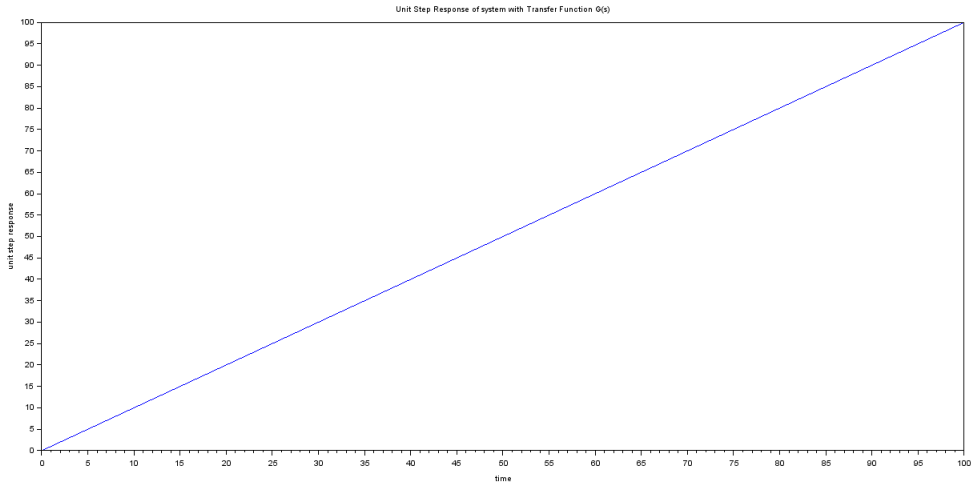

Figure 9: Unit step response of the continuous time integrator system

As we can see from the above figure, the response is just a straight line with slope 1. This is expected since the output will just be,

$$\begin{aligned} y(t) &= \int_{0^-}^{t} dt \\ &= t \end{aligned}$$

The code has been attached under Question 4 Part a in the appendix.

4.2 Part b

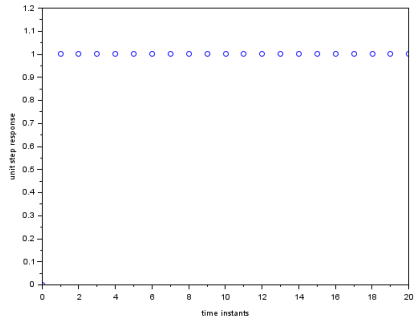

Figure 10: Unit step response of the discrete time transfer function

First define a variable z, then define a rational function 1/z and then using syslin command define the discrete time transfer function for linear system using the argument 'd'. To plot the unit step response, use the dsimul command.
As seen above, the response is all ones for positive time. This is expected since the inverse Z transform of the output is the 'heavyside step function'. This is different from the previous response, which was the integration of the input. This is also expected since the system just delays the input by one time unit.

The code has been attached under Question 4 Part b in the appendix.

4.3 Part c

Suppose we just define a polynomial function as a system and try to simulate it, using the code in Question 4 Part c of appendix. We get the following error:

```
--> y = csim('step',t,G)
WARNING: csim: Input argument #1 is assumed continuous time.
at line    59 of function csim ( D:\scilab-6.0.2\modules\cacsd\macros\csim.sci line 70 )

csim: Wrong type for input argument #1: A proper system expected
--> plot(t,y)

Undefined variable: y
```

Figure 11: Response given by the program

The difference between Part a and b is that even though the transfer function looks similar, it behaves differently in different domains. In continuous time domain(where we take the Laplace transform for analysis), the system behaves like an integrator, whereas in the discrete time domain(where we take the Z transform for analysis), the system delays the input by one unit. In the part c, the code does not compile as we had expected it to since we use the 'csim' command for compilation which deals with only continuous time transfer functions. Hence the compiler checks whether the transfer function is possible or not while simulating. Since an improper transfer function such as the one we chose above is not possible, it throws up an error. Therefore not every rational polynomial can be a transfer function of a system.

