NORTH CAROLINA STATE UNIVERSITY

Department of Mechanical and Aerospace Engineering

MAE 721 Robust Control with Convex Methods

Spring semester, 2017

Project: Controller design for latex axis flight with significant uncertainty

REPORT

By Muhammad Abdullah Naeem and Yingxi Geng

**Introduction:** A space shuttle is in its final stages of landing. In this process, a lot of uncertainty are considered. The goal of this project is to design several controllers with varied methods for lateral axis flight with the uncertainty.

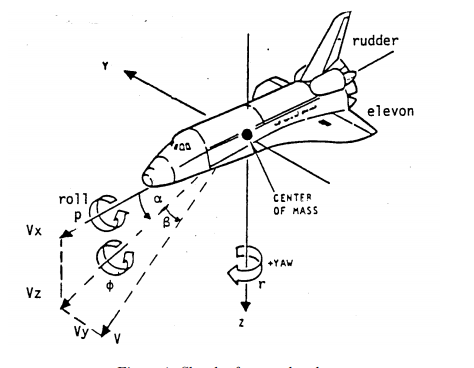


Fig 1. Sketch of space shuttle

The plant of space shuttle could be expressed by Fig 2., named olic. This olic plant contains 17 inputs and 23 outputs. The first 9 inputs and 9 outputs are for aerodynamic uncertainty, whose major source is the aerodynamic coefficients. Input channels from 10 to 15 are exogenous disturbances and channel 16 and 17 are two input to the shuttle (elevon and rudder control command signal). Output channels from 10 to 18 are weighted errors, whereas channel 19 to 23 are four measurements and one command signal.

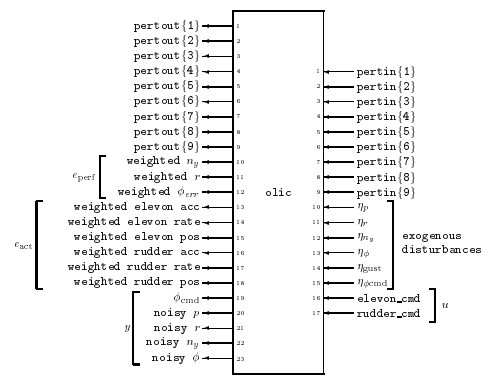


Fig 2. Plant for space shuttle

All designed controllers K in this project obtained signal from measurements and  command.

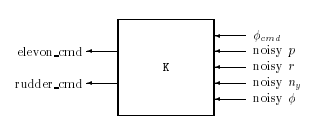


Fig 3. Diagram for controller

**Methods and results**

In order to design a controller with satisfactory performance, several steps were taken. µ analysis was used to analyze the stability and performance of each system.

**Step 1:**

Ignore the uncertainty in the aero coefficients first, design a controller to minimize the H∞ norm from the 6 exogenous signals, to the 9 errors. From the olic plant, we selected the 10th to 23th output, and 10th to 17th input. That is, leave all perturbation inputs and outputs to be zero, then use H infinity method to generate the controller K\_norob.

Main code:

minfo(olic); % check the state of olic

olic\_h=sel(olic,[10:23],[10:17]); % select input and output channel

minfo(olic\_h); % check the state of new plant

[K\_norob,CL,GAM]=hinfsyn(olic\_h,5,2,0,5,0.1);

% design controller K\_norob to minimize the H infinity norm of olic\_h

[AK,BK,CK,DK]=unpck(K\_norob);

K\_ss=ss(AK,BK,CK,DK);

kp=tf(K\_ss); % obtained the transfer function kp of K

**Step 2:**

Using this controller, k\_norob, calculate the robust stability of the system shown below, with respect to a diagonal 9 × 9 complex perturbation block, ∆ = diag [δ1, δ2, · · · , δ9].

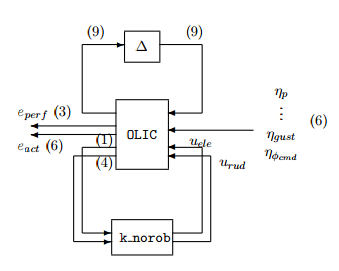


Fig 4. Closed-loop structure with perturbation

The following code performed the function described above.

clp\_h=starp(olic,K\_norob,5,2); % Form closed-loop by olic and k\_norob

omega=logspace(-2,2,40); % generate frequency variable

clp\_hg=frsp(clp\_h,omega); % complex frequency response for clp\_h

clp\_hgRS=sel(clp\_hg,1:9,1:9); % select perturbation channel

blkRS=ones(9,2); % generate perturbation block

[bnds\_h,dv\_h,sens\_h,rp\_h]=mu(clp\_hgRS,blkRS);

% calculate robust stability

vplot('liv,d',bnds\_h); % plot mu

title('Robust Stability of Closed loop');

