ME-8930

Convex Optimization Methods for Robust and Optimal Control Design

HW04

Group:

Chinmay Samak

Tanmay Samak

$$x_p = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

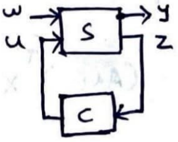
$$\dot{x}_p = \begin{bmatrix} 0 & 10 & 2 \\ -1 & 1 & 0 \\ 0 & 2 & -5 \end{bmatrix} x_p + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} u + \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} w$$

$$y = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}^T x_p + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} u$$

$$z = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}^T x_p + 2w$$

Now, we know the general form

$$\dot{x}_p = A_p x_p + B_p u + D_p w$$
$$y = C_p x_p + B_y u + D_y w$$
$$z = M_p x_p + D_z w$$



Comparing, we get

$$A_p = \begin{bmatrix} 0 & 10 & 2 \\ -1 & 1 & 0 \\ 0 & 2 & -5 \end{bmatrix} \Rightarrow \bar{A}_p = A_p + 0.2I = \begin{bmatrix} 0 & 10 & 2 \\ -1 & 1 & 0 \\ 0 & 2 & -5 \end{bmatrix} + \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}$$

$$= \begin{bmatrix} 0.2 & 10 & 2 \\ -1 & 1.2 & 0 \\ 0 & 2 & -4.8 \end{bmatrix}$$

Since we want CL poles to left of $\delta = -0.2$, we shift A_p to $A_p + 0.2I$

$$B_p = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$C_p = [1\ 0\ 0]$$

$$D_p = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$B_y = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$D_y = 0$$

$$M_p = [0\ 1\ 0]$$

$$D_z = 2$$

There exists an H_∞ controller of order $n_c \leq n_p$: $\begin{cases} \dot{x}_c = A_c x_c + B_c z \\ u = C_c x_c + D_c z \end{cases}$

such that CL system is stable and

$$\Gamma_{ee} = \max_{w \neq 0} \frac{\|y\|_{L_2}}{\|w\|_{L_2}} < \gamma \quad \text{iff}$$

$$\begin{bmatrix} B_p \\ B_y \end{bmatrix}^{\perp} \begin{bmatrix} A_p\mathbf{X} + \mathbf{X}A_p^T + D_pD_p^T & \mathbf{X}C_p^T + D_pD_y^T \\ * & D_yD_y^T - \gamma^2 I \end{bmatrix} \begin{bmatrix} B_p \\ B_y \end{bmatrix}^{\perp^T} < 0 \qquad (C_1)$$

$$\begin{bmatrix} M_p^T \\ D_z^T \end{bmatrix}^{\perp} \begin{bmatrix} \mathbf{Y}A_p + A_p^T\mathbf{Y} + C_p^TC_p & \mathbf{Y}D_p + C_p^TD_y \\ * & D_y^TD_y - \gamma^2 I \end{bmatrix} \begin{bmatrix} M_p^T \\ D_z^T \end{bmatrix}^{\perp^T} < 0 \qquad (C_2)$$

$$\begin{bmatrix} \mathbf{X} & \gamma I \\ \gamma I & \mathbf{Y} \end{bmatrix} \geq 0 \qquad (C_3)$$

$$\text{rank} \begin{bmatrix} \mathbf{X} & \gamma I \\ \gamma I & \mathbf{Y} \end{bmatrix} \leq n_p + n_c \qquad (C_4)$$

- For static state feedback (i.e. $n_c = 0$, $M_p = I$), $C_2 - C_4$ are redundant and need not be checked.
- For full order controller (i.e. $n_p = n_c$) condition (C_4) is redundant and need not be checked.

MATLAB Notation (LMI Control Toolbox)

$$\dot{x} = Ax + B_1 w + B_2 u$$
$$z = C_1 x + D_{11} w + D_{12} u$$
$$y = C_2 x + D_{21} w + D_{22} u$$

Comparing with the earlier notation:

$A = A_p$, $B_1 = D_p$, $B_2 = B_p$

$C_1 = C_p$, $D_{11} = D_y$, $D_{12} = B_y$

$C_2 = M_p$, $D_{21} = D_z$, $D_{22} = 0$

let $A = \bar{A}_p$

$B = \begin{bmatrix} D_p & B_p \end{bmatrix}$

$C = \begin{bmatrix} C_p \\ M_p \end{bmatrix}$

$D = \begin{bmatrix} D_y & B_y \\ D_z & 0 \end{bmatrix}$

By lumping matrices together:

$$\dot{x}_p = A_p x_p + \begin{bmatrix} D_p & B_p \end{bmatrix} \begin{bmatrix} w \\ u \end{bmatrix}$$

$$\begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} C_p \\ M_p \end{bmatrix} x_p + \begin{bmatrix} D_y & B_y \\ D_z & 0 \end{bmatrix} \begin{bmatrix} w \\ u \end{bmatrix}$$

H_∞ Dynamic Controller Design in MATLAB

s = ltisys (A, B, C, D)

r = [n_z, n_u]

[g_{opt}, G] = hinflmi (s, r)

$T_{ee} < \gamma^*$ controller

[A_c, B_c, C_c, D_c] = ltiss(G)

Closed-loop (CL) system:

$$A_{cl} = \begin{bmatrix} A_p + B_p D_c M_p & B_p C_c \\ B_c M_p & A_c \end{bmatrix}$$

$$B_{cl} = \begin{bmatrix} D_p + B_p D_c D_z \\ B_c D_z \end{bmatrix}$$

$$C_{cl} = \begin{bmatrix} C_p + B_y D_c M_p & B_y C_c \end{bmatrix}$$

$$D_{cl} = D_y + B_y D_c D_z$$

Verification:

H_∞ norm:

$$\left.\begin{aligned} &S_{cl} = ss(A_{cl}, B_{cl}, C_{cl}, D_{cl}) \\ &hinf\_norm = hinfnorm(S_{cl}) \\ &hinf\_norm < \gamma^* \ (g_{opt}) \end{aligned}\right\} \text{MATLAB}$$

$\hookrightarrow \Gamma_{ee} < \gamma^*$

Eigenvalues (Poles): $Re(\lambda_i(A_{cl})) < 0$ ← stability guaranteed

$Re(\lambda_i(A_{cl})) < 0.2$ ← stability with desired performance constraint guaranteed