5 Question 5

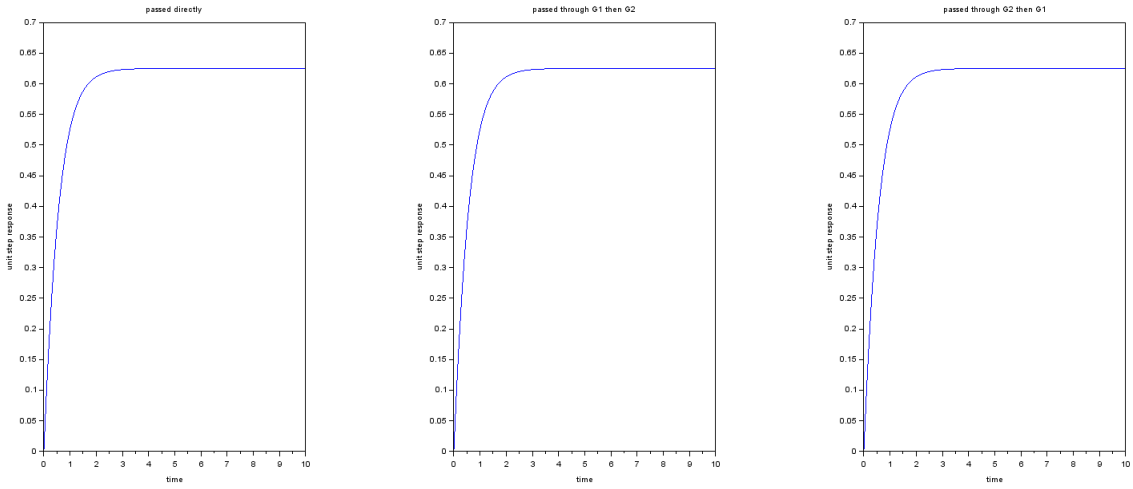

Figure 12: The unit step responses out of each of the three transfer functions

In the above figure we have assumed the value of sampling period (τ) to be 0.1 . We can see that all the three plots are same, which should be expected since they all are LTI systems and hence an interchange wont change their overall behaviour. We now analyse their responses for three different values of τ:

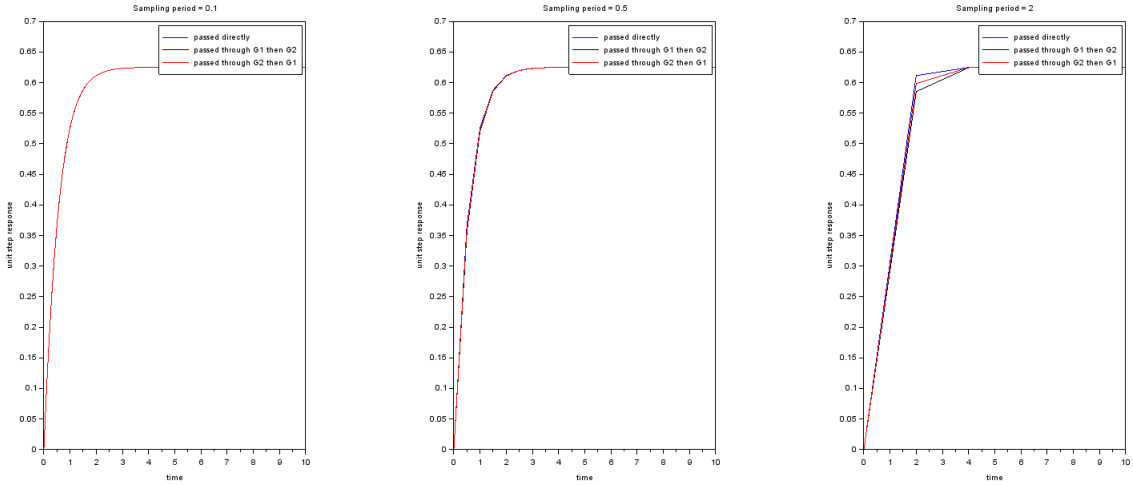

Figure 13: Unit step responses out of each of the three systems, having different values of τ

As we observe from the above figure, the error increases as the value of the sampling period increases. For $\tau = 0.1$, there is almost no visible error, and all the three plots line up well. For $\tau = 0.5$, there is a small but noticeable error. For $\tau = 2$ there is large error, as sampling period increases. This may be due to the computational error, as we break up the transfer function into two separate transfer functions which makes the respective errors combined.

