Computer Exercices

Automatique et commande numérique
ME: 326

https://github.com/KookaS/Control-Systems

Olivier Charrez
SCIPER: 271550
Quentin Rossier
SCIPER: 262706
Achraf Burrada
SCIPER: 287137

January 10, 2021

Contents

1 Analysis of Feedback Control Systems

1.1 Define a transfer function

$$G(s) = \frac{Ks*KG*Km*Ka}{(JBr*s^2+Ks)*(Rm*Jm*s^2+Rm*b*s+Km^2*KG^2*s)+Rm*Ks*JBr*s^2}$$

1.2 Time and frequency domain analysis

Give the transfer function of the flexible joint G(s) in zpk format :

We choose Dc = 1.2

```
G =

                58474
  --------------------------------
  s (s+39.09) (s^2 + 11.05s + 416)


Continuous-time transfer function.
```

1.3 Analysis of the feedback systems

1.3.1 Write the transfer function S between r and e as a function of Dc and G and give its numerical values in zpk format:

$$S = \frac{E}{R} = \frac{1}{1 + D_c G}$$

```
S =

       s (s+39.09) (s^2 + 11.05s + 416)
  ----------------------------------------
  (s+37.79) (s+5.403) (s^2 + 6.953s + 343.7)

Continuous-time transfer function.
```

1.3.2 Write the transfer function U between r and u as a function of Dc and G and give its numerical values in zpk format.

$$U = \frac{U}{R} = \frac{Dc}{1 + D_c G}$$

```
U =

     1.2 s (s+39.09) (s^2 + 11.05s + 416)
  ----------------------------------------
  (s+37.79) (s+5.403) (s^2 + 6.953s + 343.7)

Continuous-time transfer function.
```

1.3.3 Write the transfer function T between r and y as a function of Dc and G and give its numerical values in zpk format.

$$T = \frac{Y}{R} = \frac{D_c G}{1 + D_c G}$$

```
T =

                  70169
  ----------------------------------------
  (s+37.79) (s+5.403) (s^2 + 6.953s + 343.7)
Continuous-time transfer function.
```

1.3.4 Write the transfer function V between w and y as a function of Dc and G and give its numerical values in zpk format.

$$V = \frac{Y}{W} = \frac{G}{1 + D_c G}$$

```
V =

                  58474
  ----------------------------------------
  (s+37.79) (s+5.403) (s^2 + 6.953s + 343.7)
Continuous-time transfer function.
```

1.3.a Plot the step responses of the closed-loop system from reference signal to the output, from reference signal to the control signal (plant input), from reference signal to the tracking error signal (the input of the controller) and from disturbance signal to the output. All plots in one figure with appropriate scale (Use a 2 by 2 subplot).

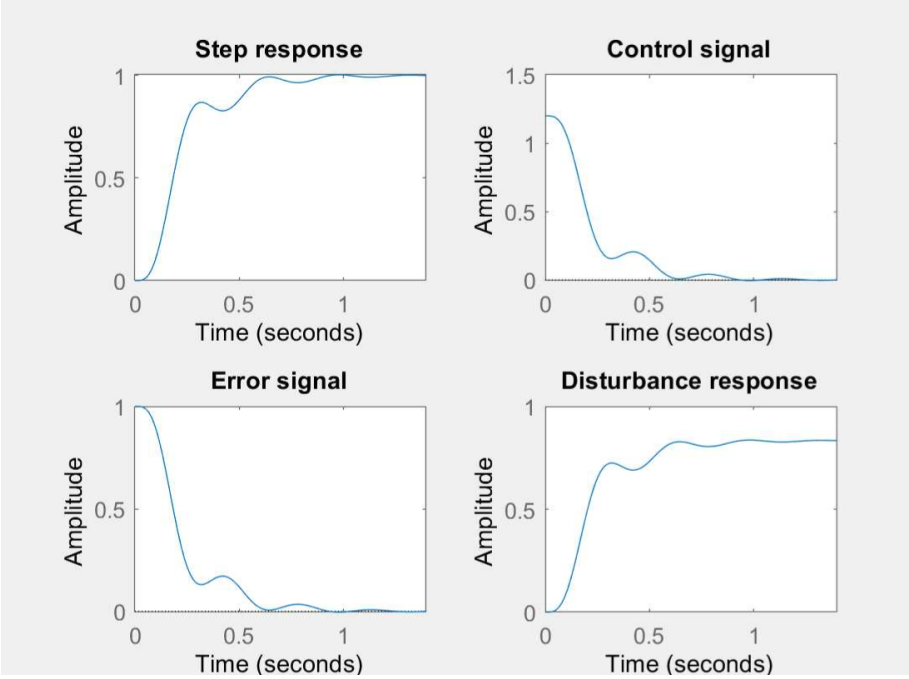

Figure 1: Step responses of the closed-loop systems

1.3.b Give the closed-loop poles.

```
ans =

 -37.7898 + 0.0000i
  -3.4767 +18.2093i
  -3.4767 -18.2093i
  -5.4030 + 0.0000i
```

1.3.c Is the closed-loop system stable? Why?

Yes, because all the poles are LHP

1.4 Computing the ultimate gain

Compute the ultimate gain using the plot of rlocus command of Matlab. Validate your result using the Routh stability criterion.

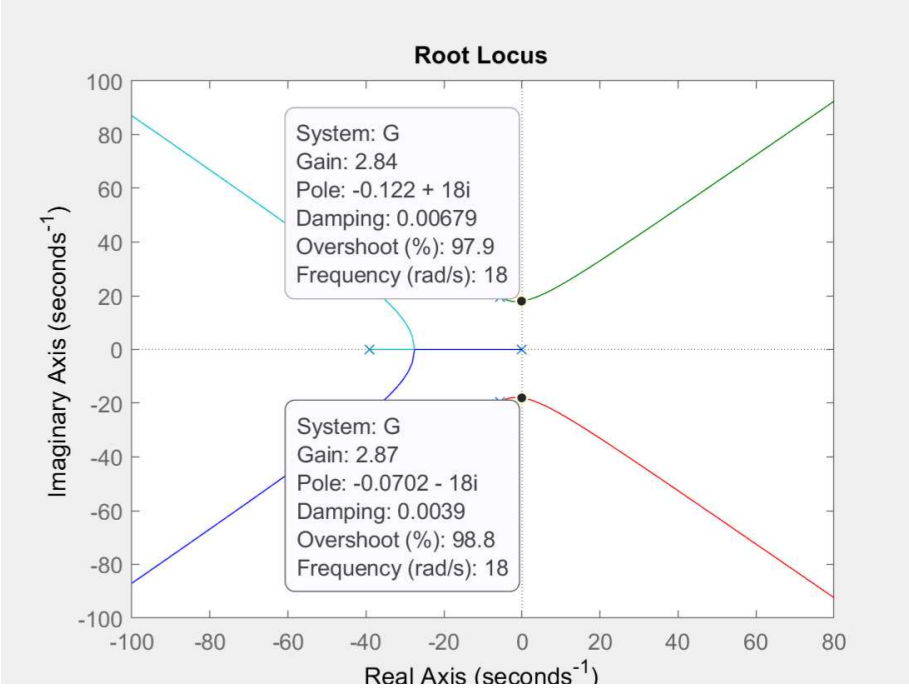

Figure 2: Root locus of G(s)

We need to look at the characteristic equation

$$1 + Dc * G = s^4 + 50.145s^3 + 847.945s^2 + 16261.45 + 5847 * k$$

Using the Routh stability criterion we found that

$$K_u = 2.9$$

witch is confirmed by the root locus

We now choose $D_c = 0.6 * K_u = 1.74$

1.5 Closed-loop step response analysis

Plot the step response of the closed-loop system (between r and y). Print the rise-time, settling time and the overshoot (use stepinfo). Compute the closed-loop bandwidth (use bandwidth). Plot the magnitude Bode diagram of the closed-loop transfer function (use bodemag) and check the correctness of the bandwidth.

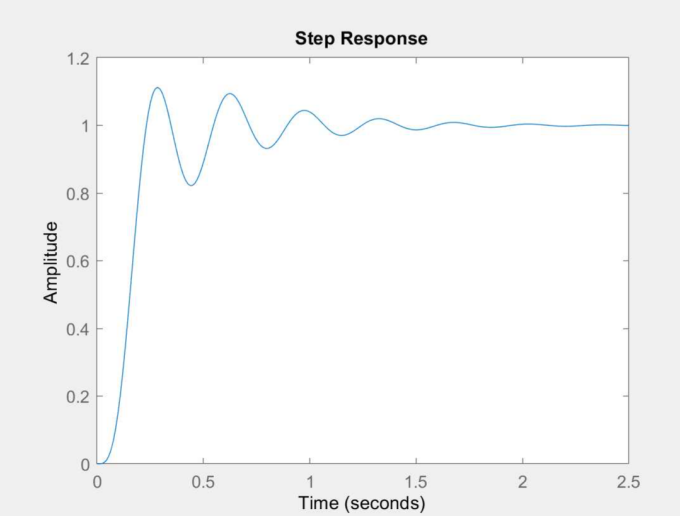

Figure 3: Step response of the closed-loop system

```
ans =

  struct with fields:

        RiseTime: 0.1262
    SettlingTime: 1.1973
     SettlingMin: 0.8222
     SettlingMax: 1.1125
       Overshoot: 11.2483
      Undershoot: 0
            Peak: 1.1125
        PeakTime: 0.2843

bandwidth =

   20.9584
```

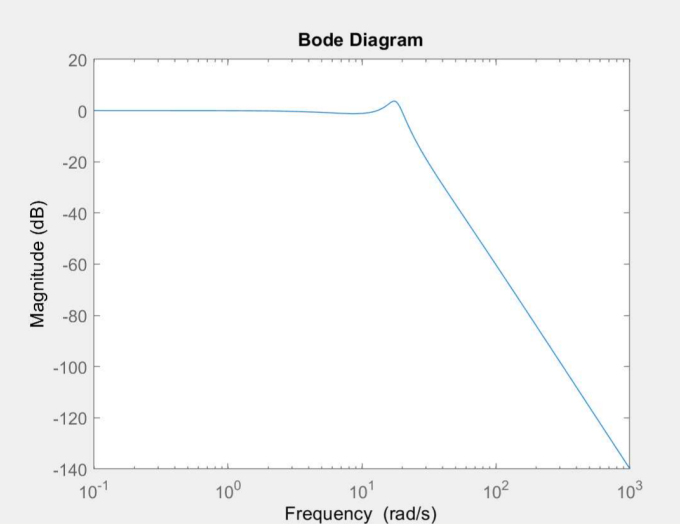

Figure 4: :Bode diagram of the closed-loop system

2 PID Controller Design

2.1 ZN First method

2.1.1 Plot the step response of G from 0 to 0.5 s together with the asymptote (or the tangent with the largest slope) for the first method of ZN.

