Class: Robust Multivariate Control
Professor: Dr. Sean Humbert
TAs: Santosh Chaganti
Student: Steve Gillet
Date: February 18, 2025
Assignment: Homework 3

1. D-Stability of a System

”In lecture it was shown that the LMI to verify stability of the system $\dot{x} = Ax$ is given by:

$$\begin{cases} P > 0 \\ A^T P + PA < 0, \end{cases}$$

which verifies that the eigenvalues of A are in the left half plane. Modifications can be made to this LMI that provide conditions to verify the eigenvalues of A are in a subset $\mathbb{D}$ of the left half plane, which is called $\mathbb{D}$-stability. The simplest case is to verify $\lambda_i(A)$ are in region that lies to the left of some value of α in the left half plane. In this case the resulting LMI conditions are:

$$\begin{cases} P > 0 \\ A^T P + PA + 2\alpha P < 0. \end{cases}$$

Code up this new LMI as a `feasp` problem using the MATLAB LMI Toolbox and use it to estimate the largest region of $\mathbb{D}$-stability for the following systems $\dot{x} = Ax$ where

$$A = \begin{pmatrix} 0 & 1 \\ -1 & -2 \end{pmatrix}, \quad A = \begin{pmatrix} -1.5 & 1 & 0.1 \\ -4 & -1 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

”

I used the following code to set up the feasability problem.

```
A = [0 1; -1 -2];
nStates = size(A, 1);

setlmis([])

```

```
P = lmivar(1, [nStates 1]);
alpha = lmivar(2, [1 1]);

lmiterm([-1 1 1 P],1,1);
lmiterm([2 1 1 P],A',1,'s');
lmiterm([2 1 1 alpha],2,1);

lmiDstab = getlmis;

[tmin, alphaFeas] = feasp(lmiDstab);
alpha = dec2mat(lmiDstab,alphaFeas,alpha);
disp(alpha);
```

The output for the first A matrix is:

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 -0.603359

Result: best value of t: -0.603359 f-radius saturation: 0.000% of R = 1.00e+09

-1 -1

The output for the second A matrix is:

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 -0.255991

Result: best value of t: -0.255991 f-radius saturation: 0.000% of R = 1.00e+09

-0.4390

2. Spectral Norm of a Matrix

"In lecture we derived the following LMI representing the inequality $\|A\|_2 = \sigma(A) < \gamma$,

$$\begin{pmatrix} \gamma^2 I & A^T \\ A & I \end{pmatrix} > 0.$$

*Code up this LMI as a **mincx** problem using the MATLAB LMI Toolbox. Assume the minimization variable is $\rho = \gamma^2$:*

$$\begin{cases} \min \rho \\ \begin{pmatrix} \rho I & A^T \\ A & I \end{pmatrix} > 0. \end{cases}$$

For the following matrices, use the code above to estimate $\sigma(A)$. Remember to back solve for $\gamma = \sqrt{\rho}$.

$$A_1 = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}, \quad A_2 = \begin{pmatrix} -1 & 1 & 0 \\ 1 & -1 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

"

I used the following Matlab code:

```
% A = [2 -1; -1 2];
A = [-1 1 0; 1 -1 1; 0 1 1];

nStates = size(A,1);

setlmis([])

rho = lmivar(2, [1 1]);

lmiterm([-1 1 1 rho],1,1);
lmiterm([2 1 1 rho],1,1);
lmiterm([2 1 2 0],A');
lmiterm([2 2 1 0],A);
lmiterm([2 2 2 0],1);

spectralLmi = getlmis;

[copt, xopt] = mincx(spectralLmi, rho);
rhoVal = dec2mat(spectralLmi, xopt, rho);
gamma = sqrt(rhoVal);
```

And got infeasibility for both matrices. Perhaps because they are both unstable, perhaps because I'm missing something in the set up. I was a bit confused on setting up the second LMI.

