EE324 Control Systems Lab

Problem Sheet 5

Sheel Shah — 19D070052

August 28, 2021

1 Question 1

1.1 Part A code

```
s = poly(0, 's')
g_temp =  10 / (s^3 + 4*s^2 + 5*s + 10)
// g_temp = g/(1+g)
// g_temp + g.g_temp = g
g = g_temp / (1 - g_temp)
g = syslin('c', g)
evans(g)
```

1.2 Part B code

```
s = poly(0, 's')
g =  (s + 1) / (s^2*(s + 3.6))
g = syslin('c', g)
evans(g)
```

1.3 Part C code

```
s = poly(0, 's')
g =  (s + 0.4) / (s^2*(s + 3.6))
g = syslin('c', g)
evans(g)
```

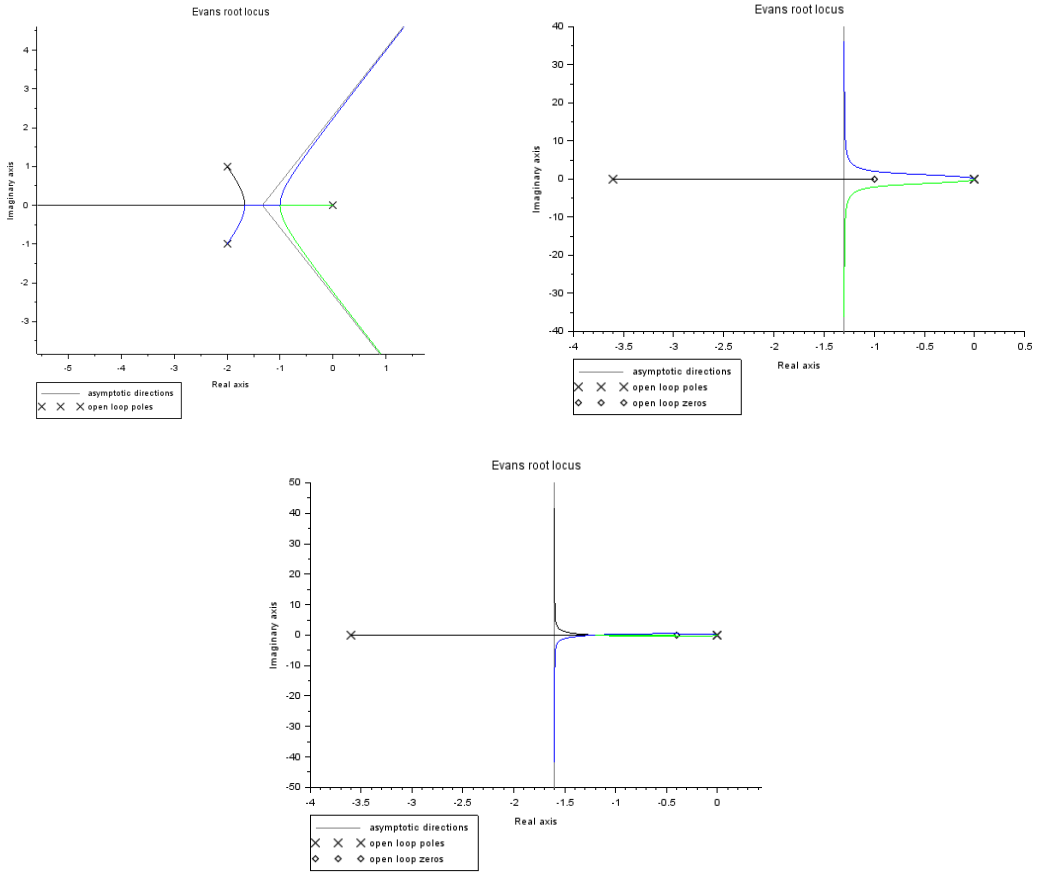

Figure 1: Answers to Part A, B, C

1.4 Part D

Code:

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


for p = -3:0.1:3
    g = (s + p) / (s * (s + 1) * (s + 2))
    poles = roots(simp(g).den)
    scatter(real(poles), imag(poles) + p)
end
```

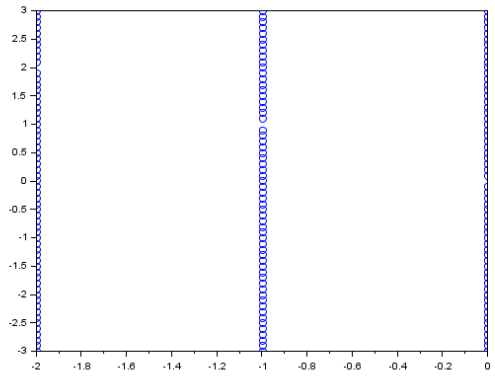

Figure 2: Scatter plot of the poles. Y-axis is the values of p

On close inspection we see that the poles get cancelled at p=0/1/2. The system is stable only for p=0.

2 Question 2

2.1 Parts A, B, C

General code used:

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

g =  // try functions here

g = syslin('c', g)

evans(g)
```