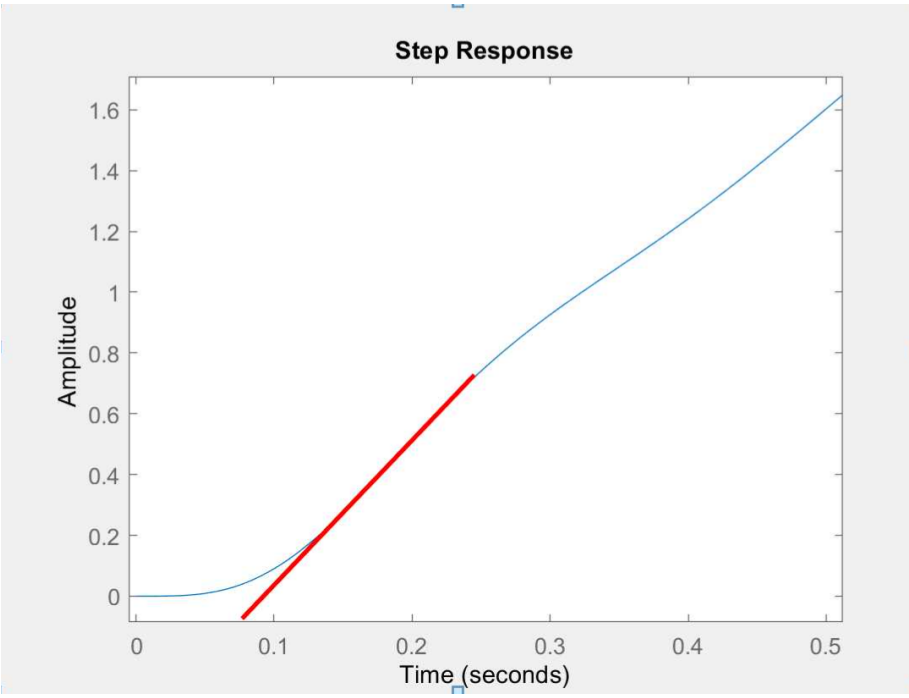

Figure 5: Step response of G with a tangent

2.1.2 Give the value of L and R:
We can approximate L and R with

$$L = 0.09$$

$$R = 4.7$$

2.1.3 Give the parameters of the PID controller:

Kp = 1.2/R*L = 2.83
Ti = 2*L = 0.18
Td = 0.5*L = 0.045

2.1.4 Is the closed-loop system stable with this PID controller?

Yes, the step response of the system converge

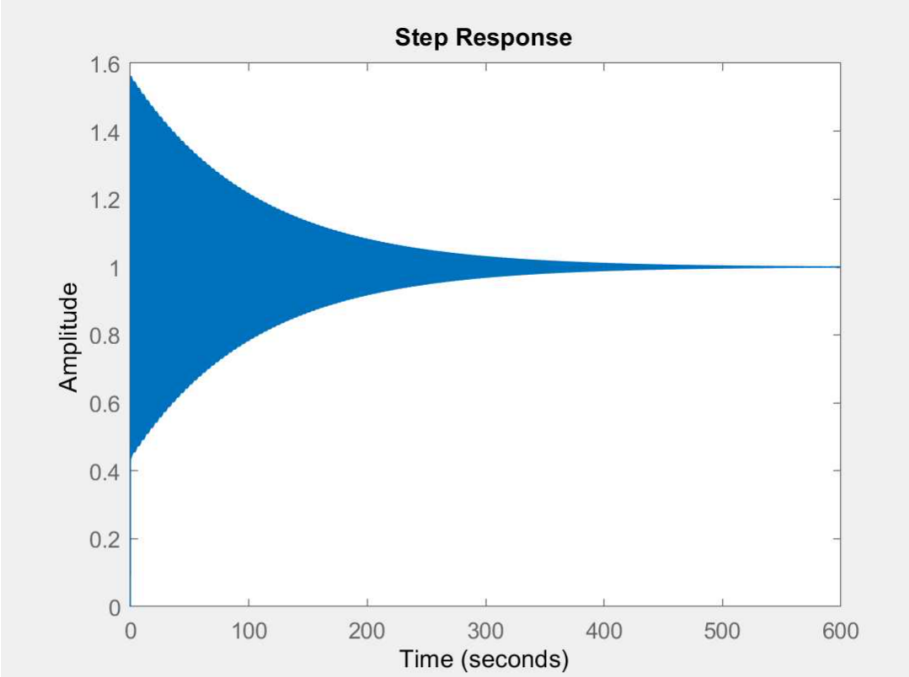

Figure 6: stable step response

2.2 ZN Second method

2.2.1 Give the value of the ultimate gain and the ultimate period:

The ultimate gain is 2.9 and the ultimate period 0.35

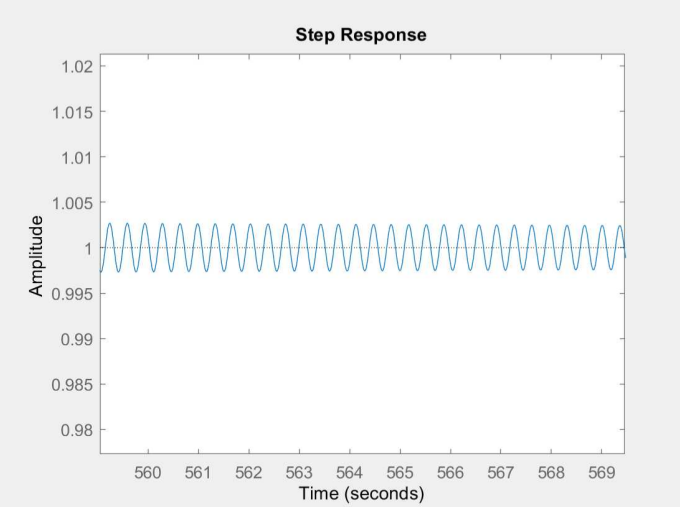

Figure 7: Step response of the closed loop system

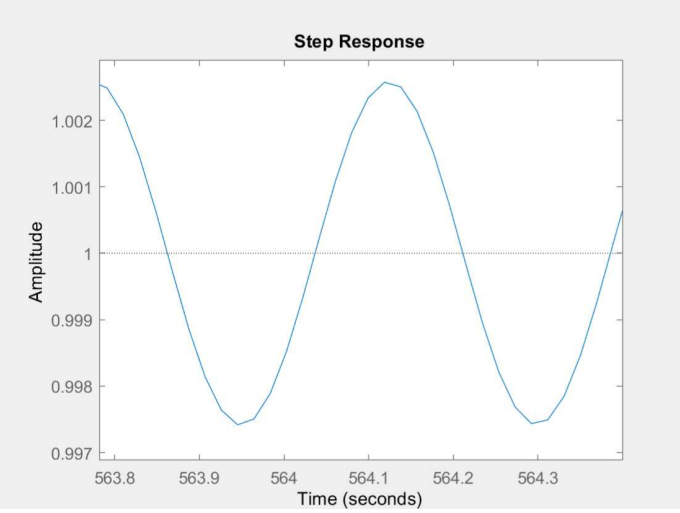

Figure 8: Step response of the closed loop system

2.2.2 Give the parameters of the PID controller:

For a PID controller we have:
Kp = 0.6*Ku = 1.7
Ti = 0.5*Pu = 0.175
Td = 0.125*Pu = 0.043

