CE3: RST controller design

Advanced Control Systems

Martin Vuichard
Thammisetty Devakumar

Professor : Karimi Alireza

Wednesday, 27 February 2019

1 Pole placement

1 .Order of A, B, d

The first lines of code gives us from G_1

- Order of A = 4
  $A(q^{-1}) = 1 - 2.9613q^{-1} + 3.8152q^{-2} - 2.5124q^{-3} + 0.7274q^{-4}$ so order 4

- Order of B = 4
  $B(q^{-1}) = 0.1081q^{-1} - 0.0493q^{-2} - 0.0502q^{-3} + 0.0815q^{-4}$ so order 4

- d = 0

And that the sampling period is $T_s = 0.02s$

2. Script of MATLAB function for pole placement

Script used for poleplace is give below.

```
function [R,S]=poleplace(B,A,Hr,Hs,P)

    [~, check0]=find(B==0);
    numberof0=size(check0,2);
    d=numberof0-1;

    degA=size(A,2)-1;
    degB=size(B,2)-1;
    degP=size(P,2)-1;
    degHr=size(Hr,2)-1;
    degHs=size(Hs,2)-1;
    degAh=degHs+degA;
    degBh=degHr+degB;
    degRh=degAh-1;
    degSh=degBh+d-1;

    Ah=conv(A,Hs);
    Bh=conv(B,Hr);

    MA=[];
    MB=[];
    for i = 1:(degBh+d)
        before=[];
        after=[];

        for z = 1:i
            if z==1
                %do nothing
            else
                before=[before; 0];
            end
        end

        if i==(degBh+d)
            %do nothing
```

```
        else
            for w = (degAh+1):(degAh+degBh+d-i)
                after=[after; 0];
            end
        end
        MA=[MA, [before; Ah'; after]];
    end

    for j = 1:(degAh)
        before=[];
        after=[];

        for z = 1:j
            if z==1
                %do nothing
            else
                before=[before; 0];
            end
        end

        if j==(degAh+d)
            %do nothing
        else
            for w = (degBh+1):(degAh+degBh+d-j)
                after=[after; 0];
            end
        end
        MB=[MB, [before; Bh'; after]];
    end

    M=[MA, MB];
    M_1=inv(M);

    for j = 1:(degAh+degBh+d-degP-1)
        P=[P, 0];
    end
    Ptrans=P';
    x=M_1*Ptrans
    Sh=[1]
    Rh=[]
    for k = 1:degSh
        Sh = [Sh, x(k+1)];
    end
    S = conv(Sh,Hs);
    for v = 1:(degRh+1)
        Rh = [Rh, x(degSh+1+v)];
    end
    R = conv(Rh,Hr);
end
```

3. The desired damping and natural frequency and the desired closed-loop characteristic polynomial

- Desired frequency = 20.904 rad/s
- Desired damping = 0.6901
- Desired characteristic polynomial is $P(q^{-1}) = 1 - 1.4306q^{-1} + 0.5617q^{-2}$

The Diophantine equations are :
$n_R = n_A - 1 = 3$
$n_S = n_B + d - 1 = 3$
$n_P \leq n_A + n_B + d - 1$ and so $n_P \leq 7$

With the specification from question 3, one decides to use $n_P = 2$
$P(q^{-1}) = 1 + p_1 q^{-1} + p_2 q^{-2}$

The maximum overshoot is 5% so $M_p = 1.05 = 1 + e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}}$ Therefore $\zeta = \sqrt{\frac{ln(0.05)^2}{\pi^2+ln(0.05)^2}} = 0.6901$

The rise time has to be $T_r = 0.1s$. Thus $0.1 = \frac{2.16\zeta+0.6}{\omega_n}$
One gets : $\omega_n = \frac{2.16\zeta+0.6}{0.1} = 20.904 rad/s$
One can deduce p_1 and p_2 :

$$p_1 = -2e^{-\zeta\omega_n T_s} cos(\omega_n T_s \sqrt{1-\zeta^2}) = -1.4306$$
$$p_2 = e^{-2\zeta\omega_n T_s} = 0.5617$$

