LMI 工具箱介绍

线性矩阵不等式（LMI）工具箱是求解一般线性矩阵不等式问题的一个高性能软件包。由于其面向结构的线性矩阵不等式表示方式，使得各种线性矩阵不等式能够以自然块矩阵的形式加以描述。一个线性矩阵不等式问题一旦确定，就可以通过调用适当的线性矩阵不等式求解器来对这个问题进行数值求解。

LMI 工具箱提供了确定、处理和数值求解线性矩阵不等式的一些工具，它们主要用于：

- 以自然块矩阵形式来直接描述线性矩阵不等式；
- 获取关于现有的线性矩阵不等式系统的信息；
- 修改现有的线性矩阵不等式系统；
- 求解三个一般的线性矩阵不等式问题；
- 验证结果。

本附录将详细介绍 LMI 工具箱提供的用于解决以上各个问题的相关函数和命令。

A.1 线性矩阵不等式及相关术语

一个线性矩阵不等式就是具有以下一般形式的一个矩阵不等式：

$$\boldsymbol{L}(\boldsymbol{x}) = \boldsymbol{L}_0 + x_1\boldsymbol{L}_1 + \cdots + x_N\boldsymbol{L}_N < \boldsymbol{0} \tag{1}$$

其中：$\boldsymbol{L}_0, \boldsymbol{L}_1, \cdots, \boldsymbol{L}_N$ 是给定的对称常数矩阵，$x_1, \cdots, x_N$ 是未知变量，称为决策变量，$\boldsymbol{x} = [x_1, \cdots, x_N]^{\mathrm{T}} \in \mathbf{R}^N$ 是由决策变量构成的向量，称为决策向量。

尽管表达式（1）是线性矩阵不等式的一个一般表示式，但在大多数实际应用中，线性矩阵不等式常常不是以一般表示式（1）的形式出现，而是具有以下形式：

$$L(X_1, \cdots, X_n) < R(X_1, \cdots, X_n)$$

其中的 $L(\cdot)$ 和 $R(\cdot)$ 是矩阵变量 $X_1, \cdots, X_n$ 的仿射函数，通过适当的代数运算，上式可以写成线性矩阵不等式的一般表示式（1）的形式。例如，在系统稳定性问题中经常遇到的 Lyapunov 矩阵不等式

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X} + \boldsymbol{X}\boldsymbol{A} < \boldsymbol{0} \tag{2}$$

也是一个线性矩阵不等式，其中的 X 是一个矩阵变量。我们以一个二阶矩阵 $\boldsymbol{A} = \begin{bmatrix} -1 & 2 \\ 0 & -2 \end{bmatrix}$ 为例，将矩阵不等式（2）写成一般表示式（1）的形式。针对二阶矩阵不

等式（2），对应的矩阵变量 X 是一个二阶的对称矩阵，$\boldsymbol{X}=\begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \end{bmatrix}$，不等式（2）中的决策变量是矩阵 $\boldsymbol{X}$ 中的独立元 x_1、x_2、x_3。根据对策性，矩阵变量 X 可以写成

$$X = x_1\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + x_2\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + x_3\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

将矩阵 A 和上式代入矩阵不等式（2），经整理，可得

$$x_1\begin{bmatrix} -2 & 2 \\ 2 & 0 \end{bmatrix} + x_2\begin{bmatrix} 0 & -3 \\ -3 & 4 \end{bmatrix} + x_3\begin{bmatrix} 0 & 0 \\ 0 & -4 \end{bmatrix} < \boldsymbol{0} \tag{3}$$

这样就将矩阵不等式（2）写成了线性矩阵不等式的表示式（1）。显然，与 Lyapunov 矩阵不等式（2）相比，表示式（3）缺少了许多控制中的直观意义。另外，（3）式涉及到的矩阵也比（2）式中的多。如果矩阵 $\boldsymbol{A}$ 是 n 阶的，则（3）式中的系数矩阵一般有 $n(n+1)/2$ 个。因此，这样的表达式在计算机中将占用更多的存储空间。由于这样的一些原因，LMI 工具箱中的函数采用线性矩阵不等式的结构表示。例如，Lyapunov 矩阵不等式（2）就以矩阵变量 $\boldsymbol{X}$ 的不等式来表示，而不是用其一般形式（3）来表示。

一般的，一个线性矩阵不等式具有块矩阵的形式，其中每一个块都是矩阵变量的仿射函数。以下通过一个例子来说明有关描述一个线性矩阵不等式的术语。

考虑 H_∞ 控制中的一个线性矩阵不等式：

$$\boldsymbol{N}^{\mathrm{T}}\begin{bmatrix} \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A} & \boldsymbol{X}\boldsymbol{C}^{\mathrm{T}} & \boldsymbol{B} \\ \boldsymbol{C}\boldsymbol{X} & -\gamma\boldsymbol{I} & \boldsymbol{D} \\ \boldsymbol{B}^{\mathrm{T}} & \boldsymbol{D}^{\mathrm{T}} & -\gamma\boldsymbol{I} \end{bmatrix}\boldsymbol{N} < \boldsymbol{0}$$

其中：$\boldsymbol{A}$、$\boldsymbol{B}$、$\boldsymbol{C}$、$\boldsymbol{D}$、$\boldsymbol{N}$ 是给定的矩阵，$\boldsymbol{X}=\boldsymbol{X}^{\mathrm{T}}\in\mathbf{R}^{n\times n}$ 和 $\gamma\in\mathbf{R}$ 是问题的变量。

- $\boldsymbol{N}$ 称为外因子，块矩阵

$$\boldsymbol{L}(\boldsymbol{X},\gamma)=\begin{bmatrix} \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A} & \boldsymbol{X}\boldsymbol{C}^{\mathrm{T}} & \boldsymbol{B} \\ \boldsymbol{C}\boldsymbol{X} & -\gamma\boldsymbol{I} & \boldsymbol{D} \\ \boldsymbol{B}^{\mathrm{T}} & \boldsymbol{D}^{\mathrm{T}} & -\gamma\boldsymbol{I} \end{bmatrix}$$

  称为内因子。外因子可以不是一个正方矩阵，它在许多问题中常常不出现。
- $\boldsymbol{X}$ 和 γ 是问题的矩阵变量。注意标量也可以看成是一个1×1维的矩阵。
- 内因子 $\boldsymbol{L}(\boldsymbol{X},\gamma)$ 是一个对称块矩阵。根据对称性，$\boldsymbol{L}(\boldsymbol{X},\gamma)$ 可以由对角线及其上方的块矩阵完全确定。
- $\boldsymbol{L}(\boldsymbol{X},\gamma)$ 中的每一块都是矩阵变量 $\boldsymbol{X}$ 和 γ 的仿射函数。这一函数由常数项和变量项这两类基本项组成，其中常数项就是常数矩阵或以一些常数矩阵组成的算术表达式，例如 $\boldsymbol{L}(\boldsymbol{X},\gamma)$ 中的 $\boldsymbol{B}$ 和 $\boldsymbol{D}$；变量项是包含一个矩阵变量的项，例如 $\boldsymbol{X}\boldsymbol{A}, -\gamma\boldsymbol{I}$ 等。

一个线性矩阵不等式不论多么复杂，都可以通过描述其中每一块的各项内容来确定这个线性矩阵不等式。

A.2　线性矩阵不等式的确定

LMI 工具箱可以处理具有以下一般形式的线性矩阵不等式：

$$\boldsymbol{N}^{\mathrm{T}}\boldsymbol{L}(\boldsymbol{X}_1,\cdots,\boldsymbol{X}_K)\boldsymbol{N}<\boldsymbol{M}^{\mathrm{T}}\boldsymbol{R}(\boldsymbol{X}_1,\cdots,\boldsymbol{X}_K)\boldsymbol{M}$$

其中：$\boldsymbol{X}_1,\cdots,\boldsymbol{X}_K$ 是具有一定结构的矩阵变量，左、右外因子 $\boldsymbol{N}$ 和 $\boldsymbol{M}$ 是具有相同维数的给定矩阵，左、右内因子 $\boldsymbol{L}(\cdot)$ 和 $\boldsymbol{R}(\cdot)$ 是具有相同块结构的对称块矩阵。

注意在线性矩阵不等式的描述中，左边总是指不等式较小的一边，例如对线性矩阵不等式 $\boldsymbol{X}>\boldsymbol{0}$，$\boldsymbol{X}$ 称为是不等式的右边，$\boldsymbol{0}$ 称为是不等式的左边，常表示成 $\boldsymbol{0}<\boldsymbol{X}$ 。

要确定一个线性矩阵不等式系统，需要做以下两步：

1．给出每个矩阵变量 $\boldsymbol{X}_1,\cdots,\boldsymbol{X}_K$ 的维数和结构；

2．描述每一个线性矩阵不等式中各个项的内容。

这个过程产生所描述线性矩阵不等式系统的一个内部表示，它以一个单一向量的形式储存在计算机内，通常用一个名字，例如 lmisys 来表示。该内部表示 lmisys 可以在后面处理这个线性矩阵不等式时调用。

下面将通过 LMI 工具箱中的一个例子来说明线性矩阵不等式系统的确定。运行 lmidem 可以看到这个例子的完整描述。

例 1：考虑一个具有 4 个输入、4 个输出和 6 个状态的稳定传递函数

$$\boldsymbol{G}(s)=\boldsymbol{C}(s\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B} \tag{4}$$

和一组具有以下块对角结构的输入/输出尺度矩阵 $\boldsymbol{D}$：

$$\boldsymbol{D}=\begin{bmatrix} d_1 & 0 & 0 & 0\\ 0 & d_1 & 0 & 0\\ 0 & 0 & d_2 & d_3\\ 0 & 0 & d_4 & d_5 \end{bmatrix} \tag{5}$$

则在具有时变不确定性系统的鲁棒稳定性分析中提出了以下问题：

寻找一个具有结构（5）的尺度矩阵 $\boldsymbol{D}$，使得 $\sup_{\omega}\left\|\boldsymbol{D}\boldsymbol{G}(j\omega)\boldsymbol{D}^{-1}\right\|<1$。

