线性矩阵不等式(LMI)的 MATLAB 求解©

作者：dynamic

时间：2008.12.10

★★

★★

\

近年来，线性矩阵不等式广泛应用于解决系统与控制中的一系列问题。随着解决线 LMI 内点法的提出以及 Matlab 中 LMI 控制工具箱的推广，LMI 这一工具已经受到人重视。LMI 控制工具箱已经成为了从控制工程到系统识别设计和结构设计等诸多领域的一个强大的设计工具。由于许多控制问题都可以转化为一个 LMI 系统的可行性问题，或者是一个具有 LMI 约束大的徒优化问题，应用 LMI 来解决系统和控制问题已经成为这些领域中的一大研究热点。

LMI 控制工具箱，采用内点法的 LMI 求解器，这些求解器比经典的凸优化算法速度有了显著提高。另方方面，它采用了有效的 LMI 结构化表示，在求解和计算领域做出了重大贡献。

LMI 基础知识

一、LMI 的一般表示

一个 LMI 就是具有形式

$$F(x)=F_0+x_1F_1+\mathrm{L}\ +x_mF_m<0 \quad \text{(式1)}$$

的表达式，式中的不等号表示矩阵 F(x)是【负定】的。

$F_i=F_i^T\in R^{n\times n}$ 是一组给定的是对称矩阵

x_i 是待确定的是变量，称为 LMI 系统的【决策变量】

$x=[x_1,\mathrm{K}\ ,x_m]^T$ 以有决策变量构成的向量，成为【决策向量】

其中 i=0,1,...,m

尽管不等式(式 1)是以很基本的形式给出，然而在许多控制系统问题中，LMI 系统却很少以这样的形式给出。问题的变量往往都是以**矩阵**的形式给出，例如 Lyapunov 矩阵不等式：

$$A^TX+XA+Q<0 \quad \text{(式2)}$$

其中，$A,Q\in R^{n\times n}$ 是给定的常数矩阵，并且 Q 为对称矩阵，$X\in R^{n\times n}$ 是对称的未知【矩阵变量】，下面通过一个实例来说明不等式(式 1)和(式 2)两种形式之间的相互转化。

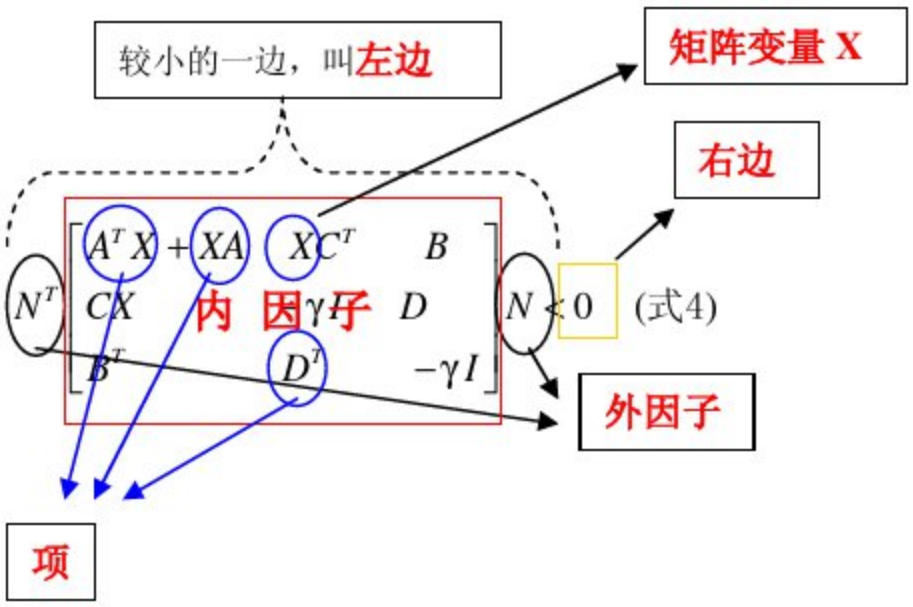
假设，常数 $A=\begin{bmatrix}-1 & 2\\ 0 & -2\end{bmatrix}$，Q 为零矩阵，变量 $X=\begin{bmatrix}x_1 & x_2\\ x_2 & x_3\end{bmatrix}$(注意 A，Q 和 X 都是对称的)，那么不等式(式 2)中的【决策变量】是【矩阵变量】X 中的独立元 x_1,x_2,x_3。将这个矩阵不等式写成一般形式(式 1)的时候，会得到下面形式：

$$x_1\begin{bmatrix}-2 & 2\\ 2 & 0\end{bmatrix}+x_2\begin{bmatrix}0 & -3\\ -3 & 4\end{bmatrix}+x_3\begin{bmatrix}0 & 0\\ 0 & -4\end{bmatrix}<0 \quad \text{(式3)}$$

