ECE 675—Linear Matrix Inequalities

Stan Žak

School of Electrical and Computer Engineering
Purdue University
zak@purdue.edu

March 9, 2022
Outline

- Motivation
- Definitions of convex set and convex function
- Linear matrix inequality (LMI)
- Canonical LMI
- Example of LMIs
- LMI solvers
The Lyapunov theorem

Lyapunov’s thm:

A constant square matrix $A \in \mathbb{R}^{n \times n}$ has its eigenvalues in the open left half-complex plane if and only if for any real, symmetric, positive definite $Q \in \mathbb{R}^{n \times n}$, the solution $P = P^\top$ to the Lyapunov matrix equation

$$A^\top P + PA = -Q$$

is positive definite.
The Lyapunov thm re-stated

**Lyapunov’s thm:**
The real parts of the eigenvalues of $A$ are all negative if and only if there exists a real symmetric positive definite matrix $P$ such that

$$A^T P + PA < 0$$

Equivalently,

$$-A^T P - PA > 0$$
\( A^T P + PA \prec 0 \) re-stated

- Let
  
  \[
  P = \begin{bmatrix}
  x_1 & x_2 & \cdots & x_n \\
  x_2 & x_{n+1} & \cdots & x_{2n-1} \\
  \vdots & \vdots & \ddots & \vdots \\
  x_n & x_{2n-1} & \cdots & x_q
  \end{bmatrix}
  \]

- Define
  
  \[
  P_1 = \begin{bmatrix}
  1 & 0 & 0 & \cdots & 0 \\
  0 & 0 & 0 & \cdots & 0 \\
  0 & 0 & 0 & \cdots & 0 \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & 0 & \cdots & 0
  \end{bmatrix}, \quad P_2 = \begin{bmatrix}
  0 & 1 & 0 & \cdots & 0 \\
  1 & 0 & 0 & \cdots & 0 \\
  0 & 0 & 0 & \cdots & 0 \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & 0 & \cdots & 0
  \end{bmatrix}
  \]
Defining $P_i$s

Note that each $P_i$ has only non-zero elements corresponding to $x_i$ in $P$.

Let

$$F_i = -A^T P_i - P_i A, \quad i = 1, 2, \ldots, q$$
Manipulate

\[
A^T P + PA = x_1 \left( A^T P_1 + P_1 A \right)
+ x_2 \left( A^T P_2 + P_2 A \right)
+ \cdots + x_q \left( A^T P_q + P_q A \right)
\]
\[
= -x_1 F_1 - x_2 F_2 - \cdots - x_q F_q
\prec 0
\]

Let \( F(x) = x_1 F_1 + x_2 F_2 + \cdots + x_q F_q \)
Lyapunov’s Inequality restated

- \( P = P^\top \succ 0 \) and \( A^\top P + P A \prec 0 \)

if and only if

\( F(x) \succ 0 \)

Equivalently,

\[
\begin{bmatrix}
P & O \\
O & -A^\top P - PA
\end{bmatrix} \succ 0
\]
Linear Matrix Inequality

- Consider \( n + 1 \) real symmetric matrices
  \[
  F_i = F_i^\top \in \mathbb{R}^{m \times m}, \quad i = 0, 1, \ldots, n
  \]
  and a vector \( \mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^\top \)
- Construct an affine function
  \[
  F(\mathbf{x}) = F_0 + x_1 F_1 + \ldots + x_n F_n
  = F_0 + \sum_{i=1}^{n} x_i F_i
  \]
Linear Matrix Inequality—Definition

\[ F(x) = F_0 + x_1 F_1 + \cdots + x_n F_n \succ 0 \]

Find a set of vectors \( x \) such that

\[ z^\top F(x) z > 0 \quad \text{for all} \quad z \in \mathbb{R}^m, \ z \neq 0, \]

that is, \( F(x) \) is positive definite
Linear Matrix Inequality—Another Definition

\[ F(x) = F_0 + x_1F_1 + \cdots + x_nF_n \succeq 0 \]

Find a set of vectors \( x \) such that

\[ z^\top F(x)z \geq 0 \text{ for all } z \in \mathbb{R}^m, \]

that is, \( F(x) \) is positive semidefinite
Convex Set

**Definition**

A set \( \Omega \subseteq \mathbb{R}^n \) is convex if for any \( x \) and \( y \) in \( \Omega \), the line segment between \( x \) and \( y \) lies in \( \Omega \), that is,

\[
\alpha x + (1 - \alpha)y \in \Omega \quad \text{for any} \quad \alpha \in (0, 1)
\]
Convex Function

Definition

A real-valued function

\[ f : \Omega \rightarrow \mathbb{R} \]

defined on a convex set \( \Omega \subseteq \mathbb{R}^n \) is convex if for all \( x, y \in \Omega \) and all \( \alpha \in (0, 1) \),

\[ f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y) \]
Definition

A convex optimization problem is the one where the objective function to be minimized is convex and the constraint set, over which we optimize the objective function, is a convex set.