3. H_∞ Norm of a System

"In this problem, you will use an alternate LMI form for the Bounded Real Lemma and implement it as a generalized eigenvalue problem (gevp) using the MATLAB LMI Toolbox. If we start with part (c) from the Bounded Real Lemma theorem from lecture and move the γ^2 term to the right side, we have

$$\begin{pmatrix} A^TP + PA + C^TC & PB + C^TD \\ B^TP + D^TC & D^TD \end{pmatrix} < \gamma^2 \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

In this form with P as the matrix variable, it can be cast as a generalized eigenvalue problem

$$\begin{cases} \min \lambda \\ A(x) < \lambda B(x) \\ B(x) > 0 \\ C(x) > 0, \end{cases}$$

where $\lambda = \gamma^2$, $A(x)$ is the left hand side of the above expression, $C(x) = P$, and

$$B(x) = \begin{pmatrix} \epsilon & 0 \\ 0 & 1 \end{pmatrix},$$

Here we need to add a small number $\epsilon \approx 0.00001$ *to the matrix so that it is positive definite for numerical stability.*

(a) Implement the above LMI as a function that accepts matrices A, B, C, *and* D *from a general state space representation* $\dot{x} = Ax + Bu, y = Cx + Du$ *of a system* G *and uses the function* `gevp` *to compute the infinity norm for the system,* $\|G\|_\infty$. *Be sure to back solve for* $\gamma = \sqrt{\lambda}$ *in your function.*

(b) Use your code from (a) to compute $\|G\|_\infty$ *for the following stable (from Problem 1) dynamical systems. You can check your results using the MATLAB function* `hinfsyn(sys)` *where* `sys` *is a state space object constructed using the command* `ss`.

$$\dot{x} = \begin{pmatrix} 0 & 1 \\ -1 & -2 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u, \quad \dot{x} = \begin{pmatrix} -1.5 & 1 & 0.1 \\ -4 & -1 & 0 \\ 1 & 0 & 0 \end{pmatrix} x + \begin{pmatrix} -0.2 \\ -1.8 \\ 0 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} x, \qquad y = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} u$$

"

This is how I set the problem up in Matlab:

```
% A = [-1.5 1 0.1; -4 -1 0; 1 0 0];
% B = [-0.2; -1.8; 0];
% C = [1 0 0; 0 1 0; 0 0 1];
% D = [1; 0; 0];

A = [0 1; -1 -2];
B = [0; 1];
C = [1 0; 0 2];
D = [0];

nStates = size(A,1);
nOutputs = size(C,1);
nInputs = size(B,1);

setlmis([]);

P = lmivar(1, [nStates 1]);

epsilon = 1e-5;

lmiterm([1 1 1 P], A', 1, 's');
lmiterm([1 1 1 0], C'*C);
lmiterm([1 1 2 P], 1, B);
lmiterm([1 1 2 0], C'*D);
lmiterm([1 2 1 P], B', 1);
lmiterm([1 2 1 0], D'*C);
```

```
lmiterm([1 2 2 0], D'*D);

lmiterm([-2 1 1 0], epsilon);
lmiterm([-2 2 2 0], 1);

lmiterm([-3 1 1 P], 1, 1);

lmisys = getlmis;

[aa, xopt] = gevp(lmisys, 2);

if isempty(xopt)
    gamma = Inf;
else
    gamma = sqrt(xopt);
end

disp(gamma);
```

And this was the result for the first system:

Solver for generalized eigenvalue minimization Iterations : Best objective value so far 1 2 3 4 5 * upper bound on optimal value set to 1.00e+08 6 7 8 9 10 * upper bound on optimal value set to 1.00e+11 11 12 13 14 15 16

Result: infeasibility Inf

And this was the result for the second system:

Solver for generalized eigenvalue minimization Iterations : Best objective value so far 1 2 3 4 5 6 * upper bound on optimal value set to 1.00e+08 7 8 9 10 11 * upper bound on optimal value set to 1.00e+11 12 13 14 15 16 17

Result: infeasibility Inf