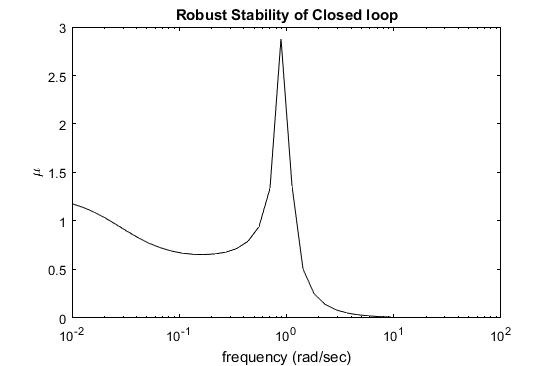


Fig 5. Robust stability of the closed-loop

The peak of the upper bound, is approximately 2.88, which occurs at =0.89 rad/sec, implies that for diagonal, real perturbations smaller than 1/2.88, the closed-loop system remains stable. The lower bound and the upper bound coincide together in the stability plot, so the difference between upper and lower bound is approximate zero.

**Step 3:**

Analyze the robust performance characteristics of the closed-loop system from µ analysis.

delsetrp\_R=[blkRS;[6 9]]; % generate perturbation and performance block

[bnds\_h2,rowd,sens,rowp]=mu(clp\_hg,delsetrp\_R); % calculate mu

vplot('liv,d',bnds\_h2); % generate mu plot

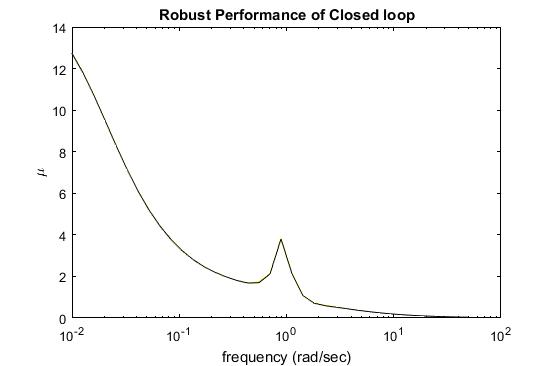


Fig. 6 Robust performance of the closed-loop

The peak of upper bound is about 3.85>1, which implies that robust performance is not achieved.

The difference between lower and upper bound are not significant.

**Step 4:**

The peak value of µ occurs at low frequency. We will take the real part of the perturbation constructed by µ at the first frequency point.

pert=unwrapp(rp\_h,blkRS);

pertatzero=xtracti(vreal(pert),1,1);

minfo(pertatzero);

norm(pertatzero);

pertatzero=pertatzero/norm(pertatzero)\*0.5;

Result:

pertatzero =

0.1576 0 0 0 0 0 0 0 0

0 0.3254 0 0 0 0 0 0 0

0 0 0.4845 0 0 0 0 0 0

0 0 0 -0.5000 0 0 0 0 0

0 0 0 0 0.1953 0 0 0 0

0 0 0 0 0 -0.0912 0 0 0

0 0 0 0 0 0 -0.4792 0 0

0 0 0 0 0 0 0 0.1140 0

0 0 0 0 0 0 0 0 -0.1737

**Step 5:**

Leave out all weighting function from olic to generate a new plant tric. This new plant is used for time-domain simulations. Let all weighting functions equal to identity matrix in order to generate the tric.

WIND

wgust = 1

% SENSOR NOISE (

wr = 1; % rads/sec

wp = wr; % rads/sec

wphi = 1; % rads

wny = 1; % ft/sec/sec

wnoise = eye(4)

% PILOT COMMAND

wphicmd =1;

% ERROR WEIGHTING

% ACTUATOR WEIGHTINGS

wact = eye(6);

% IDEAL PHI\_COMMAND RESPONSE MODEL

wmod = 1;

zmod = 1;

idmod = 1;

% PERFORMANCE VARIABLES

nyerr = 1;

cterr = 1;

baerr = 1;

fix = [0 0 1 0 0;0 1 0 -0.037 0;0 0 0 1 -1];

wperf = mmult(daug(nyerr,cterr,baerr),fix);

% PERTURBATION WEIGHTS

wr = [1 0 0;1 0 0;1 0 0;0 1 0;0 1 0;0 1 0;0 0 1;0 0 1;0 0 1];

wll =eye(3);

wlm = eye(3);

wlr = eye(3);

wl = [wll wlm wlr];

**Step 6:** (matlab code for this part is attached in Appendix-A)

First close the bottom loop of TRIC with k\_norob, and call this closed-loop system clp. Generate two systems, nominal closed-loop system and perturbed closed-loop system (shown in Fig 7.). A nominal closed-loop system is made by selecting outputs 10 − 18 and inputs 14 and 15 of clp. These channels correspond to the 9 error variables, and the gust and command input. A perturbed closed-loop system is made by first closing the top 9 inputs and outputs of clp with pertatzero, and then selecting only the 5th and 6th inputs.

|  |  |
| --- | --- |
| a)b) |  |

Fig 7. Two close-loop system: a) nominal system; b) perturbed system

Generate two inputs (shown in Fig 8.): first is randomly generated noise whose norm is less than 1. The second input is step input.

|  |  |
| --- | --- |
| a) | b) |

Fig 8. two inputs with respect to time, respectively: a) input 1-random noise signal; b) step signal

Fig 9 and Fig 10 show the five responses from 2 inputs for two systems. In Fig 9, with input 1, nominal system and perturbed system have slight differences in output lateral acceleration variable ny and coordinated-turn variable r-0.037. But responses of actuator angles and band angle tracking error ϕ – ϕideal have greater magnitude in perturbed system than nominal system.