Warning: If $f$ is a convex function, then

$$\max f(x) \quad \text{subject to } x \in \Omega$$

is NOT a convex optimization problem!
Another LMI Example

A system of LMIs,

\[ F_1(x) \succeq 0, \quad F_2(x) \succeq 0, \ldots, \quad F_k(x) \succeq 0 \]

can be represented as one single LMI

\[
F(x) = \begin{bmatrix}
F_1(x) \\
F_2(x) \\
\vdots \\
F_k(x)
\end{bmatrix} \succeq 0
\]
Yet Another LMI Example

A linear matrix inequality, involving an $m$-by-$n$ constant matrix $A$, of the form,

$$Ax \leq b$$

can be represented as $m$ LMIs

$$b_i - a_i^\top x \geq 0, \quad i = 1, 2, \ldots, m,$$

where $a_i^\top$ is the $i$-th row of the matrix $A$.
Example # 2 Contd

- View each scalar inequality as an LMI
- Represent \( m \) LMIs as one LMI,

\[
F(x) = \begin{bmatrix}
    b_1 - a_1^\top x \\
    b_2 - a_2^\top x \\
    \vdots \\
    b_m - a_m^\top x
\end{bmatrix} \succeq 0
\]
A semidefinite programming problem is a convex optimization problem of the form

\[ \text{minimize} \quad c^\top x \]
\[ \text{subject to} \quad F(x) \succeq 0 \]

Note that the linear objective function \( c^\top x \) is a convex function and the constraint set, \( \{ x : F(x) \succeq 0 \} \), is a convex set.
Notation $> \text{ versus } \geq$

- Most of the optimization solvers do not handle strict inequalities.
- Therefore, the operator $>$ is the same as $\geq$, and so $>$ implements the non-strict inequality $\geq$. 
Lemma

Let $\mathbf{P} = \mathbf{P}^\top$ be a nonsingular $n$-by-$n$ matrix and let $\mathbf{x} = \mathbf{Mz}$, where $\mathbf{M} \in \mathbb{R}^{n\times n}$ such that $\text{det} \mathbf{M} \neq 0$. Then,

$$\mathbf{x}^\top \mathbf{P} \mathbf{x} \geq 0 \quad \text{if and only if} \quad \mathbf{z}^\top \mathbf{M}^\top \mathbf{P} \mathbf{M} \mathbf{z} \geq 0,$$

that is,

$$\mathbf{P} \succeq 0 \quad \text{if and only if} \quad \mathbf{M}^\top \mathbf{P} \mathbf{M} \succeq 0.$$

Similarly

$$\mathbf{P} \succ 0 \quad \text{if and only if} \quad \mathbf{M}^\top \mathbf{P} \mathbf{M} \succ 0.$$
The Schur Complements—Background Results

- Suppose have a square block matrix

\[
\begin{bmatrix}
A & B \\
B^\top & D
\end{bmatrix},
\]

where \( A = A^\top \) and \( D = D^\top \)

- Then, by the Lemma,

\[
\begin{bmatrix}
A & B \\
B^\top & D
\end{bmatrix} \succeq 0
\]

if and only if

\[
\begin{bmatrix}
O & I \\
I & O
\end{bmatrix} \begin{bmatrix}
A & B \\
B^\top & D
\end{bmatrix} \begin{bmatrix}
O & I \\
I & O
\end{bmatrix} \succeq 0,
\]

where \( I \) is an identity matrix of appropriate dimension
Background Results—Contd.