Problem 1

CODE:

```
% PROBLEM 1


close all
clear
clc


% Define the system matrices
Ap = [0, 10, 2; -1, 1, 0; 0, 2, -5];
Ap_bar = Ap + 0.2*eye(3);
Bp = [0; 1; 0];
Cp = [1, 0, 0; 0, 0, 0];
Dp = [1; 0; 1];
By = [0; 1];
Dy = [0; 0];
Mp = [0, 1, 0];
Dz = 2;


% Convert to MATLAB notation
A = Ap_bar;
B1 = Dp;
B2 = Bp;
C1 = Cp;
D11 = Dy;
D12 = By;
C2 = Mp;
D21 = Dz;
D22 = 0;


% LTI system
S = ltisys(A, [B1 B2], [C1; C2], [D11 D12; D21 D22]);


% H-infinity LMI
[gopt, G] = hinflmi(S,[1 1])


% Controller matrices
disp('H∞ controller:')
[Ac, Bc, Cc, Dc] = ltiss(G)
```

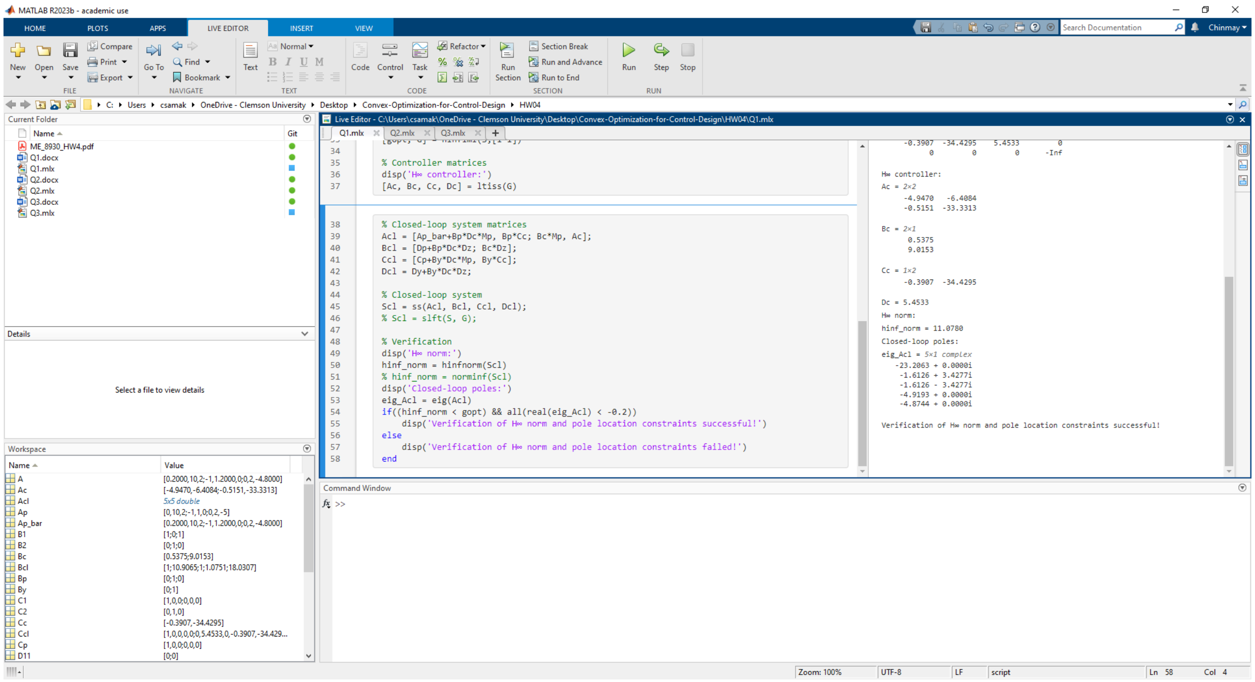
```
% Closed-loop system matrices
Acl = [Ap_bar+Bp*Dc*Mp, Bp*Cc; Bc*Mp, Ac];
Bcl = [Dp+Bp*Dc*Dz; Bc*Dz];
Ccl = [Cp+By*Dc*Mp, By*Cc];
Dcl = Dy+By*Dc*Dz;


% Closed-loop system
Scl = ss(Acl, Bcl, Ccl, Dcl);
% Scl = slft(S, G);


% Verification
disp('H∞ norm:')
hinf_norm = hinfnorm(Scl)
% hinf_norm = norminf(Scl)
disp('Closed-loop poles:')
eig_Acl = eig(Acl)
if((hinf_norm < gopt) && all(real(eig_Acl) < -0.2))
    disp('Verification of H∞ norm and pole location constraints successful!')
else
    disp('Verification of H∞ norm and pole location constraints failed!')
end
```

OUTPUT:

```
Minimization of gamma:

 Solver for linear objective minimization under LMI constraints

 Iterations   :    Best objective value so far

     1
     2
     3
     4                   21.011488
     5                   17.864947
     6                   15.385893
     7                   14.718282
     8                   14.718282
     9                   13.310861
    10                   13.310861
    11                   11.988542
    12                   11.988542
    13                   11.352617
    14                   11.352617
    15                   11.146445
    16                   11.125254
    17                   11.110262
    18                   11.110262
```

```
   19                  11.110262
***                new lower bound:    10.816366
   20                  11.081688
***                new lower bound:    11.010885

 Result:  feasible solution of required accuracy
          best objective value:    11.081688
          guaranteed relative accuracy:  6.39e-03
          f-radius saturation:  0.188% of R =  1.00e+08

 Optimal Hinf performance:  1.108e+01
gopt = 11.0815
G = 4×4
   -4.9470   -6.4084    0.5375    2.0000
   -0.5151  -33.3313    9.0153         0
   -0.3907  -34.4295    5.4533         0
         0         0         0      -Inf
H∞ controller:
Ac = 2×2
   -4.9470   -6.4084
   -0.5151  -33.3313
Bc = 2×1
    0.5375
    9.0153
Cc = 1×2
   -0.3907  -34.4295
Dc = 5.4533
H∞ norm:
hinf_norm = 11.0780
Closed-loop poles:
eig_Acl = 5×1 complex
 -23.2063 + 0.0000i
  -1.6126 + 3.4277i
  -1.6126 - 3.4277i
  -4.9193 + 0.0000i
  -4.8744 + 0.0000i
Verification of H∞ norm and pole location constraints successful!
```

