EE324 Lab 3

Shubham Kar, 180070058

January 2021

1 Aim

To analyze second order systems and form approximate second order systems from third order systems.

2 Question 1

2.1 Part a

We are given the transfer function $\left(\frac{s+5+a}{s^2+11s+30}\right)$, where a varies from -1 to 1 in steps of 0.01. The step response of the system for the various values of 'a' are shown in Figure **??**.

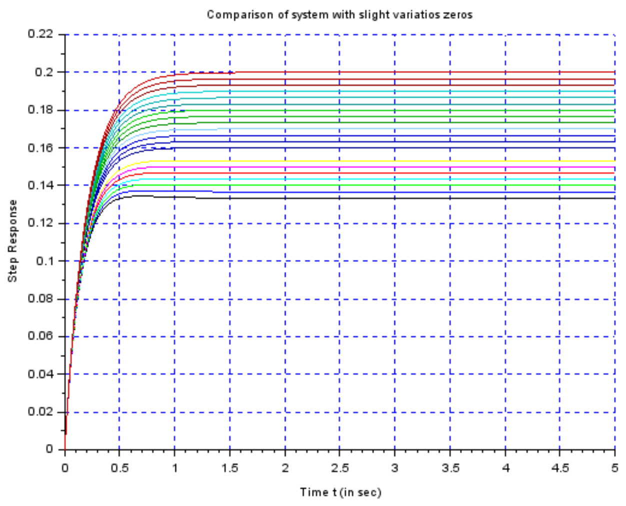


Figure 1: Comparison of the step response of the system with variation in its zero

We observe that the response remains almost the same even after pole-zero cancellation. The steady state value changes, but that is primarily due to changes in its gain due to slight changes in the zeroes. We observe no occurrence of derivative 0 at t=0 because the difference of degrees of the denominator and numerator is not more than 2.
The code for generating the same is shown below:

```
clear();
s = poly(0, 's');
tau = 0.1;
t = 0:0.001:5;
out = zeros(length(t), length(-1:tau:1));
i = 1;
for a = -1:tau:1
    G = (s + 5 + a)/(s^2 + 11*s + 30);
    G = simp(G);
    sys = syslin('c', G);
    out(:, i) = csim('step', t, sys);
    i = i + 1;
end
scf(1);
clf();
plot2d(t, out, 1:length(-1:tau:1));
xgrid(2);
xlabel("Time t (in sec)"); ylabel("Step Response");
title("Comparison of system with slight variatios zeros");
xs2png(gcf(), "Q1a.png");
```

2.2 Part b

The step response of $\left(\frac{1}{s^2-s-6}\right)$ is shown in Figure **??**.
We observe that it is unbounded due to the presence of a pole in the Right Hand complex Plane. That pole is $(+3)$. For cancelling this pole, we multiplied the transfer function with $(s-3)$ and then plotted its step response. We get the plot as shown in Figure 3.
We observe that the step response now is bounded. Adding even a small amount to the zero which is multiplied with the original transfer function makes the system unstable due to the presence of a pole in RHP. These can be easily seen in Figure 3. We see that even a small variation of 0.0001 makes the step response unbounded very quickly. Thus, we can say that to cancel an unstable pole, we need to add zeroes which exactly cancels that pole. Adding zeroes which attempt to cancel that pole won't help in stabilizing the response.
The code for reproducing the same is shown below:

```
clear();
s = poly(0, 's');
t = 0:0.01:10;
y = zeros(length(t), 2);

G1 = 1/(s^2 - s - 6);
sys1 = syslin('c', G1);
y(:, 1) = csim('step', t, sys1);

G2 = G1*(s-3);
```

