EE324 Control Systems Lab

Problem Sheet 3

Sheel Shah — 19D070052

August 14, 2021

1 Question 1

1.1 Part A

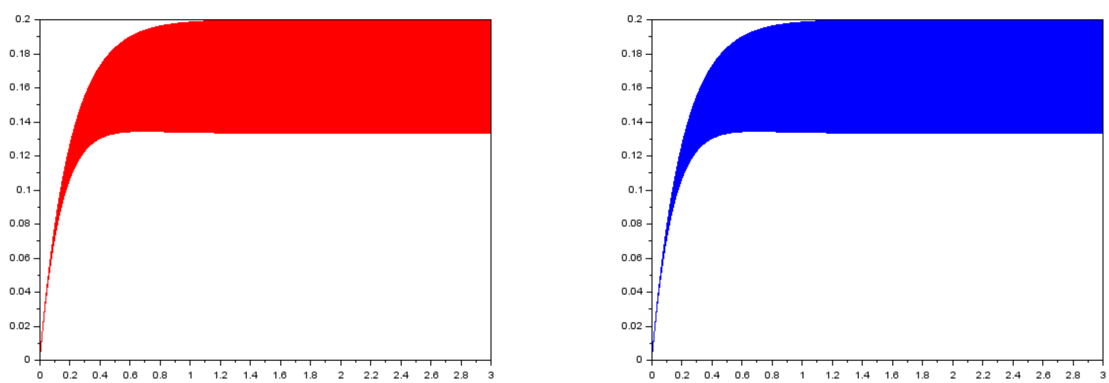

Figure 1: without simp(left), with simp(right)

We conclude that the response remains the same before and after the pole-zero cancellation

The following code is used

```
s = poly(0, 's');
t = 0:0.01:3;
a_range = -1:0.01:1;

scf(0);
for a = a_range
    // without simp
    g_temp = (s + 5 + a)/(s^2 + 11*s + 30);
    g = syslin('c', g_temp);
    step_resp1 = csim('step', t, g);
    plot(t, step_resp1, "r");
end;

scf(1);
for a = a_range
    // with simp
    g_temp = (s + 5 + a)/(s^2 + 11*s + 30);
    g = syslin('c', g_temp);
    step_resp1 = csim('step', t, simp(g));
    plot(t, step_resp1, "b");
end;
// save plots
```

1.2 Part B

Code used:

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


scf(0)

g_temp = 1/(s^2 - s - 6)

g = syslin('c', g_temp)

t = 0:0.01:5

step_resp = csim('step', t, g)

plot(t, step_resp) // step_resp of the given system


scf(1)

g_temp = (s - 3)/(s^2 - s - 6) // cancel rhp pole

g = syslin('c', g_temp)

step_resp = csim('step', t, g)

plot(t, step_resp) // step_resp with unstable pole cancelled


scf(2)

g_temp = (s - 3 - 0.01)/(s^2 - s - 6)

g = syslin('c' , g_temp)

step_resp = csim('step', t, g)

plot(t, step_resp, "r") // step_resp with slightly displaced zero

g_temp = (s - 3)/(s^2 - s - 6)

g = syslin('c', g_temp)

step_resp = csim('step', t, g)
```

```
plot(t, step_resp, "g") // step_resp with exavt zero

g_temp = (s - 3 + 0.01)/(s^2 - s - 6)

g = syslin('c', g_temp)

step_resp = csim('step', t, g)