SCREENSHOT:



MATLAB R2023b - academic use
Live Editor - C:\Users\csamak\OneDrive - Clemson University\Desktop\Convex-Optimization-for-Control-Design\HW04\Q1.mlx
Q1.mlx
Q2.mlx
Q3.mlx
% Controller matrices
disp('H∞ controller:')
[Ac, Bc, Cc, Dc] = ltiss(G)
% Closed-loop system matrices
Acl = [Ap_bar+Bp*Dc*Mp, Bp*Cc; Bc*Mp, Ac];
Bcl = [Dp+Bp*Dc*Dz; Bc*Dz];
Ccl = [Cp+By*Dc*Mp, By*Cc];
Dcl = Dy+By*Dc*Dz;
% Closed-loop system
Scl = ss(Acl, Bcl, Ccl, Dcl);
% Scl = slft(S, G);
% Verification
disp('H∞ norm:')
hinf_norm = hinfnorm(Scl)
% hinf_norm = norminf(Scl)
disp('Closed-loop poles:')
eig_Acl = eig(Acl)
if((hinf_norm < gopt) && all(real(eig_Acl) < -0.2))
disp('Verification of H∞ norm and pole location constraints successful!')
else
disp('Verification of H∞ norm and pole location constraints failed!')
end
-0.3907 -34.4295 5.4533 0
0 0 0 -Inf
H∞ controller:
Ac = 2×2
-4.9470 -6.4084
-0.5151 -33.3313
Bc = 2×1
0.5375
9.0153
Cc = 1×2
-0.3907 -34.4295
Dc = 5.4533
H∞ norm:
hinf_norm = 11.0780
Closed-loop poles:
eig_Acl = 5×1 complex
-23.2063 + 0.0000i
-1.6126 + 3.4277i
-1.6126 - 3.4277i
-4.9193 + 0.0000i
-4.8744 + 0.0000i
Verification of H∞ norm and pole location constraints successful!
Command Window

$$x = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}$$

Let $y = q_3$

$$\dot{x} = Ax + Bw \quad ; \quad y = Cx + Dw$$

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -k_1 & k_1 & 0 & 0 & 0 & 0 \\ k_1 & -k_1-k_2 & k_2 & 0 & 0 & 0 \\ 0 & k_2 & -k_2 & 0 & 0 & 0 \end{bmatrix}, \quad B_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad C = C_0 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix},$$

$$D = D_0 = 0$$

nominal values for k_1 & k_2

$k_1, k_2 \in [1-\alpha, 1+\alpha]$; $\alpha = 0.1 \leftarrow$ perturbation/uncertainty

$\Rightarrow P = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}$

$\Rightarrow k_1, k_2 \in [0.9, 1.1]$

$A = A_0 + k_1 A_1 + k_2 A_2 \leftarrow$ Affine system representation

$$A_0 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\leftarrow$ nominal part of system without any uncertainties / variations

$$A_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\leftarrow$ part of system with uncertainties or variations w.r.t k_1

$$A_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \end{bmatrix}$$

$\leftarrow$ part of system with uncertainties or variations w.r.t. k_2

Check open-loop system stability using "quadstab" function
↳ Confirm that the open-loop uncertain system is NOT quadratically stable.

Design LQR stabilizing controller with unit weights (Q, R) for the nominal system

$$K = -\mathrm{lqr}(\underbrace{A}_{A_0+A_1+A_2},\ \underbrace{B}_{B_0},\ \mathrm{eye}(6),\ 1)$$

} nominal system

State-feedback control law

$$u = Kx$$

Closed-loop system:

$$A_{cL} = A + BK$$

$$= (A_0 + k_1 A_1 + k_2 A_2) + BK \quad \ldots \text{(affine representation)}$$

$$B_{cL} = 0 \quad \ldots \text{(state feedback)}$$

Check closed-loop system stability using "quadstab" function

To determine maximum region of quadratic stability of the closed-loop system, use "quadstab" with option(1) = 1, i.e., quadstab($\underbrace{\ldots}_{\text{system}}$, [1 0 0]) ← ONLY for affine systems

↓

get the expansion factor

↓

compute maximum α for each parameter

↳ For given problem, ~~since~~ we have the same $\alpha = \alpha_{max}$ for both parameters (k_1 & k_2) such that the closed loop uncertain system is stable for all perturbations in the interval $[1-\alpha_{max}, 1+\alpha_{max}]$

1 ← nominal values of k_1 & k_2

↓

Compute updated lower & upper bounds for k_1 & k_2 using α_{max}.

Confirm that the closed-loop uncertain system is quadratically stable for all upper & lower bounds of the stiffness values. using "quadstab".