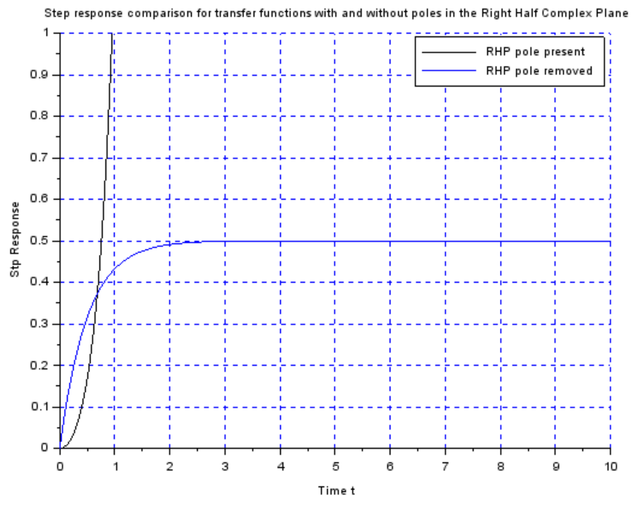


Figure 2: Step response with pole on the RHP

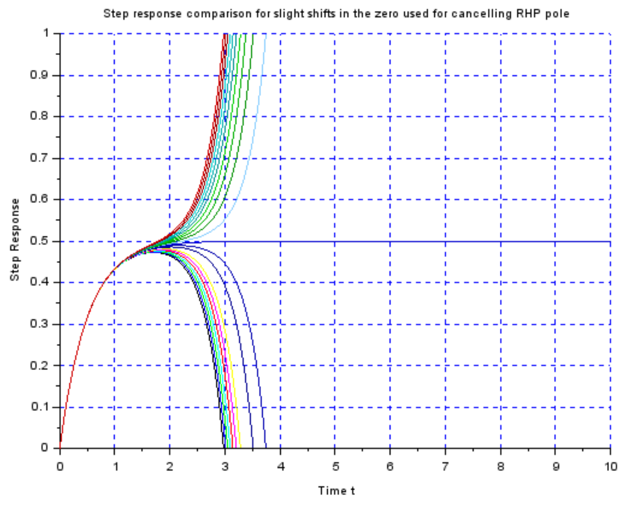


Figure 3: Step response without a pole on the RHP

```
G2 = simp(G2);
sys2 = syslin('c', G2);
y(:, 2) = csim('step', t, sys2);

tau = 0.0001;
max_a = 0.001
```

```
y_new = zeros(length(t), length(0:tau:max_a));
i = 1;
for a=-max_a:tau:max_a
    G3 = G1*(s - 3 + a);
    G3 = simp(G3);
    sys3 = syslin('c', G3);
    y_new(:, i) = csim('step', t, sys3);
    i = i + 1;
end

scf(1);
clf();
plot2d(t, y, 1:2);
a = gca();
a.data_bounds = [0, 0; 10, 1];
legend(["RHP pole present", "RHP pole removed"]);
xlabel("Time t"), ylabel("Stp Response");
title("Step response comparison for transfer functions with and ↩
    without poles in the Right Half Complex Plane");
xgrid(2);
xs2png(gcf(),"Q1b_1.png");

scf(2);
clf();
plot2d(t, y_new, 1:length(0:tau:1));
a = gca();
a.data_bounds = [0, 0; 10, 1];
xlabel("Time t"); ylabel("Step Response");
title("Step response comparison for slight shifts in the zero used for↩
     cancelling RHP pole");
xgrid(2);
xs2png(gcf(),"Q1b_2.png");
```

3 Question 2

3.1 Part a

We are given that

$$H(s) = \frac{85}{(s^3 + 7s^2 + 27s + 85)} = \frac{17}{(0.2s + 1)(s^2 + 2s + 17)} \tag{1}$$

We see that if we just consider the second order part of the transfer function, then the undamped natural frequency is $\omega_n = \sqrt{17}$ and the damping ratio is $\zeta = \frac{1}{\sqrt{17}}$. And observing hte additional pole, we have that $\tau = 0.2$. So, we have that

$$\left|\frac{1}{\tau}\right| \geq 5|\omega_n \zeta|$$

So, we can ignore the extra pole at (-5) as it attains its steady state fairly quickly that the second order system. The step responses of the approximated second order system and the original system is shown in Figure 4.
The plot for the same is shown below:

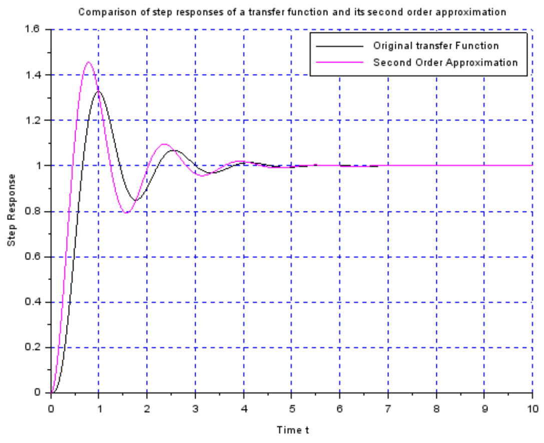


Figure 4: Comparison of approximated second order system with original system

```
clear();
s = poly(0,'s');
t = 0:0.01:10;
y = zeros(length(t),2);

G1 = 85/(s^3 + 7*s^2 + 27*s + 85);
G2 = 17/(s^2 + 2*s + 17);
sys1 = syslin('c', G1);
sys2 = syslin('c', G2);
y(:, 1) = csim('step', t, sys1);
y(:, 2) = csim('step', t, sys2);
scf(1);
clf();
plot2d(t, y, 1:2);
a = gca();
a.children(1).children(1).foreground = 6;
legend(["Original transfer Function", "Second Order Approximation"]);
xlabel("Time t"); ylabel("Step Response");
title("Comparison of step responses of a transfer function and its ↩
    second order approximation");
xgrid(2);
xs2png(gcf(), "Q2a.png");
```

3.2 Part b

We now have:

$$H(s) = \frac{(s+0.01)}{(s^3+2.02s^2+5.04s+0.1)} = \frac{(s+0.01)}{(s+0.02)(s^2+2s+5)} \tag{2}$$

We observe that the zero and one of the poles are very close to each other. They almost cancel each other out and therefore, we can suitably approximate $H(s)$ by neglecting this pole-zero pair. However, we need to account for he change in the DC gain. This will result in multiplying the second order part of the transfer function with 0.5 to get the approximated second order system. The step responses for both are shown in Figure 5.

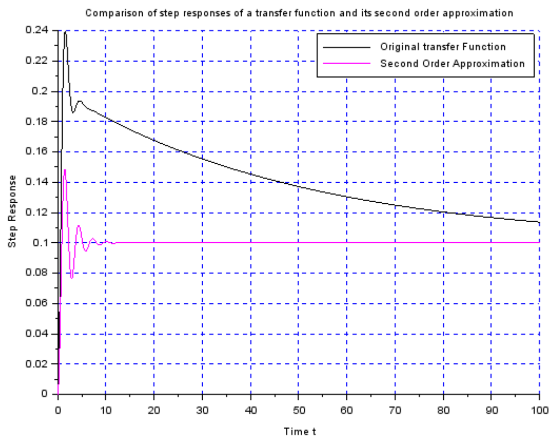


Figure 5: Comparison of approximated and and original step responses

We see that the approximation does not hold good for the transient response but holds good in the steady state time frame.
The code for producing the same is shown below:

```
clear();
s = poly(0,'s');
t = 0:0.01:100;
y = zeros(length(t),2);

G1 = (s+0.01)/(s^3 + 2.02*s^2 + 5.04*s + 0.1);
G2 = 1/(2*s^2 + 2*s + 10);
sys1 = syslin('c', G1);
sys2 = syslin('c', G2);
y(:, 1) = csim('step', t, sys1);
y(:, 2) = csim('step', t, sys2);
scf(1);
clf();
plot2d(t, y, 1:2);
a = gca();
a.children(1).children(1).foreground = 6;
legend(["Original transfer Function", "Second Order Approximation"]);
xlabel("Time t"); ylabel("Step Response");
title("Comparison of step responses of a transfer function and its ↩
    second order approximation");
xgrid(2);
xs2png(gcf(), "Q2b.png");
```

