

EE752 Homework#2 Solution

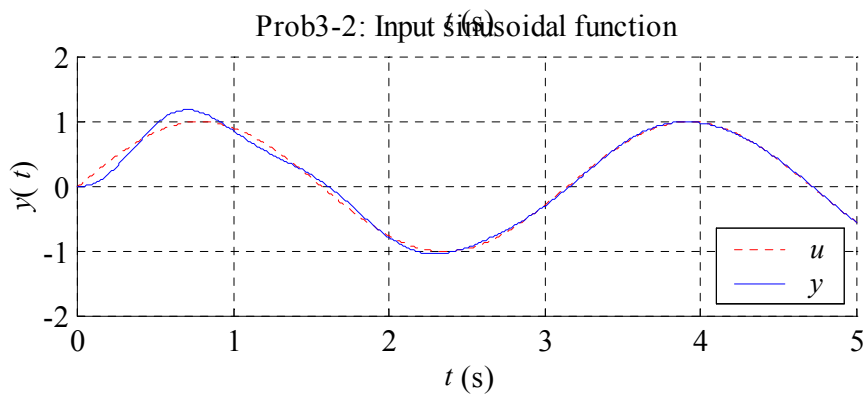
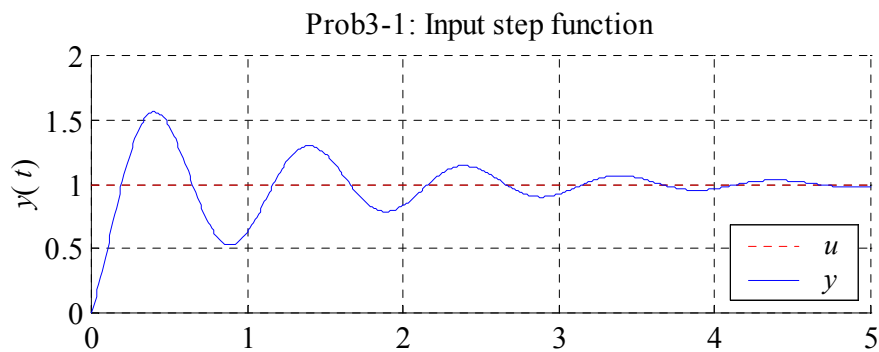
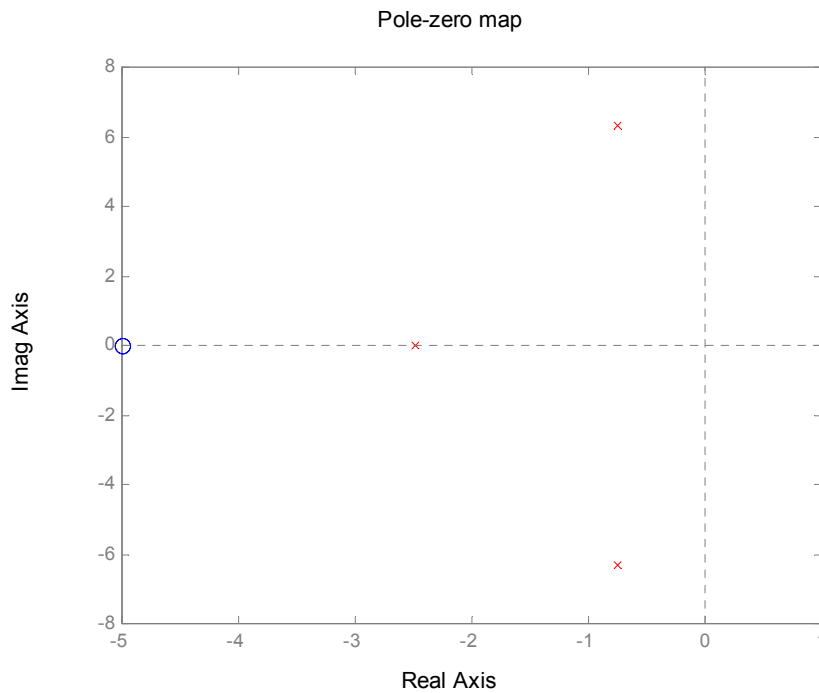
3-2

From problem 2-5, we set $\omega_0 = 2$.

$$\text{Then, } S(s) = \frac{1}{1+P(s)C(s)} = \frac{s(s^2+4)}{s^3+k_c s^2+(4+2k_c z_c)s+k_c z_c^2} \cdot (\text{with } \omega_0 = 2)$$

From the constraints, $k_c > 1.5$.