Problem 2

CODE:

```
% PROBLEM 2


% Clear workspace
close all
clear
clc


% System matrices in affine form
A0 = [0 0 0 1 0 0; 0 0 0 0 1 0; 0 0 0 0 0 1; 0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0
0];
A1 = [0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0; -1 1 0 0 0 0; 1 -1 0 0 0 0; 0 0 0 0
0 0];
A2 = [0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0; 0 -1 1 0 0 0; 0 1 -1 0
0 0];
B0 = [0; 0; 0; 1; 0; 0];
C0 = [0 0 1 0 0 0];
D0 = 0;


% Nominal system parameters
k1_nominal = 1;
k2_nominal = 1;


% Uncertain LTI system in affine form
S0 = ltisys(A0, B0, C0, D0, 1);
S1 = ltisys(A1, zeros(size(B0)), zeros(size(C0)), zeros(size(D0)), 0);
S2 = ltisys(A2, zeros(size(B0)), zeros(size(C0)), zeros(size(D0)), 0);


% Uncertainty bounds
alpha = 0.1;
LB = [k1_nominal - alpha, k2_nominal - alpha];
UB = [k1_nominal + alpha, k2_nominal + alpha];


% Parameter vector
P = pvec('box', [LB(1), UB(1); LB(2), UB(2)]);


% Affine system
affsys = psys(P, [S0, S1, S2]);
```

```matlab
% Confirm that the open-loop uncertain system is not quadratically stable
result = quadstab(affsys)
if result < 0
    disp('The open-loop uncertain system is quadratically stable.');
else
    disp('The open-loop uncertain system is NOT quadratically stable.');
end
```

```matlab
% Nominal system representation
A = A0 + A1 + A2;
B = B0;

% LQR control with unit weights
K = -lqr(A, B, eye(6), 1) % Q = I, R = 1

% Closed loop system considering state-feedback control law u = Kx
Acl = A + B*K

% Uncertain closed-loop LTI system in affine form
S0cl = ltisys(Acl, zeros(size(B0)), C0, D0, 1);
S1cl = ltisys(A1, zeros(size(B0)), zeros(size(C0)), zeros(size(D0)), 0);
S2cl = ltisys(A2, zeros(size(B0)), zeros(size(C0)), zeros(size(D0)), 0);

% Uncertainty bounds
alpha = 0.1;
LB = [k1_nominal - alpha, k2_nominal - alpha];
UB = [k1_nominal + alpha, k2_nominal + alpha];

% Parameter vector
P = pvec('box', [LB(1), UB(1); LB(2), UB(2)]);

% Affine closed loop system
affsys_cl = psys(P, [S0cl, S1cl, S2cl]);

% Determine if the closed-loop uncertain system is quadratically stable
result = quadstab(affsys_cl)
if result < 0
    disp('The closed-loop uncertain system is quadratically stable.');
```

```
else
    disp('The closed-loop uncertain system is NOT quadratically stable.');
end
```

```
% Determine the maximum region of quadratic stability of the closed-loop
uncertain system
expansion_factor = quadstab(affsys_cl, [1 0 0]) % Compute expansion factor


% Find the maximum α = α_max such that the uncertain system is quadratically
stable for all stiffness perturbations in the interval [nominal−α_max,
nominal+α_max]
k1_side = UB(1)-LB(1);
k2_side = UB(2)-LB(2);
k1_side_scaled = expansion_factor*k1_side;
k2_side_scaled = expansion_factor*k2_side;
k1_side_diff = k1_side_scaled-k1_side;
k2_side_diff = k2_side_scaled-k2_side;
LB_stable_1 = LB(1)-(k1_side_diff/2);
UB_stable_1 = UB(1)+(k1_side_diff/2);
LB_stable_2 = LB(2)-(k2_side_diff/2);
UB_stable_2 = UB(2)+(k2_side_diff/2);
alpha_max_1 = UB_stable_1 - k1_nominal % (or k1_nominal - LB_stable_1) α_max for
k1
alpha_max_2 = UB_stable_2 - k2_nominal % (or k2_nominal - LB_stable_2) α_max for
k2


% Uncertainty bounds for which the closed-loop system is stable
LB_stable = [k1_nominal - alpha_max_1, k2_nominal - alpha_max_2];
UB_stable = [k1_nominal + alpha_max_1, k2_nominal + alpha_max_2];


% Parameter vector
P_stable = pvec('box', [LB_stable(1), UB_stable(1); LB_stable(2), UB_stable(2)]);


% Affine stable closed loop system
affsys_stable = psys(P_stable, [S0cl, S1cl, S2cl]);


% Confirm that the closed-loop uncertain system is quadratically stable for all
upper and lower bounds of the stiffness values
result = quadstab(affsys_stable)
if result < 0
    disp('The closed-loop uncertain system is quadratically stable for all upper
and lower bounds of the stiffness values.');
```

```
else
    disp('The closed-loop uncertain system is NOT quadratically stable for all upper and lower bounds of the stiffness values.');
end
```