4. R and S Polynomials

Using the poleplace, we get the following R and S polynomials
$R(q^{-1}) = 12.4043 - 29.1102q^{-1} + 42.7814q^{-2} - 40.3897q^{-3} + 15.7686q^{-4}$
$S(q^{-1}) = 1 + 0.1896q^{-1} + 1.0674q^{-2} - 0.4897q^{-3} - 1.7673q^{-4}$

5.Achieved closed-loop poles

Achieved P using the script *conv(B,R)+conv(A,S)* is given below, same as the desired one.

```
P =     1.0000    -1.4306     0.5617     0.0000    -0.0000     0.0000
```

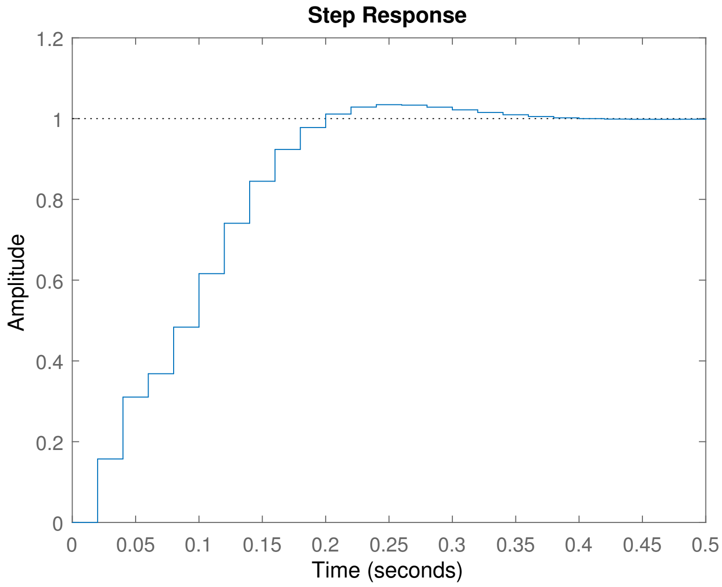
5.T polynomial

T = sum(R) = 1.4543

$T(q^{-1}) = R(1)$ because we want the same dynamics for tracking and regulation and because there is an integrator in the controller ($H_s(q^{-1}) = 1 - q^{-1}$) so $T(q^{-1}) = 1.4543$

2 Analysis of the closed-loop system

1. Plot the tracking step response. Do you obtain the exact values for the desired rise-time and overshoot? Why?



Tracking step response for G1

Results of step info is given below,

```
     RiseTime: 0.1400
 SettlingTime: 0.3200
  SettlingMin: 0.9237
  SettlingMax: 1.0345
    Overshoot: 3.4507
   Undershoot: 0
         Peak: 1.0345
     PeakTime: 0.2400
```

We do not obtain the desired response even though the characteristic polynomial is estimated to meet the specifications. For example, the rise time specification is 0.1s, however, the step response give a rise time of 0.14 which is beyond the specification. Overshoot is within the desired limit, but not exactly same. The discrepancy could be due to following two factors

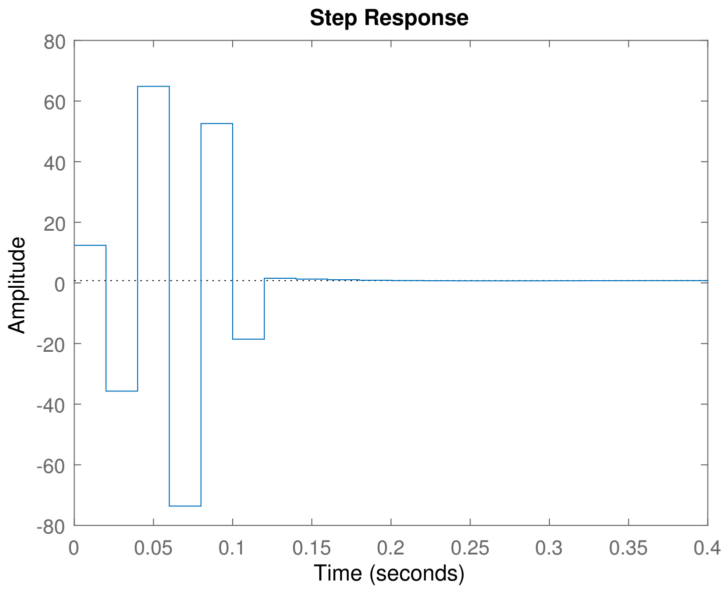
1. The formulas used in the derivation of characteristic polynomial are only approximations. This may be a reason for the discrepancy.
2. Further, we use a discrete time system step response. We see that the duration of the step is 20ms which is the sampling time. Hence, this error could also be partly due to the discretization.