2.2.3 Plot the step response of the closed-loop system from the reference signal to the output:

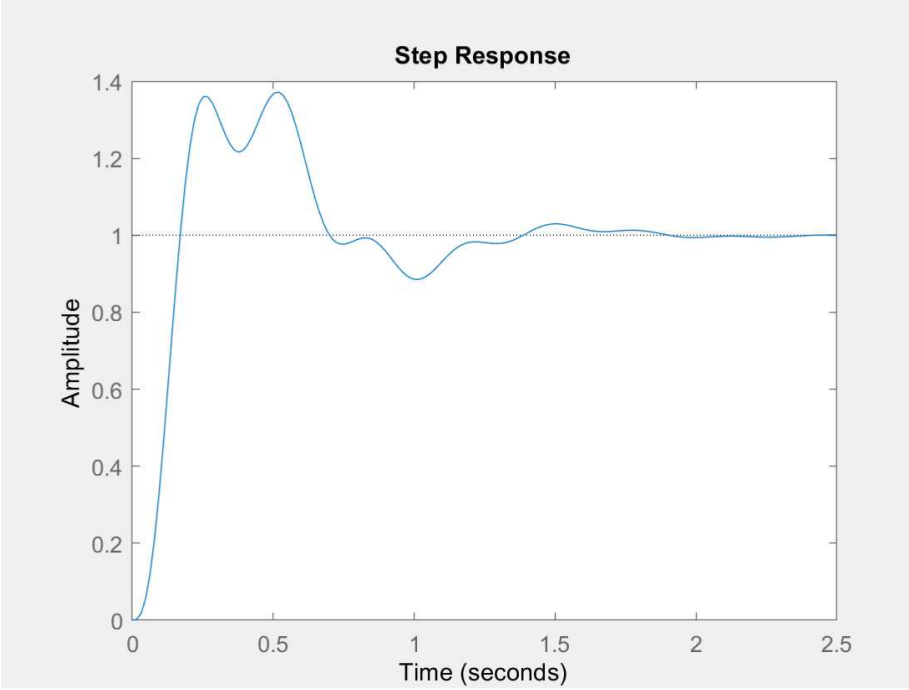

Figure 9: Step response of the closed loop system

2.2.4 Plot the step response of the closed-loop system from the disturbance signal to the output:

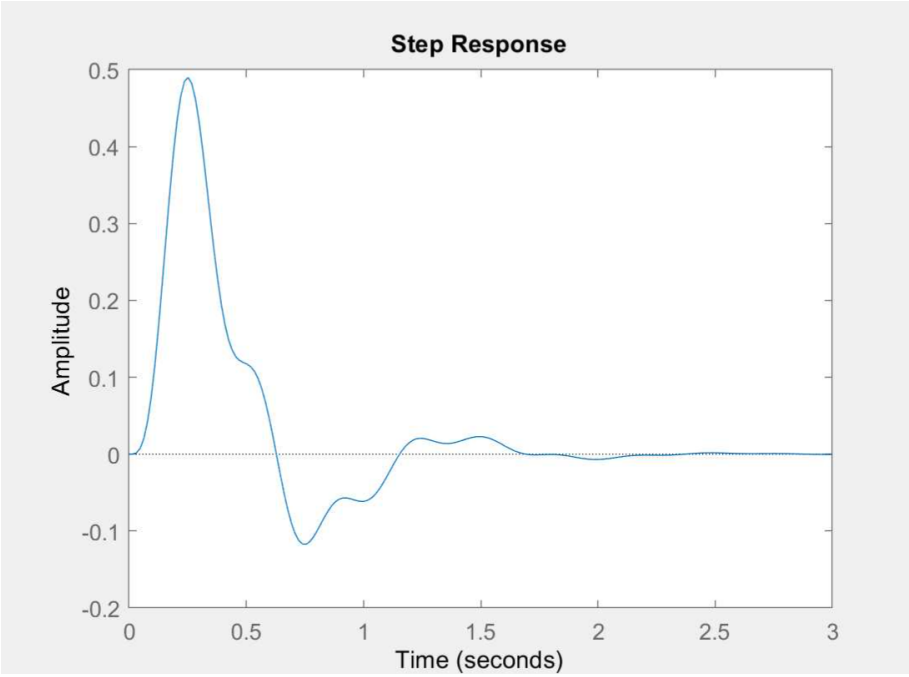

Figure 10: Step response of the closed loop system from the disturbance signal to the output

2.3 Cascade Controller

2.3.1 Plot the step response of G_1 and give the step information. From the step response, estimate the steady-state gain, the damping factor and the natural frequency of an approximate second-order model.

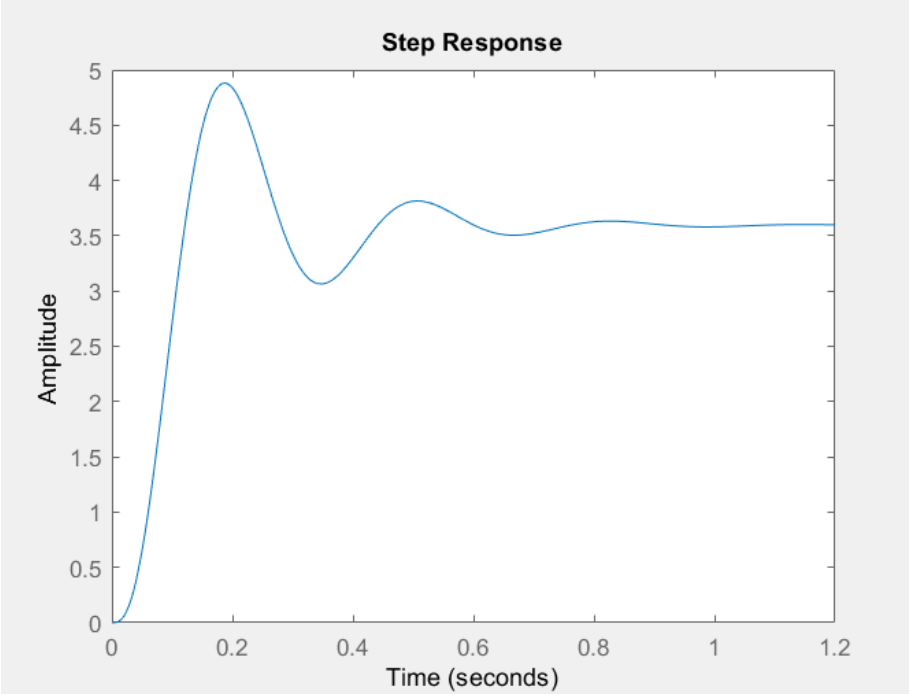

Figure 11: Step response of G_1

From the figure 11 we can deduce the following characteristics:
steady-state gain:

$K = 3.6$
$\gamma = K/\alpha = 3.6$
overshoot: M_p =(4.85-3.6)/3.6=0.347
t_p=0.19
$\zeta = \sqrt{\frac{(lnMp)^2}{(\pi^2+(lnMp)^2)}} = 0.32$
$\omega_n = \frac{\pi}{tp*\sqrt{(1-\zeta^2)}} = 17.45$

2.3.2 Give the reference model for the inner loop. Give the PID controller parameters of the inner loop.

We choose a first order model for the MRC method. M(s) and τ_m denotes the inner controller here:

$M(s) = \frac{1}{\tau_m * s+1}$
$\tau_m = \frac{1}{BW} = \frac{1}{50} = 0.02$
$D'_c(s) = \frac{s^2+2\omega_n\zeta s+\omega_n^2}{\gamma*\tau_m*s*\omega_n^2} = \frac{s^2+11.17s+304.5}{21.92s}$
$k_P = \frac{2\zeta}{\gamma*\omega_n*\tau_m} = 0.4867$
$k_I = \frac{1}{\gamma*\tau_m} = 13.9$
$k_D = \frac{1}{\gamma*\omega_n^2*\tau_m} = 0.044$

2.3.3 Give the reference model for the outer loop. Give the proportional controller for the outer loop.

M(s) and τ_m denotes the outer controller here:

$\tau_m = \frac{1}{BW} = \frac{1}{5} = 0.2$

$D_c(s) = \frac{M(s)}{\frac{1}{s}*(1-M(s))} = K_P = 5$

2.3.4 Give the transfer function between R and Y in terms of Dc, Dc' and G. Give also the transfer function with its numerical values in zpk format:

$\frac{Y}{R} = \frac{D_c D_c' G_1}{s(1+D_c' G_1)+D_c D_c' G_1}$

```
ans =

                13335 (s+39.09)^2 (s^2 + 11.17s + 304.5)
  ------------------------------------------------------------------
  (s+39.09)^2 (s+5.546) (s^2 + 12.44s + 283.3) (s^2 + 32.16s + 2585)

Continuous-time zero/pole/gain model.
```

2.3.5 Give the transfer function between W and Y in terms of Dc, Dc' and G. Give also the transfer function with its numerical values in zpk format
$\frac{Y}{W} = \frac{G_1}{s+G_1 D_c'+G_1 D_c' D_c}$

```
ans =

                          58474 s (s+39.09)^2 (s^2 + 11.05s + 416)^2
  ------------------------------------------------------------------------------------------
  (s+46.17) (s+39.09)^2 (s^2 + 3.901s + 263.1) (s^2 + 11.05s + 416)^2 (s^2 + 0.0768s + 401.1)

Continuous-time zero/pole/gain model.
```

2.3.6 Plot the closed-loop output for tracking a step reference signal for the cascade controller and the ZN controller in the same figure. Plot the closed-loop output for the rejection of a step disturbance for the cascade controller and the ZN controller in the same figure

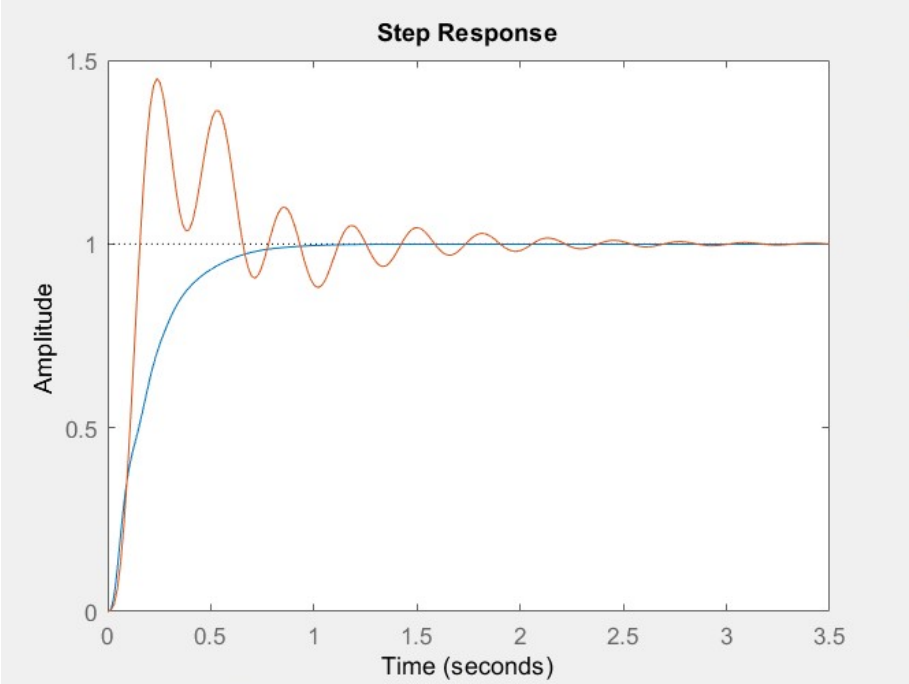

Figure 12: Input step response of the closed loop system with a controller | ZN model (orange) and Cascade controller (blue)