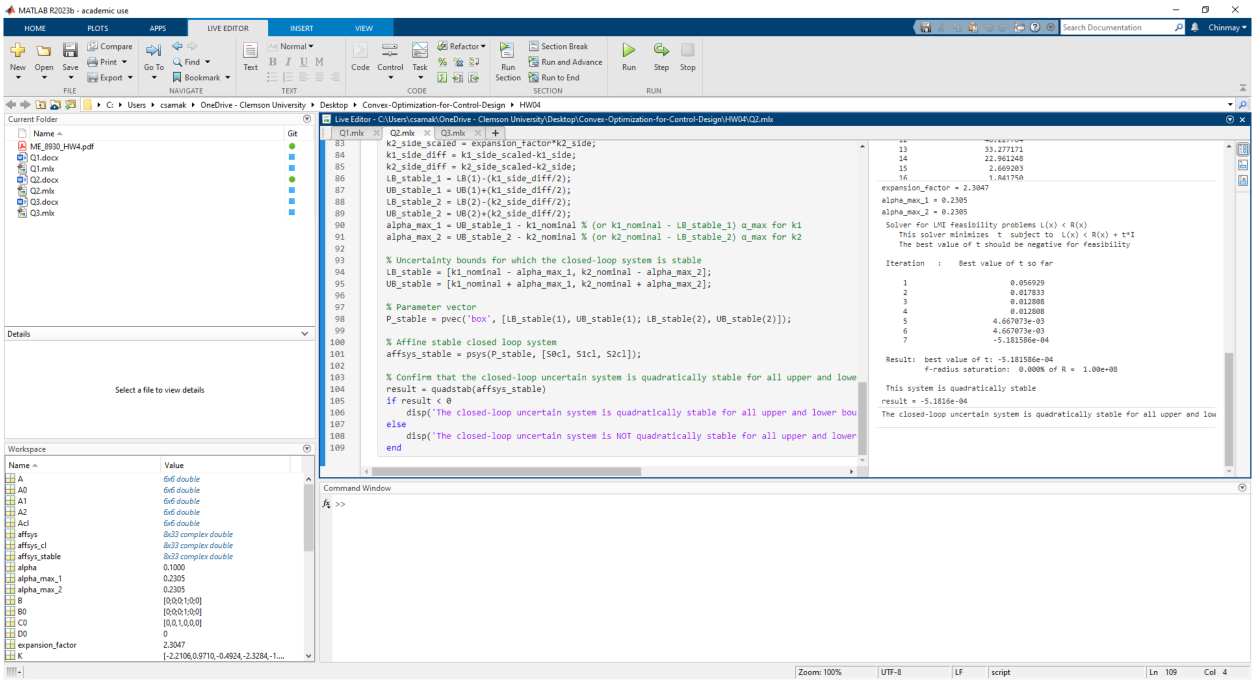
OUTPUT:

```
Solver for LMI feasibility problems L(x) < R(x)
    This solver minimizes  t  subject to  L(x) < R(x) + t*I
    The best value of t should be negative for feasibility

 Iteration   :    Best value of t so far

     1                        0.108566
     2                        0.013617
     3                        0.012804
     4                    8.313415e-03
     5                    8.313415e-03
     6                    9.585323e-04
     7                    9.585323e-04
     8                    3.127972e-04
     9                    3.127972e-04
    10                    2.848959e-04
    11                    2.848959e-04
    12                    2.341144e-04
    13                    2.256522e-04
    14                    2.256522e-04
    15                    2.111485e-04
* switching to QR
    16                    2.111485e-04
    17                    2.071661e-04
    18                    2.071661e-04
    19                    2.067091e-04
    20                    2.064719e-04
    21                    2.064719e-04
    22                    2.064440e-04
    23                    2.064440e-04
    24                    2.064329e-04
    25                    2.064329e-04
    26                    2.064329e-04
    27                    2.064329e-04
***                 new lower bound: 2.064286e-04

 Result:  best value of t: 2.064329e-04
          guaranteed absolute accuracy:  4.27e-09
          f-radius saturation:  1.383% of R =  1.00e+08

 Marginal infeasibility: these LMI constraints may be
          feasible but are not strictly feasible

 This system is not quadratically stable
result = 2.0643e-04
The open-loop uncertain system is NOT quadratically stable.
K = 1×6
   -2.2106    0.9710   -0.4924   -2.3284   -1.3671   -1.3048
```

```
Acl = 6×6
         0         0         0    1.0000         0         0
         0         0         0         0    1.0000         0
         0         0         0         0         0    1.0000
   -3.2106    1.9710   -0.4924   -2.3284   -1.3671   -1.3048
    1.0000   -2.0000    1.0000         0         0         0
         0    1.0000   -1.0000         0         0         0
 Solver for LMI feasibility problems L(x) < R(x)
    This solver minimizes  t  subject to  L(x) < R(x) + t*I
    The best value of t should be negative for feasibility

 Iteration   :    Best value of t so far

     1                        0.051882
     2                       -0.127668

 Result:  best value of t:    -0.127668
          f-radius saturation:  0.000% of R =  1.00e+08

 This system is quadratically stable
result = -0.1277
The closed-loop uncertain system is quadratically stable.
 Solver for generalized eigenvalue minimization
 Iterations   :    Best objective value so far
      1
     2
     3
     4
     5
     6
     7
     8                 309.375000
     9                 146.808105
    10                 101.297593
    11                  69.895339
    12                  48.227784
    13                  33.277171
    14                  22.961248
    15                   2.669203
    16                   1.841750
    17                   1.270807
    18                   0.876857
    19                   0.491696
    20                   0.491696
    21                   0.491696
    22                   0.486779
    23                   0.481911
    24                   0.477092
    25                   0.472321
    26                   0.467598
    27                   0.462922
    28                   0.458293
    29                   0.458293
    30                   0.453710
    31                   0.449173
***                  new lower bound:     0.313060
    32                   0.449173
```

```
    33                   0.436412
***                new lower bound:     0.317039
    34                   0.436412
    35                   0.434547
    36                   0.434547
***                new lower bound:     0.375793
    37                   0.434547
***                new lower bound:     0.405170
    38                   0.433892
***                new lower bound:     0.420215

 Result:  feasible solution
          best value of t:     0.433892
          guaranteed absolute accuracy:  1.37e-02
          f-radius saturation:  0.000% of R =  1.00e+08
 Termination due to SLOW PROGRESS:
          the gen. eigenvalue t decreased by less than
          1.000% during the last 5 iterations.
 Quadratic stability established on  230.4722%  of the
 prescribed parameter box
expansion_factor = 2.3047
alpha_max_1 = 0.2305
alpha_max_2 = 0.2305
 Solver for LMI feasibility problems L(x) < R(x)
    This solver minimizes  t  subject to  L(x) < R(x) + t*I
    The best value of t should be negative for feasibility

 Iteration   :    Best value of t so far

     1                        0.056929
     2                        0.017833
     3                        0.012808
     4                        0.012808
     5                    4.667073e-03
     6                    4.667073e-03
     7                    -5.181586e-04

 Result:  best value of t: -5.181586e-04
          f-radius saturation:  0.000% of R =  1.00e+08

 This system is quadratically stable
result = -5.1816e-04
The closed-loop uncertain system is quadratically stable for all upper and
lower bounds of the stiffness values.
```

SCREENSHOT:


MATLAB R2023b - academic use
Live Editor - C:\Users\csamak\OneDrive - Clemson University\Desktop\Convex-Optimization-for-Control-Design\HW04\Q2.mlx
Q1.mlx
Q2.mlx
Q3.mlx
k2_side_scaled = expansion_factor*k2_side;
k1_side_diff = k1_side_scaled-k1_side;
k2_side_diff = k2_side_scaled-k2_side;
LB_stable_1 = LB(1)-(k1_side_diff/2);
UB_stable_1 = UB(1)+(k1_side_diff/2);
LB_stable_2 = LB(2)-(k2_side_diff/2);
UB_stable_2 = UB(2)+(k2_side_diff/2);
alpha_max_1 = UB_stable_1 - k1_nominal % (or k1_nominal - LB_stable_1) α_max for k1
alpha_max_2 = UB_stable_2 - k2_nominal % (or k2_nominal - LB_stable_2) α_max for k2
% Uncertainty bounds for which the closed-loop system is stable
LB_stable = [k1_nominal - alpha_max_1, k2_nominal - alpha_max_2];
UB_stable = [k1_nominal + alpha_max_1, k2_nominal + alpha_max_2];
% Parameter vector
P_stable = pvec('box', [LB_stable(1), UB_stable(1); LB_stable(2), UB_stable(2)]);
% Affine stable closed loop system
affsys_stable = psys(P_stable, [S0cl, S1cl, S2cl]);
% Confirm that the closed-loop uncertain system is quadratically stable for all upper and lowe
result = quadstab(affsys_stable)
if result < 0
disp('The closed-loop uncertain system is quadratically stable for all upper and lower bou
else
disp('The closed-loop uncertain system is NOT quadratically stable for all upper and lower
end
13 33.277171
14 22.961248
15 2.669203
expansion_factor = 2.3047
alpha_max_1 = 0.2305
alpha_max_2 = 0.2305
Solver for LMI feasibility problems L(x) < R(x)
This solver minimizes t subject to L(x) < R(x) + t*I
The best value of t should be negative for feasibility
Iteration : Best value of t so far
1 0.056929
2 0.017833
3 0.012808
4 0.012808
5 4.667073e-03
6 4.667073e-03
7 -5.181586e-04
Result: best value of t: -5.181586e-04
f-radius saturation: 0.000% of R = 1.00e+08
This system is quadratically stable
result = -5.1816e-04
The closed-loop uncertain system is quadratically stable for all upper and low
Command Window
Workspace
expansion_factor 2.3047
alpha 0.1000
alpha_max_1 0.2305
alpha_max_2 0.2305