2. Can this controller be implemented on the real system? Why?

No, this controller can't be implemented on real system. This is because the absolute value of the input signal is very high (-73 to +54). Further, infinity norm of the system is much higher than 20dB.

In addition, the modulus margin is only 0.1245 where as we should have atleast 0.5. For all these reasons the controller can't be implemented in real system.

Step response of the input system is given below for reference.



Step response of input sensitivity function

3. Does this controller stabilize the other plant models?

No, this controller doesn't stabilize all models. The controller is used to simulate the step response of closed loop system. We find that the models G0, G3 become unstable with this controller where as G1 and G2 are stable.

3 Robust RST controller with Q-parameterization

Explain the details of your RST controller design by Q-parameterization. Give the Matlab code and the final RST controller parameters.

Weighting filters for the performance and stability are taken for the results of Section 2, reproduced below for reference. From the question 2.2, we get the filter W_1

$$W_1(q^{-1}) = \frac{1 + 20q^{-1}}{2(1 + 0.00001q^{-1})}$$

From question 2.1, we get the filter W_2 :

$$W_2 = \frac{0.3993 - 1.117q^{-1} + 1.238q^{-2} - 0.6121q^{-3} + 0.09655q^{-4}}{1 - 2.933q^{-1} + 3.661q^{-2} - 2.246q^{-3} + 0.5848q^{-4}}$$

To meet stability and performance for all models, the following model will be minimized :

$$|||W_1 S| + |W_2 T|||_\infty$$

Since modulus margin is incorporated in W1 filter, we don't explicitly need the following constraint. However, this is also given in the constrains function for the optimization for completeness sake. which will be subject to :

$$||M_m S(Q)||_\infty < 1$$

For implementation on real system we need the infinity norm of input sensitivity function to be less than 20dB. Hence we use

$$||U(Q)||_\infty < 10$$

However, using < 10 for the input sensitivity sometimes results in input value being marginally higher than 20dB. Hence, we use a value of 7.5 in the MATLAB to ensure that the input sensitivity is < 20dB always.
In addition, to include integrator in the controller, we add an equality constraint on Q, i.e Q(1) = 0. Similarly, to open the loop at Nyquist frequency we add additional equality constraint on Q i.e Q(-1) = 0.
The solver settings for the fmincon are selected to guarantee convergence for the given problem. Also the number of permissible iterations are increased so that we get a feasible solution for the problem. Implementation of fmincon is given in detail in the MATLAB code below.

```
Ts=0.02;
G1=G{2};

B=G1.b;
A=G1.f;
W2 = tf([0.09655, -0.6121, 1.238, -1.117, 0.3993], [0.5848, -2.246, 3.661,
    -2.933, 1],Ts,'Variable','z^-1');
W1s = tf([1 20], [2 0.00002]);
W1 = c2d(W1s, Ts, 'tustin');
Hs=[1, -1];
Hr=[1, 1];
[R,S]=poleplace(B,A,Hr,Hs,P)
% Check poleplace
P=conv(B,R)+conv(A,S)
T=sum(R);
CL=tf(conv(T,B),P,Ts,'Variable','z^-1');
U=tf(conv(A,R), P,Ts,'Variable','z^-1');
bodemag(U)
step(U)
stepinfo(U)
figure()
step(CL)
stepinfo(CL)
Mm = 1/norm(Stf, Inf)

%%
Q0 = ones(1, 10);

Mm=0.5;%Modulus Margin
obj = @(Q) q33(Q, A, B, S, R, Hs, Hr, P, Ts, W1, W2, CL);
con = @(Q) constraints(Q, A, B, S, R, Hs, Hr, P, Ts, W1, W2, CL);
options = optimoptions('fmincon','Display','iter','Algorithm','active-set');
options.MaxFunctionEvaluations = 3000;
Q = fmincon(obj, Q0, [], [], [], [], [], [], con, options);

temp = conv(A, Q);
Rq = [R, zeros(1, length(temp) - length(R))] + temp;