可以证明：这样一个问题可以转化成一个线性矩阵不等式系统的可行性问题，即寻找两个对称矩阵 $\boldsymbol{X}\in\mathrm{R}^{6\times6}$ 和 $\boldsymbol{S}=\boldsymbol{D}^{\mathrm{T}}\boldsymbol{D}\in\mathbf{R}^{4\times4}$，使得

$$\begin{bmatrix} \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}+\boldsymbol{C}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{C} & \boldsymbol{X}\boldsymbol{B}\\ \boldsymbol{B}^{\mathrm{T}}\boldsymbol{X} & -\boldsymbol{S} \end{bmatrix}<\boldsymbol{0} \tag{6}$$

$$\boldsymbol{X}>\boldsymbol{0} \tag{7}$$

$$\boldsymbol{S}>\boldsymbol{I} \tag{8}$$

用命令 lmivar 和 lmiterm 给出线性矩阵不等式系统（6）~（8）的内部描述如下：

```
setlmis([])
```

```
X=lmivar(1,[6 1])
S=lmivar(1,[2 0;2 1])

% 1st LMI
lmiterm([1 1 1 X],1,A,'s')
lmiterm([1 1 1 S],C',C)
lmiterm([1 1 2 X],1,B)
lmiterm([1 2 2 S],-1,1)

% 2nd LMI
lmiterm([-2 1 1 X],1,1)

% 3rd LMI
lmiterm([-3 1 1 S],1,1)
lmiterm([3 1 1 0],1)

lmisys=getlmis
```

其中：函数 lmivar 定义了两个矩阵变量 ***X*** 和 ***S***，lmiterm 则描述了每一个线性矩阵不等式中各项的内容。getlmis 回到了这个线性矩阵不等式系统的内部表示 lmisys，lmisys 也称为是储存在机器内部的线性矩阵不等式系统的名称。以下将详细介绍这几个函数的功能和用法。

setlmis 和 getlmis

一个线性矩阵不等式系统的描述以 setlmis 开始，以 getlmis 结束。当要确定一个新的系统时，输入：

```
setlmis([])
```

如果需要将一个线性矩阵不等式添加到一个名为 lmiso 的现有的线性矩阵不等式系统中，则输入：

```
setlmis(lmiso)
```

当线性矩阵不等式系统被完全确定好后，输入：

```
lmisys=getlmis
```

该命令返回这个线性矩阵不等式系统的内部表示 lmisys。

lmivar

函数 lmivar 用来描述出现在线性矩阵不等式系统中的矩阵变量，每一次只能描述一个矩阵变量。矩阵变量的描述包括该矩阵变量的结构。该函数的一般表达式是：

```
X=lmivar(type,struct)
```

这一函数定义了一个新的矩阵变量 $\boldsymbol{X}$，X 是该矩阵变量的变量名。函数中的第一个输入量 type 确定了矩阵变量 X 的类型，第二个输入量 struct 进一步根据变量 $\boldsymbol{X}$ 的类型给出该变量的结构。变量的类型分成三类：

Type =1：对称块对角结构。这种结构对应于具有以下形式的矩阵变量：

$$\begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_2 & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{D}_r \end{bmatrix}$$

其中对角线上的每一个矩阵块 $\boldsymbol{D}_j$ 是方阵，它可以是零矩阵、对称矩阵或数量矩阵。这种结构也包含了通常意义的对称矩阵和数量矩阵（分别相当于只有一块）。此时，struct 是一个 $r\times 2$ 维的矩阵。如果该矩阵的第 i 行是 (m, n)，则其中的 m 表示对称矩阵块 $\boldsymbol{D}_i$ 的阶数，而 n 只能取 1、0 或 –1，其中 $n=1$ 表示 $\boldsymbol{D}_i$ 是一个满的对称矩阵（或无结构的对称矩阵），$n=0$ 表示 $\boldsymbol{D}_i$ 是一个数量矩阵，$n=-1$ 表示 $\boldsymbol{D}_i$ 是一个零矩阵。

Type =2：长方型结构。这种结构对应于任意的长方矩阵。此时，struct= (m, n) 表示矩阵的维数。

Type =3：其他结构。这种结构用来描述更加复杂的矩阵，也可以用于描述矩阵变量之间的一些关联。$\boldsymbol{X}$ 的每一个元或者是 0，或者是 $\pm x_n$，其中 x_n 是第 n 个决策变量。相应的，struct 是一个和变量 $\boldsymbol{X}$ 有相同维数的矩阵，其中的每一个元取值如下：

$$\text{struct}(i,\ j)=\begin{cases} 0, & \text{如果}\boldsymbol{X}(i,\ j)=0 \\ n, & \text{如果}\boldsymbol{X}(i,\ j)=x_n \\ -n, & \text{如果}\boldsymbol{X}(i,\ j)=-x_n \end{cases}$$

例 2：考虑具有三个矩阵变量 $\boldsymbol{X}_1$、$\boldsymbol{X}_2$ 和 $\boldsymbol{X}_3$ 的线性矩阵不等式系统，其中

- $\boldsymbol{X}_1$ 是一个 3×3 维的对称矩阵；
- $\boldsymbol{X}_2$ 是一个 2×4 维的长方矩阵；
- $\boldsymbol{X}_3=\begin{bmatrix} \Delta & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \delta_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \delta_2\boldsymbol{I}_2 \end{bmatrix}$，其中 Δ 是 5×5 维的对称矩阵，δ_1 和 δ_2 是两个标量，$\boldsymbol{I}_2$ 表示 2×2 维的单位矩阵。

可以应用 lmivar 来定义这些矩阵变量：

```
setlmis([])
X1=lmivar(1,[3 1])
X2=lmivar(2,[2 4])
X3=lmivar(1,[5 1;1 0;2 0])
```

lmiterm

在确定了矩阵变量之后，还需要确定每一个线性矩阵不等式中各项的内容。线性矩阵不等式的项指构成这个线性矩阵不等式的块矩阵中的加项。这些项可以分成三类：

1．常数项；

2．变量项，即包含了矩阵变量的项，例如（3）式中的 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}$ 和 $\boldsymbol{C}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{C}$ 。一般的变量项具有形式 $\boldsymbol{P}\boldsymbol{X}\boldsymbol{Q}$ ，其中的 $\boldsymbol{X}$ 是一个变量， $\boldsymbol{P}$ 和 $\boldsymbol{Q}$ 是给定的矩阵，分别称为该变量项的左系数和右系数；

3．外因子。

在描述一个具有多个块的线性矩阵不等式时，LMI 工具箱提供了这样的功能，即只需要确定对角线上和对角线上方的项的内容，或者只描述对角线上和对角线下方的项的内容，其他部分项的内容可以根据线性矩阵不等式的对称性得到。

用命令 lmiterm 每次可以确定线性矩阵不等式的一个项的内容。例如，对线性矩阵不等式

$$\begin{bmatrix} \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}+\boldsymbol{C}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{C} & \boldsymbol{X}\boldsymbol{B} \\ \boldsymbol{B}^{\mathrm{T}}\boldsymbol{X} & -\boldsymbol{S} \end{bmatrix} < \boldsymbol{0}$$

可以用以下一组命令来描述：

```
lmiterm([1 1 1 X],1,A,'s')
lmiterm([1 1 1 S],C',C)
lmiterm([1 1 2 X],1,B)
lmiterm([1 2 2 S],-1,1)
```

这些命令依次描述了项 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}$、$\boldsymbol{C}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{C}$、$\boldsymbol{X}\boldsymbol{B}$ 和 $-\boldsymbol{S}$ 。在每一条命令中，第 1 项是一个四元向量，它刻画了所描述的项所在的位置和特征：

- 第 1 个元表示所描述的项属于哪一个线性矩阵不等式。值 m 表示第 m 个不等式的左边，-m 表示第 m 个不等式的右边。
- 第 2 和第 3 个元表示所描述的项所在块的位置。例如，向量[1 1 2 1]表示所描述的项位于第一个线性矩阵不等式左边内因子的块（1，2）中。第 2 和第 3 个元均取零表示所描述的项在外因子中。
- 最后一个元表明了所描述的项是常数项还是变量项。如果是变量项，则进一步说明涉及哪一个变量。0 表示常数项，k 表示所描述的项包含第 k 个矩阵变量 $\boldsymbol{X}_k$ ，-k 则表示包含矩阵变量 $\boldsymbol{X}_k$ 的转置 $\boldsymbol{X}_k^{\mathrm{T}}$（在例 1 中， $\boldsymbol{X}$ 是第 1 个变量， $\boldsymbol{S}$ 是第 2 个变量，它们按确定的先后顺序排列）。

lmiterm 的第 2 项和第 3 项包含了数据（常数项的值，外因子，变量项 $\boldsymbol{P}\boldsymbol{X}\boldsymbol{Q}$ 或 $\boldsymbol{P}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Q}$ 中的左、右系数）。第 4 项是可选择的，且只能是's'。

在描述项的内容时，有一些简化的方法。

1．零块可以省略描述；

2．可以通过在命令 lmiterm 中外加一个分量's'，使得可以只用一个命令 lmiterm 就能

描述一个变量项与该变量项的转置的和。例如，上面的第一个命令描述了 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}$。

3．可以用一个标量值来表示一个数量矩阵，即用 α 表示数量矩阵 $\alpha\boldsymbol{I}$，其中 α 是一个标量。如例 1 中的第 3 个不等式 $\boldsymbol{S}>\boldsymbol{I}$ 被描述成

```
lmiterm([-3 1 1 S],1,1)
lmiterm([3 1 1 0],1)
```

为了便于阅读，也可以用线性矩阵不等式和矩阵变量的名称来表示对应的线性矩阵不等式和矩阵变量。矩阵变量的变量名可以用命令 lmivar 来赋值，线性矩阵不等式的名称则可以用函数 newlmi 来确定。这些标识符可以用在命令 lmiterm 中以表示相应的线性矩阵不等式或矩阵变量。对例 1 中的线性矩阵不等式系统，采用名称的相应描述如下：

```
setlmis([])
X=lmivar(1,[6 1])
S=lmivar(1,[2 0;2 1])

BRL=newlmi
lmiterm([BRL 1 1 X],1,A,'s')
lmiterm([BRL 1 1 S],C',C)
lmiterm([BRL 1 2 X],1,B)
lmiterm([BRL 2 2 S],-1,1)

Xpos=newlmi
lmiterm([-Xpos 1 1 X],1,1)

Slmi=newlmi
lmiterm([-Slmi 1 1 S],1,1)
lmiterm([Slmi 1 1 0],1)

lmisys=getlmis
```

其中：X 和 S 分别表示变量 $\boldsymbol{X}$ 和 $\boldsymbol{S}$，而 BRL、Xpos 和 Slmi 则分别表示第 1、第 2 和第 3 个线性矩阵不等式。-Xpos 指的是第 2 个线性矩阵不等式的右边，-X 表示变量 $\boldsymbol{X}$ 的转置。

lmiedit

线性矩阵不等式编辑器 lmiedit 是一个图形用户界面，它可以按符号方式直接确定线性矩阵不等式系统。输入

```
lmiedit
```

出现一个具有一些可编辑文本区域和各种按钮的窗户。按以下步骤来确定一个线性矩阵不

等式系统：

1．在文本区域的上半部分给出每一个矩阵变量的描述（名字和结构），其结构是通过类型（S 表示对称块矩阵，R 表示无结构的长方矩阵，G 表示其他结构矩阵）和一个“附加”的结构矩阵（类似于 lmivar 中的 struct）来刻画的。在文本编辑区，使用一行描述一个变量。