However, for responses from input 2 in Fig 10, two systems seem to have similar results, especially in the responses of actuator angles.

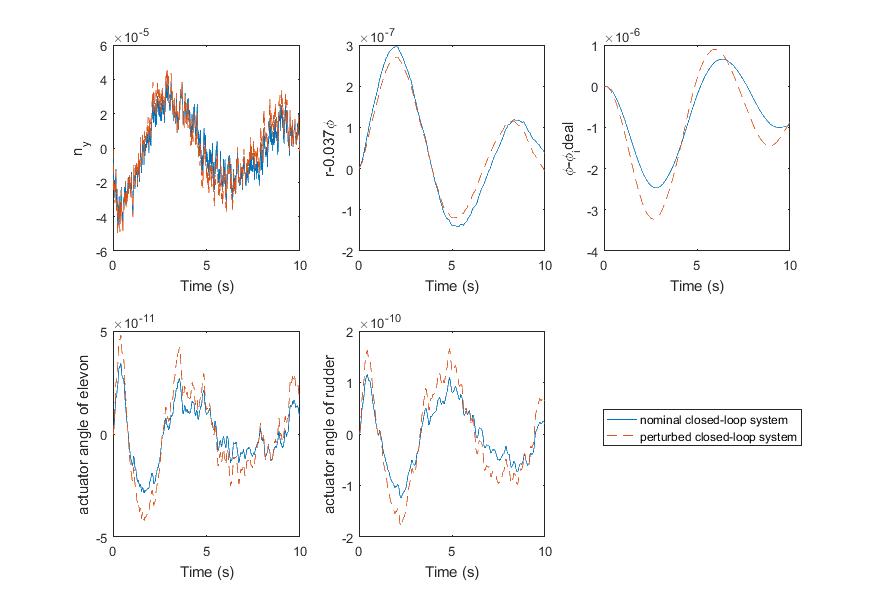


Fig 9. Reponses of two systems from input 1

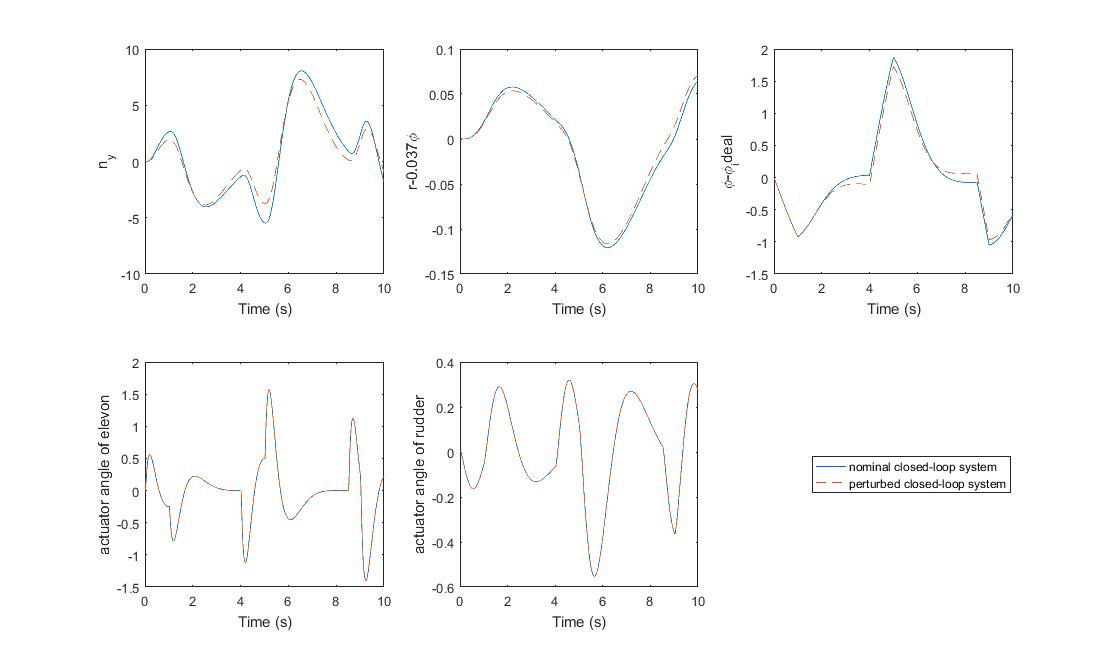


Fig 10. Responses of two systems from input 2

**Step 7:**

Use DK iteration to design a controller for olic.