```

```
temp = conv(B, Q);
nq = length(temp);
Sq = [S, zeros(1, nq - length(S))] - temp;
Pq=conv(B,Rq)+conv(A,Sq);
Tq = sum(Rq);

Stfq = tf(conv(A,Sq),Pq,Ts,'Variable','z^-1');
Utfq = tf(conv(A,Rq),Pq,Ts,'Variable','z^-1');
save('RST_Devakumar_Martin','Rq','Sq','Tq','-ascii','-tabs')
%%
% Check closeloop response
CLq=tf(conv(Tq,B),Pq,Ts,'Variable','z^-1');
figure()
step(CL)
stepinfo(CL)

% Check input
figure()
bodemag(Utfq)
figure()
step(Utfq)
stepinfo(Utfq)

Mmq = 1/norm(Stfq, 'Inf');
%%
function objective = q33(Q, A, B, S, R, Hs, Hr, P, Ts, W1, W2, CL)
    temp = conv(B, Q);
    nq = length(temp);
    Sq = [S, zeros(1, nq - length(S))] - temp;
    Stf = tf(conv(A,Sq),P,Ts,'Variable','z^-1');
    objective = norm(W2*CL + W1*Stf, 'Inf');
end

function [cons, cons_eq] = constraints(Q, A, B, S, R, Hs, Hr, P, Ts, W1, W2, CL)
    temp = conv(A, Q);
    Rq = [R, zeros(1, length(temp) - length(R))] + temp;
    U = tf(conv(A,Rq),P,Ts,'Variable','z^-1');
    cons = norm(U, 'Inf') - 7.5;

    cons_eq = [sum(Q); sum(Q(1:2:end)) - sum(Q(2:2:end))];

    temp = conv(B, Q);
    nq = length(temp);
    Sq = [S, zeros(1, nq - length(S))] - temp;
    Stf = tf(conv(A,Sq),P,Ts,'Variable','z^-1');
    cons = [cons; -1/norm(Stf, 'Inf')+0.5];
end
```

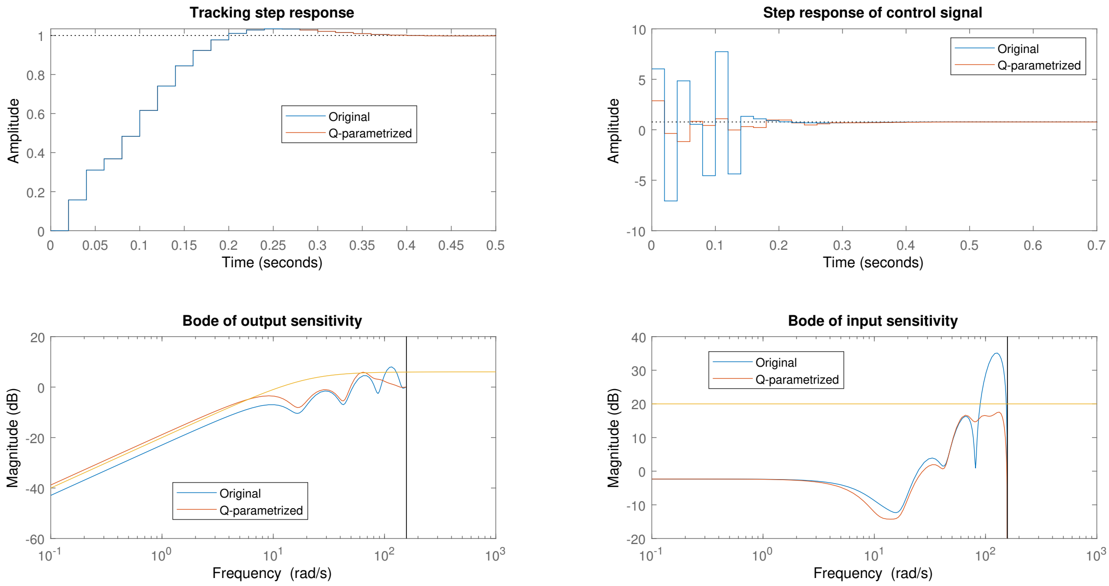
*Final RST control parameters are given below