2．在文本区的下半部分，按 MATLAB 的表示方式给出要描述的线性矩阵不等式。例如，线性矩阵不等式

$$\begin{bmatrix} \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A} & \boldsymbol{X}\boldsymbol{B} \\ \boldsymbol{B}^{\mathrm{T}}\boldsymbol{X} & -\boldsymbol{I} \end{bmatrix} < \boldsymbol{0}$$

可以通过输入

```
[A'*X+X*A X*B;B'*X -1]<0
```

来描述。其中 X 是文本区上半部分描述矩阵变量 $\boldsymbol{X}$ 的变量名。一个线性矩阵不等式的描述可能需要几行，但一行中最多只能描述一个线性矩阵不等式。

完成了线性矩阵不等式系统的描述后，可以通过按相应的按钮来完成以下的任务：

- 显示用于描述线性矩阵不等式的 lmivar/lmiterm 命令串（按钮 view commands）；反之，通过单击按钮 describe…可以将用一串 lmivar/lmiterm 命令定义的线性矩阵不等式系统按 MATLAB 表示式显示。
- 将线性矩阵不等式的符号描述存为一个 MATLAB 语句串（按钮 save）。以后可以通过按钮 load 重新显示这种描述。
- 可以从一个文件读一串 lmivar/lmiterm 命令（按钮 read），然后通过单击“describe the matrix variables”或“describe the LMIs …”显示出由这些命令确定的线性矩阵不等式系统的符号表示。
- 写一串用于描述一个特殊线性矩阵不等式系统的 lmivar/lmiterm 命令（按钮 write）。
- 通过按钮 create 产生线性矩阵不等式系统的内部表示，结果用一个以线性矩阵不等式命名的 MATLAB 变量记录（如果线性矩阵不等式系统名是 mylmi，则其内部表示用 MATLAB 变量 mylmi 记录）。内部表示 mylmi 可以被线性矩阵不等式求解器或任何其他的线性矩阵不等式函数调用。

如同命令 lmiterm 一样，可以应用简捷的方法来输入线性矩阵不等式的表示式。例如零块可以简单地输入 0，而不必定义其维数，类似地，单位矩阵只需输入数字 1 等。

lmiedit 尽管很一般，但它没有 lmiterm 灵活。以下是 lmiedit 的一些局限性：

- 在矩阵变量的两边不能使用括号。例如当 X 是一个变量名时，

  ```
  (A*X+B)'*C+C'*(A*X+B)
  ```

  是不允许的，而

  ```
  (A+B)'*X+X'*(A+B)
  ```

则是可以的。

- 不允许出现循环和条件语句。
- 当把 lmiterm 命令转换成一个线性矩阵不等式的符号描述时，如果 lmiterm 的第 1 个分量不能确认就将出错。使用由 newlmi 和 lmivar 提供的线性矩阵不等式和变量标识符可以避免这样的问题。

图 A.1 给出了用 lmiedit 描述例 1 中的线性矩阵不等式系统的窗口。

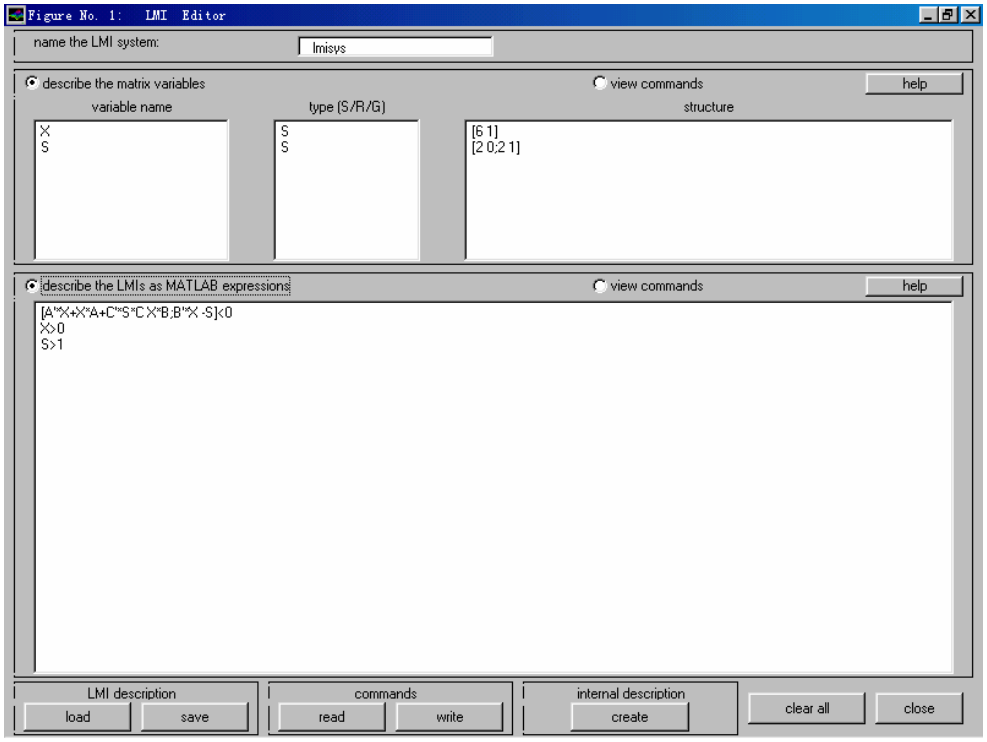

图 A.1　lmiedit 的图形界面

A.3　信息提取

线性矩阵不等式系统的完整描述是以一个叫做内部表示的向量储存在机器内的。LMI 工具箱提供了三个函数 lmiinfo、lminbr 和 matnbr，它们可以从内部表示向量中提取线性矩阵不等式的相关信息，并以用户可读的方式显示出来。

lmiinfo

lmiinfo 是一种交互式工具，用以反映有关线性矩阵不等式系统的一些信息。这些信息包括线性矩阵不等式的个数、矩阵变量的个数和它们的结构、每一个线性矩阵不等式块中

项的内容等。为了调用 lmiinfo，输入

```
lmiinfo(lmisys)
```

其中的 lmisys 是由 getlmis 产生的线性矩阵不等式系统的内部表示。

lminbr 和 matnbr

这两个函数给出了系统中线性矩阵不等式的个数和矩阵变量的个数。例如，为了得到矩阵变量的个数，输入

```
matnbr(lmisys)
```

A.4 线性矩阵不等式求解器

LMI 工具箱提供了用于求解以下三个问题的线性矩阵不等式求解器（其中 $\boldsymbol{x}$ 表示决策变量向量，即矩阵变量 $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_k$ 中的独立变元构成的向量）。

- 可行性问题：

寻找一个 $\boldsymbol{x} \in \mathbf{R}^N$ （或等价的：具有给定结构的矩阵 $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_k$），使得满足线性矩阵不等式系统

$$\boldsymbol{A}(\boldsymbol{x}) < \boldsymbol{B}(\boldsymbol{x})$$

相应的求解器是 feasp。

- 具有线性矩阵不等式约束的一个线性目标函数的最小化问题：

$$\min_{\boldsymbol{x}} \boldsymbol{c}^{\mathrm{T}} \boldsymbol{x}$$

$$\text{s.t.} \quad \boldsymbol{A}(\boldsymbol{x}) < \boldsymbol{B}(\boldsymbol{x})$$

相应的求解器是 mincx。

- 广义特征值的最小化问题：

$$\min_{\boldsymbol{x}} \lambda$$

$$\text{s.t.} \quad \boldsymbol{C}(\boldsymbol{x}) < \boldsymbol{D}(\boldsymbol{x}) \tag{9}$$

$$\mathbf{0} < \boldsymbol{B}(\boldsymbol{x}) \tag{10}$$

$$\boldsymbol{A}(\boldsymbol{x}) < \lambda \boldsymbol{B}(\boldsymbol{x}) \tag{11}$$

相应的求解器是 gevp。

以下详细介绍这三个求解器的功能和使用方法。

feasp

求解器 feasp 的一般表达式如下：

```
[tmin,xfeas]=feasp(lmisys,options,target)
```

求解器 feasp 是通过求解如下的一个辅助凸优化问题

$$\min t$$

$$\text{s.t.} \quad \boldsymbol{A}(\boldsymbol{x}) - \boldsymbol{B}(\boldsymbol{x}) \le t\boldsymbol{I}$$

来求解线性矩阵不等式系统 lmisys 的可行性问题。

这个凸优化问题的全局最优值用 tmin 表示，作为求解器 feasp 输出的第一个分量。如果 tmin<0，则系统 lmisys 是可行的。当系统 lmisys 为可行时，求解器 feasp 输出的第二个分量 xfeas 给出了该线性矩阵不等式系统决策变量的一个可行解。进而，应用 dec2mat 可以得到系统 lmisys 矩阵变量的一个可行解。

求解器 feasp 的输入变量 target 为 tmin 设置了目标值，使得只要 tmin<target，则优化迭代过程就结束。target=0 是 feasp 的默认值。

可选择的输入量 options 是一个 5 维向量，它用来描述优化迭代过程中的一些控制参数：

- options(1)：该分量不用。
- options(2)：该参数设定优化迭代过程中允许的最大迭代次数（该参数的默认值是 100）。
- options(3)：该参数设定了可行域的半径。options(3)= $R > 0$ 表示限制决策变量在球体

$$\sum_{i=1}^{N} x_i^2 < R^2$$

中，或者说向量 xfeas 的欧氏范数不超过 R。该参数的默认值是 $R = 10^9$。

可行域半径的设定可以避免产生具有很大数值的解 $\boldsymbol{x}$，同时也可以加快计算过程，改进数值稳定性。

- options(4)：该参数用于加快迭代过程的结束，它提供了反映优化过程中迭代速度和解的精度之间的一个折中指标。当该参数取值为一个正整数 J 时，表示在最后的 J 次迭代中，如果每次迭代后 t 的减小幅度不超过 1%，则优化迭代过程就停止。该参数的默认值是 10。
- options(5)：options(5)=1 表示不显示迭代过程中的数据，options(5)=0（默认值）则相反。