Use dkit command in Matlab direction. Choose fitting scaling D to be first order. Do three times iteration to get followed controller:

NOMINAL\_DK=olic;

NMEAS\_DK=5;

NCONT\_DK=2;

BLK\_DK=[ones(9,2);[6 9]];

OMEGA\_DK=logspace(-2,2,40);

DISCRETE\_DK=0;

DK\_DEF\_NAME='dkit\_induce';

dkit;

From the summary of three times Iteration, the peak µ value decreased to 1.084, which is close to 1, as we desired. Also from the result of D-K iteration, a controller k\_dk3 was generated automatically by Matlab.

Iteration Summary

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

Iteration # 1 2 3

Controller Order 23 41 41

Total D-Scale Order 0 18 18

Gamma Acheived 6.156 1.392 1.105

Peak mu-Value 1.979 1.200 1.084

**Step 8:** generate new closed-loop system by controller k\_dk3 with olic. Analyze the robust stability and performance.

% robust stability

load('k\_dk3.mat')

clp\_dk=starp(olic,k\_dk3,5,2);

omega=logspace(-5,5,40);

clp\_fdk=frsp(clp\_dk,omega);

clp\_hgdk=sel(clp\_fdk,1:9,1:9);

delsetrs\_C=ones(9,2);

[bnds\_h2,dv\_h2,sens\_h2,rp\_h2]=mu(clp\_hgdk,delsetrs\_C);

vplot('liv,d',bnds\_h2);

title('Robust Stability of Closed loop');

% robust performance

figure(2)

delsetrp\_R=[delsetrs\_C;[6 9]];

[bnds\_h2,rowd2,sens2,rowp2]=mu(clp\_fdk,delsetrp\_R);

vplot('liv,d',bnds\_h2);

Fig 11. and Fig 12. show the robust stability and performance for the new closed-loop system with k\_dk3 and olic.

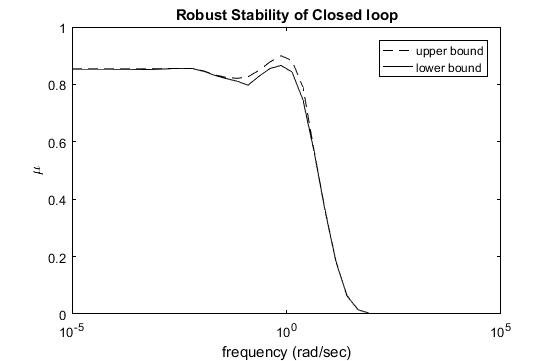


Fig 11. Robust stability for new closed-loop system

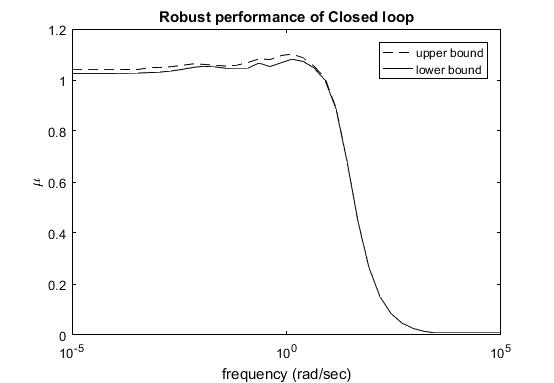


Fig 12. Robust performance for new closed-loop system

From Fig. 11, it could be observed that the peak of upper bound is 0.9, lower than 1, which means the system is stable with any perturbation lower than 1/0.9. The lower and upper bound here have slightly different. From Fig. 12, the peak of the upper bound is about 1.09, which is close to 1. So this new closed-loop system has better robust performance than old closed-loop system generated in problem 2 and 3.

**Step 9:** (Matlab code in this part is attached in Appendix-B)

Use controller k\_dk3 to replace the controller k\_norob in problem 6 to generate two new nominal and perturbed systems. Plot the responses of same inputs in problem 6 (Figure 13 and 14).