In other words,

\[
\begin{bmatrix}
A & B \\
B^T & D
\end{bmatrix} \succeq 0 \text{ if and only if } \begin{bmatrix}
D & B^T \\
B & A
\end{bmatrix} \succeq 0.
\]
The Schur Complements

- Given a square block matrix,
  \[ \begin{bmatrix}
  A_{11} & A_{12} \\
  A_{21} & A_{22}
  \end{bmatrix}, \]

  where \( A_{11} \) and \( A_{22} \) are square and symmetric submatrices, and \( A_{12} = A_{21}^\top \)

- Suppose that the matrix \( A_{11} \) is invertible

- Then
  \[
  \begin{bmatrix}
  I & O \\
  -A_{21}A_{11}^{-1} & I
  \end{bmatrix}
  \begin{bmatrix}
  A_{11} & A_{21}^\top \\
  A_{21} & A_{22}
  \end{bmatrix}
  \begin{bmatrix}
  I & -A_{11}^{-1}A_{21}^\top \\
  O & I
  \end{bmatrix}
  = \begin{bmatrix}
  A_{11} & O \\
  O & A_{22} - A_{21}A_{11}^{-1}A_{21}^\top
  \end{bmatrix}
  \]
Let
\[
\Delta_{11} = A_{22} - A_{21} A_{11}^{-1} A_{21}^\top.
\]
The matrix $\Delta_{11}$ is called the Schur complement of $A_{11}$.

By the Lemma
\[
\begin{bmatrix}
A_{11} & A_{21} \\
A_{21} & A_{22}
\end{bmatrix} \succeq 0 \text{ if and only if } \begin{bmatrix}
A_{11} & O \\
O & \Delta_{11}
\end{bmatrix} \succeq 0
\]

That is,
\[
\begin{bmatrix}
A_{11} & A_{21} \\
A_{21} & A_{22}
\end{bmatrix} \succeq 0 \text{ if and only if } A_{11} \succ 0 \text{ and } \Delta_{11} \succ 0
The Schur Complement of $\mathbf{A}_{22}$

- Let
  \[ \Delta_{22} = \mathbf{A}_{11} - \mathbf{A}_{12} \mathbf{A}_{22}^{-1} \mathbf{A}_{12}^\top. \]
  The matrix $\Delta_{22}$ is called the Schur complement of $\mathbf{A}_{22}$.
- By the Lemma
  \[
  \begin{bmatrix}
  \mathbf{A}_{11} & \mathbf{A}_{12} \\
  \mathbf{A}_{12}^\top & \mathbf{A}_{22}
  \end{bmatrix} \succ 0 \text{ if and only if } \begin{bmatrix}
  \Delta_{22} & \mathbf{0} \\
  \mathbf{0} & \mathbf{A}_{22}
  \end{bmatrix} \succ 0
  \]
  That is,
  \[
  \begin{bmatrix}
  \mathbf{A}_{11} & \mathbf{A}_{12} \\
  \mathbf{A}_{12}^\top & \mathbf{A}_{22}
  \end{bmatrix} \succ 0 \text{ if and only if } \mathbf{A}_{22} \succ 0 \text{ and } \Delta_{22} \succ 0
  \]
Solving LMIs

- \( F(x) = F_0 + x_1 F_1 + \cdots + x_n F_n \succeq 0 \) is called the canonical representation of an LMI.
- The LMIs in the canonical form are very inefficient from a storage view-point as well as from the efficiency of the LMI solvers view-point.
- The LMI solvers use a structured representation of LMIs.
LMI Solvers

- Three types of LMI solvers
- To test whether or not there exists a solution $x$ to $F(x) \succ 0$ is called a feasibility problem
- Minimization of a linear objective under LMI constraints
- Generalized eigenvalue minimization problem
Solving the feasibility problem

- Can solve LMIs of the form

\[ N^T \mathcal{L}(X_1, \ldots, X_k) N \prec M^T \mathcal{R}(X_1, \ldots, X_k) M \]

- \( X_1, \ldots, X_k \) — matrix variables
- \( N \) — left outer factor, \( M \) — right outer factor
- \( \mathcal{L}(X_1, \ldots, X_k) \) — left inner factor, \( \mathcal{R}(X_1, \ldots, X_k) \) — right inner factor
The term “left-hand side” refers to what is on the “smaller” side of the inequality $0 \prec \mathbf{X}$.