6 Appendix : Codes

6.1 Question 1

6.1.1 Part b

```
a = 83;
b = 16;

s = poly(0,'s');

G = a/(s+b);

sys = syslin('c', G);

t_const = 1/b;

t_st2 = log(50)/b;

t_rise = log(9)/b;

t_rise_st = log(10/9)/b;
t_rise_end = log(10)/b;

t = linspace(0, 1, 1000);

y1 = linspace(0, 6, 100);
x1 = t_rise_st*ones(y1);
x2 = t_rise_end*ones(y1);
x3 = t_const*ones(y1);
x4 = t_st2*ones(y1);


xpts = [t_const, t_st2, t_rise_st, t_rise_end];

y_at_t_const = 5.188*((%e)-1)/(%e);
ypts = [y_at_t_const, 0.98*5.188, 0.1*5.188, 0.9*5.188];
ypts2 = [0,0,0,0]

labels = ['time constant','2% settling time','start of rise','end of rise'];

plot(t, csim('step', t, G), 'k-');

plot(x1, y1, 'b--');
plot(x2, y1, 'b--');
plot(x3, y1, 'b--');
plot(x4, y1, 'b--');

plot(xpts, ypts, 'o');
xstring(xpts, ypts2, labels);

xtitle ("Unit Step Response of system with Transfer Function G(s)", "time" , "unit step response" );
```

6.1.2 Part c

```
a = 83;
b = 16;

s = poly(0,'s');

t_rise_array = [];
for x = a:a:100*a
    G = x/(s+b);
    sys = syslin('c', G);
    t_rise = log(9)/b;
    t_rise_array($+1) = t_rise;
end

plot(a:a:100*a, t_rise_array);
xtitle ("Variation of Rise Time with the parameter a", "a" , "rise time" );
```

6.1.3 Part d

```
a = 83;
b = 16;

s = poly(0,'s');

t_rise_array = [];
for x = b:b:100*b
    G = a/(s+x);
    sys = syslin('c', G);
    t_rise = log(9)/x;
    t_rise_array($+1) = t_rise;
end

plot(b:b:100*b, t_rise_array);

xtitle ("Variation of Rise Time with the parameter b", "b" , "rise time" );
```

6.2 Question 2

6.2.1 Part a

```
s = poly(0,'s');
//nat_freq = 1, and damping ratio = 0.5
P= 1/(s^2 + s + 1);
sys = syslin('c',G);

t =[0:0.01:20];
y = csim('step', t, P);

plot(t,y);
xtitle ("Step response of system (s)", "time (t)" , "y(t)" );
```

6.2.2 Part b

```
// Multiple plots in single figure
s = poly(0,'s');
```

```
t = [0:0.01:15];

for i = 0:8
    num_subplot = 331+i;
    subplot(num_subplot);
    P = 1/(s^2 + s*i*2*0.25 + 1);
    sys = syslin('c',P);
    plot(t,csim('step', t, P));
    x = string(i*0.25);
    xtitle('damping ratio = '+ x);
    xlabel('time');
    ylabel('unit step response');
end
```

6.2.3 Part c

```
// Overlapping figures with different colors
s = poly(0,'s');
t = [0:0.01:16];
colours = ['r' 'g' 'b' 'c' 'm' 'y' 'k' 'r' 'g'];

for i=0:8
    P = 1/(s^2 + s*i*2*0.25 + 1);
    P = syslin('c',P);
    plot(t, csim('step', t, P), colours(i+1));
end

xtitle ("Variation of Unit Step Response with Damping Ratio", "time" , "unit step response" );
```

6.3 Question 3

6.3.1 Part a

```
s = poly(0,'s');

G1 = 1/(s + 1);
G2 = 1/(s^2 + 4*s + 1);

sys1 = syslin('c',G1);
sys2 = syslin('c',G2);

t = [0:0.01:20];

plot(t,csim('step', t, G1), 'r');
plot(t,csim('step', t, G2), 'g');

xtitle ("Unit Step Response for both the systems", "time" , "unit step response" );
legend ("First Order Response" , "Second Order Response");

var = gca();
var.data_bounds = [0,0; 20,1.2];
```