$$\dot{x}_p = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x_p + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} w$$

$$y = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x_p + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

$$z = \begin{bmatrix} 1 & 0 \end{bmatrix} x_p + \begin{bmatrix} 0 & 1 \end{bmatrix} w$$

System matrices.

$$A_p = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B_p = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad D_p = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

$$C_p = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B_y = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad D_y = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$M_p = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D_z = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

$r = [n_z, n_u]$

Lumped system matrices

$A = A_p$

$B = [D_p \quad B_p]$

$C = [C_p \quad M_p]$

$$D = \begin{bmatrix} D_y & B_y \\ D_z & 0 \end{bmatrix}$$

Open-loop system

$S_{ol} = ss(A, B, C, D)$

Inputs to system:

$$w = \begin{cases} 0.1 & ; \ 0 \leq t \leq 1 \\ 0 & ; \ \text{otherwise} \end{cases}$$

$u = 0 \quad \forall t$

w is a 2×1 vector since the disturbance input $w(t)$ consists of plant disturbance and measurement disturbance

Simulate open-loop system using "lsim" for $t = 0$ to $t = 10$ seconds

H_∞ Dynamic Controller Design in MATLAB:

$s = \text{ltisys}(A, B, C, D)$

$r = [n_z, n_u]$

$[g_{opt}, G] = \text{hinflmi}(s, r)$

$\Gamma_{ee} < \gamma^*$

controller

$[A_c, B_c, C_c, D_c] = \text{ltiss}(G)$

Closed-loop system:

$$A_{cl} = \begin{bmatrix} A_p + B_p D_c M_p & B_p C_c \\ B_c M_p & A_c \end{bmatrix}.$$

$$B_{cl} = \begin{bmatrix} D_p + B_p D_c D_z \\ B_c D_z \end{bmatrix}$$

$$C_{cl} = \begin{bmatrix} C_p + B_y D_c M_p & B_y C_c \end{bmatrix}$$

$$D_{cl} = D_y + B_y D_c D_z$$

Examination/validation of system:

- Stability: Eigenvalues (Poles): $Re(\lambda_i(A_{cl})) < 0$ ← stability guaranteed

- Performance: H_∞ norm (energy-to-energy gain Γ_{ee}): $S_{cl} = ss(A_{cl}, B_{cl}, C_{cl}, D_{cl})$
  hinf_norm = hinfnorm (S_{cl})
  hinf_norm $\lessapprox$ g_{opt} } MATLAB
  ↑ γ^* $(\Gamma_{ee} < \gamma^*)$

Inputs to closed-loop system:

$$w = \begin{cases} 0.1 ; & 0 \leq t \leq 1 \\ 0 ; & \text{otherwise} \end{cases}$$

$$u = 0 \quad \forall t$$

w is a 2×1 vector since the disturbance input $w(t)$ consists of plant disturbance and measurement disturbance

Simulate closed-loop system using "lsim" for $t = 0$ to $t = 10$ seconds

Problem 3

CODE:

```
% PROBLEM 3


% Clear workspace
close all
clear
clc


% Define the system matrices
Ap = [0 1; 0 0];
Bp = [0; 1];
Dp = [0 0; 1 0];
Cp = [0 1; 0 0];
By = [0; 1];
Dy = [0 0; 0 0];
Mp = [1 0];
Dz = [0 1];


% Lumped system matrices
A = Ap;
B = [Dp Bp];
C = [Cp; Mp];
D = [Dy By; Dz 0]


% State-space system
Sol = ss(A, B, C, D)
% sys_openloop = ss(Ap, Dp, Cp, 0)


% Define the lumped (disturbance + control) input
t = 0:0.01:10;
w_amplitude = 0.1;
w_duration = 1;
w_pulse = w_amplitude * (t >= 0 & t <= w_duration);
u = zeros(size(w_pulse));
w = [w_pulse; w_pulse; u];
% w = [w_pulse; w_pulse];


% Simulate the open-loop system response
```

```
[y_ol, t_out, x_ol] = lsim(Sol, w', t);
% [y_ol, t_out, x_ol] = lsim(Sol, w, t);


% Plot the results
figure;
sgtitle('Open-Loop System Response');
subplot(4, 1, 1);
plot(t, w(2, :), t, w(3, :));
legend('W', 'U');
subplot(4, 1, 2);
plot(t, x_ol(:, 1), t, x_ol(:, 2));
legend('X1', 'X2');
subplot(4, 1, 3);
plot(t, y_ol(:, 1), t, y_ol(:, 2));
legend('Y1', 'Y2');
subplot(4, 1, 4);
plot(t, y_ol(:, 3));
legend('Z');
```

```
% H-infinity LMI
S = ltisys(A, B, C, D);
[gopt, G] = hinflmi(S,[1 1])


% Controller matrices
disp('H∞ controller:')
[Ac, Bc, Cc, Dc] = ltiss(G)
```

```
% Closed-loop system matrices
disp('Closed-loop system:')
Acl = [Ap+Bp*Dc*Mp, Bp*Cc; Bc*Mp, Ac]
Bcl = [Dp+Bp*Dc*Dz; Bc*Dz]
Ccl = [Cp+By*Dc*Mp, By*Cc]
Dcl = Dy+By*Dc*Dz


% Closed-loop system
Scl = ss(Acl, Bcl, Ccl, Dcl);
% Scl = slft(S, G);


% Verification
disp('H∞ norm:')
hinf_norm = hinfnorm(Scl)
```

```
% hinf_norm = norminf(Scl)
disp('Closed-loop poles:')
eig_Acl = eig(Acl)
if((hinf_norm < gopt) && all(real(eig_Acl) < 0.0))
    disp('Verification of H∞ norm and pole location constraints successful!')
else
    disp('Verification of H∞ norm and pole location constraints failed!')
end
```