In $\mathbf{X} \succ 0$, the matrix $\mathbf{X}$ is on the right-hand side—it is on the “larger” side of the inequality.
A general structure for finding a feasible soln

setlmis([])

lmivar

lmiterm
.
.
.
.

lmiterm

getlmis

feasp

dec2mat
\[ X = \text{lmivar}(\text{type}, \text{structure}) \]

- The input type specifies the structure of the variable \( X \)
- Three structures of matrix variables
  - type=1—symmetric block diagonal matrix variable
  - type=2—full rectangular matrix variable
  - type=3—other cases
Second input of $X=\text{lmivar}(\text{type,structure})$

- Additional info on the structure of the matrix variable $X$
- Example

$$X = \begin{bmatrix}
D_1 & O & \cdots & O \\
O & D_2 & \cdots & O \\
\vdots & \vdots & \ddots & \vdots \\
O & O & \cdots & D_r
\end{bmatrix}$$

- Each $D_i$ is a square symmetric matrix—type=1
- $r$ blocks—structure is $r \times 2$
The input structure

- The first component of each row of the input structure—corresponding block size
- The second element of each row—the block type

$X=\text{lmivar}(1, \begin{bmatrix} 3 & 1 \end{bmatrix})$
full symmetric $3 \times 3$ matrix variable

$X=\text{lmivar}(2, \begin{bmatrix} 2 & 3 \end{bmatrix})$
rectangular $2 \times 3$ matrix variable
Scalar block

\[
S = \begin{bmatrix}
  s_1 & 0 & | & 0 & 0 \\
  0 & s_1 & | & 0 & 0 \\
  0 & 0 & | & s_2 & s_3 \\
  0 & 0 & | & s_3 & s_4 \\
\end{bmatrix},
\]

- \( S = \text{lmivar}(1, [2 \ 0; 2 \ 1]) \) describes a scalar block matrix,
- \( D_1 = s_1 I_2 \)
- The second block is a 2 \( \times \) 2 symmetric full block
limiterm(termid,A,B,flag)

- termid—row with four elements specify the terms of each LMI of the LMI system
- termid(1)=n to specify the left-hand side of the $n$-th LMI
- termid(1)=-n to specify the right-hand side of the $n$-th LMI
- termid(2,3)=[i j] specifies the term of the $(i,j)$ block of the LMI specified by the first component
More on \texttt{Imiterm(termid,A,B,flag)}

- termid(4)=0 for the constant term
- termid(4)=X for the variable term in the form $AXB$
- termid(4)=-X for the variable term in the form $AX^TB$
Second and third inputs in \texttt{Imiterm(termid,A,B,flag)}

- A and B give the value of the constant outer factors in the variable terms, \( AXB \) or in \( AX^\top B \)
- The flag input to \texttt{Imiterm} serves as a compact way to specify the expression

\[
AXB + (AXB)^\top
\]
Using flag in \texttt{lmiterm(termid,A,B,flag)}

- flag='s' use for symmetrized expression
- \[ PA + A^\top P \prec 0 \]
  \begin{align*}
  &\text{\texttt{lmiterm([1 1 1 P],1,A)}} \\
  &\text{\texttt{lmiterm([1 1 1 -P],A',1)}}
  \end{align*}
- Note that \[ PA + A^\top P = PA + (PA)^\top \]
- \texttt{lmiterm([1 1 1 P],1,A,'s')}
\[ \text{feas} \text{ solves the auxiliary convex} \]

\[
\begin{align*}
\text{minimize} & \quad t \\
\text{subject to} & \quad L(x) \preceq R(x) + tI.
\end{align*}
\]
$P = \text{dec2mat}(\text{lmis}, \text{xfeas}, P)$

- The system of LMIs is feasible if the minimal $t < 0$
- $P = \text{dec2mat}(\text{lmis}, \text{xfeas}, P)$ converts the output of the LMI solver into matrix variables
Minimizer of a Linear Objective Subject to LMI Constraints

- Invoked using the function \texttt{mincx}

\[
\begin{align*}
\text{minimize} & \quad c^\top x \\
\text{subject to} & \quad A(x) \prec B(x).
\end{align*}
\]

- \(A(x) \prec B(x)\) is a shorthand notation for general structured LMI systems
State/Output Feedback Control

LTI System with output feedback control:

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx \\
u &= -Ky
\end{align*}
\]

Goal: Design $K$ to ensure asymptotic stability of $(A - BKC)$

- Matrix inequality for output-feedback controller design:

\[
(A - BKC)^\top P + P(A - BKC) \prec 0, \quad P \succ 0
\]