plot(t, step_resp, "b") // step_resp with slightly displaced zero
```

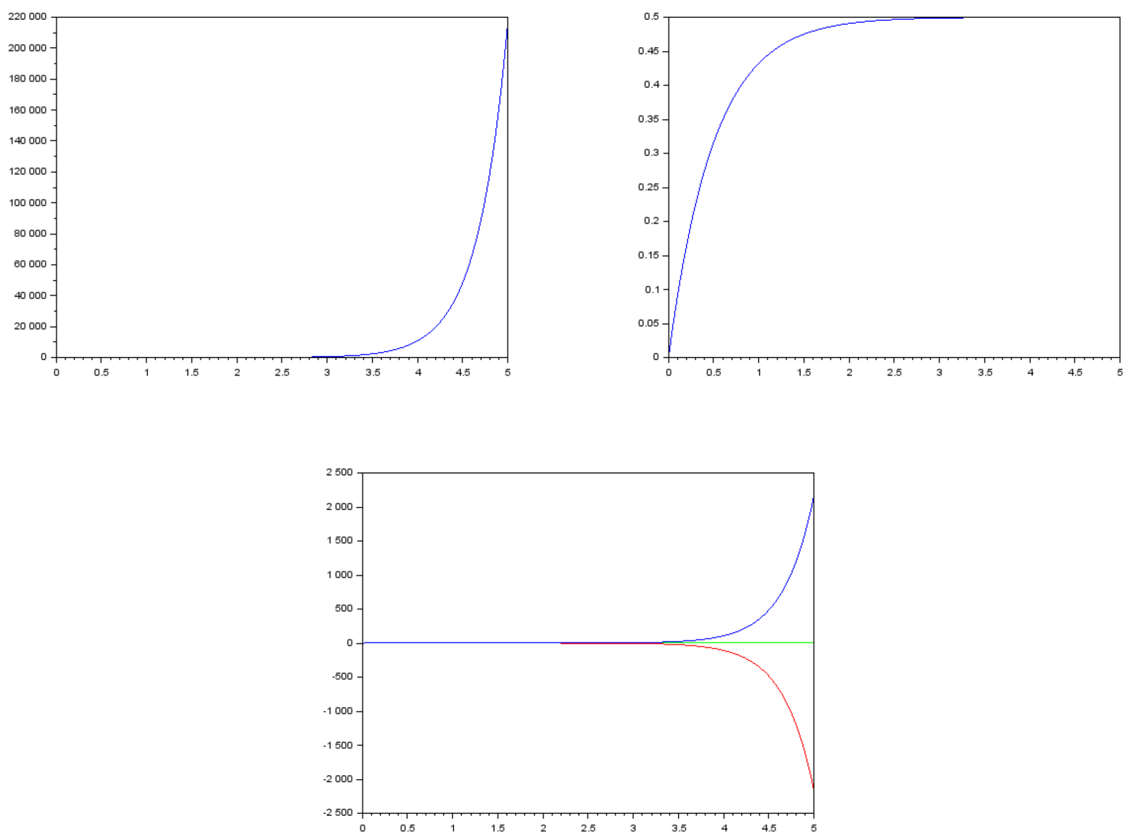

Figure 2: original system, system with pole-zero cancellation, zero displaced

We can hence conclude that the statement given in the problem sheet is false. An exact pole zero cancellation of an unstable pole can indeed render the system stable

2 Question 2

2.1 Part A

The following code is used to find the roots of the system

```
s = poly(0,'s')
scf(0)
g_temp = 85/(s^3 + 7*s^2 + 27*s + 85)
roots(g_temp.den)
```

The roots are $-5, -1+4j, -1-4j$

Hence the pole that is very far away from the imaginary axis is ignored. The new transfer function is found by:

```
g_new_temp = g_temp * (s + 5)
g_new_temp = g_new_temp / 5
// cancel pole and scale aptly
```

Plots are generated as:

```
g = syslin('c', g_temp)
g_new = syslin('c', g_new_temp)
t = 0:0.01:5
step_resp = csim('step', t, g)
plot(t, step_resp, 'r')
step_resp = csim('step', t, g_new)
plot(t, step_resp, 'b')
```

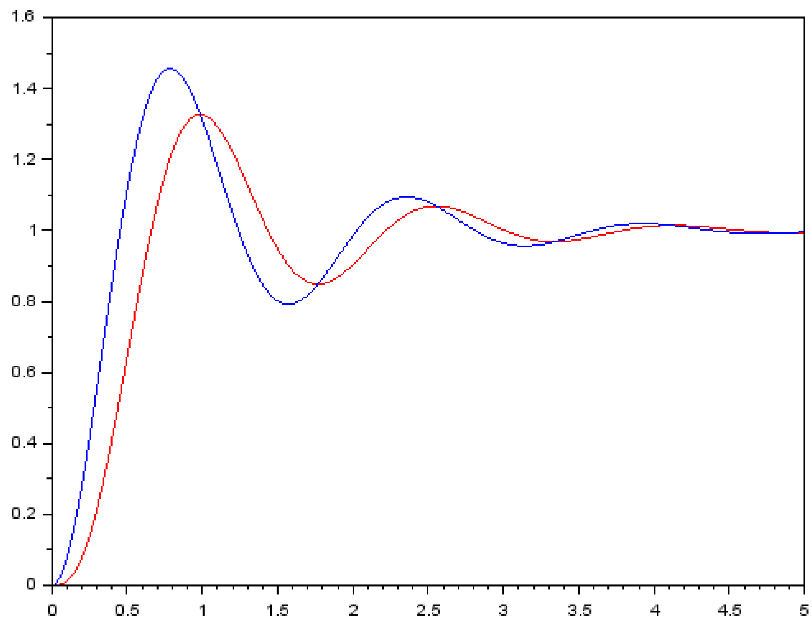

Figure 3: Red: original system, Blue: approximation

2.2 Part B

Using a similar approach as part A, we see that the system has:

$$Zero: -0.01, Poles: -0.02, -1+2j, -1-2j$$

We can approximate this by assuming that the zero cancels the pole at -0.02.

The following code was used:

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

g_temp = (s + 0.01)/(s^3 + (101/50)*s^2 + (126/25)*s + 0.1)

roots(g_temp.den)

g = syslin('c', g_temp)

t = 0:0.1:200

step_resp = csim('step', t, g)

plot(t, step_resp, 'r')

g_new_temp = (s + 0.02)/(s^3 + (101/50)*s^2 + (126/25)*s + 0.1)

g_new_temp = g_new_temp * (0.01/0.02)

g_new = syslin('c', g_new_temp)

step_resp = csim('step', t, g_new)

plot(t, step_resp, 'b')
```