将 options(i)设置为零相当于将相应的控制参数设置为默认值，也可以通过忽略该输入变量来接受默认值。

例 3：求满足 $\boldsymbol{P} > \boldsymbol{I}$ 的对称矩阵 $\boldsymbol{P}$，使得

$$\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_1 < \boldsymbol{0} \qquad (12)$$

$$\boldsymbol{A}_2^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_2 < \boldsymbol{0} \qquad (13)$$

$$\boldsymbol{A}_3^{\mathrm{T}}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_3 < \boldsymbol{0} \qquad (14)$$

其中：

$$\boldsymbol{A}_1=\begin{bmatrix}-1 & 2\\ 1 & -3\end{bmatrix},\quad \boldsymbol{A}_2=\begin{bmatrix}-0.8 & 1.5\\ 1.3 & -2.7\end{bmatrix},\quad \boldsymbol{A}_3=\begin{bmatrix}-1.4 & 0.9\\ 0.7 & -2.0\end{bmatrix}$$

为了调用 feasp，我们首先确定线性矩阵不等式系统：

```
setlmis([])
P=lmivar(1,[2 1])

lmiterm([1 1 1 P],1,A1,'s')          % LMI #1
lmiterm([2 1 1 P],1,A2,'s')          % LMI #2
lmiterm([3 1 1 P],1,A3,'s')          % LMI #3
lmiterm([-4 1 1 P],1,1)              % LMI #4: P
lmiterm([4 1 1 0],1)                 % LMI #4: I
lmis=getlmis
```

然后调用 feasp 来求该线性矩阵不等式系统的一个可行决策变量：

```
[tmin,xfeas]=feasp(lmis)
```

得到 tmin = –3.1363 。因此，线性矩阵不等式系统 lmis 是可行的。应用 dec2mat

```
PP=dec2mat(lmis,xfeas,P)
```

得到问题的可行矩阵变量值：

$$\boldsymbol{P}=\begin{bmatrix}270.8 & 126.4\\ 126.4 & 155.1\end{bmatrix}$$

在求解这个可行性问题的过程中，也可以附加一些约束，例如，要求矩阵 $\boldsymbol{P}$ 的 Frobenius 范数不超过 10，且 tmin ≤ –1 。可以通过调用

```
[tmin,xfeas]=feasp(lmis,[0,0,10,0,0],-1)
```

来达到这些附加要求。相应的结果是 tmin = –1.1745 ，相应的矩阵 $\boldsymbol{P}$ 的最大特征值是 $\lambda_{\max}(\boldsymbol{P})=9.6912$ 。

mincx

求解器 mincx 的一般表达式如下：

```
[copt,xopt]=mincx(lmisys,c,options,xinit,target)
```

问题中的线性矩阵不等式系统由 lmisys 表示，向量 $\boldsymbol{c}$ 和决策变量向量 $\boldsymbol{x}$ 有相同的维数。对于由矩阵变量表示的线性目标函数，可以应用函数 defcx 来得到适当的向量 c。函数 mincx 返回到目标函数 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$ 的全局最优值 copt 和决策变量的最优解 xopt，相应的矩阵变量的最优解可以应用函数 dec2mat 从 xopt 得到。

函数 mincx 的输入量中除了 lmisys 和 c 以外，其他的输入是可选择的。xinit 是最优解 xopt 的一个初始猜测（可以从矩阵变量 $\boldsymbol{X}_1,\cdots,\boldsymbol{X}_k$ 的给定值，通过使用 mat2dec 来导出 xinit）。当输入的 xinit 不是一个可行解时，它将被忽略；否则，则有可能加快问题求解的过程。

target 是目标函数的一个设定目标，只要某个可行的 $\boldsymbol{x}$ 满足 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \le$ target，求解过程就停止。

options 是一个 5 维向量，它用来描述优化迭代过程中的一些控制参数：

- options(1)：该参数确定了最优值 copt 所要求的精度（默认值是 10^{-2}）。
- options(2)：该参数设定优化迭代过程中允许的最大迭代次数（默认值是 100）。
- options(3)：该参数设定了可行域的半径。有求解器 feasp 中的相应参数相同。
- options(4)：该参数用于加快迭代过程的结束。当该参数取值为一个正整数 J 时，表示在最后的 J 次迭代中。如果每次迭代后，目标函数 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$ 的减小幅度在给定的精度内，则优化迭代过程就停止。该参数的默认值是 5。
- options(5)：options(5)=1 表示不显示迭代过程中的数据，options(5)=0（默认值）则相反。

将 options(i)设置为零相当于将相应的控制参数设置为默认值，也可以通过忽略该输入变量来接受默认值。

以下的例子说明了求解器 mincx 的使用方法。

例 4：考虑优化问题

$$\min_{\boldsymbol{X}} \mathrm{Trace}(\boldsymbol{X})$$

$$\text{s.t.} \quad \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}+\boldsymbol{X}\boldsymbol{B}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{Q}<\boldsymbol{0}$$

其中：$\boldsymbol{X}$ 是一个对称的矩阵变量，

$$\boldsymbol{A}=\begin{bmatrix}-1 & -2 & 1\\ 3 & 2 & 1\\ 1 & -2 & -1\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}1\\ 0\\ 1\end{bmatrix},\quad \boldsymbol{Q}=\begin{bmatrix}1 & -1 & 0\\ -1 & -3 & -12\\ 0 & -12 & -36\end{bmatrix}$$

根据矩阵的 Schur 补性质，本例中的优化问题等价于

$$\min_{\boldsymbol{X}} \mathrm{Trace}(\boldsymbol{X})$$

$$\text{s.t.} \quad \begin{bmatrix}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}+\boldsymbol{Q} & \boldsymbol{X}\boldsymbol{B}\\ \boldsymbol{B}^{\mathrm{T}}\boldsymbol{X} & -\boldsymbol{I}\end{bmatrix}<\boldsymbol{0}$$

由于 Trace($\boldsymbol{X}$) 是 $\boldsymbol{X}$ 的元的一个线性函数，因此以上的优化问题是一个具有线性矩阵不等式约束的线性目标函数的最小化问题，从而可以应用求解器 mincx 来求解这个问题。以下给出用 mincx 来求解该问题的过程。

1．定义线性矩阵不等式约束

```
setlmis([])
```

```
X=lmivar(1,[3 1])       %  变量 X，满对称的

lmiterm([1 1 1 X],1,A,'s')
lmiterm([1 1 1 0],Q)
lmiterm([1 2 2 0],-1)
lmiterm([1 2 1 X],B',1)

LMIs=getlmis
```

2．将目标函数 Trace($\boldsymbol{X}$) 写成 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$，其中 $\boldsymbol{x}$ 是矩阵变量 $\boldsymbol{X}$ 中的独立元所构成的向量。由于引进向量 $\boldsymbol{c}$ 的目的是要选择 $\boldsymbol{X}$ 的对角元，因此它可以作为相应于 $\boldsymbol{X}=\boldsymbol{I}$ 的决策向量得到，即

```
c=mat2dec(LMIs,eye(3))
```

事实上，函数 defcx 将提供一个确定这样的目标函数的更加系统化的方法。

3．调用 mincx 计算最小值 xopt，目标函数的全局最小值 copt=c′*xopt。

```
options=[1e-5,0,0,0,0]
[copt,xopt]=mincx(LMIs,c,options)
```

其中 1e-5 给定了所要求的关于 copt 的计算精度。

作为求解器 mincx 运行的结果，以下的信息将出现在屏幕上：

```
 Solver for linear objective minimization under LMI constraints

 Iterations   :    Best objective value so far

     1
     2                  -8.511476
     3                 -13.063640
***                  new lower bound:   -34.023978
     4                 -15.768450
***                  new lower bound:   -25.005604
     5                 -17.123012
***                  new lower bound:   -21.306781
     6                 -17.882558
***                  new lower bound:   -19.819471
     7                 -18.339853
```

```
***                 new lower bound:   -19.189417
     8                 -18.552558
***                 new lower bound:   -18.919668
     9                 -18.646811
***                 new lower bound:   -18.803708
    10                 -18.687324
***                 new lower bound:   -18.753903
    11                 -18.705715
***                 new lower bound:   -18.732574
    12                 -18.712175
***                 new lower bound:   -18.723491
    13                 -18.714880
***                 new lower bound:   -18.719624
    14                 -18.716094
***                 new lower bound:   -18.717986
    15                 -18.716509
***                 new lower bound:   -18.717297
    16                 -18.716695
***                 new lower bound:   -18.716873

 Result:  feasible solution of required accuracy
           best objective value:   -18.716695
           guaranteed relative accuracy: 9.50e-006
           f-radius saturation:  0.000% of R = 1.00e+009
```

迭代的次数和当前这次迭代时 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$ 的最佳值分别在左列和右列中。注意，在第一次迭代中没有一个对应的目标函数值，这表明满足约束条件的可行解 $\boldsymbol{x}$ 只是在第二次迭代时才被找到。

4．mincx 也给出决策变量 xopt 的最优值。相应的矩阵变量最优值由以下函数给出

```
Xopt=dec2mat(LMIs,xopt,X)
```

由此可以得到：

$$\boldsymbol{X}_{\mathrm{opt}}=\begin{bmatrix}-6.3542 & -5.8895 & 2.2046\\ -5.8895 & -6.2855 & 2.2201\\ 2.2046 & 2.2201 & -6.0771\end{bmatrix}$$

gevp

求解器 gevp 的一般表达式如下：

```
[lopt,xopt]=gevp(lmisys,nlfc,options,linit,xinit,target)
```

如果问题的线性矩阵不等式约束是可行的，则 gevp 给出了优化问题的全局最小值 lopt 和决策向量 $\boldsymbol{x}$ 的最优解 xopt。相应的矩阵变量的最优解可以应用 dec2mat 得到。

输入分量 lmisys 表示当 $\lambda=1$ 时由（6）~（8）构成的线性矩阵不等式系统。包含 λ 的线性矩阵不等式系统称为线性分式约束。线性分式约束（8）的个数用 nlfc 表示。其他的输入分量都是可选择的。

如果 $(\lambda_0, \boldsymbol{x}_0)$ 是一个可行解，则通过令 linit= λ_0、xinit= $\boldsymbol{x}_0$，将 $(\lambda_0, \boldsymbol{x}_0)$ 设置为 gevp 的初始值。当 $(\lambda_0, \boldsymbol{x}_0)$ 不是可行解时，这样的初始值设置不会被接受。target 的设定值表明了只要当一个可行解 $(\lambda, \boldsymbol{x})$ 满足 $\lambda \le$ target 时，迭代过程就停止。