$R(q^{-1}) = 2.8873204 + 1.1847284q^{-1} - 1.9809926q^{-2} - 2.2513287q^{-3} + -1.0357708q^{-4} + 1.2805150q^{-5} + 1.2111561q^{-6} + 0.52531398q^{-7} - 0.34475234q^{-8} - 0.34056213q^{-9} - 0.20757673q^{-10} + 0.39504454q^{-11} + 0.19778955q^{-12} - 00.66537616q^{-13}$

$S(q^{-1}) = 1 + 1.2185562q^{-1} + 0.36943198q^{-2} - 0.62517487q^{-3} + -0.95709192q^{-4} - 0. + 0243635q^{-5} - 1.1290375e-01q^{-6} + 7.8106878e-02q^{-7} - 3.6255078e-03q^{-8} - 1.3460451e-01q^{-9} - 1.5899906e-01q^{-10} - 7.7716544e-02q^{-11} - 9.9993498e-04q^{-12} + 7.4573959e-03q^{-13}$

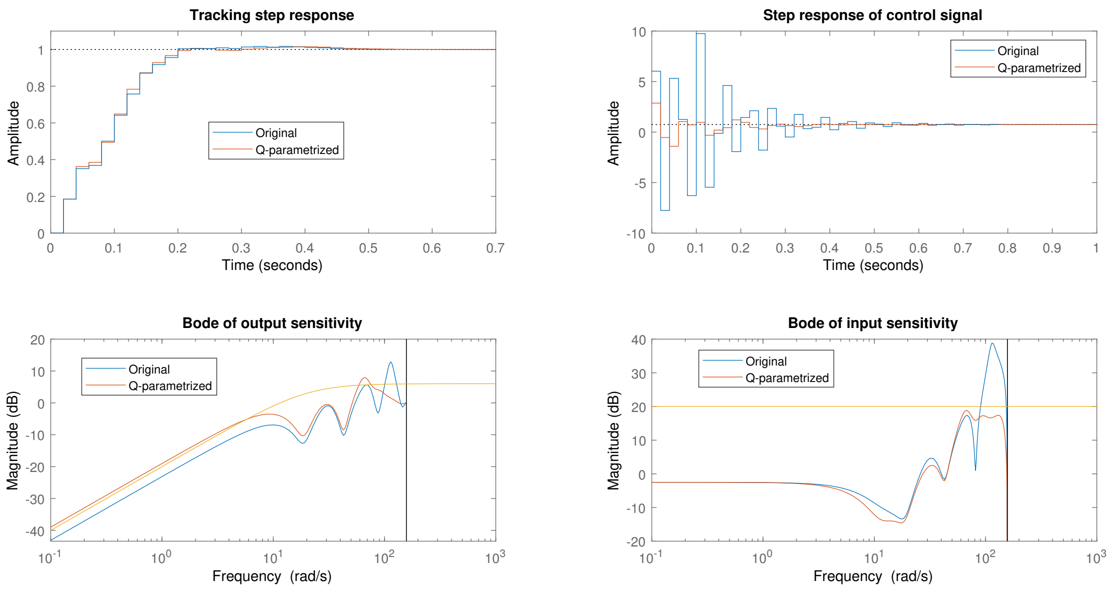
$T(q^{-1}) = 1.4543470$

- Plot of tracking step, input step and sensitivity functions for model G1 is given below