4 Question 3

4.1 Part a

The system given was:

$$H(s) = \frac{9}{(s^2 + 2s + 9)} \tag{3}$$

The poles of $H(s)$ are $(-1 \pm 3i)$. Now, a zero at (-1) is added to $H(s)$. This maintains the DC gain of the system and we can properly see compare the changes in the various parameters of the system. The step responses of the 2 systems are shown in Figure 6.

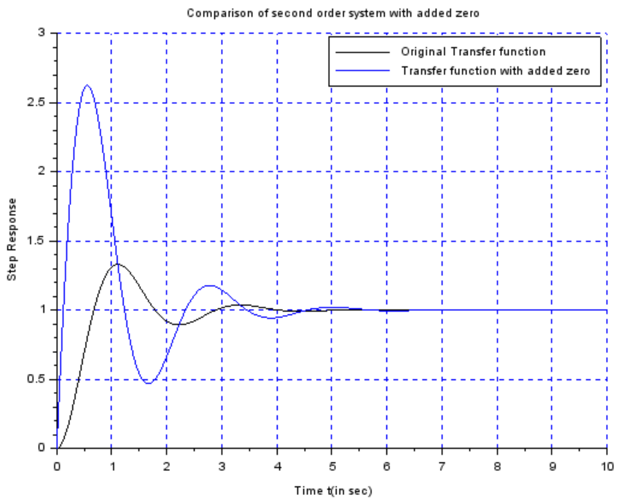


Figure 6: Comparison of the step response of under damped system before and after adding a zero

The response parameters are shown below in Table 1.

Parameters	Original System	System with added zero
Percentage Overshoot	32.93%	162.31%
Rise Time	0.455 sec	0.096 sec
Peak Time	1.111 sec	0.555 sec
Settling Time	2.786 sec	4.376 sec

Table 1: Step response parameters for under damped systems with added zeroes

The code for generating the same is shown below:

```
clear();
s = poly(0, 's');
t = 0:0.001:9;
y = zeros(length(t), 2);

G1 = 9/(s^2 + 2*s + 9);
//G1 = 1/(s+1);
sys1 = syslin('c', G1);
//gs = trfmod(sys1, 'f');
G2 = G1*(s+1);
sys2 = syslin('c', G2);

trfmod(sys1, 'p');
trfmod(sys2,'p');
y(:, 1) = csim('step', t, sys1);
y(:, 2) = csim('step', t, sys2);
parameters = zeros(2, 4);
for i=1:2
    parameters(i, 1) = (max(y(:, i)) - 1);

    temp = find(y(:, i)>0.9);
    t2 = t(temp(1));
    temp = find(y(:, i)>0.1);
    t1 = t(temp(1));
    parameters(i, 2) = (t2 - t1);

    parameters(i, 3) = t(find(y(:, i) == max(y(:, i))));

    temp = find(y(:, i)>0.98);
    parameters(i, 4) = t(temp(1));
end
disp(parameters(1,:));
disp(parameters(2,:));
scf(1);
clf();
plot2d(t, y, 1:2);
legend(["Original Transfer function", "Transfer function with added ↩
    zero"]);
a = gca();
xlabel("Time t(in sec)"); ylabel("Step Response");
title("Comparison of second order system with added zero");
//a.children(1).children(1).foreground = 6;
xgrid(2);
xs2png(gcf(),"Q3a.png")
```

4.2 Part b

Now, we add poles to the system $H(s)$. The first pole we add is (-0.8), which is closer to the origin than the other poles and is on the left-hand real axis. Next, we normalize the DC gain to be 1 so that its easier to visually compare.
Next, we do the same using a pole at (-1.2). The step responses are shown in Figure 7.
The step response parameters are shown in Table 2. The code for the same is shown below:

```
clear();
s = poly(0, 's');
t = 0:0.001:20;
y = zeros(length(t), 3);
```