Choose $k_c = 4$, $z_c = 5$, check the pole positions which are satisfied.



MATLAB code:

```
clear all;
% Plant model:
num_P=1;
den_P=[1 0]; % P(s)=1/s
sys_P=tf(num_P, den_P);
% Controller model:
Kc=4; % controller parameter to be designed
z_c=5; % controller parameter, assumed
w=2;
num_C=Kc*[1 2*z_c z_c^2];
den_C=[1 0 w^2];
sys_C=tf(num_C, den_C);
sys_ol=series(sys_C, sys_P); % open-loop transfer function
sys_cl=feedback(sys_ol,1); % closed-loop transfer function
figure(1); % plot the closed-loop poles and zeros
pzmap(sys_cl);
figure(2); % plot the time responses
subplot(2,1,1)
t = 0:0.01:5;
u = ones(size(t));
Y=lsim(sys_cl,u,t); grid on;zoom on; hold on;
plot(t, u, 'r:', t, Y, 'b-');
title('Prob3-1: Input step function', 'FontName','times new roman',
'FontSize', 12);
xlabel('\it t (s)', 'FontName','times new roman', 'FontSize', 12);
ylabel('\it y({\it t})', 'FontName','times new roman', 'FontSize',
12);
set(gca, 'FontName', 'times new roman');
set(gca, 'FontSize', 12);
legend('\it u', '\it y', 4);
subplot(2,1,2)
u = sin(2*t);
Y=lsim(sys_cl,u,t); grid on;zoom on; hold on;
plot(t, u, 'r:', t, Y, 'b-');
title('Prob3-2: Input sinusoidal function', 'FontName','times new
roman',
'FontSize', 12);
xlabel('\it t (s)', 'FontName','times new roman', 'FontSize', 12);
ylabel('\it y({\it t})', 'FontName','times new roman', 'FontSize',
12);
set(gca, 'FontName', 'times new roman');
set(gca, 'FontSize', 12);
legend('\it u', '\it y', 4);
```

4-1

(a) Characteristic Equation $\chi(s) = a_0s^3 + a_1s^2 + a_2s + a_3$. Set up four polynomials according to Kharitanov's theorem.

$$a_1(s) = 0.1 + s + 3s^2 + 4s^3$$

$$a_2(s) = 0.5 + 2s + s^2 + s^3$$

$$a_3(s) = 0.1 + 2s + 3s^2 + s^3$$

$$a_4(s) = 0.5 + s + s^2 + 4s^3$$

Using Routh-Hurwitz test, $a_1(s)$, $a_2(s)$ and $a_3(s)$ are stable.

$$a_4(s) = 4s^3 + s^2 + s + 0.5$$

$$s^3 \quad 4 \quad 1 \quad 0$$

$$s^2 \quad 1 \quad 0.5$$

$$s^1 \quad -1 \quad 0$$

$$s^0 \quad -0.5$$

Then, the system is not robustly stable with the given parameter uncertainties.

$$(b) \chi(s) = a_0s^3 + a_1s^2 + a_2s + a_3$$

$$\chi(s) = a_0s^3 + a_1s^2 + a_2s + a_3$$

$$s^3 \quad 1+3x^2 \quad a_2$$

$$s^2 \quad 1+2x^2 \quad a_3$$

$$s^1 \quad \frac{a_2(1+2x^2) - a_3(1+3x^2)}{1+2x^2}$$

Define $\alpha = a_2(1+2x^2) - a_3(1+3x^2)$, then for every $x \in (-1,1)$

$$\alpha > (1+2x^2) - 0.5(1+3x^2) = 0.5 + 0.5x^2 > 0$$

Thus, the system is robustly stable under the defined parameter uncertainties.

4-2

Objective: In this problem, the controller has a fixed form. And for each K , the closed loop system has some ability of rejecting the uncertainty to some extent. Here we need to find the best K such that the closed loop system with the specified controller is the best with respect to the stability under the largest uncertainty.