Response of the model G1 for original and Q-parametrized RST

- Plot of tracking step, input step and sensitivity functions for model G0



Response of the model G0 for original and Q-parametrized RST

- Plot of tracking step, input step and sensitivity functions for model G2

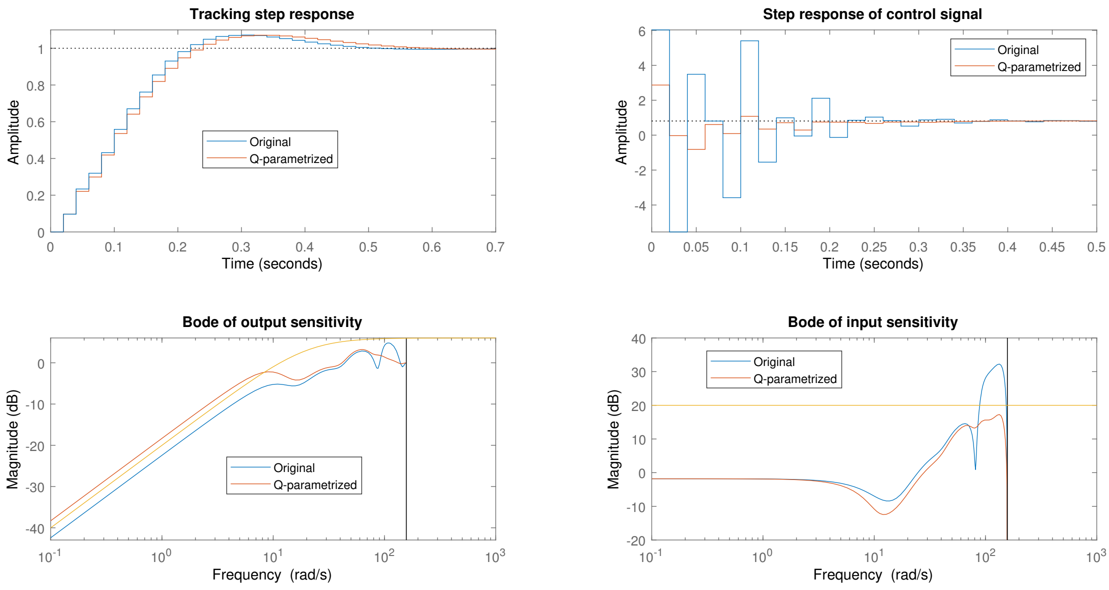
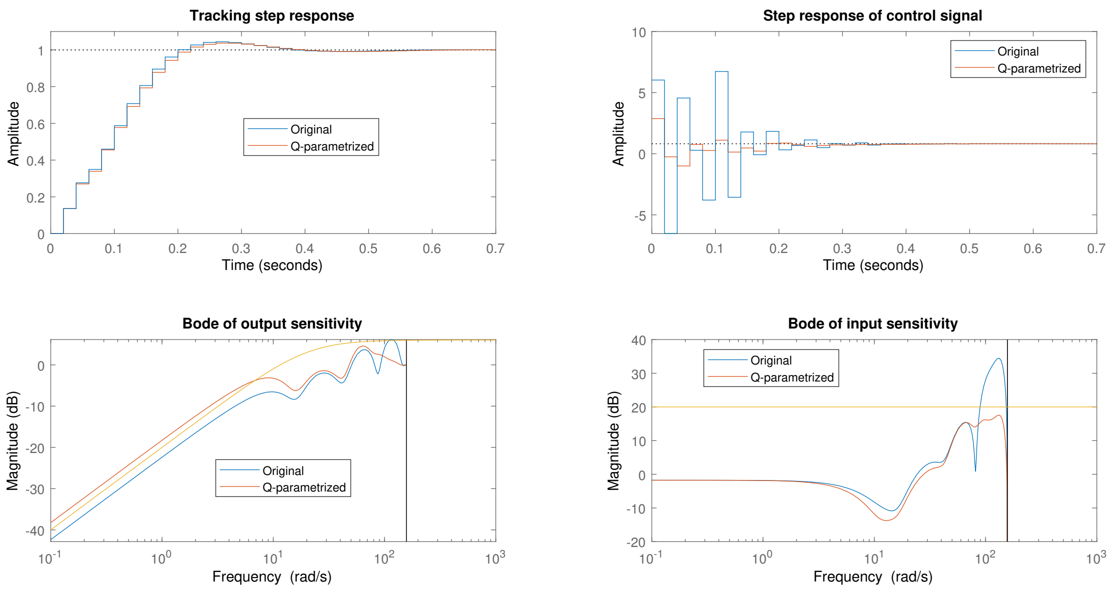


Figure 1: **Response of the model G3 for original and Q-parametrized RST**



Response of the model G2 for original and Q-parametrized RST

- Plot of tracking step, input step and sensitivity functions for model G3

Comparison of parameters for both controllers

	Original RST	Q-Parameterized RST
Infinity norm in input sensitivity, dB	57.2301	7.4861
Infinity norm in output sensitivity, dB	2.5145	1.9524
Rise time, s	0.14	0.14
Settling time, s	0.32	0.32
Overshoot %	3.45	3.45