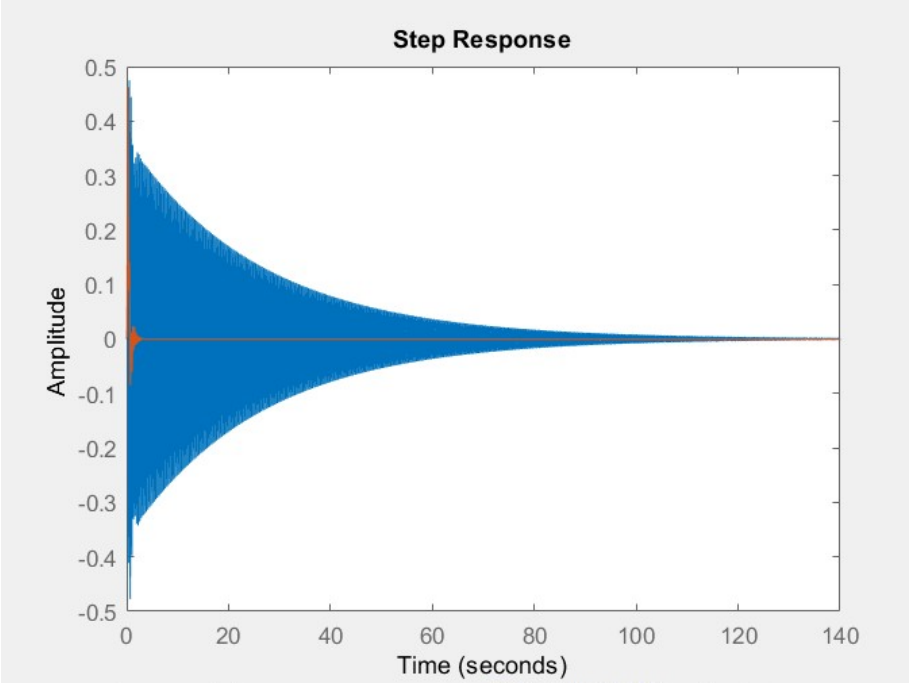

Figure 13: Disturbance step response of the closed loop system with a controller | ZN model (orange) and Cascade controller (blue)

3 Loop Shaping Method

3.1 Proportional controller

Give the value of kP and explain how you compute it from the Bode diagram of G. Give the values of gain margin and phase margin using the Bode diagram of the open-loop transfer function. Check your results using the command margin of Matlab.

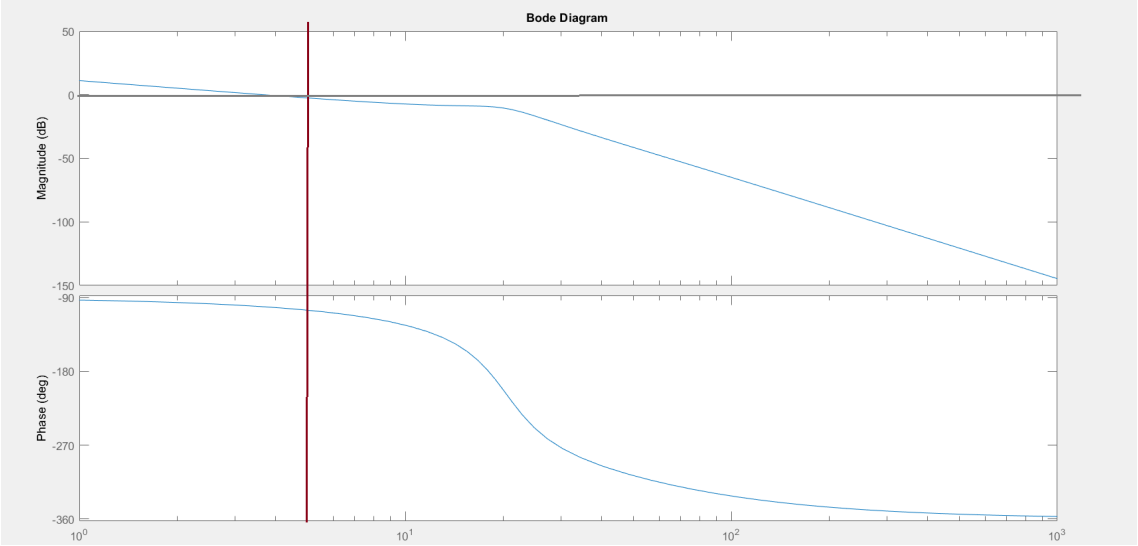

Figure 14: Bode of G

Gain at 5 [rad/s] = -2.5 [dB]
We need to compensate a Gain =2.5 db to reach the ωc =5 rad/s
We find $k_P = 10^{\frac{Gain}{20}}$ then $k_P = 1.33$
Gm=6.79 [dB]
Pm=75 [°]

As PM doesn't change with a constant gain, we take it directly from the diagram at ωc =5 rad/s