可选择的输入量 options 是一个 5 维向量，它用来描述优化迭代过程中的一些控制参数：

- options(1)：该参数设定了最优值 lopt 所要求的精度（默认值是 10^{-2}）。
- options(2)：该参数设定优化迭代过程中允许的最大迭代次数（默认值是 100）。
- options(3)：该参数设定了可行域的半径。与求解器 feasp 中的相应参数相同。
- options(4)：该参数用于加快迭代过程的结束。当该参数取值为一个正整数 J 时，表示在最后的 J 次迭代中，如果每次迭代后 λ 的减小幅度在给定的精度内，则优化迭代过程就停止。该参数的默认值是 5。
- options(5)：options(5)=1 表示不显示迭代过程中的数据，options(5)=0（默认值）则相反。

将 options(i)设置为零相当于将相应的控制参数设置为默认值，也可以通过忽略该输入变量来接受默认值。

对广义特征值的最小化问题，在调用求解器 gevp 时，须遵循以下规则：

- 确定包含 λ 的线性矩阵不等式： $\boldsymbol{A}(\boldsymbol{x}) < \boldsymbol{B}(\boldsymbol{x})$（注意没有 λ）；
- 总是把 $\boldsymbol{A}(\boldsymbol{x}) < \boldsymbol{B}(\boldsymbol{x})$ 放在线性矩阵不等式系统的最后；
- 要求有约束 $\boldsymbol{0} < \boldsymbol{B}(\boldsymbol{x})$，或者保证 $\boldsymbol{0} < \boldsymbol{B}(\boldsymbol{x})$ 成立的任何其他约束。

根据以上要求，对于

$$\boldsymbol{B}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{B}_1(\boldsymbol{x}) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{B}_1(\boldsymbol{x}) > \boldsymbol{0}$$

的广义特征值优化问题，我们就不能直接应用求解器 gevp 来求解。为了克服这一困难，可以通过引进矩阵变量 $\boldsymbol{Y}$，并用

$$\boldsymbol{A}(\boldsymbol{x}) < \begin{bmatrix} \boldsymbol{Y} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{Y} < \lambda \boldsymbol{B}_1(\boldsymbol{x}), \quad \boldsymbol{B}(\boldsymbol{x}) > \boldsymbol{0}$$

来代替

$$\boldsymbol{A}(\boldsymbol{x}) < \lambda \boldsymbol{B}(\boldsymbol{x}), \quad \boldsymbol{B}(\boldsymbol{x}) > \boldsymbol{0}$$

而等价的新问题可以用 gevp 来求解。

例 5：对以下的 3 个系统

$$\dot{\boldsymbol{x}}(t)=\boldsymbol{A}_i\boldsymbol{x}(t),\quad (i=1,2,3),$$

其中矩阵 $\boldsymbol{A}_i,(i=1,2,3)$ 由例 3 给出。问题是寻找一个单一的 Lyapunov 函数 $V(\boldsymbol{x})=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{x}$ 来验证给定的这 3 个系统的稳定性，同时使得衰减率 $-\dfrac{\mathrm{d}V(\boldsymbol{x})}{\mathrm{d}t}$ 最大化。这样一个问题等价于如下的优化问题：

$$\begin{aligned}&\min\ \alpha\\ \text{s.t.}\quad &\boldsymbol{I}<\boldsymbol{P}\\ &\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A}_1<\alpha\boldsymbol{P}\\ &\boldsymbol{A}_2^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A}_2<\alpha\boldsymbol{P}\\ &\boldsymbol{A}_3^{\mathrm{T}}\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A}_3<\alpha\boldsymbol{P}\end{aligned}$$

以下应用求解器 gevp 来求解该问题，为此，首先确定线性矩阵不等式系统：

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

lmiterm([1 1 1 0],1)                % P>I:      I
lmiterm([-1 1 1 P],1,1)             % P>I:      P
lmiterm([2 1 1 P],1,A1,'s')         % LFC #1 (lhs)
lmiterm([-2 1 1 P],1,1)             % LFC #1 (rhs)
lmiterm([3 1 1 P],1,A2,'s')         % LFC #2 (lhs)
lmiterm([-3 1 1 P],1,1)             % LFC #2 (rhs)
lmiterm([4 1 1 P],1,A3,'s')         % LFC #3 (lhs)
lmiterm([-4 1 1 P],1,1)             % LFC #3 (rhs)
lmis=getlmis
```

进而，通过以下命令来调用 gevp：

```
[alpha,popt]=gevp(lmis,3)
```

得到 alpha = -0.122 是该问题的最优值，因此相应的最大衰减率是 0.122，最优解是

$$\boldsymbol{P}=\begin{bmatrix}5.58 & -8.35\\ -8.35 & 18.64\end{bmatrix}$$

如何从决策变量到矩阵变量以及从矩阵变量到决策变量

当线性矩阵不等式由相应的矩阵变量描述时，线性矩阵不等式求解器涉及的是由这些

矩阵变量中的独立元所组成的决策向量 $\boldsymbol{x}$。两个函数 mat2dec 和 dec2mat 可以实现这两种变量之间的转换。

考虑一个具有三个矩阵变量 $\boldsymbol{X}_1$、$\boldsymbol{X}_2$、$\boldsymbol{X}_3$ 的线性矩阵不等式系统。给定这些变量的特定值 X1、X2、X3，那么由 mat2dec 可以得到相应的决策向量的值：

```
xdec=mat2dec(lmisys,X1,X2,X3)
```

如果 lmisys 后分量的个数和线性矩阵不等式系统 lmisys 中的矩阵变量个数不符，则系统会提示一个出错信息。

这个函数在线性矩阵不等式求解器 mincx 或 gevp 的初始化中也是很有用的。例如，给定 $\boldsymbol{X}_1$、$\boldsymbol{X}_2$、$\boldsymbol{X}_3$ 的一个初始猜测值，mat2dec 就形成了相应决策向量的初始值 xinit。

反之，给定决策向量的一个值 xdec，那么可以通过函数 dec2mat 给出相应的第 k 个矩阵的取值。例如，以下的表示式可以给出第 2 个矩阵变量的取值：

```
X2=dec2mat(lmisys,xdec,2)
```

函数 dec2mat 中的最后一个分量表明了要求的是第 2 个矩阵变量，这里也可以用 lmivar 定义的相应矩阵变量的变量名。

矩阵变量和决策变量的总数分别由 matnbr 和 decnbr 给出。另外，函数 decinfo 提供了决策变量和矩阵变量之间关系的一些详细信息。

A.5 结果验证

LMI 工具箱提供了两个函数用于分析和验证一个线性矩阵不等式优化问题的结果。对一个给定的决策向量的值，例如由线性矩阵不等式求解器给出的可行或最优解向量，函数 evallmi 求出线性矩阵不等式系统中所有变量项的值，进而应用 showlmi 给出特定线性矩阵不等式的左边和右边。

在例 4 中，我们可以这样来验证由 mincx 得到的最优解 xopt 是否满足给定的约束条件：

```
evlmi=evallmi(LMIs,xopt)
[lhs,rhs]=showlmi(evlmi,1)
```

第一个命令表示对给定的决策变量值 xopt，求取系统的值，第二个命令则显示第 1 个线性矩阵不等式的左边和右边的矩阵值。这个不等式的成立与否可以通过

```
eig(lhs-rhs)
```

来检验，得到的结果是

```
ans=
  -2.0387e-04
  -3.9333e-05
  -1.8917e-07
  -4.6680e+01
```

由此可以看到 lhs-rhs 是负定的，因此 xopt 满足第 1 个线性矩阵不等式。

A.6　修改一个线性矩阵不等式系统

LMI 工具箱提供了函数 dellmi、delmvar 和 setmvar，它们可以用来修改一个已经确定的线性矩阵不等式系统。以下来说明这些函数的用法。

Dellmi

用 dellmi 可以从一个线性矩阵不等式系统中删除一个完整的线性矩阵不等式。例如对例 1 中的线性矩阵不等式系统，假定它由 lmisys 描述，现在要删除矩阵变量 $\boldsymbol{X}$ 的正定性约束，这可以通过以下的命令来实现：

```
newsys=dellmi(lmisys,2)
```

其中第 2 个分量表明了要删除的是哪一个线性矩阵不等式。所导出的由两个线性矩阵不等式构成的系统由 newsys 表示。

原来系统中的线性矩阵不等式标识符（由位置排列确定的）不因删除了其中的一些不等式后而改变它们在新的线性矩阵不等式系统中的标识符。因此，在例 1 中删除了第 2 个线性矩阵不等式后，

$$\boldsymbol{S} > \boldsymbol{I}$$

仍然是第 3 个线性矩阵不等式，尽管它在新的系统中已位于第 2 的位置。为了避免引起混淆，更安全的方法是使用由 newlmi 给定的线性矩阵不等式名。如在例 1 中，BRL、Xpos 和 Slmi 依次是三个线性矩阵不等式的名称，则 Slmi 仍是通过

```
newsys=dellmi(lmisys,Xpos)
```

删除了第 2 个不等式后所得到的新系统中线性矩阵不等式 $\boldsymbol{S} > \boldsymbol{I}$ 的名称。

delmvar

另一种修改一个线性矩阵不等式系统的方式是删除一个矩阵变量，即在所有包含该变量的变量项中删除该矩阵变量。delmvar 提供了这样一种修改功能。例如，考虑线性矩阵

不等式系统

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}+\boldsymbol{B}\boldsymbol{W}+\boldsymbol{W}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}+\boldsymbol{I}<\boldsymbol{0}$$

其中：$\boldsymbol{X}=\boldsymbol{X}^{\mathrm{T}}\in\mathbf{R}^{4\times4}$ 和 $\boldsymbol{W}\in\mathbf{R}^{2\times4}$ 是这个线性矩阵不等式中的两个矩阵变量。这个线性矩阵不等式可以用以下的命令来描述：

```
setlmis([])
X=lmivar(1,[4 1])     %  X
W=lmivar(2,[2 4])     %  S

lmiterm([1 1 1 X],1,A,'s')
lmiterm([1 1 1 W],B,1,'s')
lmiterm([1 1 1 0],1)

lmisys=getlmis
```

为了删除变量 $\boldsymbol{W}$，输入命令

```
newsys=delmvar(lmisys,W)
```

得到的 newsys 描述了 Lyapunov 不等式

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}+\boldsymbol{I}<\boldsymbol{0}$$

注意：delmvar **将自动删除所有的只依赖所删除矩阵变量的不等式。矩阵变量的变量名不会因删除个别的矩阵变量而改变。**

删除一个矩阵变量实际上等价于让这个矩阵变量等于零，这一功能也可以由下面的 setmvar 方便地实现。

setmvar

函数 setmvar 用于给某个矩阵变量赋值，一旦赋值，该变量就从原来的问题中消失了，包含该变量的所有项都成为常数项。另外，这个函数在优化问题中也是很有用的，例如可以通过固定某个变量，然后对其他变量进行优化。

考虑例 1，我们想要知道 $\boldsymbol{G}$ 的最大增益是否小于 1，即 $\|\boldsymbol{G}\|_{\infty}<1$ 是否成立。这相当于令尺度矩阵 $\boldsymbol{D}$（或等价的，$\boldsymbol{S}=\boldsymbol{D}^{\mathrm{T}}\boldsymbol{D}$）等于单位矩阵的某个倍数。注意到 $\boldsymbol{S}>\boldsymbol{I}$，因此可以选择 $\boldsymbol{S}=2\boldsymbol{I}$。为了让 $\boldsymbol{S}$ 取这样一个值，输入