Those four Kharitanov polynomials for the numerator of the plant are:

$$N_1(s) = 1.25s^3 - 3.6s^2 + 3s + 0.6$$

$$N_2(s) = 0.95s^3 - 4s^2 + 3.5s + q$$

$$N_3(s) = 0.95s^3 - 3.6s^2 + 3.5s + 0.6$$

$$N_4(s) = 1.25s^3 - 4s^2 + 3s + q$$

And the four Kharitanov polynomials for the denominator of the plant are:

$$D_1(s) = 0.9s^4 + 12.5s^3 + 24s^2 + 16s - 0.1$$

$$D_2(s) = 1.1s^4 + 11.5s^3 + 20s^2 + 24s + 0.1$$

$$D_3(s) = 0.9s^4 + 11.5s^3 + 24s^2 + 24s - 0.1$$

$$D_4(s) = 1.1s^4 + 12.5s^3 + 20s^2 + 16s + 0.1$$

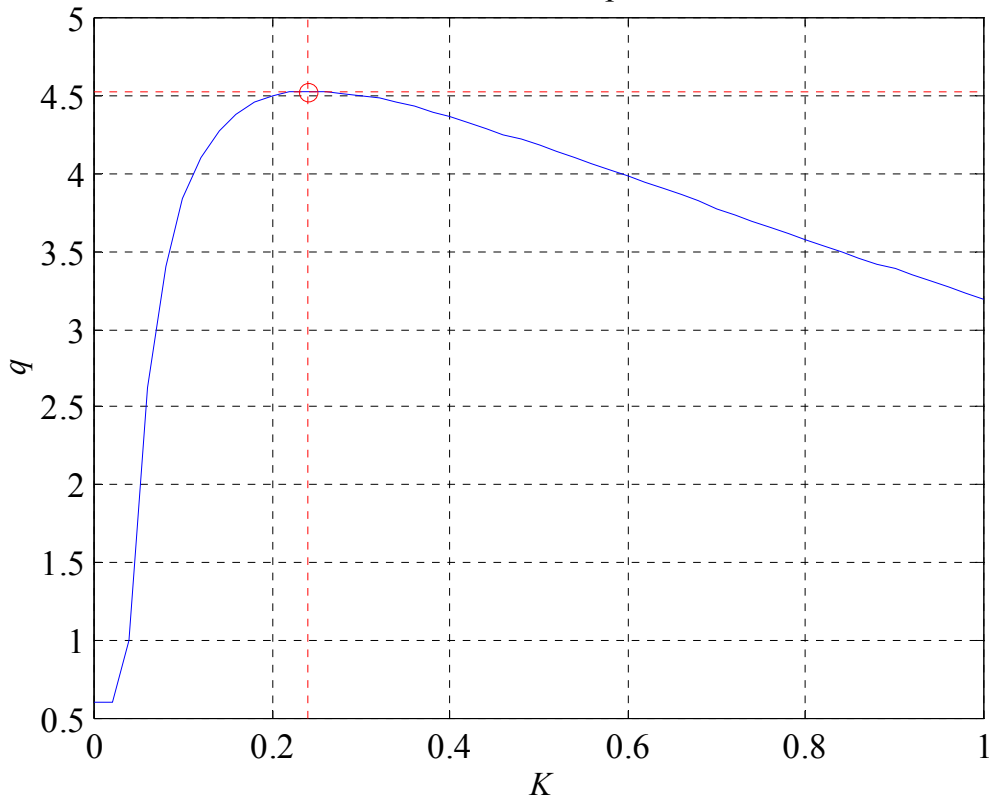
The controller: $N_c(s) = K(s+1)$, $D_c(s) = s$

According to Theorem 4.5, 16 polynomials are obtained by $N_c(s)N_{i1}(s) + D_c(s)D_{i2}(s)$,

where $i1 \in \{1, 2, 3, 4\}$ and $i2 \in \{1, 2, 3, 4\}$.

The stability of each polynomials can be examined by checking its roots – if the greatest real part among all roots is less than 0, then this polynomial is stable, otherwise, unstable. The closed loop system is stable if and only if all 16 polynomials are stable.