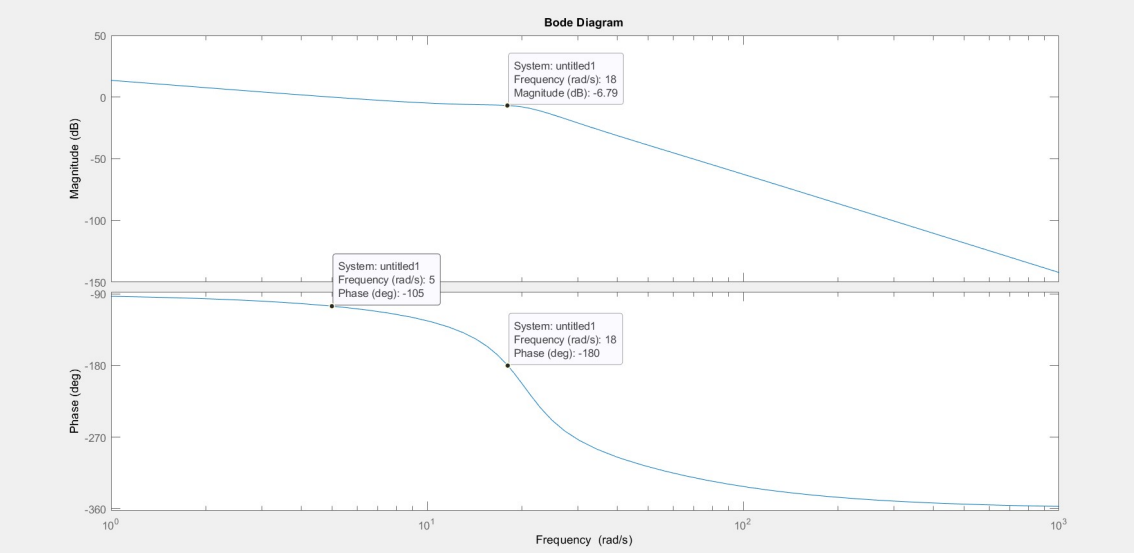

Figure 15: Bode and Phase Diagram of the System $k_P * G$

3.2 Lead-Lag controller

3.2.1 Give the controller and explain in details how did you compute it.

We add an integrator to reject a step disturbance input

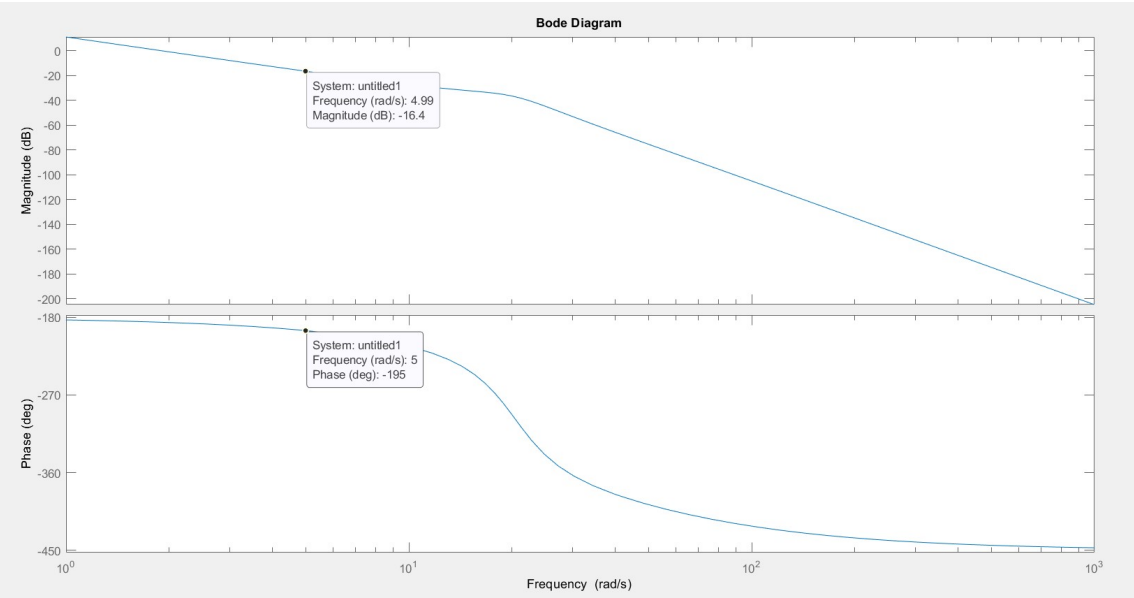

Figure 16: Bode Diagram of G(s)/s

We need to compensate 16.4 [dB] in magnitude and 70 [degrees] in Phase
So

$$\sqrt{(}c) = 6.6$$

$$c = 43.65$$

$$P = tan(70°) = 2.75$$

By resolving the equation $\alpha = 32.9$ et $\tau = 0,04$
We have

$$D_c(s) = \frac{1 + 1.316s}{s(1 + 0.04s)}$$

3.2.2 Check your results (closed-loop bandwidth, phase margin).

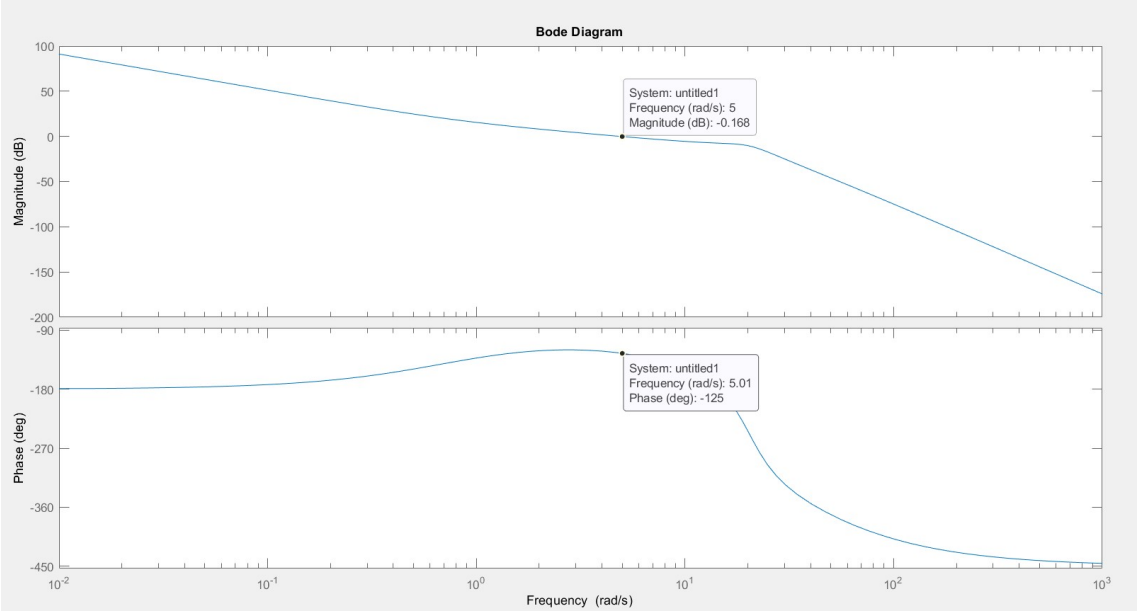

Figure 17: Bode Diagram of Dc(s)*G(s)

It's clear $\omega_c = 5[rad/s]$ and $PM = 55[°]$

3.2.3 Compute the modulus margin from the magnitude Bode diagram of the closed-loop sensitivity function.

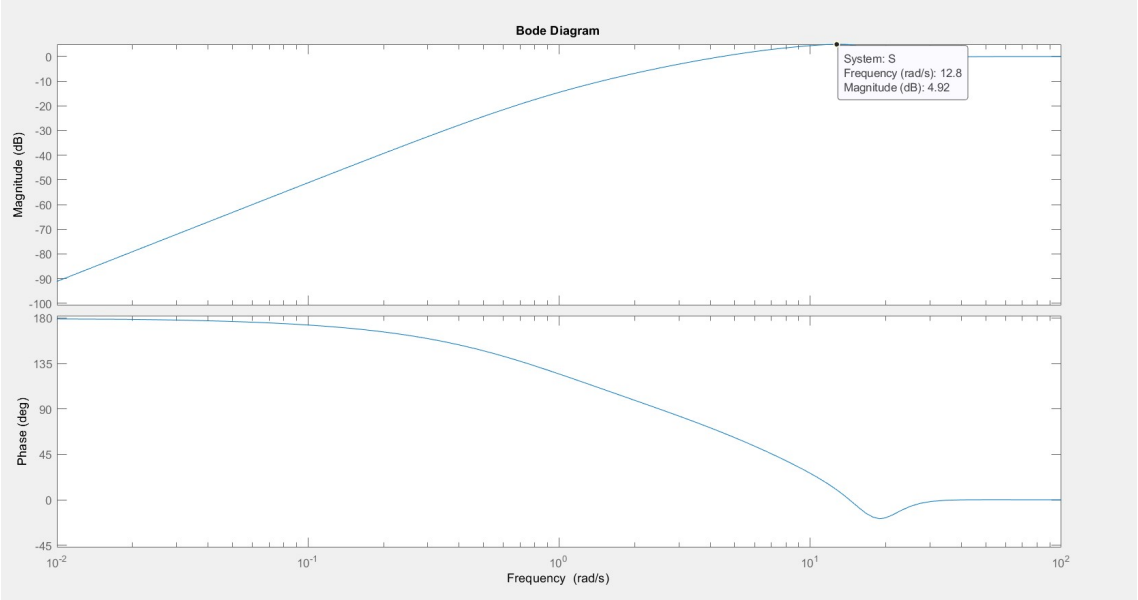

Figure 18: Bode Diagram of S=1/(1+G*Dc)

1/m [dB] = 4.92
Then m = 0.57

3.3 Comparison with cascade controller

3.3.1 Compare the tracking step response of the lead-lag controller and the cascade controller (plot both responses in the same figure).

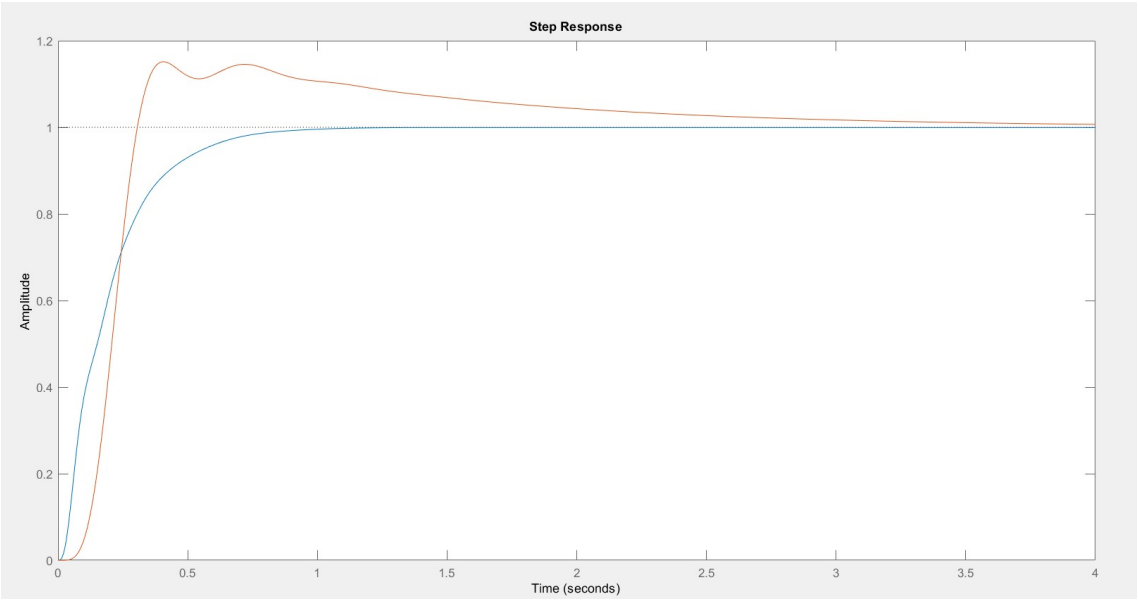

Figure 19: Input step response of Lead-Lag(orange) and Cascade(bleu)

3.3.2 Compare the disturbance step response of the lead-lag controller and the cascade controller (plot both responses in the same figure).

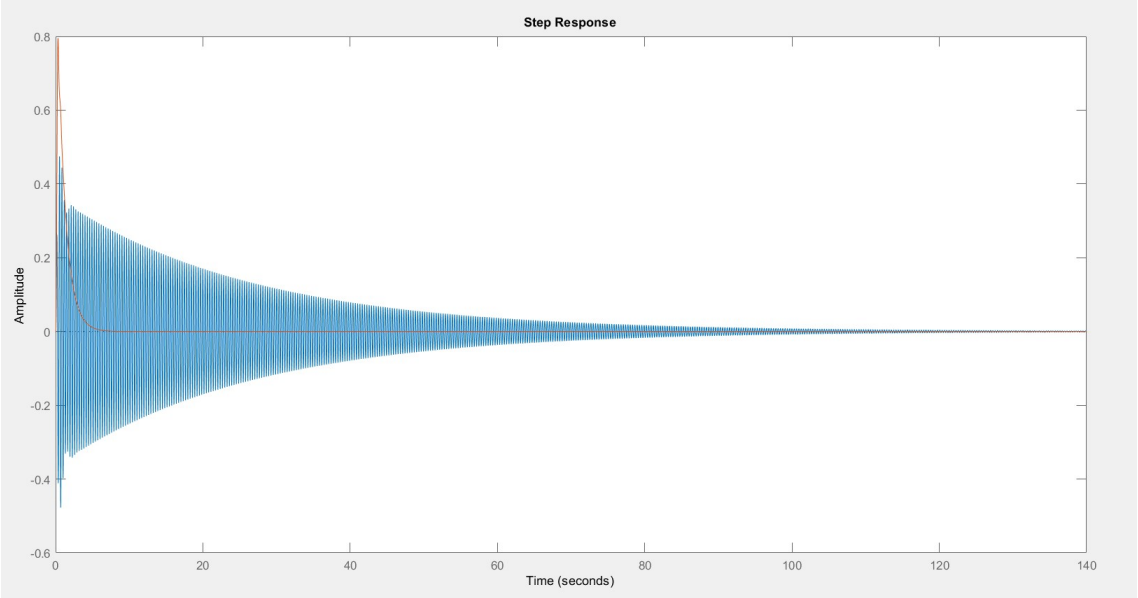

Figure 20: Disturbance step response of Lead-Lag(orange) and Cascade(bleu)