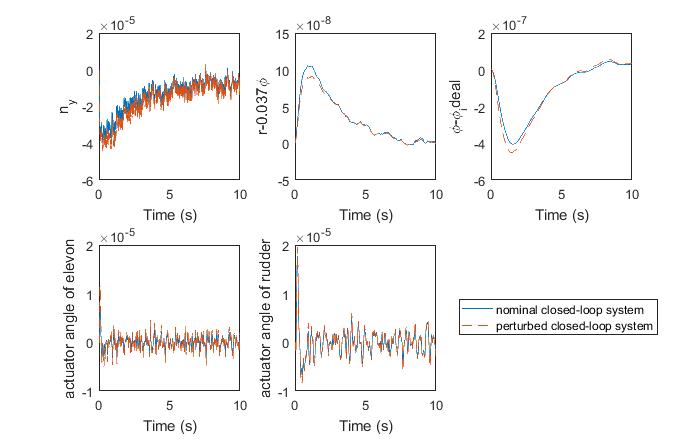


Fig 13. Responses from input 1 to both closed-loop system

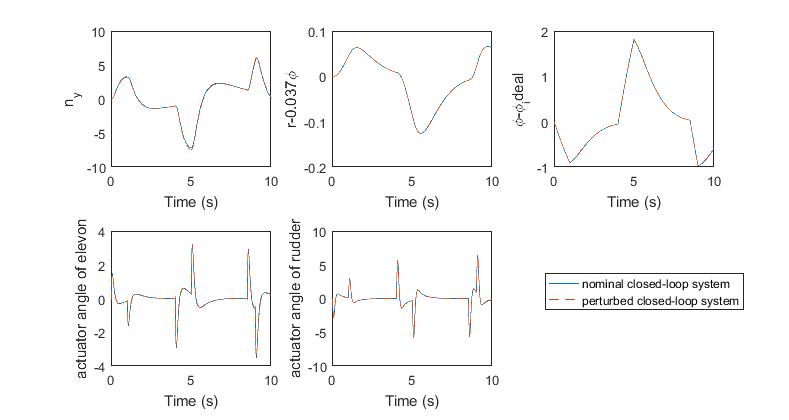


Fig 14. Reponses from input 2 to two closed-loop system

It could be observed from Fig. 13 and 14 that responses are more precise than old systems in Fig. 9 and 10.

**Step 10: comparison: is the perturbed response better?**

By compared Fig. 9, 10, 13, and 14, it could be observed that by using controller obtained from D-K iteration, the responses of nominal and perturbed system are almost same. It implies that the perturbed system respond same with nominal system in with new designed controller, whereas they all perform better than old ones. So our goal is achieved, that is, designing a controller which could be undisturbed by perturbation.

**Step 11: (matlab code is attached in Appendix-C)**

Design a stable controller that deals with the time-varying uncertainties for all perturbation. Use H infinity optimiazation to find a controller and there should exist a constant multiplier matrix D. If D satisfies the following condition, then the system is stable.

1. Set D=identity, the dimension of D should be same with that of uncertainty (9\*9).
2. Use H infinity design to generate a controller k and  (same as problem 1)
3. Release D to a unknown diagonal matrix with same dimention of 1. Let PP=D-1\*P\*D.
4. Set G=lft(PP,k), and find a D from followed equation.



1. Manipulate the equation





1. If there is a D satisfies this equation, then the system constructed by PP and k is stable.

Result:

D=

0.87 0 0 0 0 0 0 0 0

0 0.988 0 0 0 0 0 0 0

0 0 0.438 0 0 0 0 0 0

0 0 0 0.655 0 0 0 0 0

0 0 0 0 0.901 0 0 0 0

0 0 0 0 0 0.404 0 0 0

0 0 0 0 0 0 0.259 0 0

0 0 0 0 0 0 0 0.569 0

0 0 0 0 0 0 0 0 0.219

Controller k is attached in Appendix-D

Appendix-A: Matlab code (problem 6)

% problem 6

% close bottom loop of TRIC with K

clp=starp(tric,K,5,2);

nom\_clp=sel(clp,[10:18],[14,15]);

[AN,BN,CN,DN]=unpck(nom\_clp);

N=ss(AN,BN,CN,DN);

NN=tf(N);

starp\_clp=starp(pertatzero,clp,9,9);

pert\_clp=sel(starp\_clp,[1:9],[5:6]);

[AS,BS,CS,DS]=unpck(pert\_clp);

S=ss(AS,BS,CS,DS);

SS=tf(S);

t=0:0.01:9.99;

randsig = siggen('rand(1)',t);

ll=randsig(1:1000,1);

eta\_gust = lsim(tf(20,[1,20]),ll,0:0.01:9.99);

gust = lsim(tf([1],[1]),eta\_gust,0:0.01:9.99);

figure(1)

plot(t,gust)

title('signal gust versus time')

xlabel('Time (s)')

ylabel('gust')

%%

% command aove here worked

figure(1)

%vplot('iv,d',gust);

comm = vpck([0;1;1;-1;-1;0],[0;1;4;5;8.5;9]);

vplot('iv,d',comm);

comm = vinterp(comm,0.01,10,1);

com=vunpck(comm);

vplot('iv,d',comm);

title('signal comm versus time')

xlabel('Time (s)')

ylabel('comm')

input1 = gust;

%input2 = abv(0,com);

input2=com;

[Y1,T1]=lsim(NN,[input1 zeros(1000,1)],0:0.01:9.99);

[Y2,T2]=lsim(NN,[zeros(1001,1) input2],0:0.01:10)

[Y3,T3]=lsim(SS,[input1 zeros(1000,1)],0:0.01:9.99);

[Y4,T4]=lsim(SS,[zeros(1001,1) input2],0:0.01:10);

% input 1 for both system

figure(3)

subplot(2,3,1)

plot(T1,Y1(:,1),T3,Y3(:,1),'--')

xlabel('Time (s)')

ylabel('n\_y')

subplot(2,3,2)

plot(T1,Y1(:,2),T3,Y3(:,2),'--')

xlabel('Time (s)')

ylabel('r-0.037\phi')

subplot(2,3,3)

plot(T1,Y1(:,3),T3,Y3(:,3),'--')

xlabel('Time (s)')

ylabel('\phi-\phi\_ideal')

subplot(2,3,4)

plot(T1,Y1(:,6),T3,Y3(:,6),'--')

xlabel('Time (s)')

ylabel('actuator angle of elevon')

subplot(2,3,5)

plot(T1,Y1(:,9),T3,Y3(:,9),'--')

xlabel('Time (s)')

ylabel('actuator angle of rudder')

legend('nominal closed-loop system','perturbed closed-loop system')