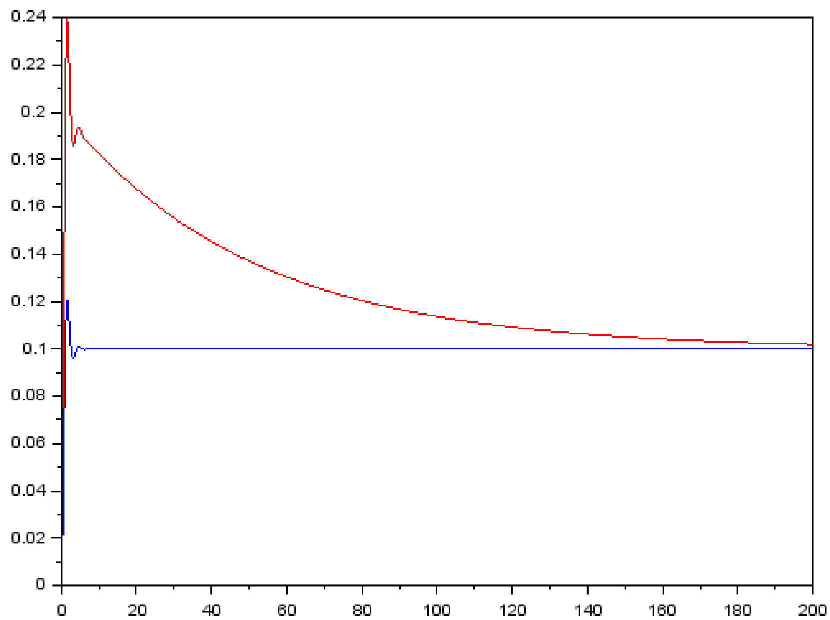

Figure 4: Red: original system, Blue: approximation

Helper functions for Questions 3 and 4

```
function rise_time = risetime(step_resp, t)

    ss_value = step_resp(length(step_resp))

    time_index = 1

    while step_resp(time_index) < 0.1*ss_value

        time_index = time_index + 1

    end

    rise_time_start = t(time_index)

    while step_resp(time_index) < 0.9*ss_value

        time_index = time_index + 1

    end

    rise_time_end = t(time_index)

    rise_time = rise_time_end - rise_time_start

endfunction


function percent_os = percentos(step_resp, t)

    ss_value = step_resp(length(step_resp))

    max_value = max(step_resp)

    percent_os = (max_value - ss_value) * 100 / ss_value

endfunction


function settling_time = settlingtime(step_resp, t)

    ss_value = step_resp(length(step_resp))

    time_index = length(step_resp)

    while (abs(resp(time_index) - ss_value) / ss_value) < 0.2
```

```
        time_index = time_index - 1
    end
    settling_time = t(time_index)
endfunction

function peak_time = peaktime(step_resp, t)
    max_value = max(step_resp)
    time_index = 1
    while step_resp(time_index) < max_value
        time_index = time_index + 1
    end
    peak_time = t(time_index)
endfunction
```

3 Question 3

3.1 Part A

Code:

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

g_temp = 9/(s^2 + 2*s + 9)

g = syslin('c', g_temp)

trfmod(g, 'f') // see w_n, zeta

roots(g_temp.den) // poles: -1. + 2.8284271i, -1. - 2.8284271i


g_new_temp = (s + 1) * g_temp // zero at -1

g_new = syslin('c', g_new_temp)


t = 0:0.01:5

step_resp = csim('step', t, g)

step_resp_new = csim('step', t, g_new)


plot(t, step_resp, 'r')

plot(t, step_resp_new, 'b')


disp(risetime(step_resp, t))

disp(risetime(step_resp_new, t))


disp(percentos(step_resp, t))

disp(percentos(step_resp_new, t))
```

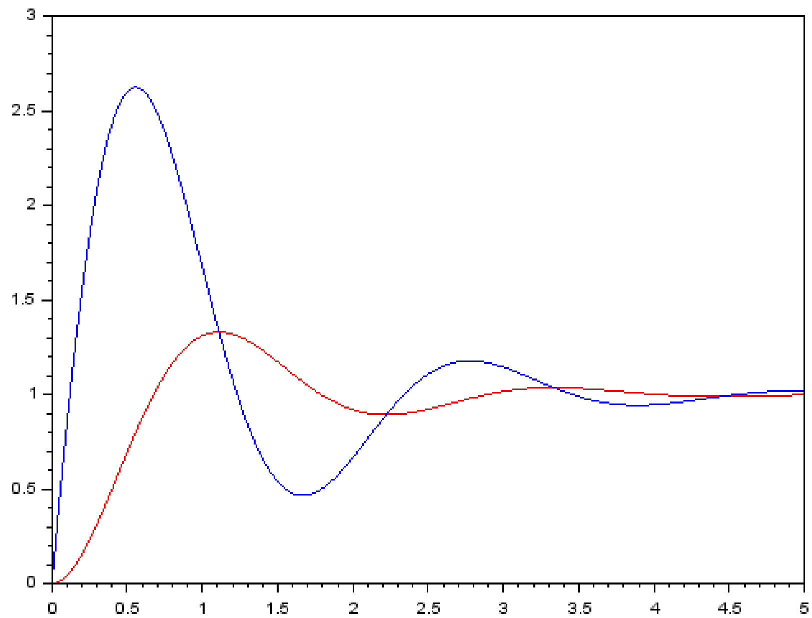

Figure 5: Red: original system, Blue: with zero

Risetime goes from 0.46s to 0.09s, Percent OS goes from 33% to 157%

3.2 Part B

Code:

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

g_temp = 9/(s^2 + 2*s + 9)

g = syslin('c', g_temp)

trfmod(g, 'f') // see w_n, zeta

roots(g_temp.den) // poles: -1. + 2.8284271i, -1. - 2.8284271i


g_new_temp1 = g_temp * 0.5 / (s + 0.5) // extra pole closer to origin

g_new1 = syslin('c', g_new_temp1)

g_new_temp2 = g_temp * 3 / (s + 3) // extra pole away from origin

g_new2 = syslin('c', g_new_temp2)


t = 0:0.01:15

step_resp = csim('step', t, g)

step_resp_new1 = csim('step', t, g_new1)

step_resp_new2 = csim('step', t, g_new2)


plot(t, step_resp, 'r')

plot(t, step_resp_new1, 'g')

plot(t, step_resp_new2, 'b')


disp(risetime(step_resp, t))

disp(risetime(step_resp_new1, t))

disp(risetime(step_resp_new2, t))
```

```
disp(percentos(step_resp, t))
disp(percentos(step_resp_new1, t))
disp(percentos(step_resp_new2, t))
```