4 State-Space Method

4.1 State-space model of the flexible joint

1. Compute the state-space model of the system (A, B, C, D)

$$\ddot{\theta}(-K_G^2 J_{mot} - J_{mod} - J_{Br}) = \dot{\theta}(\frac{K_G^2 K_m^2}{R_m} + b) + u\frac{K_G K_m K_a}{R_m} + \ddot{\alpha} J_{Br}$$

$$J_{Br}(\ddot{\alpha} + \ddot{\theta}) + K_s \alpha = 0$$

These formulas give:

$$\ddot{\theta} = \dot{\theta}\frac{-K_G^2 K_m^2 - bR_m}{(K_G^2 J_{mot} + J_{mod})R_m} + u\frac{K_G K_m K_a}{(K_G^2 J_{mot} + J_{mod})R_m} + \alpha\frac{K_s}{K_G^2 J_{mot} + J_{mod}}$$

$$\ddot{\alpha} = -\dot{\theta}\frac{-K_G^2 K_m^2 - bR_m}{(K_G^2 J_{mot} + J_{mod})R_m} - u\frac{K_G K_m K_a}{(K_G^2 J_{mot} + J_{mod})R_m} - \alpha(\frac{K_s}{K_G^2 J_{mot} + J_{mod}} + \frac{K_s}{J_{Br}})$$

The state space matrices can be deduced:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

$$A = \begin{bmatrix} \frac{-K_G^2 K_m^2 - bR_m}{(K_G^2 J_{mot} + J_{mod})R_m} & 0 & 0 & \frac{K_s}{(K_G^2 J_{mot} + J_{mod})} \\ 1 & 0 & 0 & 0 \\ \frac{K_G^2 K_m^2 + bR_m}{(K_G^2 J_{mot} + J_{mod})R_m} & 0 & 0 & -(\frac{K_s}{K_G^2 J_{mot} + J_{mod}} + \frac{K_s}{J_{Br}}) \\ 0 & 0 & 1 & 0 \end{bmatrix} x = \begin{bmatrix} \dot{\theta} \\ \theta \\ \dot{\alpha} \\ \alpha \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{K_G K_m K_a}{(K_G^2 J_{mot} + J_{mod})R_m} \\ 0 \\ -\frac{K_G K_m K_a}{(K_G^2 J_{mot} + J_{mod})R_m} \\ 0 \end{bmatrix} C = \begin{bmatrix} 0 & 1 & 0 & 1 \end{bmatrix} D = 0$$

2. Is the system controllable?

   $\begin{bmatrix} B & AB & ... & A^{n-1}B \end{bmatrix}$ is nonsingular so yes it is controllable:

```
Co = ctrb(A, B)
>> Co =

   1.0e+07 *

    0.0000   -0.0009    0.0359   -1.3263
```

```
         0    0.0000   -0.0009    0.0359
   -0.0000    0.0009   -0.0300    1.0330
         0   -0.0000    0.0009   -0.0300
```

3. Is the system observable?

$\begin{bmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{bmatrix}$ is non singular so yes it is observable:

```
Ob = obsv(A, C)
>> Ob =

         0    1.0000         0    1.0000
    1.0000         0    1.0000         0
         0         0         0 -324.3243
         0         0 -324.3243         0
```

4.2 State-space controller design

1. Compute a state feedback control using the LQR method.

   The values below are selected to have a decent response time arbitrarily. Q is made to match some matrix proportion and R has been chosen to have a response time of approximately 5 seconds with an amplitude of 1e-03. If we take a higher R then the amplitude might be higher but at the cost of a really slow controller, vice-versa if we take a lower R. This value seemed the most correct one for this application.

```
Q = C^T C;
R = 1e-2;
K = lqr(A, B, Q, R)
>> K =

    0.7113   10.0000    0.5566    0.0828
```

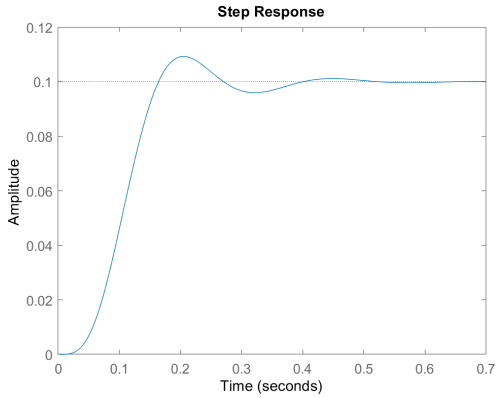

Figure 21: Step response with LQR

2. Design a state estimator using the pole placement technique (use acker command). Note that the poles of the estimator (observer) should be faster than the dominant control poles.

   All states are measurable and the state-space model is controllable.
   The estimator equation:

   $\dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x})$

   We select estimator poles 10 times higher than the control poles found above, we don't want the estimator poles slower than the controller poles otherwise they will be the dominant ones and will shape the response of the system. By doing so we can obtain the estimator gain:

```
sys_lqr = ss(A-B*K, B, C-D*K, D);
Pc = pole(sys_lqr);
Pe = Pc * 10;
L=acker(A.',C.',Pe).'
>> L =

   1.0e+07 *

    1.0534
    0.0142
   -1.0307
   -0.0141
```

3. Compute the feed-forward gain $\bar{N}$ to have zero steady-state error for a step reference signal.

   In order to have no steady-state error, the reference need to be scaled by just the right amount with a feed-forward gain:

$$\bar{N} = [C(-A + BK)^{-1}B]^{-1}$$

```
N = inv(C * inv(-A + B.*K) * B)
>> N =

   10.0000
```

Here is the control law with the reference normalization:

$u = K\hat{x} + \bar{N}r$

Put them all together and we obtain the following steady-state matrices were the state variables and estimated state variables are combined:

$$A_{cl} = \begin{bmatrix} A & -BK \\ LC & A - BK - LC \end{bmatrix} B_{cl} = \begin{bmatrix} B\bar{N} \\ B\bar{N} \end{bmatrix} C_{cl} = \begin{bmatrix} C & 0 \end{bmatrix}$$

We can see on figure 22 that indeed there are no error at the steady-state compared to the unit step input. This method is very nice to remove the steady-state error but is not robust in case of changes in the parameters of the system.

4. Compute the transfer function between the reference signal r and the output y.

   A state space can be easily transformed to a transfer function:

```
sys_est = ss(Acl, Bcl, Ccl, D);
Tyr = tf(sys_est)
>> Tyr =
```

```
                5.847e05 s^4 + 4.563e08 s^3 + 1.541e11 s^2 + 3.383e13 s + 3.419e15
-----------------------------------------------------------------------------------------------------------
s^8 + 858.5 s^7 + 3.272e05 s^6 + 8.055e07 s^5 + 1.11e10 s^4 + 6.246e11 s^3 + 1.892e13 s^2 + 3.721e14 s + 3.419e15
```

5. Compute the transfer function between the reference signal r and the input u.

   To do so we use a litle trick where we use the inital transfer function Y/U with the new transfer function Y/R to obtain U/R, or we can tweak the state space to compute this output:

```
Ccl = [zeros(size(C)), -K];
sys_est = ss(Acl, Bcl, Ccl, D);
Tur = tf(sys_est)
>> Tur =
```

```
      -279 s^7 - 2.356e05 s^6 - 8.791e07 s^5 - 2.118e10 s^4 - 2.78e12 s^3 - 1.302e14 s^2 - 2.77e15 s - 3.419e16
-----------------------------------------------------------------------------------------------------------
s^8 + 858.5 s^7 + 3.272e05 s^6 + 8.055e07 s^5 + 1.11e10 s^4 + 6.246e11 s^3 + 1.892e13 s^2 + 3.721e14 s + 3.419e15
```

6. Plot the output y(t) and the input u(t) when a unit step signal is applied to r(t).

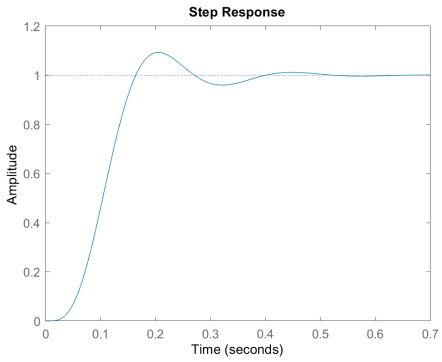

Figure 22: Step response of y(t) for step signal to r(t)

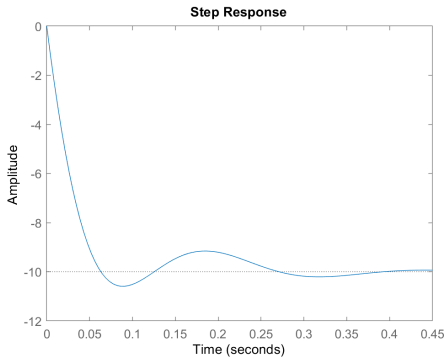

Figure 23: Step response of u(t) for step signal to r(t)

4.3 State-space controller with integrator

1. Find the augmented state-space model of the plant.

$$\bar{A} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \bar{B} = \begin{bmatrix} B \\ 0 \end{bmatrix} \bar{C} = \begin{bmatrix} C & 0 \end{bmatrix}$$

2. Design a state feedback controller by the pole placement method.

   With the given transfer function we can easily find the poles to place. The other poles need to be faster in order to not disturb the steady state response. We took arbitrarily 2, 4 and 6 times faster poles.

```
H = tf([1, 2*0.8*10, 10^2], 1);
Pi = tzero(H);
m = real(max(Pi));
Pbar = [Pi; 2*m; 4*m; 6*m];
K=acker(Abar,Bbar,Pbar)
```

   On figure 24 we have the following response of the closed loop system. First of all we see some overshoot that may happen because of the integrator. Also we notice that the steady-state response goes to 0 and that's due to the fact that the integrator corrects the state until equilibrium. This method is more robust than using an input normalization and just needs a measurable output.