Answers:

Part A: $(s * s)/((s-1) * (s-2) * (s+1) * (s+2))$

Part B: $1/((s+1)^5)$

Part C: $1/((s+1)^3)$

2.2 Part D

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

// step 1:

// g =  1 / ((s - 1) * (s + 1) * (s - 2) * (s + 2))


// step 2

// g = 1 / (4 - 5 * s^2 + s^4)

// g = 1 / (4 + 5 * s^2 + s^4)


// step 3

g = 1 / (4 + 5 * (s - 1)^2 + (s - 1)^4)

g = syslin('c', g)

evans(g)
```

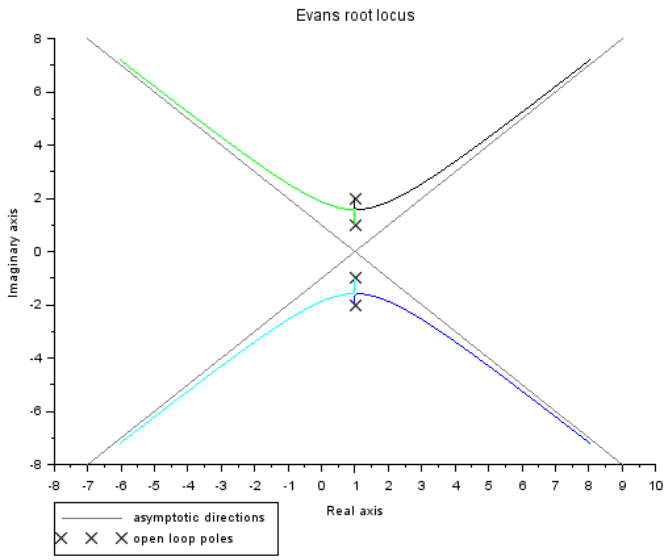

Figure 3: Root locus for part D

3 Question 3

The approach was to vary Kp and find the risetime in a brute force manner, whilst ensuring that the system was stable.

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


function cl_tf_k = cltfk(K)
    // use as tf = cltf(kp)
    G = 1 / (s*(s^2 + 3*s +5))
    cl_tf_k = K * G / (1 + K * G)
endfunction


function rise_time = risetime(step_resp, t)
    ss_value = mean(step_resp(1*0.75: 1))
    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
```

```
for kp = 0.1:0.05:20
    G = cltfk(kp)
    t = 0: 0.01: 100
    G_cont = syslin("c", G)
    step_resp = csim("step", t, G_cont)
    plot(t, step_resp)
    // ensure stability
    l = length(step_resp)
    initial_peak = max(step_resp(1:l/4))
    last_peak = max(step_resp(l*0.75:l))
    if initial_peak > mean(step_resp(l*0.75: l))
        if last_peak > initial_peak
            disp("unstable", kp)
            break
        end
    end
    disp(risetime(step_resp, t))
end
```

The minimum possible rise time is 0.57 seconds, and is attained at Kp = 15.

4 Question 4

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


function cl_tf_k = cltfk(K)
    // use as tf = cltf(kp)
    G = 0.11 * (s + 0.6) / (6 * s^2 + 3.6127 * s + 0.0572)
    cl_tf_k = K * G / (1 + K * G)
endfunction


// root locus
scf(0)
G = 0.11 * (s + 0.6) / (6 * s^2 + 3.6127 * s + 0.0572)
G_cont = syslin("c", G)
evans(G_cont)


// step response
scf(1)
G = cltfk(100)
t = 0: 0.01: 10
G_cont = syslin("c", G)
step_resp = csim("step", t, G_cont)
plot(t, step_resp)
disp(step_resp(length(step_resp)))