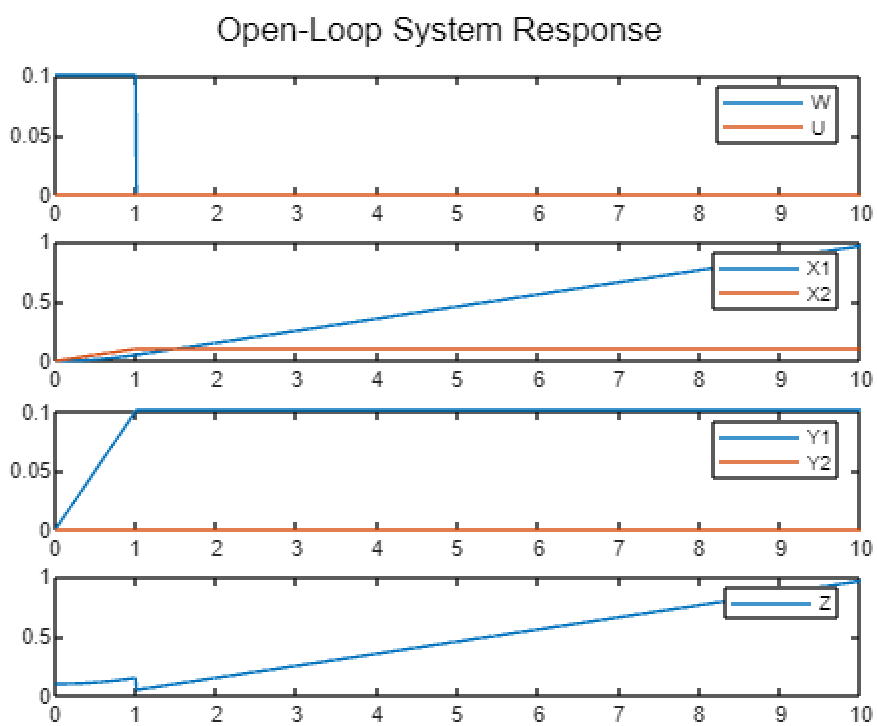
```
% Define the lumped (disturbance + control) input
t = 0:0.01:10;
w_amplitude = 0.1;
w_duration = 1;
w_pulse = w_amplitude * (t >= 0 & t <= w_duration);
u = zeros(size(w_pulse));
w = [w_pulse; w_pulse];


% Simulate the closed-loop system response
[y_cl, t_out, x_cl] = lsim(Scl, w', t);


% Plot the results
figure;
sgtitle('Closed-Loop System Response');
subplot(3, 1, 1);
plot(t, w(2, :));
legend('W');
subplot(3, 1, 2);
plot(t, x_cl(:, 1), t, x_cl(:, 2), t, x_cl(:, 3));
legend('X1', 'X2', 'X3');
subplot(3, 1, 3);
plot(t, y_cl(:, 1), t, y_cl(:, 2));
legend('Y1', 'Y2');
```

OUTPUT:

```
D = 3×3
     0     0     0
     0     0     1
     0     1     0
Sol =

  A =
       x1   x2
   x1   0    1
   x2   0    0
```

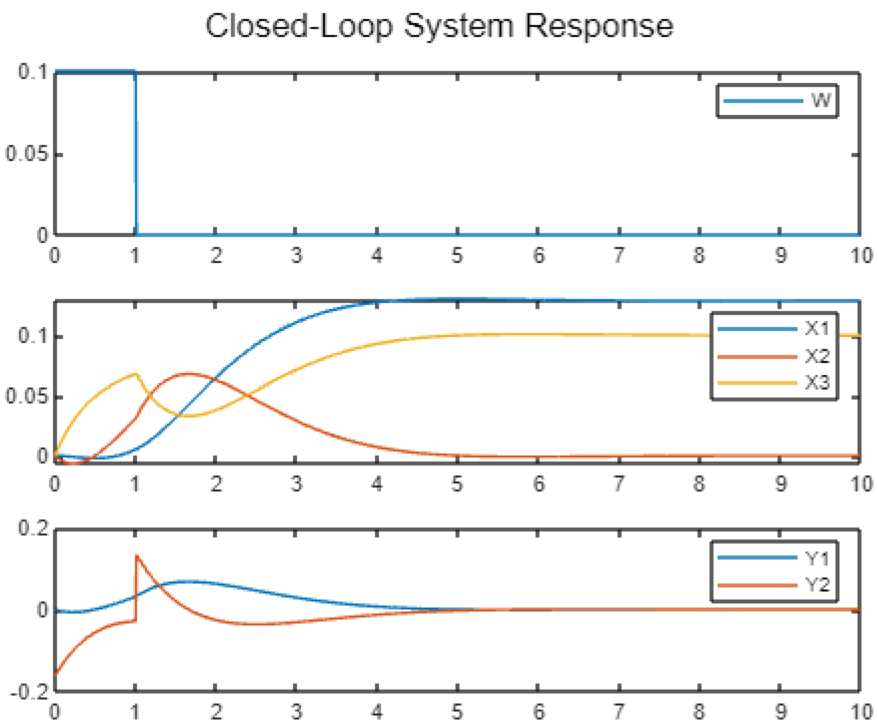
```
  B =
         u1   u2   u3
   x1     0    0    0
   x2     1    0    1

  C =
         x1   x2
   y1     0    1
   y2     0    0
   y3     1    0

  D =
         u1   u2   u3
   y1     0    0    0
   y2     0    0    1
   y3     0    1    0

Continuous-time state-space model.
```