Curve of q with respect to K



MATLAB code:

```
clear all;
close all;
%%%%% Construct denominator Kharitanov polynomials %%%%%
% r0 ... rN
D(1,:)=[0.9, 12.5, 24, 16, -0.1]; % - + + - -
D(2,:)=[1.1, 11.5, 20, 24, 0.1 ]; % + - - + +
D(3,:)=[0.9, 11.5, 24, 24, -0.1]; % - - + + -
D(4,:)=[1.1, 12.5, 20, 16, 0.1 ]; % + + - - +
%%%%% Construct numerator Kharitanov %%%%%%%%%
%%%%% polynomials which are independent of q %%%%%%%%%
% q0 ... qN
N(1,:)=[1.25, -3.6, 3.0, 0.6]; % + + - -
N(3,:)=[0.95, -3.6, 3.5, 0.6]; % - + + -
%%%%% For each K between 0 and 1 solve for %%%%%%%%%
%%%%% largest allowable q (denoted by q1) %%%%%%%%%
K=0:0.02:1; % Construct a vector of K covering the range of interest
for j=1:length(K), % This loop iterates on K
qmax=100; % Initial guess for upper bound of q,
% which must make the system unstable
qmin=0.6; % Initial guess for lower bound of q
% Polynomials for the controller: K(s+1)/s
Nc=K(j)*[1 1];
Dc=[1 0];
while (qmax - qmin) > 0.001, % Determines precision of the solution
q=qmax; % assume that q equal to its upper bound
% Kharitanov Polynomials depending on q:
% q0 ... qN
N(2,:)=[0.95, -4.0, 3.5, q ]; % - - + +
N(4,:)=[1.25, -4.0, 3.0, q ]; % + - - +
% Construct 16 characteristic polynomials and obtain the roots:
for i1=1:4,
for i2=1:4,
CurrentPoly=polyadd(conv(Nc, N(i1,:)), conv(Dc, D(i2,:)));
% Pick and store the maximum real part of all roots
% of current polynomial:
max_realpart_of_roots((i1-1)*4+i2)= ...
```

```

max(real(roots(CurrentPoly)));
% This indexing yields indices from 1 to 16
end
end

% Check stability of all 16 polynomials
% and reduce range for next search of q accordingly (binary search):
if max(max_realpart_of_roots) >= 0 % If unstable
previous_qmax=qmax; % store current qmax for future use
qmax=(qmax+qmin)/2; % decrease qmax for next search
else % If stable
qmin=qmax; % increase qmin for next search
qmax=(previous_qmax+qmax)/2; % increase qmax and
end
error=qmax-qmin;
end
q_K(j)=q; % Store the q value corresponding to current K, i.e., K(j)
end
% Plot the curve of q with respect to K:
figure(1)
plot(K, q_K);grid on;zoom on;
axis([0 1 0.5 5]);
title('Curve of {\itq} with respect to {\itK}',...
'FontName','times new roman','FontSize', 12);
xlabel('{\it K}', 'FontName','times new roman', 'FontSize', 12);
ylabel('{\it q}', 'FontName','times new roman', 'FontSize', 12);
set(gca,'FontName','times new roman');
set(gca,'FontSize',12);
hold on;
% Find Kmax where q reaches its maximum point:
[maxq, the_index]=max(q_K); % Find the maximum point in vector q_K
maxq % maxq = 4.5298
Kmax=K(the_index) % Find the corresponding K: K(the_index) = 0.24
line([0 1], [maxq maxq], 'LineStyle', ':', 'Color', [1 0 0]);
line([Kmax Kmax], [0.5 5], 'LineStyle', ':', 'Color', [1 0 0]);
plot(Kmax, maxq, 'ro');
%%%%%%%% Of Course finer search on K would give more precise answer
%%%%%%%%%%%%

Matlab m-file of the polyadd function called by the main program: polyadd.m
function[poly]=polyadd(poly1,poly2)
%Copyright 1996 Justin Shriver
%polyadd(poly1,poly2) adds two polynomials possibly of uneven length
if length(poly1)<length(poly2)
short=poly1;
long=poly2;
else
short=poly2;
long=poly1;
end
mz=length(long)-length(short);
if mz>0
poly=[zeros(1,mz),short]+long;
else
poly=long+short;
end
end

```