// marginal stability
```

```
scf(2)
for kp = 0.8:0.01:0.87
    G = cltfk(-kp)
    t = 0: 10: 10000
    G_cont = syslin("c", G)
    step_resp = csim("step", t, G_cont)
    plot(t, step_resp)
end
```

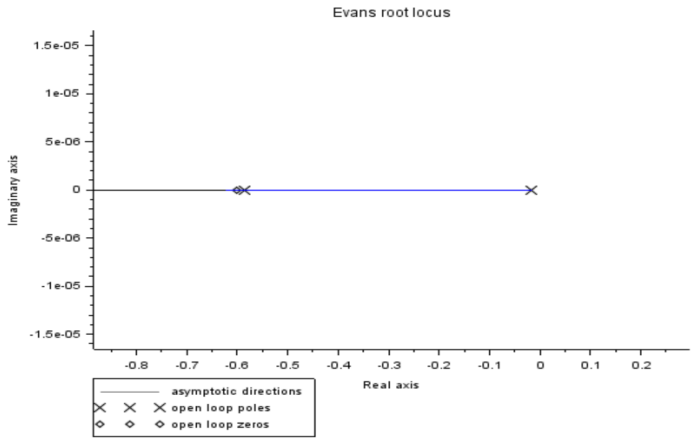

Figure 4: Root locus

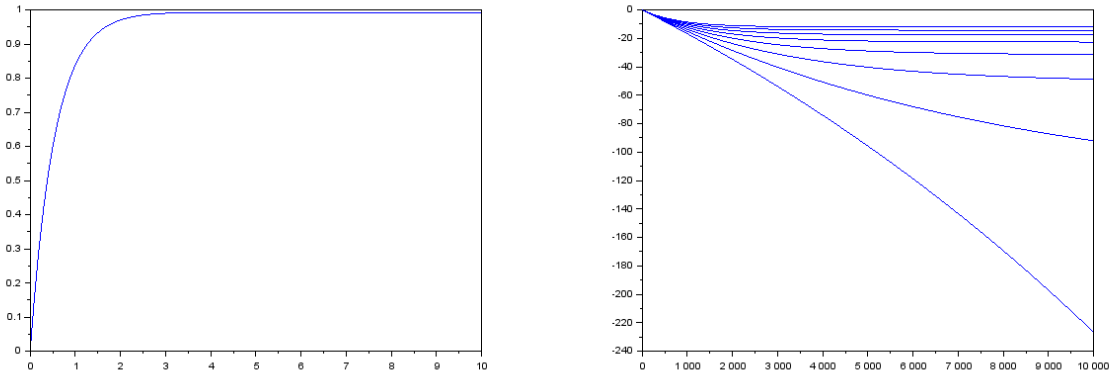

Figure 5: Step Response, Marginal stability is around Kp = -0.86

5 Question 5

Code:

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


function cl_tf_k = cltfk(K, G)
    cl_tf_k = K * G / (1 + K * G)
endfunction


g1 = 1 / ((s + 1) * (s + 2) * (s + 50))
g1 = syslin("c", g1)


g2 = 1 / ((s + 1) * (s + 2))
g2 = syslin("c", g2)


scf(0)
evans(g1)
scf(1)
evans(g2)


for k = 1: 1: 100
    g2_new = cltfk(k, g2)
    approximate_poles = roots(g2_new.den)
    g1_new = cltfk(k, g1)
    actual_poles = roots(g1_new.den)
    difference = abs(
```

```
        actual_poles(2) - approximate_poles(1)
    ) / abs(actual_poles(1))
    disp(difference, k)
end
```

Allowing a relative difference of upto 0.1, we get the max Kp as 32.

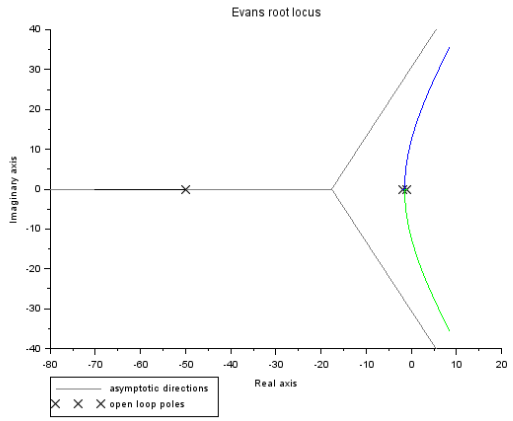

Figure 6: Root Loci of the original system

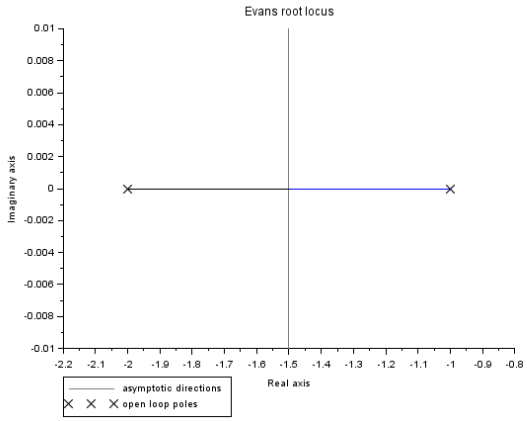

Figure 7: Root Loci of the approximated system