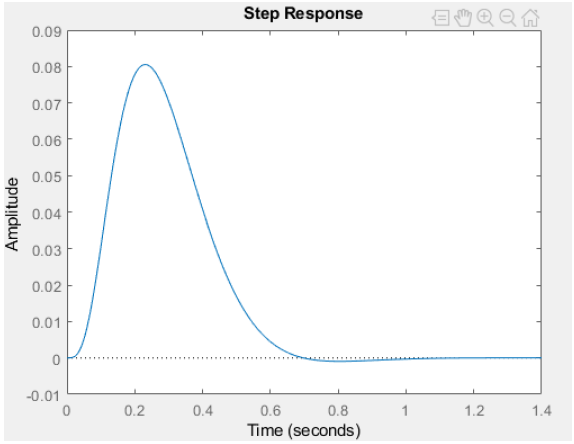

Figure 24: Step response with Integrator and state feedback

3. Design a state estimator by the pole placement technique. Note that only the states of the plant model are estimated.

   In here we implement a state estimator on top the the integrator and feedback control. The poles of the estimator are higher than the control loop poles because we don't want it to influence too much system.

```
L= acker(A',C',[10*Pi; 20*m; 40*m]).'
```

4. Compute the transfer function between r and y.

   $(1)\dot{x} = Ax + Bu$
   $(2)\dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x})$
   $(3)\dot{x_I} = r - y$
   $(4)u = -K_0\hat{x} - K_1 xI$
   $(5)y = Cx$

   These equations gives us the following system:

```
K0 = K(1:4);
K1 = K(5);
```

$$A_{ie} = \begin{bmatrix} A & -BK_0 & -BK_1 \\ LC & A - BK_0 - LC & -BK_1 \\ -C & \mathbf{0} & 0 \end{bmatrix} \quad B_{ie} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ 1 \end{bmatrix} \quad C_{ie} = \begin{bmatrix} C & \mathbf{0} & 0 \end{bmatrix}$$

```
sys_ie = ss(Aie, Bie, Cie, D);
Hyr = tf(sys_ie)
```

```
Hyr =

                2.458e06 s^4 + 1.573e09 s^3 + 3.391e11 s^2 + 3.193e13 s + 1.258e15

-----------------------------------------------------------------------------------------------------

s^9 + 752 s^8 + 2.141e05 s^7 + 3.138e07 s^6 + 2.633e09 s^5 + 1.266e11 s^4 + 3.404e12 s^3 + 4.967e13 s^2

                                                                    + 3.774e14 s + 1.258e15
```

5. Compute the transfer function between the reference signal r and the input u.

```
Cie = [zeros(size(C)), -K];
sys_ie = ss(Aie, Bie, Cie, D);
Hur = tf(sys_ie)
```

```
Hur =

42.03 s^8 + 2.901e04 s^7 + 7.185e06 s^6 + 8.604e08 s^5 + 5.426e10 s^4 + 1.637e12 s^3 + 2.713e13 s^2

                                                                    + 3.5e14 s - 109.9

-----------------------------------------------------------------------------------------------------

s^9 + 752 s^8 + 2.141e05 s^7 + 3.138e07 s^6 + 2.633e09 s^5 + 1.266e11 s^4 + 3.404e12 s^3 + 4.967e13 s^2

                                                                    + 3.774e14 s + 1.258e15
```

6. Plot the control signal u(t) and the output y(t) when a unit step signal is applied to the reference.

   On the response of figure 25 we can see that the tracking is accurate and converges toward the step input without residual error. On figure 26 we can see the influence of the fast poles and that their effect diminishes quickly.

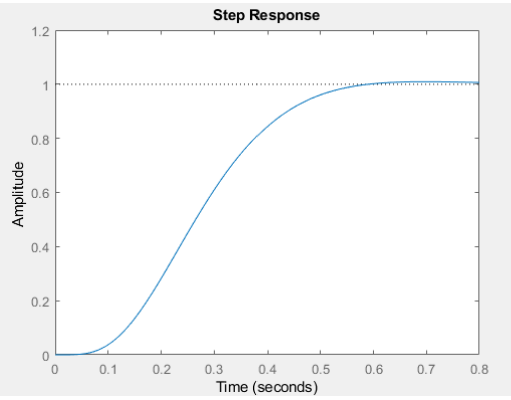

Figure 25: Step response of y(t) for step signal to r(t)

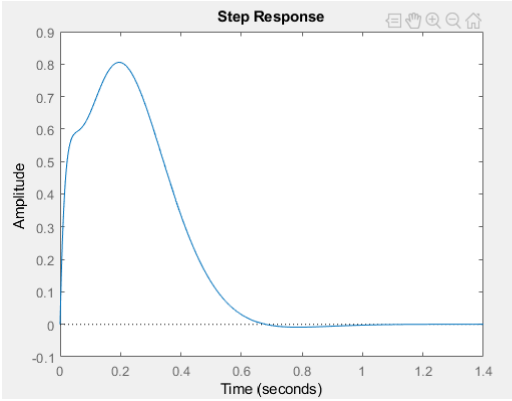

Figure 26: Step response of u(t) for step signal to r(t)

5 Digital Control

5.1 Discrete-time model

1. Assume that a closed-loop bandwidth of 10 rad/s is desired for the position control of the flexible joint system. Find an appropriate sampling period Ts for the closed-loop output.

   10 rad/s is equal to $\frac{10}{2*\pi}$ [1/s] or [Hz]. To obtain such a frequency we need a $T_s = \frac{2*\pi}{10}$. With the Shannon theorem we need to take a sampling time 2 times lower to avoid aliasing.

   $$T_s = \frac{2 * \pi}{10 * 2} = 0.31416[s]$$

   In order to have a good controller the sampling frequency should be 20-50 times higher than the closed loop frequency. Let's take a ratio of 40:

   $$T_s = \frac{2 * \pi}{10 * 40} = 0.0157[s]$$

2. Compute a discrete time model for the system using zero-order hold, zero-pole matching and Tustin methods. Use c2d command.

   The closed loop system selected is the state space model with an LQR.

   ```
   Ts = 2*pi/10/40;
   s = tf('s');
   zoh = c2d(Gcl,Ts,'zoh');
   zpm = c2d(Gcl,Ts,'matched');
   tustin = c2d(Gcl,Ts,'tustin');
   ```

3. Compare the step response of the continuous-time system with that of discrete-time models.

   If we don't take a time sampling with the 20-50 ratio explained above then we have a rough approximation of the dynamic. Here are the different step simulation with the T_s found above.

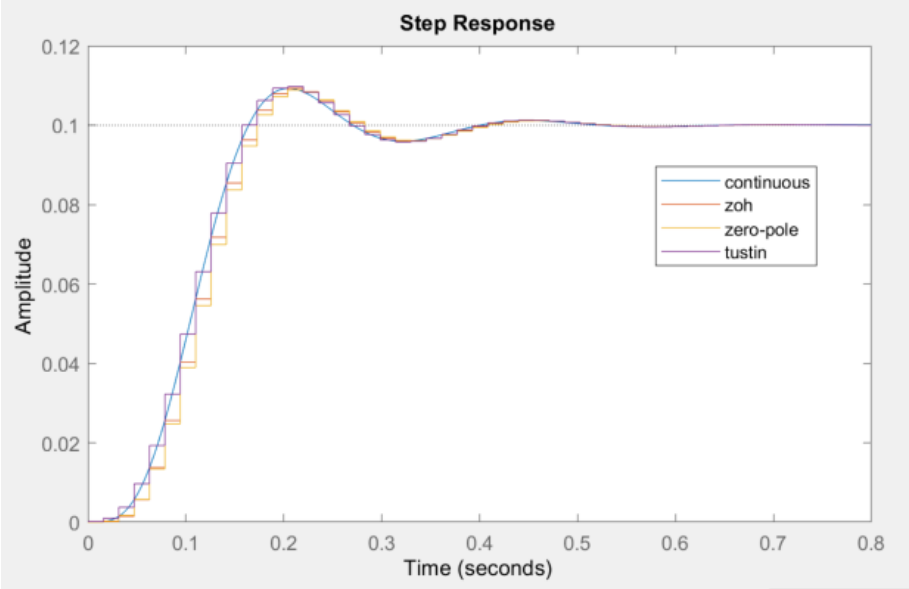

Figure 27: Step responses for discrete time models

We can see that all three responses are quite accurate, tustin being the more accurate one and zero-pole the least favorable.

4. Compare the Bode diagram of the continuous-time system with that of discrete-time models.

   Near the cutoff frequency the behaviour of the three changes quite a lot. The best approximation of the continuous model is with ZOH. Strangely the phase of the Tustin approximation is off by a lot but with a lead of 360 degrees which means a full cycle ahead. This can be observed on the figure 27 where we see that it is always above the other two approximations, which means that the signal is always one step ahead of the two others which lags behind the continuous signal.