```
newsys=setmvar(lmisys,S,2)
```

其中的第 2 个分量是相关矩阵变量的变量名 S，第 3 个分量是应赋给变量 $\boldsymbol{S}$ 的值。这里 2 是 $2\boldsymbol{I}$ 的简捷表示。新系统 newsys 中的线性矩阵不等式是

$$\begin{bmatrix}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}+\boldsymbol{X}\boldsymbol{A}+2\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C} & \boldsymbol{X}\boldsymbol{B}\\ \boldsymbol{B}^{\mathrm{T}}\boldsymbol{X} & -2\boldsymbol{I}\end{bmatrix}<\boldsymbol{0}$$

$$\boldsymbol{X} > \boldsymbol{0}$$

$$2\boldsymbol{I} > \boldsymbol{I}$$

容易看到最后一个不等式不依赖任何变量，而且是成立的，因此可以将它从系统中删除。这可以通过

```
newsys=dellmi(newsys,3)
```

或者

```
newsys =dellmi(newsys,Slmi)
```

来实现。其中 Slmi 是由 newlmi 给定的第 3 个线性矩阵不等式的名称。

A.7　一些进一步的功能

结构矩阵变量

在前面介绍的 lmivar 使用方法中，我们知道可以用第 1 和第 2 类型的变量来描述具有对称结构和一般长方结构的矩阵变量。为了描述一些具有更复杂结构的矩阵变量或变量间的相互关系，则需要利用第 3 种类型的变量以及用决策变量来确定矩阵变量中的独立元。

对第三种类型的变量结构，每一个元被确定为 0 或 $\pm x_n$，其中 x_n 是第 n 个决策变量。为了描述第三种类型的变量 $\boldsymbol{X}$，首先需要确定变量 $\boldsymbol{X}$ 中包含了多少个独立的变元，这些独立变元构成了包含在变量 $\boldsymbol{X}$ 中的那部分决策变量。如果在定义变量 $\boldsymbol{X}$ 之前已经有 n 个决策变量，将变量 $\boldsymbol{X}$ 中涉及到的 p 个独立变元表示成 $x_{n+1},\cdots,x_{n+p}$，那么可以应用这些 $x_{n+1},\cdots,x_{n+p}$ 来定义变量 $\boldsymbol{X}$ 的结构。为了有助于描述这一类型的矩阵变量，lmivar 提供了两个额外的输出，它的一般表达式是

```
[X,n,sX]=lmivar(type,struct)
```

其中，n 表示到目前为止使用的决策变量的总数，sX 表明了变量 $\boldsymbol{X}$ 中的每一个元依赖于决策变量 $x_1,\cdots,x_n$ 的哪一个元。

下面举例说明如何应用 lmivar 来描述第三类的矩阵变量。首先考虑不相关矩阵变量的情况。

例 6：考虑具有结构

$$\boldsymbol{X}=\begin{bmatrix}\boldsymbol{X}_1 & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{X}_2\end{bmatrix}$$

的矩阵变量 $\boldsymbol{X}$，其中 $\boldsymbol{X}_1$ 和 $\boldsymbol{X}_2$ 分别是 2×3 维和 3×2 维的长方矩阵。这样的一个矩阵变量 $\boldsymbol{X}$ 可以描述如下：

1．定义长方矩阵变量 $\boldsymbol{X}_1$ 和 $\boldsymbol{X}_2$

```
setlmis([])
[X1,n,sX1]=lmivar(2,[2 3])
[X2,n,sX2]=lmivar(2,[3 2])
```

输出 sX1 和 sX2 给出了包含在变量 $\boldsymbol{X}_1$ 和 $\boldsymbol{X}_2$ 中的决策变量：

```
>>sX1

sX1=
     1   2   3
     4   5   6

>>sX2

sX2=
7  8
9  10
    11  12
```

例如，sX2(1, 1)=7 表示 $\boldsymbol{X}_2$ 的第(1, 1)处的元是第 7 个决策变量。

2．利用 Type 3 来定义矩阵变量 $\boldsymbol{X}$ ，并且用 $\boldsymbol{X}_1$ 和 $\boldsymbol{X}_2$ 来定义其结构：

```
[X,n,sX]=lmivar(3,[sX1,zeros(2);zeros(3),sX2])
```

这样得到的变量 X 具有所要的结构。

```
>>sX

sX=
    1   2   3   0   0
    4   5   6   0   0
    0   0   0   7   8
    0   0   0   9  10
    0   0   0  11  12
```

例 7：假设问题变量包含一个 3×3 的对称矩阵 $\boldsymbol{X}$ 和一个 3×3 的 Toepliz 矩阵

$$\boldsymbol{Y}=\begin{bmatrix} y_1 & y_2 & y_3 \\ y_2 & y_1 & y_2 \\ y_3 & y_2 & y_1 \end{bmatrix}$$

矩阵变量 $\boldsymbol{Y}$ 只有 3 个独立的变元，因此涉及 3 个决策变量。由于 $\boldsymbol{Y}$ 和 $\boldsymbol{X}$ 是相互独立的，因此和 $\boldsymbol{Y}$ 相关的决策变量可以记为 $n+1$、$n+2$、$n+3$，其中 n 是与 $\boldsymbol{X}$ 相关的决策变量的个

数。这个数可以通过定义矩阵变量 $\boldsymbol{X}$ 的命令得到：

```
setlmis([])
[X,n]=lmivar(1,[3 1])
```

其中输出[X, n]中的第 2 个分量给出了到目前为止使用的决策变量的总数（这里 n=6）。有了这个数以后， $\boldsymbol{Y}$ 可以定义如下：

```
Y=lmivar(3,n+[1 2 3;2 1 2;3 2 1])
```

或等价的，

```
Y=lmivar(3,toeplitz(n+[1 2 3]))
```

其中的 toeplitz 是一个标准的 MATLAB 函数。为了验证，也可以使用 decinfo 来显示 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$ 中决策变量的分布。

```
lmis=getlmis
decinfo(lmis,X)

ans=
1  2  4
2  3  5
4  5  6

decinfo(lmis,Y)

ans=
7  8  9
8  7  8
9  8  7
```

下面是一个关于相互关联的矩阵变量的例子。

例 8：考虑具有以下结构的三个矩阵变量 $\boldsymbol{X}$、$\boldsymbol{Y}$ 和 $\boldsymbol{Z}$ ：

$$\boldsymbol{X}=\begin{bmatrix} x & 0 \\ 0 & y \end{bmatrix},\quad \boldsymbol{Y}=\begin{bmatrix} z & 0 \\ 0 & t \end{bmatrix},\quad \boldsymbol{Z}=\begin{bmatrix} 0 & -x \\ -t & 0 \end{bmatrix}$$

其中： x、y、z、t 是独立的标量变量。为了描述这样的三个矩阵变量，首先定义两个独立的矩阵变量 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$ （两个都是第 1 种类型的）：

```
setlmis([])
[X,n,sX]=lmivar(1,[1 0;1 0])
[Y,n,sY]=lmivar(1,[1 0;1 0])
```

其中 lmivar 输出中的第 3 个分量给出了 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$ 中的每个元对决策变量 $(x_1, x_2, x_3, x_4) = (x, y, z, t)$ 的依赖关系：

```
sX=
1  0
0  2

sY=
    3  0
    0  4
```

使用 lmivar 的类型 3，可以确定矩阵变量 $\boldsymbol{Z}$ 的结构：

```
[Z,n,sZ]=lmivar(3,[0 -sX(1,1);-sY(2,2) 0])
```

由于 sX(1, 1)等于 x_1，而 sY(2, 2)等于 x_4，因此上面的命令定义了

$$\boldsymbol{Z} = \begin{bmatrix} 0 & -x_1 \\ -x_4 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -x \\ -t & 0 \end{bmatrix}$$

复线性矩阵不等式

前面讨论的线性矩阵不等式都是针对实矩阵的，不能直接用来处理包含复矩阵的线性矩阵不等式问题。注意到一个复的埃尔米特矩阵（Hermitian matrix） $\boldsymbol{L}(\boldsymbol{x})$ 满足 $\boldsymbol{L}(\boldsymbol{x}) < \boldsymbol{0}$ 当且仅当

$$\begin{bmatrix} \mathrm{Re}(\boldsymbol{L}(\boldsymbol{x})) & \mathrm{Im}(\boldsymbol{L}(\boldsymbol{x})) \\ -\mathrm{Im}(\boldsymbol{L}(\boldsymbol{x})) & \mathrm{Re}(\boldsymbol{L}(\boldsymbol{x})) \end{bmatrix} < \boldsymbol{0}$$

因此，复线性矩阵不等式可以转换成实线性矩阵不等式来讨论。

以下提出一个将复线性矩阵不等式转换为实线性矩阵不等式的方法：

- 把每一个复矩阵变量 $\boldsymbol{X}$ 分解成

  $$\boldsymbol{X} = \boldsymbol{X}_1 + j\boldsymbol{X}_2$$

  其中 $\boldsymbol{X}_1$ 和 $\boldsymbol{X}_2$ 是实的。
- 把每一个复系数矩阵 $\boldsymbol{A}$ 分解成

  $$\boldsymbol{A} = \boldsymbol{A}_1 + j\boldsymbol{A}_2$$

  其中 $\boldsymbol{A}_1$ 和 $\boldsymbol{A}_2$ 是实的。
- 进行所有复矩阵的乘积运算。对每一个线性矩阵不等式的实部和复部，得到其关于 $\boldsymbol{X}_1$、$\boldsymbol{X}_2$ 的仿射表示式，进而得到等价的实线性矩阵不等式表示式。

对于没有外因子的线性矩阵不等式，在不等式中用 $\begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_2 \\ -\boldsymbol{X}_2 & \boldsymbol{X}_1 \end{bmatrix}$ 来替代矩阵变量 $\boldsymbol{X}$ =

$\boldsymbol{X}_1 + j\boldsymbol{X}_2$，用$\begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 \\ -\boldsymbol{A}_2 & \boldsymbol{A}_1 \end{bmatrix}$来替代给定的矩阵$\boldsymbol{A} = \boldsymbol{A}_1 + j\boldsymbol{A}_2$。例如对以下的线性矩阵不等式系统

$$\boldsymbol{M}^{\mathrm{H}}\boldsymbol{X}\boldsymbol{M} < \boldsymbol{X}, \quad \boldsymbol{X} = \boldsymbol{X}^{\mathrm{H}} > \boldsymbol{I}$$

它的等价的实线性矩阵不等式系统表示是

$$\begin{bmatrix} \boldsymbol{M}_1 & \boldsymbol{M}_2 \\ -\boldsymbol{M}_2 & \boldsymbol{M}_1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_2 \\ -\boldsymbol{X}_2 & \boldsymbol{X}_1 \end{bmatrix} \begin{bmatrix} \boldsymbol{M}_1 & \boldsymbol{M}_2 \\ -\boldsymbol{M}_2 & \boldsymbol{M}_1 \end{bmatrix} < \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_2 \\ -\boldsymbol{X}_2 & \boldsymbol{X}_1 \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_2 \\ -\boldsymbol{X}_2 & \boldsymbol{X}_1 \end{bmatrix} > \boldsymbol{I}$$

其中：$\boldsymbol{M} = \boldsymbol{M}_1 + j\boldsymbol{M}_2$，$\boldsymbol{X} = \boldsymbol{X}_1 + j\boldsymbol{X}_2$，$\boldsymbol{M}_i$、$\boldsymbol{X}_i$是实矩阵。

如果假定$\boldsymbol{M} \in \mathbf{C}^{5\times 5}$，则以上考虑的复线性矩阵不等式可以描述如下：