% input 2 for both system

figure(4)

subplot(2,3,1)

plot(T2,Y2(:,1),T4,Y4(:,1),'--')

xlabel('Time (s)')

ylabel('n\_y')

subplot(2,3,2)

plot(T2,Y2(:,2),T4,Y4(:,2),'--')

xlabel('Time (s)')

ylabel('r-0.037\phi')

subplot(2,3,3)

plot(T2,Y2(:,3),T4,Y4(:,3),'--')

xlabel('Time (s)')

ylabel('\phi-\phi\_ideal')

subplot(2,3,4)

plot(T2,Y2(:,6),T4,Y4(:,6),'--')

xlabel('Time (s)')

ylabel('actuator angle of elevon')

subplot(2,3,5)

plot(T2,Y2(:,9),T4,Y4(:,9),'--')

xlabel('Time (s)')

ylabel('actuator angle of rudder')

legend('nominal closed-loop system','perturbed closed-loop system')

Appendix-B: Matlab code (problem 9)

% generate two new system

% nominal system

clp2=starp(tric,k\_dk3,5,2);

nom\_clp2=sel(clp2,[10:18],[14,15]);

[AN2,BN2,CN2,DN2]=unpck(nom\_clp2);

N2=ss(AN2,BN2,CN2,DN2);

NN2=tf(N2);

% perturbed plant

starp\_clp2=starp(pertatzero,clp2,9,9);

pert\_clp2=sel(starp\_clp2,[1:9],[5:6]);

[AS2,BS2,CS2,DS2]=unpck(pert\_clp2);

S2=ss(AS2,BS2,CS2,DS2);

SS2=tf(S2);

% problem 9

% gust

t=0:0.002:10;

randsig2 = siggen('rand(1)',t);

ll2=randsig2(1:5000,1);

eta\_gust2 = lsim(tf(20,[1,20]),ll2,0:0.002:9.998);

gust2 = lsim(tf([1],[1]),eta\_gust2,0:0.002:9.998);

% comm

figure(1)

%vplot('iv,d',gust);

comm = vpck([0;1;1;-1;-1;0],[0;1;4;5;8.5;9]);

vplot('iv,d',comm);

comm2= vinterp(comm,0.002,9.998,1);

com2=vunpck(comm2);

vplot('iv,d',comm2);

input11 = gust2;

input22=com2;

[Y11,T11]=lsim(NN2,[input11 zeros(5000,1)],0:0.002:9.998);

[Y22,T22]=lsim(NN2,[zeros(5000,1) input22],0:0.002:9.998);

[Y33,T33]=lsim(SS2,[input11 zeros(5000,1)],0:0.002:9.998);

[Y44,T44]=lsim(SS2,[zeros(5000,1) input22],0:0.002:9.998);

figure(3)

subplot(2,3,1)

plot(T11,Y11(:,1),T33,Y33(:,1),'--')

xlabel('Time (s)')

ylabel('n\_y')

subplot(2,3,2)

plot(T11,Y11(:,2),T33,Y33(:,2),'--')

xlabel('Time (s)')

ylabel('r-0.037\phi')

subplot(2,3,3)

plot(T11,Y11(:,3),T33,Y33(:,3),'--')

xlabel('Time (s)')

ylabel('\phi-\phi\_ideal')

subplot(2,3,4)

plot(T11,Y11(:,6),T33,Y33(:,6),'--')

xlabel('Time (s)')

ylabel('actuator angle of elevon')

subplot(2,3,5)

plot(T11,Y11(:,9),T33,Y33(:,9),'--')

xlabel('Time (s)')

ylabel('actuator angle of rudder')

legend('nominal closed-loop system','perturbed closed-loop system')

% input 2 for both system

figure(4)

subplot(2,3,1)

plot(T22,Y22(:,1),T44,Y44(:,1),'--')

xlabel('Time (s)')

ylabel('n\_y')

subplot(2,3,2)

plot(T22,Y22(:,2),T44,Y44(:,2),'--')

xlabel('Time (s)')

ylabel('r-0.037\phi')

subplot(2,3,3)

plot(T22,Y22(:,3),T44,Y44(:,3),'--')

xlabel('Time (s)')

ylabel('\phi-\phi\_ideal')

subplot(2,3,4)

plot(T22,Y22(:,6),T44,Y44(:,6),'--')

xlabel('Time (s)')

ylabel('actuator angle of elevon')

subplot(2,3,5)

plot(T22,Y22(:,9),T44,Y44(:,9),'--')

xlabel('Time (s)')

ylabel('actuator angle of rudder')

legend('nominal closed-loop system','perturbed closed-loop system')