- Simpler case: state-feedback $(C = I)$

\[
(A - BK)^\top P + P(A - BK) \prec 0, \quad P \succ 0
\]
Simpler Case: State-Feedback Control

\[(A - BK)^T P + P(A - BK) \prec 0, \ P \succ 0\]

- **To-do:** Find \( K, P \)
- **Problem:** Bi-linear matrix inequality in \( K \) and \( P \)
- **Technique \#2:** Congruence transformation with \( S \triangleq P^{-1} \) and \( Z \triangleq KS \)
- **New inequalities**

\[SA^T + AS - SK^T B^T - BKS \prec 0\]

- **LMIs:**

\[
\underbrace{SA^T + AS}_{\text{linear in } S} - \underbrace{Z^T B^T - BZ}_{\text{linear in } Z} \prec 0, \ P \succ 0
\]

- Get back \( P = S^{-1}, \ K = ZP \)
General Structure of CVX Code in MATLAB as summarized by Dr. Ankush Chakrabarty

```matlab
cvx_begin sdp quiet
  % sdp: semi-definite programming mode
  % quiet: no display during computing
  include CVX [variables]
  % very intuitive variable initialization
  % for example: variable P(3,3) symmetric
  minimize([cost]) % convex function
  subject to
    [affine constraints] % preferably non-strict inequalities

cvx_end

disp(cvx_status) % solution status
```
cvx_begin sdp

% Variable definition
variable S(n, n) symmetric
variable Z(m, n)

% LMIs
S >= eps*eye(n)

cvx_end
sys.K = Z/S; % compute K matrix
Output-Feedback Control

\[ A^T P + PA - C^T K^T B^T P - PBKC \prec 0, \ P \succ 0 \]

- **To-do**: Find \( K, P \)
- **Problem**: Bi-linear matrix inequality in \( K \) and \( P \)
- **Technique #3**: Choose \( M \) such that \( BM = PB \) and \( N \triangleq MK \)
- **New inequalities**: \( A^T P + PA - C^T K^T MB^T - BMKC \prec 0 \)
- **Linear matrix (in)equalities**:
  \[
  \underbrace{A^T P + PA - C^T N^T B^T - BNC}_{\text{linear in } P} \prec 0, \ BM = PB, \ P \succ 0 \]

- Get back \( K = M^{-1} N \) (\( M \) is invertible if \( B \) has full column rank)
Cool fact: CVX/YALMIP can handle *equality constraints*!

```matlab
cvx_begin sdp quiet

% Variable definition
variable P(n, n) symmetric
variable N(m, p)
variable M(m, m)

% LMIs
P*sys.A + sys.A'*P - sys.B*N*sys.C ... 
- sys.C'*N'*sys.B' <= -eps*eye(n)
sys.B*M == P*sys.B
P >= eps*eye(n);

cvx_end
sys.K = M\N % compute K matrix
```
Observer Design

Plant model:

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &=Cx
\end{align*}
\]

Linear observer:

\[
\dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x})
\]

**Goal:** Design \( L \) to ensure asymptotic stability of the error dynamics

- **Matrix inequality for observer design:**

\[
(A - LC)^\top P + P(A - LC) \prec 0, \quad P = P^\top \succ 0
\]
Observer Design—Contd.

\[ A^\top P + PA - C^\top L^\top P - PLC \prec 0, \ P \succ 0 \]

- To-do: Find \( L, \ P \)
- Problem: Bi-linear matrix inequality in \( L \) and \( P \)
- **Technique #1**: Choose \( Y = PL \)
- LMIs:

\[
\begin{aligned}
A^\top P + PA - C^\top Y^\top - YC &\prec 0, \ P \succ 0 \\
\text{linear in } P &\quad \text{linear in } Y
\end{aligned}
\]

- For robustness of solution, rewrite as

\[
A^\top P + PA - C^\top Y^\top - YC + 2\alpha P \preceq 0, \ P \succ 0
\]

with fixed \( \alpha > 0 \)
- Get back \( L = P^{-1}Y \) (\( P \succ 0 \), hence invertible)
cvx_begin sdp

% Variable definition
variable P(n, n) symmetric
variable Y(n, p)

% LMIs
P*sys.A + sys.A'*P - Y*sys.C - sys.C'*Y' + P <= 0
P >= eps*eye(n) % eps is a very small number in MATLAB

cvx_end
sys.L = P\Y; % compute L matrix