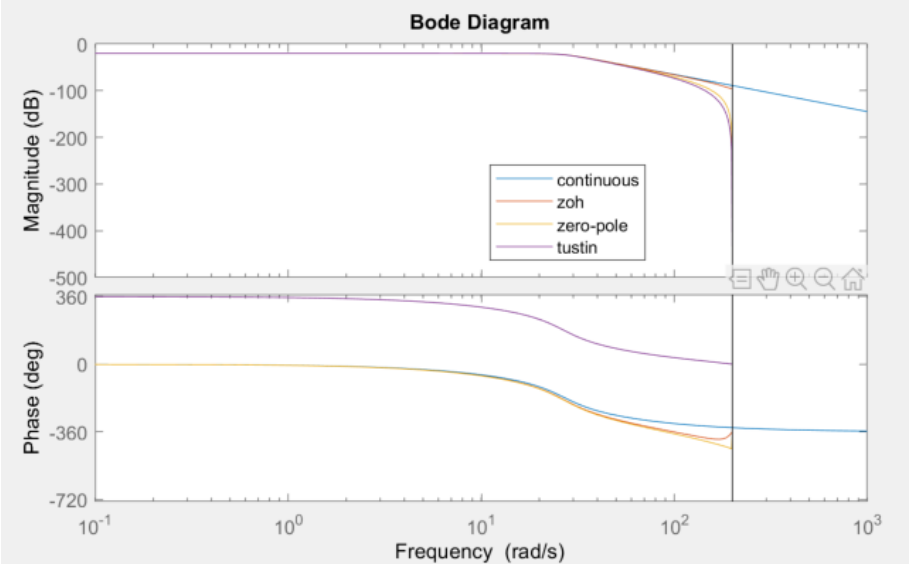

Figure 28: Bode for discrete time models

5.2 RST controller design

1. Compute the desired closed-loop polynomial.

```
% desired continuous poles
Hc = tf([1, 2*0.8*10, 10^2], 1)
Pc = zero(Hc)
z = tf('z');
% desired discrete poles
Hd = (z - exp(Pc(1)*Ts))*(z - exp(Pc(2)*Ts))
Pd = tfdata(Hd, 'v')

>> Hd =

  z^2 - 1.756 z + 0.7778

Sample time: unspecified
Discrete-time transfer function.


>> Pd =

    1.0000   -1.7560    0.7778
```

2. Use the discretized model using the zero-order hold method. Extract the coefficients of numerator and denominator in two column vectors B and A.

```
zoh = c2d(G,Ts,'zoh');
```

```
sys = tf(zoh);
[B, A] = tfdata(sys, 'v')

>> A =

    1.0000   -2.8601    3.1576   -1.5716    0.2935


>> B =

   1.0e-03 *

         0    0.1159    0.9946    0.7788    0.0556
```

3. In order to have an integrator in the controller, compute the coefficients of A'=AHs using the convolution command conv.

   An integrated is added in the controller with $H_S = 1 - q^{-1}$. An other term $H_R = 1 + q^{-1}$ is added to open the loop at the Nyquist frequency.

```
A = conv(A,[1 -1])
B = conv(B, [1 1])

>> A =

    1.0000   -3.8601    6.0177   -4.7292    1.8651   -0.2935


>> B =

         0    0.0001    0.0011    0.0018    0.0008    0.0001
```

4. Construct the Sylvester matrix and compute R and S polynomials (do not forget to put back the integrator in the controller). Compute the polynomial T.

   The Sylvester matrix depends on the size of A, B, H_R and H_R. In here the matrices A and B are already augmented with H_R and H_R:

```
nA = length(A)-1
nB = length(B)-1
d = nA - nB
nR = nA -1
nS = nB + d - 1
```

```
>> nA =

     5


>> nB =

     5


>> d =

     0


>> nR =

     4


>> nS =

     4
```

```
M = [[A';0;0;0;0] [0;A';0;0;0], [0;0;A';0;0],[0;0;0;A';0],[0;0;0;0;A'],
[B';0;0;0;0],[0;B';0;0;0], [0;0;B';0;0], [0;0;0;B';0], [0;0;0;0;B']]
```

```
>> M =

 1.0000       0       0       0       0       0       0       0       0       0
-3.8601  1.0000       0       0       0  0.0001       0       0       0       0
 6.0177 -3.8601  1.0000       0       0  0.0011  0.0001       0       0       0
-4.7292  6.0177 -3.8601  1.0000       0  0.0018  0.0011  0.0001       0       0
 1.8651 -4.7292  6.0177 -3.8601  1.0000  0.0008  0.0018  0.0011  0.0001       0
-0.2935  1.8651 -4.7292  6.0177 -3.8601  0.0001  0.0008  0.0018  0.0011  0.0001
      0 -0.2935  1.8651 -4.7292  6.0177       0  0.0001  0.0008  0.0018  0.0011
      0       0 -0.2935  1.8651 -4.7292       0       0  0.0001  0.0008  0.0018
      0       0       0 -0.2935  1.8651       0       0       0  0.0001  0.0008
      0       0       0       0 -0.2935       0       0       0       0  0.0001
```

Now that we have the Slvester matrix we can find S and R:

```
x = inv(M) * [Pd';0;0;0;0;0;0;0]   % [1; s1; s2; ...; r0; r1; ...]

S = [];
R = [];
for i = 1:nS+1
    S(i) = x(i);
end

for i = 1:nR+1
    R(i) = x(i + nS+1);
```

```
end

>> S =

    1.0000    2.0046    1.8332    0.6724    0.0427


>> R =

   1.0e+03 *

    0.8584   -2.4887    2.6847   -1.2747    0.2258
```

Because we have an integrator in the controller T has the following value:

```
T = evalfr(tf(R, 1), 1)   % with integrator in the controller

>> T =

    5.5977
```

5. Compute the closed-loop poles and compare them with the desired ones.

   The closed loop poles are found as follow:

```
P=conv(A,S)+conv(B,R)
nP = length(P)-1
if nP > nA + nB + d -1
    error('P does not fit the required size')
end

>> P =

    1.0000   -1.7560    0.7778   -0.0000    0.0000    ...
        0.0000   -0.0000    0.0000   -0.0000    0.0000

>> Pd =

    1.0000   -1.7560    0.7778
```

We can see that the generated poles of the system after tuning gives the same poles as the desired placed poles found at first. It means that the system found is correct.

6. Compute the output of the system when a unit step is applied to the input of T.

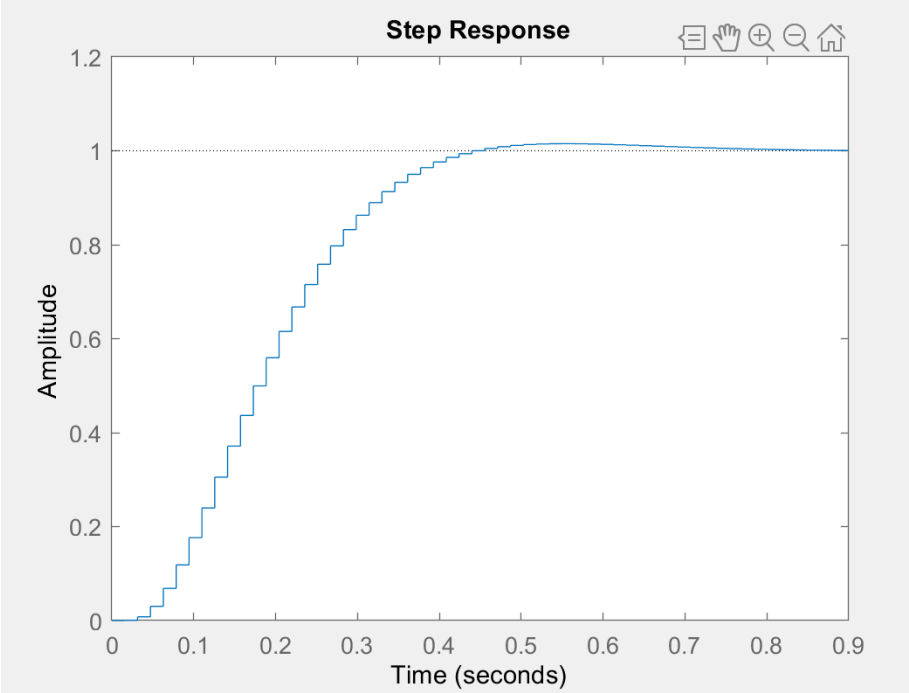

Figure 29: Step responses for RST

7. Compare this controller with the stat-space controller in terms of tracking a unit step reference.

   We can see that the state-space controller on figure 25 with an integrator is similar to the RST integrator on figure 29. The dynamics are similar expect that one is continuous and the other one is discrete.

5.3 Global comparison

Compare this controller with the loop shaping, cascade and state-space controllers in terms of the complexity of the controller structure, the clarity of the design methods, their advantages and disadvantages.

The cascade controller is very simple with a good dynamic. The MRC approach is simple for designing the controller, it just requires some desired transient performances. Furthermore each loop needs its own controller so the process can become tedious.

The loop shaping method is more difficult because it requires multiple processes to go through to identify the correct parameters. Only one controller for the plant is designed and it must satisfy some desired characteristics. It is a more complex approach than cascade but allows a more flexible controller to achieve a certain phase margin, crossover frequency or noise attenuation.

The state space controller is even more complex than the two other approaches because the whole system is expressed not by transfer functions but by state variables

and their differential equations. Once the state space is done, the manipulation of the system becomes easier to do. This approach is better when the system becomes more complex and allows multi dimensions. Designing an estimator or integrator is just applying the correct matrices and solving the equations for the correct poles.

The RST method is the only discrete controller that has been studied here. Its design is more complex to grasp than cascade or loop-sharing but is more simple to do than state-space. The characteristics of the controller rest upon the poles and zeros of the plant and some desired poles for the dynamic. The computation is straightforward and easy to do.