Appendix-C Matlab code for problem 11

% problem 11

[mattype,rowd,cold,num]=minfo(olic);

% define the first generation of D

D=eye(9);

bnds=0;

eps=1e-5;

tol=1;

while tol>eps

omega=logspace(-2,2,40);

blk=ones(9,2);

olicnew=mmult([inv(D) zeros(9,rowd-9);zeros(rowd-9,9) eye(rowd-9,rowd-9)],olic,[D zeros(9,cold-9);zeros(cold-9,9) eye(cold-9,cold-9)]);

khinf=hinfsyn(olicnew,5,2,0,20,0.1);

clp=starp(olic,khinf,5,2);

clpfre=frsp(clp,omega);

clpnew=sel(clpfre,[1:9],[1:9]);

[bndstv,rowdtv,senstv,rowptv,rowgtv]=mu(clpnew,blk);

[dl,dr]=muunwrap(rowdtv,blk);

[maxG,index]=max(bndstv(1:end-1,1));

D=dl((index-1)\*9+1:index\*9,1:9);

tol=abs(bnds-max(bndstv(1:end-1,1)));

bnds=max(bndstv(1:end-1,1));

end

Appendix-D controller k for problem 11

From input 1 to output...

4.907 s^22 + 3826 s^21 + 1.326e06 s^20 + 2.597e08 s^19 + 3.093e10 s^18 + 2.278e12 s^17

+ 1.054e14 s^16 + 3.165e15 s^15 + 6.245e16 s^14 + 7.982e17 s^13

+ 6.465e18 s^12 + 3.064e19 s^11 + 8.583e19 s^10 + 1.511e20 s^9 + 1.721e20 s^8

+ 1.27e20 s^7 + 5.956e19 s^6 + 1.692e19 s^5 + 2.609e18 s^4 + 1.677e17 s^3

+ 2.902e15 s^2 + 1.736e13 s + 2.665e10

1: ---------------------------------------------------------------------------------------

s^23 + 968.8 s^22 + 4.156e05 s^21 + 1.024e08 s^20 + 1.575e10 s^19 + 1.547e12 s^18

+ 9.655e13 s^17 + 3.801e15 s^16 + 9.498e16 s^15 + 1.473e18 s^14

+ 1.325e19 s^13 + 6.723e19 s^12 + 2.004e20 s^11 + 3.726e20 s^10

+ 4.509e20 s^9 + 3.569e20 s^8 + 1.802e20 s^7 + 5.518e19 s^6 + 9.21e18 s^5

+ 6.576e17 s^4 + 1.536e16 s^3 + 1.503e14 s^2 + 6.011e11 s + 6.281e08

-0.6538 s^22 - 621.6 s^21 - 2.605e05 s^20 - 6.226e07 s^19 - 9.181e09 s^18

- 8.484e11 s^17 - 4.825e13 s^16 - 1.664e15 s^15 - 3.545e16 s^14

- 4.633e17 s^13 - 3.522e18 s^12 - 1.509e19 s^11 - 3.699e19 s^10

- 5.643e19 s^9 - 5.591e19 s^8 - 3.623e19 s^7 - 1.511e19 s^6 - 3.877e18 s^5

- 5.544e17 s^4 - 3.564e16 s^3 - 7.158e14 s^2 - 5.656e12 s - 1.545e10

2: -----------------------------------------------------------------------------------

s^23 + 968.8 s^22 + 4.156e05 s^21 + 1.024e08 s^20 + 1.575e10 s^19 + 1.547e12 s^18

+ 9.655e13 s^17 + 3.801e15 s^16 + 9.498e16 s^15 + 1.473e18 s^14

+ 1.325e19 s^13 + 6.723e19 s^12 + 2.004e20 s^11 + 3.726e20 s^10

+ 4.509e20 s^9 + 3.569e20 s^8 + 1.802e20 s^7 + 5.518e19 s^6 + 9.21e18 s^5

+ 6.576e17 s^4 + 1.536e16 s^3 + 1.503e14 s^2 + 6.011e11 s + 6.281e08

From input 2 to output...

-209.4 s^21 - 1.46e05 s^20 - 4.455e07 s^19 - 7.396e09 s^18 - 7.059e11 s^17

- 3.834e13 s^16 - 1.28e15 s^15 - 2.734e16 s^14 - 3.719e17 s^13 - 3.197e18 s^12

- 1.618e19 s^11 - 4.932e19 s^10 - 9.618e19 s^9 - 1.244e20 s^8 - 1.073e20 s^7

- 6.001e19 s^6 - 2.027e19 s^5 - 3.608e18 s^4 - 2.518e17 s^3 - 4.578e15 s^2

- 2.946e13 s - 5.452e10

1: ---------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08

-56.48 s^21 - 4.989e04 s^20 - 1.918e07 s^19 - 4.115e09 s^18 - 5.262e11 s^17

- 3.991e13 s^16 - 1.719e15 s^15 - 4.313e16 s^14 - 6.454e17 s^13

- 5.436e18 s^12 - 2.465e19 s^11 - 6.245e19 s^10 - 9.797e19 s^9 - 9.966e19 s^8

- 6.64e19 s^7 - 2.878e19 s^6 - 7.809e18 s^5 - 1.177e18 s^4 - 7.012e16 s^3

- 7.905e14 s^2 + 8.947e11 s + 2.888e10

2: --------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08

From input 3 to output...