Model Properties

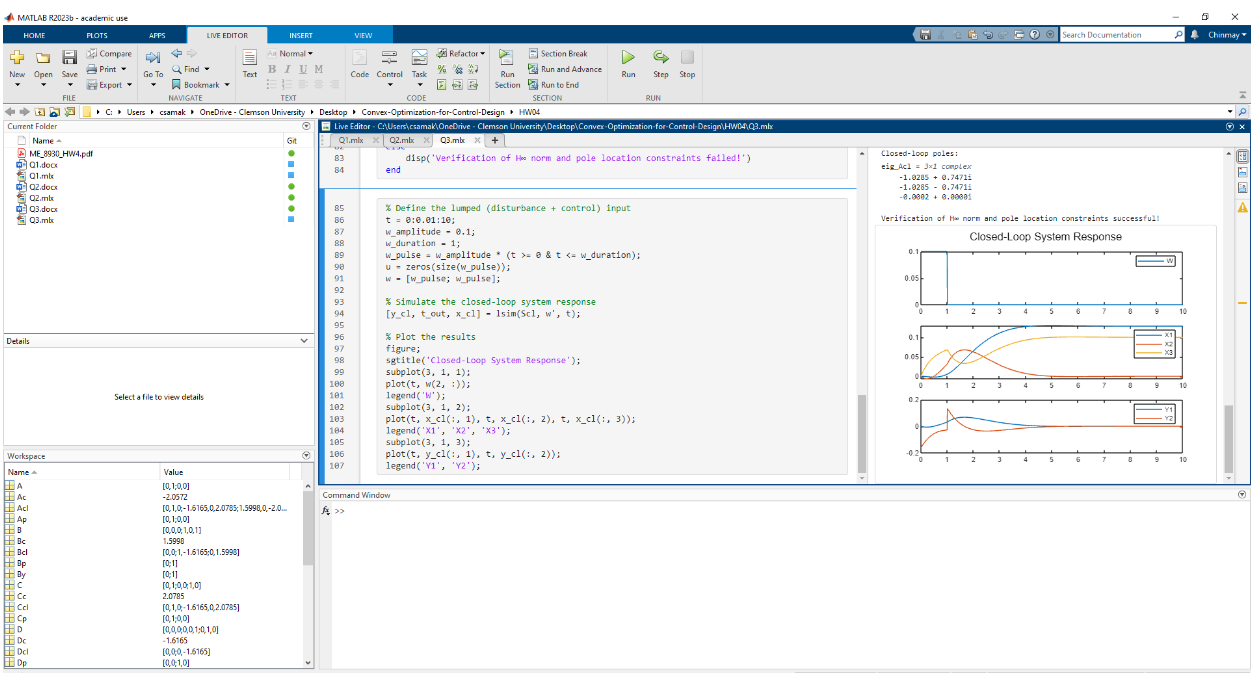



```
 Minimization of gamma:

 Solver for linear objective minimization under LMI constraints

 Iterations    :    Best objective value so far

     1
     2                      4.331487
     3                      2.480378
     4                      2.094585
     5                      1.984537
```

```
     6                   1.984537
     7                   1.891906
     8                   1.891906
***                 new lower bound:     0.208335
     9                   1.693528
    10                   1.693528
***                 new lower bound:     0.773250
    11                   1.636185
***                 new lower bound:     1.240010
    12                   1.636185
***                 new lower bound:     1.567929
    13                   1.622326
    14                   1.620133
***                 new lower bound:     1.604742

 Result:  feasible solution of required accuracy
          best objective value:     1.620133
          guaranteed relative accuracy:  9.50e-03
          f-radius saturation:  0.399% of R =  1.00e+08

 Optimal Hinf performance:  1.620e+00
gopt = 1.6197
G = 3×3
   -2.0572    1.5998    1.0000
    2.0785   -1.6165         0
         0         0      -Inf
H∞ controller:
Ac = -2.0572
Bc = 1.5998
Cc = 2.0785
Dc = -1.6165
Closed-loop system:
Acl = 3×3
         0    1.0000         0
   -1.6165         0    2.0785
    1.5998         0   -2.0572
Bcl = 3×2
         0         0
    1.0000   -1.6165
         0    1.5998
Ccl = 2×3
         0    1.0000         0
   -1.6165         0    2.0785
Dcl = 2×2
         0         0
         0   -1.6165
H∞ norm:
hinf_norm = 1.6185
Closed-loop poles:
eig_Acl = 3×1 complex
  -1.0285 + 0.7471i
  -1.0285 - 0.7471i
  -0.0002 + 0.0000i
Verification of H∞ norm and pole location constraints successful!
```


Closed-Loop System Response
W
X1
X2
X3
Y1
Y2

SCREENSHOT:


MATLAB R2023b - academic use
Live Editor - C:\Users\csamak\OneDrive - Clemson University\Desktop\Convex-Optimization-for-Control-Design\HW04\Q3.mlx
Q1.mlx
Q2.mlx
Q3.mlx
disp('Verification of H∞ norm and pole location constraints failed!')
end
% Define the lumped (disturbance + control) input
t = 0:0.01:10;
w_amplitude = 0.1;
w_duration = 1;
w_pulse = w_amplitude * (t >= 0 & t <= w_duration);
u = zeros(size(w_pulse));
w = [w_pulse; w_pulse];
% Simulate the closed-loop system response
[y_cl, t_out, x_cl] = lsim(Scl, w', t);
% Plot the results
figure;
sgtitle('Closed-Loop System Response');
subplot(3, 1, 1);
plot(t, w(2, :));
legend('W');
subplot(3, 1, 2);
plot(t, x_cl(:, 1), t, x_cl(:, 2), t, x_cl(:, 3));
legend('X1', 'X2', 'X3');
subplot(3, 1, 3);
plot(t, y_cl(:, 1), t, y_cl(:, 2));
legend('Y1', 'Y2');
Closed-loop poles:
Verification of H∞ norm and pole location constraints successful!
Closed-Loop System Response
Command Window