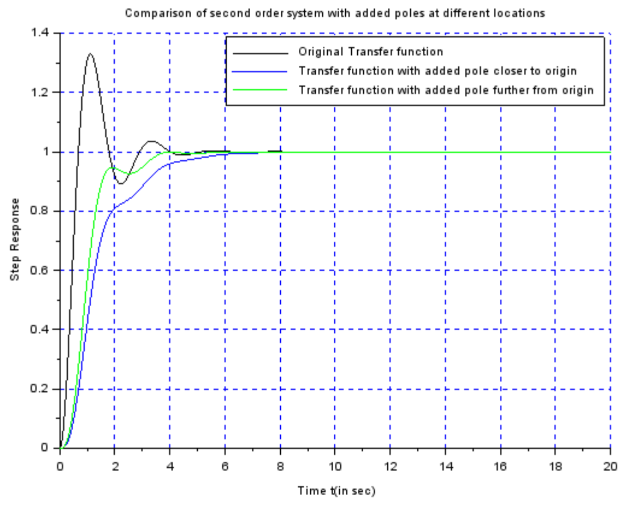


Figure 7: Comparison of step responses after adding poles at specific locations

Parameters	Original System	Added pole closer to origin	Added pole farther from origin
Percentage Overshoot	32.93%	0%	0.025%
Rise Time	0.455 sec	2.646 sec	1.095 sec
Peak Time	1.111 sec	–	6.188 sec
Settling Time	2.786 sec	5.145 sec	3.401 sec

Table 2: Step response parameters for under damped systems with added poles

```
G1 = 9/(s^2 + 2*s + 9);
//G1 = 1/(s+1);
sys1 = syslin('c', G1);
//gs = trfmod(sys1, 'f');
G2 = (G1*0.8)/(s+0.8);
sys2 = syslin('c', G2);
G3 = (1.2*G1)/(s+1.2);
sys3 = syslin('c', G3);
y(:, 1) = csim('step', t, sys1);
y(:, 2) = csim('step', t, sys2);
y(:, 3) = csim('step', t, sys3);
parameters = zeros(3, 4);
for i=1:3
    parameters(i, 1) = (max(y(:, i)) - 1);

    temp = find(y(:, i)>0.9);
    t2 = t(temp(1));
    temp = find(y(:, i)>0.1);
    t1 = t(temp(1));
    parameters(i, 2) = (t2 - t1);

    parameters(i, 3) = t(find(y(:, i) == max(y(:, i))));
    temp = find(y(:, i)<0.98);
    parameters(i, 4) = t(temp(length(temp)));
end
disp(parameters(1,:));
disp(parameters(2,:));
disp(parameters(3,:));
scf(1);
clf();
plot2d(t, y, 1:3);
legend(["Original Transfer function", "Transfer function with added ↩
    pole closer to origin", "Transfer function with added pole further↩
     from origin"]);
a = gca();
//a.children(1).children(1).foreground = 6;
xlabel("Time t(in sec)"); ylabel("Step Response");
title("Comparison of second order system with added poles at different↩
     locations");
xgrid(2);
xs2png(gcf(), "Q3b.png");
```

4.3 Part c

We observe that additional zeroes in the system can significantly increase the percentage overshoot of the under damped system. It also decreases the rise time and therefore, the peak time too. The settling time increases in the case observed. So, we see that adding a zero may create larger disturbances in the transient phase. This greater disturbance at the start creates larger ripples which result in a longer settling time for the system with added zero. It is basically like the addition of the derivative of the original step response to itself. This results in addition of the original response at points close to the rising peak as the derivative is positive there. This in turn, increases settling time too due to additional disturbances.
Upon adding poles closer to the origin, we observe that there is no percentage overshooting and the rise time and the settling time increase as well. Since we don't observe any overshoot, peak time is not mentioned in Table 2. This should also theoretically occur as the disturbance to the original under damped system due to the pole closer to the origin should outweigh the peak observed in the under damped case as the magnitude of the extra pole response getting added is more significant than the real parts of the other two poles at larger t. The opposite case will be observed if the pole added is farther from the origin. This pole should attain its steady state at a time before the other pole responses observe their peaks or steady state values. This results in a lesser disturbance to the original under damped system and we observe lower rise and settling time than the previos case. We alos observe a slight(negligible) peak here.