```
M1=real(M),  M2=imag(M)
bigM=[M1 M2;-M2 M1]
setlmis([])

% declare bigX=[X1 X2;-X2 X1] with X1=X1' and X2+X2'=0:

[X1,n1,sX1]=lmivar(1,[5 1])
[X2,n2,sX2]=lmivar(3,skewdec(5,n1))
bigX=lmivar(3,[sX1 sX2;-sX2 sX1])

% describe the real counterpart of the complex LMI system:

lmiterm([1 1 1 0],1)
lmiterm([-1 1 1 bigX],1,1)
lmiterm([2 1 1 bigX],bigM',bigM)
lmiterm([-2 1 1 bigX],1,1)

lmis=getlmis
```

结构矩阵变量 bigX=$\begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_2 \\ -\boldsymbol{X}_2 & \boldsymbol{X}_1 \end{bmatrix}$按以下步骤确定：

1．将$\boldsymbol{X}_1$确定为实的对称矩阵变量，保存它的结构描述 sX1 和$\boldsymbol{X}_1$中使用的决策变量个数 n1。

2．使用 skewdec，将$\boldsymbol{X}_2$确定为一个斜对称矩阵变量。函数 skewdec(5,n1)确定了依赖于决策变量 n1+1, n1+2, ... 的5×5维斜对称矩阵。

3．使用第 3 种类型的矩阵变量，利用$\boldsymbol{X}_1$和$\boldsymbol{X}_2$的结构 sX1, sX2 定义 bigX 的结构。

为 mincx 确定目标函数 $c^{\mathrm{T}}x$

线性矩阵不等式求解器 mincx 是在线性矩阵不等式约束下求线性目标函数 $c^{\mathrm{T}}x$ 的最小值，其中 x 是由决策变量构成的向量。在多数控制问题中，这样的目标函数是由线性矩阵不等式系统中的矩阵变量而不是决策变量表示的，例如， $\mathrm{Trace}(X)$，其中 X 是一个对称矩阵变量； $u^{\mathrm{T}}Xu$，其中 u 是一个给定的向量。

当目标函数是矩阵变量的一个仿射函数时，函数 defcx 可以为 c 的确定提供一个方便的方法。以下通过一个例子来说明这个方法。

考虑线性目标函数

$$\mathrm{Trace}(X)+x_0^{\mathrm{T}}Px_0$$

其中 X 和 P 是两个对称矩阵变量， x_0 是一个给定向量。如果 lmisys 是所考虑的线性矩阵不等式系统的内部表示， x_0、X、P 由以下命令确定：

```
x0=[1 1]
setlmis([])
X=lmivar(1,[3 0])
P=lmivar(1,[2 1])
      :
      :
lmisys=getlmis
```

使得 $c^{\mathrm{T}}x=\mathrm{Trace}(X)+x_0^{\mathrm{T}}Px_0$ 的向量 c 可以用以下命令描述：

```
n=decnbr(lmisys)
c=zeros(n,1)

for j=1:n,
   [Xj,Pj]=defcx(lmisys,j,X,P)
   c(j)=trace(Xj)+x0'*Pj*x0
end
```

第 1 个命令给出了问题中的决策变量的个数，第 2 个命令则确定了向量 c 的维数。循环语句 for 完成以下的运算：

1．取 $x_j=1$， x 的其它元等于零，求矩阵变量 X 和 P，该运算由 defcx 完成。defcx 的输入除了 lmisys 和 j 以外，还有包含在目标函数中的矩阵变量的变量名 X 和 P，其输出是 Xj 和 Pj。

2．对 X =Xj 和 P =Pj，求相应的目标函数值。根据定义得到 c 的第 j 个元。

在以上考虑的例子中，相应的结果是：

```
c=
```

```
3
1
2
1
```

其他的目标函数可以通过采用以下的一般框架类似地处理：

```
n=decnbr(LMI system)
c=zeros(n,1)
for j=1:n,
   [matrix values]=defcx(LMI system,j,
matrix identifiers)
   c(j)=objective(matrix values)
end
```

A.8 系统模型描述

考虑具有以下形式的线性时不变模型

$$\begin{aligned} \boldsymbol{E}\dot{\boldsymbol{x}}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t) \end{aligned} \tag{15}$$

其中： $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{E}$ 是已知的实矩阵，且 $\boldsymbol{E}$ 是可逆的， $\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{y}(t)$ 分别是系统的状态、输入和输出。模型（15）在描述一些参数依赖的系统模型时是特别有用的。另外，当矩阵 $\boldsymbol{E}$ 的逆是病态时，模型（15）可以避免求矩阵 $\boldsymbol{E}$ 的逆。最后，许多动态系统也可以很自然地写成模型（15）的形式，例如，二阶系统

$$\begin{aligned} m\ddot{x} + f\dot{x} + kx &= u \\ y &= x \end{aligned}$$

可以写成以下具有形式（15）的模型：

$$\begin{bmatrix} 1 & 0 \\ 0 & m \end{bmatrix} \dot{\xi}(t) = \begin{bmatrix} 0 & 1 \\ -k & -f \end{bmatrix} \xi(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \xi(t)$$

其中 $\xi(t) = [x(t) \quad \dot{x}(t)]^{\mathrm{T}}$ 。

状态空间模型（15）可以用一个单一的 MATLAB 矩阵 SYSTEM 来表示，它具有以下的结构形式：

$$\left[\begin{array}{cc|c} & & n \\ \boldsymbol{A}+j(\boldsymbol{E}-\boldsymbol{I}) & \boldsymbol{B} & 0 \\ \boldsymbol{C} & \boldsymbol{D} & \vdots \\ & & 0 \\ \hline \multicolumn{2}{c|}{\boldsymbol{0}} & -\mathrm{Inf} \end{array}\right]$$

其中右上角的 n 表示状态的维数。

LMI 工具箱提供了函数 ltisys 和 ltiss，用以产生矩阵 SYSTEM 以及从 SYSTEM 得到状态空间模型的系数矩阵。例如，

```
sys=ltisys(-1,1,1,0)
```

给出了描述系统

$$\dot{x}=-x+u,\quad y=x$$

的 SYSTEM 矩阵。为了从 SYSTEM 矩阵 sys 得到状态空间模型的系数矩阵 $\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}$，可以应用以下的命令：

```
[A,B,C,D]=ltiss(sys)
```

对于一个单输入单输出系统，函数 ltitf 给出了矩阵 SYSTEM 的传递函数表示。反之，若 $G(s)=n(s)/d(s)$，则命令

```
sys=ltisys('tf',n,d)
```

给出了系统 $G(s)$ 的一个状态空间实现。其中 n 和 d 分别是多项式 $n(s)$ 和 $d(s)$ 的向量表示。

sinfo(sys)给出了 SYSTEM 矩阵 sys 所表示的系统的状态，输入和输出向量的维数。spol(sys)则给出了系统的极点。函数 ssub 给出了由系统的特定输入和输出组成的新系统的 SYSTEM 矩阵。例如系统 $\boldsymbol{G}$ 有两个输入和三个输出，则从系统 $\boldsymbol{G}$ 的第 1 个输入到第 2、第 3 个输出的新系统可以由以下命令得到：

```
ssub(G,1,2:3)
```

其中 G 是系统 $\boldsymbol{G}$ 的 SYSTEM 矩阵表示。函数 sinv 给出系统 $\boldsymbol{G}(s)$ 的逆 $\boldsymbol{H}(s)=\boldsymbol{G}^{-1}(s)$，其中假定 $\boldsymbol{G}(s)$ 的状态空间模型表示中的矩阵 $\boldsymbol{D}$ 是可逆的。函数 sbalanc 可用来平衡一个线性时不变系统的状态空间实现，即寻找一个对角尺度相似变换矩阵，以降低矩阵 $\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}$ 的范数。

一个复杂的系统可以通过一些更加简单的环节经串联、并联和反馈等得到。LMI 工具箱提供了函数来产生由各个环节的 SYSTEM 矩阵经串联、并联和反馈等得到的系统的 SYSTEM 矩阵。

设 g1 和 g2 分别是系统 $\boldsymbol{G}_1(s)$ 和 $\boldsymbol{G}_2(s)$ 的 SYSTEM 矩阵，则 sadd 给出了

$\boldsymbol{G}_1(s)+\boldsymbol{G}_2(s)$ 的 SYSTEM 矩阵，而 `smult(g1,g2)`给出了传递函数$\boldsymbol{G}_2(s)\boldsymbol{G}_1(s)$（注意顺序）的 SYSTEM 矩阵。这两个函数最多可以有 10 个输入分量，其中最多可以有一个是参数依赖的系统。

`sdiag(g1,g2)`给出了传递函数$\boldsymbol{G}(s)=\text{diag}\{\boldsymbol{G}_1(s),\boldsymbol{G}_2(s)\}$的 SYSTEMS 矩阵。

对于下图表示的反馈关联系统

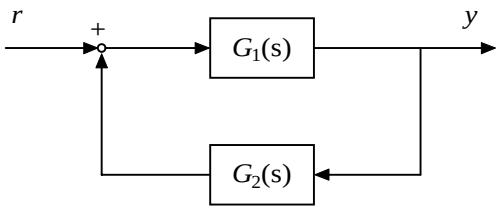

图 1 反馈关联系统

函数 `sloop` 给出了从$\boldsymbol{r}$到$\boldsymbol{y}$的闭环系统的 SYSTEM 矩阵。

函数 `slft` 则给出了以下更一般的反馈关联系统从$\begin{bmatrix}\boldsymbol{w}_1\\ \boldsymbol{w}_2\end{bmatrix}$到$\begin{bmatrix}\boldsymbol{z}_1\\ \boldsymbol{z}_2\end{bmatrix}$的闭环传递函数的 SYSTEM 矩阵表示：

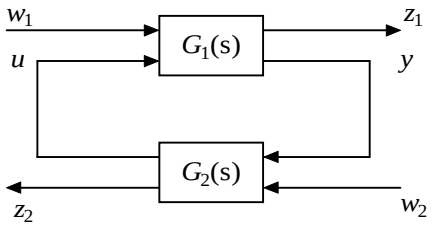

图 2 关联系统

若$\boldsymbol{u}\in\mathbf{R}^2,\boldsymbol{y}\in\mathbf{R}^3$，则描述以上反馈关联系统的命令是