-344.4 s^21 - 2.595e05 s^20 - 8.633e07 s^19 - 1.598e10 s^18 - 1.755e12 s^17

- 1.141e14 s^16 - 4.403e15 s^15 - 1.053e17 s^14 - 1.52e18 s^13 - 1.328e19 s^12

- 6.23e19 s^11 - 1.63e20 s^10 - 2.631e20 s^9 - 2.743e20 s^8 - 1.857e20 s^7

- 8.017e19 s^6 - 2.11e19 s^5 - 3.063e18 s^4 - 1.969e17 s^3 - 4.095e15 s^2

- 3.408e13 s - 9.96e10

1: ---------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08

340.3 s^21 + 3.156e05 s^20 + 1.283e08 s^19 + 2.942e10 s^18 + 4.093e12 s^17

+ 3.46e14 s^16 + 1.702e16 s^15 + 4.656e17 s^14 + 7.419e18 s^13 + 6.378e19 s^12

+ 2.852e20 s^11 + 7.138e20 s^10 + 1.106e21 s^9 + 1.109e21 s^8 + 7.212e20 s^7

+ 2.994e20 s^6 + 7.579e19 s^5 + 1.054e19 s^4 + 6.415e17 s^3 + 1.172e16 s^2

+ 8.055e13 s + 1.766e11

2: ---------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08

From input 4 to output...

-0.07635 s^21 - 58.4 s^20 - 1.98e04 s^19 - 3.77e06 s^18 - 4.327e08 s^17

- 3.028e10 s^16 - 1.316e12 s^15 - 3.697e13 s^14 - 6.744e14 s^13

- 7.993e15 s^12 - 5.955e16 s^11 - 2.598e17 s^10 - 6.187e17 s^9 - 8.818e17 s^8

- 7.721e17 s^7 - 4.031e17 s^6 - 1.167e17 s^5 - 1.526e16 s^4 - 2.793e14 s^3

- 3.321e11 s^2 + 2.171e10 s + 1.123e08

1: --------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08

0.08212 s^21 + 77.64 s^20 + 3.232e04 s^19 + 7.657e06 s^18 + 1.116e09 s^17

+ 1.014e11 s^16 + 5.625e12 s^15 + 1.88e14 s^14 + 3.922e15 s^13 + 5.077e16 s^12

+ 3.946e17 s^11 + 1.744e18 s^10 + 4.173e18 s^9 + 5.967e18 s^8 + 5.244e18 s^7

+ 2.754e18 s^6 + 8.068e17 s^5 + 1.09e17 s^4 + 2.767e15 s^3 + 2.524e13 s^2

+ 7.222e10 s - 4.739e07

2: ---------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08

From input 5 to output...

-0.4768 s^21 - 370.3 s^20 - 1.279e05 s^19 - 2.495e07 s^18 - 2.963e09 s^17

- 2.181e11 s^16 - 1.013e13 s^15 - 3.064e14 s^14 - 6.117e15 s^13

- 7.931e16 s^12 - 6.542e17 s^11 - 3.172e18 s^10 - 9.034e18 s^9 - 1.579e19 s^8

- 1.748e19 s^7 - 1.206e19 s^6 - 4.831e18 s^5 - 9.746e17 s^4 - 7.156e16 s^3

- 1.251e15 s^2 - 7.213e12 s - 9.333e09

1: --------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08

-0.02853 s^21 - 25.96 s^20 - 1.032e04 s^19 - 2.304e06 s^18 - 3.093e08 s^17

- 2.488e10 s^16 - 1.134e12 s^15 - 2.76e13 s^14 - 3.697e14 s^13 - 2.158e15 s^12

- 3.27e15 s^11 + 1.4e16 s^10 + 3.489e16 s^9 - 4.858e16 s^8 - 2.504e17 s^7

- 3.67e17 s^6 - 2.514e17 s^5 - 7.231e16 s^4 - 5.595e15 s^3 - 8.149e13 s^2

- 2.281e11 s + 9.71e08

2: ---------------------------------------------------------------------------------------

s^22 + 966.8 s^21 + 4.137e05 s^20 + 1.016e08 s^19 + 1.554e10 s^18 + 1.516e12 s^17

+ 9.352e13 s^16 + 3.614e15 s^15 + 8.775e16 s^14 + 1.298e18 s^13

+ 1.065e19 s^12 + 4.593e19 s^11 + 1.085e20 s^10 + 1.556e20 s^9 + 1.397e20 s^8

+ 7.743e19 s^7 + 2.537e19 s^6 + 4.443e18 s^5 + 3.25e17 s^4 + 7.64e15 s^3

+ 7.5e13 s^2 + 3.004e11 s + 3.14e08