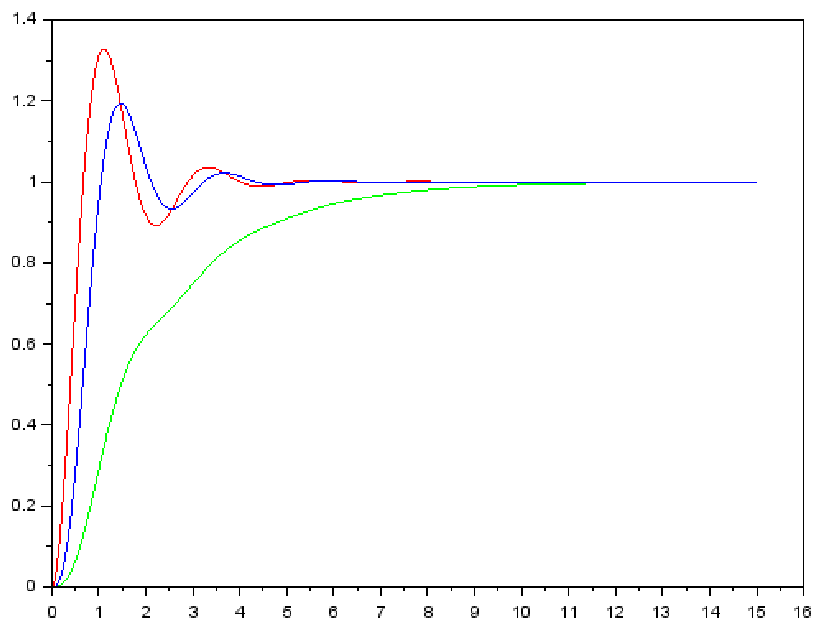

Figure 6: Red: original system, Green: pole near origin, Blue: pole away from origin

Risetimes are 0.46, 4.18, 0.63, Percent OS are 33%, 0%, 19%

3.3 Part C

On addition of a zero: rise time decreases, percent OS increases.

On addition of a pole:

If the pole is closer to the origin, percent OS becomes 0, otherwise it decreases

If the pole is closer to the origin, the rise time increases a lot, otherwise it increases only a little.

4 Question 4

Code:

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

zetas = 0:0.7:1.4

for zeta = zetas

    g_temp = 1/(s^2 + 2*zeta*s + 1)

    g = syslin('c', g_temp)

    t = 0:0.01:30

    step_resp = csim('step', t, g)

    disp(zeta, "rise time", risetime(step_resp, t))

    disp(zeta, "percent OS", percentos(step_resp, t))

    disp(zeta, "settling time", settlingtime(step_resp, t))

    disp(zeta, "peak time", peaktime(step_resp, t))

end
```

Parameter: $\zeta = 0, 0.7, 1.4$

Rise Time: 0.82, 2.13, 5.37

Percent OS: 100, 4.599, 0

Settling Time: -, 2.23, 4.29

Peak Time: 3.14, 4.4, -