```
slft(g1,g2,2,3)
```

这个函数在H_∞控制中计算线性分式关联时是很有用的。

LMI：Linear Matrix Inequality，就是线性矩阵不等式。

在 Matlab 当中，我们可以采用图形界面的 lmiedit 命令，来调用 GUI 接口，但是我认为采用程序的方式更方便（也因为我不懂这个 lmiedit 的 GUI）。

对于 LMI Lab， 其中有三种求解器（solver）： feasp，mincx 和 gevp。

每个求解器针对不同的问题：

feasp：解决可行性问题（feasibility problem），例如：A(x)<B(x)。

mincx：在线性矩阵不等式的限制下解决最小化问题（Minimization of a linear objective under LMI constraints），例如最小化 c'x，在限制条件 A(x) < B(x)下。

gevp：解决广义特征值最小化问题。例如：最小化 lambda，在 0<B(x),A(x)<lamba*B(x) 限制条件下。

要解决一个 LMI 问题，首要的就是要把线性矩阵不等式表示出来。

对于以下类型的任意的 LMI 问题

N' * L(X1, . . . , XK) * N < M' * R(X1, . . . , XK) * M

其中 X1, . . . , XK 是结构已经事先确定的矩阵变量。左侧和右侧的外部因子（outer factors）N 和 M 是给定的具有相同维数的矩阵。

左侧和右侧的内部因子（inner factors）L(.)和 R(.)是具有相同结构的对称块矩阵。每一个块由 X1, . . . , XK 以及它们的转置组合而成形成的。

解决 LMI 问题的步骤有两个：

1、定义维数以及每一个矩阵的结构，也就是定义 X1, . . . , XK。

2、描述每一个 LMI 的每一项内容（Describe the term content of each LMI）

此处介绍两个术语：

矩阵变量（Matrix Variables）：例如你要求解 X 满足 A(x)<B(x)，那么 X 就叫做矩阵变量。

项（Terms）：项是常量或者变量（Terms are either constant or variable）。

常项（Constant Terms）是确定的矩阵。可变项（Variable Terms)是哪些含有矩阵变量的项，例如：X*A, X*C'。如果是 X*A + X*C'，那么记得要把它当成两项来处理。

好了废话不说了，让我们来看个例子吧（下面是一线性时滞系统）。

500)this.width=500;" border=0>

针对这个式子，如果存在满足如下 LMI 的正矩阵(positive-define)的 Q，S1，S2 和矩阵 M，那么我们就称作

该系统为 H-inf 渐进稳定的，并且 gammar 是上限。

500)this.width=500;" border=0>

该论文的地址为：论文原文地址

该论文的算例为：

500)this.width=500;" border=0>

我们要实现的就利用 LMI 进行求解，验证论文结果。

首先我们要用 setlmis([])命令初始化一个 LMI 系统。

接下来，我们就要设定矩阵变量了。采用函数为 lmivar

语法：X = lmivar(type,struct)

type=1: 定义块对角的对称矩阵。 每一个对角块或者是全矩阵<任意对称矩阵>，标量<单位矩阵的乘积>，或者是零阵。

如果 X 有 R 个对角块，那么后面这个 struct 就应该是一个 Rx2 阶的的矩阵，在此矩阵中，struct(r,1)表示第 r 个块的大小，struct(r,2) 表示第 r 个块的类型<1--全矩阵，0--标量，-1--零阵）。

比如一个矩阵有两个对角块，其中一个是 2x2 的全对称矩阵，第二个是 1x1 的一个标量，那么该矩阵变量应该表示为 X = lmivar(1, [2 1; 1 0]) 。

type=2: mxn 阶的矩阵，只需要写作 struct = [m,n]即可。

type=3: 其它类型。针对类型 3，X 的每一个条目（each entry of X）被定义为 0 或者是+(-)xn，此处 xn 代表了第 n 个决策变量。

那么针对我们的例子，我们如此定义变量：

```
% Q is a symmetric matrix, has a block size of 2 and this block is symmetric

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

% S1 a symmeric matrix, size 2

S1 = lmivar(1, [2 1]);
```

% S2 is 1 by 1 matrix

S2 = lmivar(1, [1 0]);

% Type of 2, size 1 by 2

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

定义完成变量之后，我们就该用 lmiterm 来描述 LMI 中的每一个项了。Matlab 的官方文档提示我们，如果要描述一个 LMI 只需要描述上三角或者下三角元素就可以了，否则会描述成另一个 LMI。

When describing an LMI with several blocks, remember to specify only the terms in the blocks on or below the diagonal (or equivalently, only the terms in blocks on or above the diagonal).

语法为：lmiterm(termID,A,B,flag)

termID 是一个四维整数向量，来表示该项的位置和包含了哪些矩阵变量。

termID(1)可以为+p 或者-p，+p 代表了这个项位于第 p 个线性矩阵不等式的左边，-p 代表了这个项位于第 p 个线性矩阵不等式的右边。注意：按照惯例来讲，左边通常指较小的那边。

termID(2:3):

1、对于外部变量来说，取值为[0,0];

2、对于左边或者右边的内部变量来说，如果该项在(i,j)位置，取值[i,j]

termID(4):

1、对于外部变量，取值为 0

2、对于 A*X*B，取值 X

3、对于 A*X'*B，取值-X

flag(可选，值为 s）:

因为：(A*X*B) + (A*X*B)T = A*X*B + B'*X'*A'，所以采用 s 来进行简写。

比如：针对 A*X + X'*A'

我们采用笨方法：

lmiterm([1 1 1 X],A,1)

lmiterm([1 1 1 -X],1,A')

那么简写就是 lmiterm([1 1 1 X],A,1,'s')

接下来我们就看该论文中的算例吧：(1,1)位置是

-Q+Bd*S2*Bd'+Ad*S1*Ad';

我们应该表示为：

```
% pos in (1, 1)
lmiterm([1 1 1 Q], -1, 1);
lmiterm([1 1 1 S2], Bd, Bd');
lmiterm([1 1 1 S1], Ad, Ad');
```

其它位置仿照写就行了，不懂了多看帮助文档。

把每一个项都定义以后，要记得

```
lmis = getlmis;
[tmin, feas] = feasp(lmis)
```

getlmis:是在完成定义变量和项之后，LMI 系统的内部表示就可以通过此命令获得（After completing the description of a given LMI system with lmivar and lmiterm, its internal representation lmisys is obtained with the command）。

feasp 是调用 feasp 求解器，看有没有可行解。feas 就是可行解。

下面我把代码贴上去，那些常数矩阵都在此源程序中定义了。

```
A = [2 1; 0 1];
Ad = [0.2 0.1; 0 0.1];
B1 = [0.1 0.1]';
B2 = [1 1]';
Bd = [0.1 0.1]';
C = [1, 1];
Cd = [0.1, 0.1];
D11 = 0.1;
D12 = 1;
Dd = 0.1;
gammar = 1;
% Initial a LMI system
setlmis([]);
% Define Variables
% Q is a symmetric matrix, has a block size of 2 and this block is symmetric
Q = lmivar(1, [2 1]);
```

```
% S1 a symmeric matrix, size 2
S1 = lmivar(1, [2 1]);
% S2 is 1 by 1 matrix
S2 = lmivar(1, [1 0]);
% Type of 2, size 1 by 2
M = lmivar(2, [1 2]);
% Q, S1, S2 > 0
lmiterm([-2 1 1 Q], 1, 1);
lmiterm([-3 1 1 S1], 1, 1);
lmiterm([-4 1 1 S2], 1, 1);
% pos in (1, 1)
lmiterm([1 1 1 Q], -1, 1);
lmiterm([1 1 1 S2], Bd, Bd');
lmiterm([1 1 1 S1], Ad, Ad');
% pos (1, 2)
lmiterm([1 1 2 Q], A, 1);
lmiterm([1 1 2 M], B2, 1);
% pos(1, 3)
lmiterm([1 1 3 0], B1);
% pos(1, 4)
lmiterm([1 1 4 S2], Bd, Dd');
lmiterm([1 1 4 S1], Ad, Cd');
% pos(2, 2)
lmiterm([1 2 2 Q], -1, 1);
% pos(2, 4)
lmiterm([1 2 4 Q], 1, C');
lmiterm([1 2 4 -M], 1, D12');
% pos(2, 5)
lmiterm([1 2 5 -M], 1, 1);
```

```
% pos(2, 6)
lmiterm([1 2 6 Q], 1, 1);
% pos(3, 3)
lmiterm([1 3 3 0], -(gammar^2));
% pos(3, 4)
lmiterm([1 3 4 0], D11');
% pos(4, 4)
lmiterm([1 4 4 0], -1);
lmiterm([1 4 4 S1], Cd, Cd');
lmiterm([1 4 4 S2], Dd, Dd');
lmiterm([1 5 5 S2], -1, 1);
lmiterm([1 6 6 S1], -1, 1);
lmis = getlmis;
[tmin, feas] = feasp(lmis)
```

运行后，就调用 dec2mat 把决策变量转化为矩阵形式。

```
Q = dec2mat(lmis, feas, Q)
Q =
    1.9253   -2.2338
   -2.2338    9.1054
```

可以看到，和论文中的一样。