5 Question 4

5.1 Part a

The un-damped, under damped and the over damped step responses are shown in Figure 8. The parameters of the step responses are shown in Table 3. //Next,

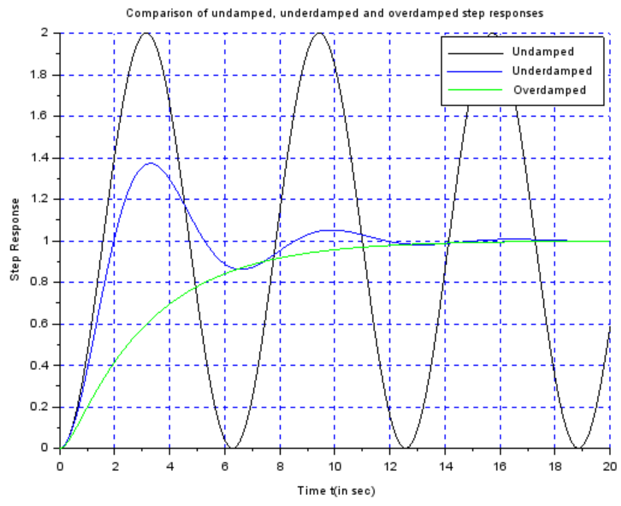


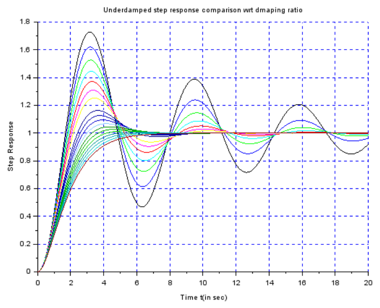
Figure 8: Comparison of un-damped, under damped and over damped systems

Parameters	Un Damped	Under Damped	Over Damped
Percentage Overshoot	–%	37.232%	0%
Rise Time	– sec	1.32 sec	7.43 sec
Peak Time	2.5 sec	3.29 sec	– sec
Settling Time	– sec	8.3 sec	12.37 sec

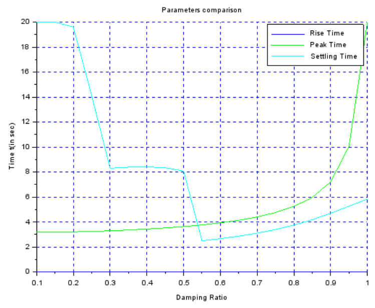
Table 3: Step response parameters for un damped, under damped and over damped systems

the damping ratio was varied from 0.1 to 1 in steps of 0.05. The results are shown in Figure 9.
We observe that percentage overshoot decreases monotonically while settling time decreases relatively with increasing damping ratio ζ. However, the peak and the rise times increase monotonically with increasing damping ratio.
Th code for generating the same results are shown below:

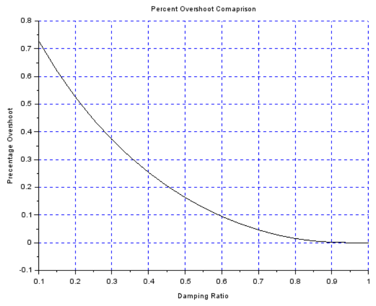
```
clear();
w_n = 1;
zeta = [0, 0.3, 1.7];
s = poly(0, 's');
t = 0:0.01:20;
y = zeros(length(t), 3);

for i=1:3
    G = (w_n^2)/(s^2+(2*zeta(i)*w_n*s)+(w_n^2));
    sys = syslin('c', G);
    y(:, i) = csim('step', t, sys);
end
```