6.3.2 Part b

```
s = poly(0,'s');
```

```
//repeated poles
G = 1/(s^2 + 2*s +1);

sys = syslin('c',G);
t = [0:0.01:20];

plot(t,csim('step', t, G), 'r');

var = gca();
var.data_bounds = [0,0; 20,1.2];
xtitle ("Unit Step Response for both the systems", "time" , "unit step response" );
legend ("Second Order Response with repeated roots");
```

6.4 Question 4

6.4.1 Part a

```
s = poly(0,'s');

G = 1/s;
sys = syslin('c',G);

t = [0:0.1:100];
y = csim('step', t, G);

plot(t,y);

xtitle ("Unit Step Response of system with Transfer Function G(s)", "time" , "unit step response" );
```

6.4.2 Part b

```
z = poly(0,'z');

G = 1/z;
G = syslin('d',G);
G = tf2ss(G);

ts = 1;
t = 0:ts:20;

u = ones(1,length(t));

ds = dsimul(G,u);

plot(t, ds, 'o');

var = gca();
var.data_bounds = [0,0; 20,1.2];
xtitle ('',"time instants" , "unit step response" );
```

6.4.3 Part c

```
s = poly(0,'s');

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

```
t = [0:0.01:20];

y = csim('step',t,G)
plot(t,y)
```

6.5 Question 5

6.5.1 Part a

```
s = poly(0,'s');
t = [0:0.1:10];


G = (s+5)/[(s+4)*(s+2)];
G = syslin('c',G);
G1 = (s+5)/(s+4);
G1 = syslin('c',G1);
G2 = (1)/(s+2);
G2 = syslin('c',G2);

subplot(131);
plot(t, csim('step', t, G));
xlabel('time');
ylabel('unit step response');
xtitle("passed directly");

subplot(132);
plot(t, csim(csim('step', t, G1), t, G2));
xlabel('time');
ylabel('unit step response');
xtitle("passed through G1 then G2");

subplot(133);
plot(t, csim(csim('step', t, G2), t, G1));
xlabel('time');
ylabel('unit step response');
xtitle("passed through G2 then G1");
```

6.5.2 Part b

```
s = poly(0,'s');

G = (s+5)/[(s+4)*(s+2)];
G = syslin('c',G);
G1 = (s+5)/(s+4);
G1 = syslin('c',G1);
G2 = (1)/(s+2);
G2 = syslin('c',G2);


subplot(131)
t = [0:0.1:10];
plot(t, csim('step', t, G));
xlabel('time');
ylabel('unit step response');
```

```
subplot(132)
t = [0:0.5:10];
plot(t, csim('step', t, G));
xlabel('time');
ylabel('unit step response');

subplot(133);
t = 0:2:10;
plot(t, csim('step', t, G));
xlabel('time');
ylabel('unit step response');

// ---------------------------------------

subplot(131);
t = [0:0.1:10];
plot(t, csim(csim('step', t, G1), t, G2), 'k');
xlabel('time');
ylabel('unit step response');

subplot(132);
t = [0:0.5:10];
plot(t, csim(csim('step', t, G1), t, G2), 'k');
xlabel('time');
ylabel('unit step response');

subplot(133);
t = [0:2:10];
plot(t, csim(csim('step', t, G1), t, G2), 'k');
xlabel('time');
ylabel('unit step response');

// ---------------------------------------

subplot(131);
t = [0:0.1:10];
plot(t, csim(csim('step', t, G2), t, G1), 'r');
xlabel('time');
ylabel('unit step response');
xtitle("Sampling period = 0.1");
legend ("passed directly" , "passed through G1 then G2", "passed through G2 then G1");

subplot(132);
t = [0:0.5:10];
plot(t, csim(csim('step', t, G2), t, G1), 'r');
xlabel('time');
ylabel('unit step response');
xtitle("Sampling period = 0.5");
legend ("passed directly" , "passed through G1 then G2", "passed through G2 then G1");

subplot(133);
t = 0:2:10;
plot(t, csim(csim('step', t, G2), t, G1), 'r');
```

```
xlabel('time');
ylabel('unit step response');
xtitle("Sampling period = 2");
legend ("passed directly" , "passed through G1 then G2", "passed through G2 then G1");
```