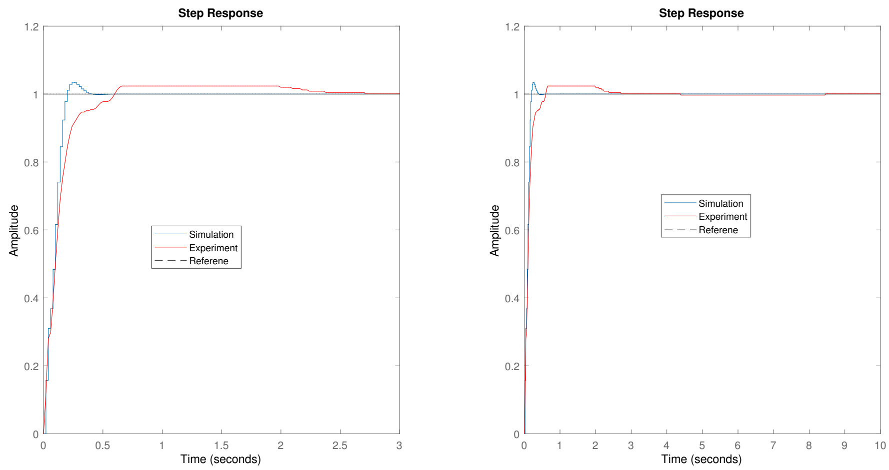


Figure 2: Tracking response superimposed with simulation result

RST controller in final form

R(q^{-1}) $= 2.8873204 + 1.1847284q^{-1} - 1.9809926q^{-2} - 2.2513287q^{-3} + -1.0357708q^{-4} + 1.2805150q^{-5} + 1.2111561q^{-6} + 5.2531398e - 01q^{-7} - 3.4475234e - 01q^{-8} - 3.4056213e - 01q^{-9} - 2.0757673e - 01q^{-10} + 3.9504454e - 01q^{-11} + 1.9778955e - 01q^{-12} - 6.6537616e - 02q^{-13}$

S(q^{-1}) $= 1 + 1.2185562q^{-1} + 3.6943198e - 01q^{-2} - 6.2517487e - 01q^{-3} + -9.5709192e - 01q^{-4} - 6.0243635e - 01q^{-5} - 1.1290375e - 01q^{-6} + 7.8106878e - 02q^{-7} - 3.6255078e - 03q^{-8} - 1.3460451e - 01q^{-9} - 1.5899906e - 01q^{-10} - 7.7716544e - 02q^{-11} - 9.9993498e - 04q^{-12} + 7.4573959e - 03q^{-13}$

T(q^{-1}) $= 1.4543470$

Experimental results

The data is normalized by dividing the experimental results by a factor of 10. Further, the start time is adjusted to match with time ’0’. Tracking response superimposed with simulation is given in figure 2 and input comparison is given in 3

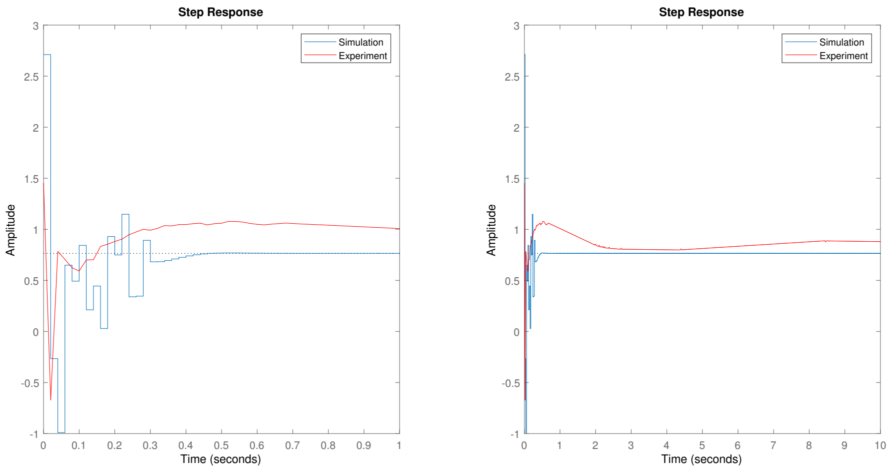


Figure 3: Input response superimposed with simulation result