(a) Step responses comparison for varying damping ratios



(b) Varying rise, settling and peak times with damping ratios



(c) Percentage Overshoot with varying damping ratio

Figure 9: Comparison of parameters with varying damping ratios

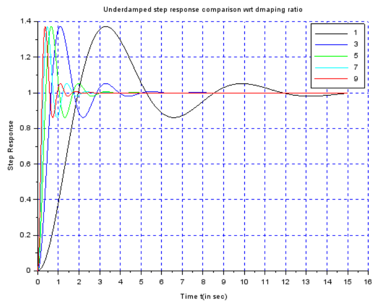
```
scf(1);
clf();
plot2d(t, y, 1:3);
legend(["Undamped", "Underdamped", "Overdamped"]);
xlabel("Time t(in sec)"); ylabel("Step Response");
title("Comparison of undamped, underdamped and overdamped step ↩
    responses");
xgrid(2);
xs2png(gcf(), "Q4a.png");

// Underdamped Parameters
percent_overshoot = (max(y(:, 2)) - 1)*100;

temp = find(y(:, 2)>0.9);
t2 = t(temp(1));
temp = find(y(:, 2)>0.1);
t1 = t(temp(1));
rise_time = (t2 - t1);

peak_time = t(find(y(:, 2) == max(y(:, 2))));

temp = find(y(:, 2)<0.98);
settling_time = t(temp(length(temp)));
```

```
mprintf("Percent Overshoot = %0.3f\n", percent_overshoot);
mprintf("Rise Time = %0.3f\n", rise_time);
mprintf("Peak Time = %0.3f\n", peak_time);
mprintf("Settling Time = %0.3f\n", settling_time);
mprintf("\n");

// Overdamped Parameters
percent_overshoot = (max(y(:, 3)) - 1)*100;

temp = find(y(:, 3)>0.9);
t2 = t(temp(1));
temp = find(y(:, 3)<0.1);
t1 = t(temp(1));
rise_time = (t2 - t1);

//peak_time = t(find(y(:, 3) == max(y(:, 3))));

temp = find(y(:, 3)<0.98);
settling_time = t(temp(length(temp)));

mprintf("Percent Overshoot = %0.3f\n", percent_overshoot);
mprintf("Rise Time = %0.3f\n", rise_time);
//mprintf("Peak Time = %0.3f\n", peak_time);
mprintf("Settling Time = %0.3f\n", settling_time);

pts = 0.1:0.05:1;
parameters = zeros(length(pts), 4);
i = 1;
y = zeros(length(t), length(pts));
for zeta=0.1:0.05:1
    G = (w_n^2)/(s^2+(2*zeta*w_n*s)+(w_n^2));
    sys = syslin('c', G);
    y(:, i) = csim('step', t, sys);

    parameters(i, 1) = (max(y(:, i)) - 1);

    temp = find(y(:, i)>0.9);
    t2 = t(temp(1));
    temp = find(y(:, i)<0.1);
    t1 = t(temp(1));
    parameters(i, 2) = (t2 - t1);

    parameters(i, 3) = t(find(y(:, i) == max(y(:, i))));

    temp = find(y(:, i)<0.98);
    parameters(i, 4) = t(temp(length(temp)));

    i = i+1;
end
scf(2);
clf();
plot2d(pts, parameters(:, 2:4), 2:4);
legend(["Rise Time", "Peak Time", "Settling Time"]);
xlabel("Damping Ratio"); ylabel("Time t(in sec)");
title("Parameters comparison");
xgrid(2);
xs2png(gcf(), "Q4a_param.png");
scf(3);
clf();
plot2d(t, y, 1:length(pts));
title("Underdamped step response comparison wrt dmaping ratio");
xlabel("Time t(in sec)"); ylabel("Step Response");
xgrid(2);
xs2png(gcf(), "Q4a_plot_compare.png");
scf(4);
clf();
plot2d(pts, parameters(:, 1));
title("Percent Overshoot Comaprison");
```

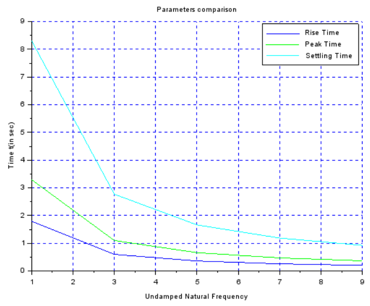
```
xlabel("Damping Ratio"); ylabel("Precentage Overshoot");
xgrid(2);
xs2png(gcf(), "Q4a_percent.png");
```

5.2 Part b

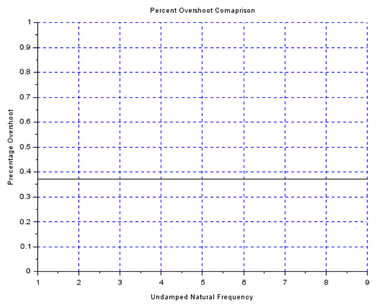
The un-damped natural frequency was varied from 1 to 9 in steps of 2. The damping ratio ζ was maintained at 0.3 This yielded the step responses and parameters as shown in Figure 10.



(a) Step responses comparison for varying natural frequency



(b) Varying rise, settling and peak times with natural frequency



(c) Percentage Overshoot with varying natural frequency

Figure 10: Comparison of parameters with varying natural frequency

We observe that percentage overshoot remains the same across all the plots. However, the rise, settling and peak times fall off monotonically with increasing natural frequencies. This should also happen theoretically as the natural frequency affects the envelope of the under damped step response and increasing that frequency just decreases the time constant of the envelope which is exponential. This in turn results in the envelope achieving its steady state quicker than before and therefore, the rise, peak and the settling times increase. This

frequency however does not have any affect on the maximum magnitude achievable by the under damped system which results in the percentage overshoot remaining the same for all the plots.
The code for the above is shown below:

```
clear();
w_n = 1:2:9;
zeta = 0.3;
s = poly(0, 's');
t = 0:0.001:15;
y = zeros(length(t), length(w_n));
parameters = zeros(length(w_n), 4);

for i=1:length(w_n)
    G = (w_n(i)^2)/(s^2+(2*zeta*w_n(i)*s)+(w_n(i)^2));
    sys = syslin('c', G);
    y(:, i) = csim('step', t, sys);

    parameters(i, 1) = (max(y(:, i)) - 1);

    temp = find(y(:, i)>0.9);
    t2 = t(temp(1));
    temp = find(y(:, i)<0.1);
    t1 = t(temp(1));
    parameters(i, 2) = (t2 - t1);

    parameters(i, 3) = t(find(y(:, i) == max(y(:, i))));

    temp = find(y(:, i)<0.98);
    parameters(i, 4) = t(temp(length(temp)));
end
scf(1);
clf();
plot2d(w_n, parameters(:, 2:4), 2:4);
legend(["Rise Time", "Peak Time", "Settling Time"]);
xlabel("Undamped Natural Frequency"); ylabel("Time t(in sec)");
title("Parameters comparison");
xgrid(2);
xs2png(gcf(), "Q4b_param.png");
scf(2);
clf();
plot2d(t, y, 1:length(w_n));
title("Underdamped step response comparison wrt dmaping ratio");
legend(["1", "3", "5", "7", "9"]);
xlabel("Time t(in sec)"); ylabel("Step Response");
xgrid(2);
xs2png(gcf(), "Q4b_plot_compare.png");
scf(3);
clf();
plot2d(w_n, parameters(:, 1));
title("Percent Overshoot Comaprison");
a = gca();
a.data_bounds = [1,0;9,1];
xlabel("Undamped Natural Frequency"); ylabel("Precentage Overshoot");
xgrid(2);
xs2png(gcf(), "Q4b_percent.png");
```