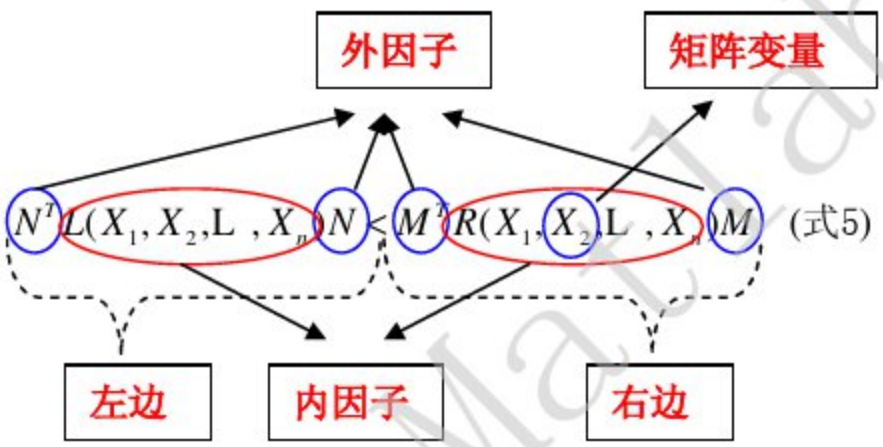
可以看出，不等式(式 3)所涉及的矩阵要比不等式(式 2)中的多。如矩阵 A 是 n 阶矩阵，那么不等式(式 3)中的系数矩阵一般要有 n(n+1)/2 个，要占用更多的存储空间。此外，不等式(式 3)已经不再具有不等式(式 2)所具有的控制中的直观含义，因此，LMI 工具箱中的函数一般采用线性矩阵不等式块结构来表示，并且其中每一个块都是矩阵变量的仿射函数。

二、描述 LMI 的相关术语

以 H_∞ 控制中的一个典型线性矩阵不等式为例来进行说明：



其中， $X = X^T \in R^{n\times n}, \gamma \in R$ 是上述 LMI 的【变量】，而 A、B、C、D 和 N 都是给定的【常数矩阵】。或者用更一般形式的表示 LMI 系统



其中 X_k 是 LMI 系统的第 k 个矩阵变量或标量变量，L 和 R 是关于 X_k 的分块矩阵，N 和 M 是给定的常数矩阵。

那么我们通过下列语句来描述这个 LMI：

(1)我们约定左边是不等式较小的一边，右边是不等式较大的一边。比如 X>0，X 是不等式的右边，而 0 是右边，**也就是说 LMI 系统一般总是使用“<”号表示不等式**，与优化工具箱相似。还有就是标量在 Matlab 中认为是 1×1 的对称矩阵，例如(式 4)中的 γ 。

(2) (式 5)中的 N 和 M 称为 LMI 的【外因子】，是具有相同维数的给定矩阵，它们可以不是方阵，在一般的问题中都是不出现的。

(3)(式 5)中的 L 和 R，或者说不等号两边的中间位置的大分块矩阵叫做【内因子】，L 和 R 是具有相同块结构的对称块矩阵。

(3)内因子是一个分块对称矩阵，它的分块结构是由其对角块的数目来描述的，并可完全由其对角线以上或以下

的那些部分来决定，**故在 Matlab 编程时，内因子只需要并且只能写出它一半**。

(4)内因子中的子块矩阵中每一个表达式都称为 LMI 的【项】，它们是关于矩阵变量 X 和矩阵标量 γ 的仿射表达式，并且可以分解为一些基本项之和，例如(式 4)中的内因子的第(1,1)块就包含两个基本项：$A^T X$ 和 XA 。

(5) 内因子中的每项都有常数项和变量项之分，所谓常数项即(式 4)中所示的 B 和 D，而变量项则至少包含一个矩阵变量，例如 XA 以及 $-\gamma I$ 等。

三、3 类标准的 LMI 控制问题

1.可行性问题

寻找一个 $X \in R^n$，使的满足 LMI 系统成立，这个问题的求解器为 feasp。

2.线性目标最小化问题

$$\min_x c^T x$$
$$s.t.\ A(x) < B(x)$$

此类问题的求解器为 mincx。

3.广义特征值最小化问题

$$\min_x \lambda$$
$$s.t.\ C(x) < D(x)$$
$$0 < B(x)$$
$$A(x) < \lambda B(x)$$

相应的求解器是 gevp。

三个解算器的功能和具体使用方法在后面会详细介绍。

LMI 工具箱介绍和使用

一、LMI 工具箱概述

LMI 工具箱支持如下功能：

1.系统描述

LMI 系统的描述既可以通过交互式 GUI 界面 lmiedit 来进行直观的矩阵描述，又可以通过命令 lmivar 和 lmiterm 来进行逐项的描述。

2.信息检

交互式函数 lmiinfo 提供有 lmiedit、lmivar 和 lmiterm 所创建的 LMI 系统的量化检索功能。也可以通过 lmiedit 使得由一系列 lmivar 和 lmiterm 命令所建立的 LMI 系统可视化。

3.问题求解

一般的 LMI 求解器都可以针对上面说到的三类标准 LMI 问题进行求解，三个求解器可以解决一般结构的 LMI 系统和矩阵变量，并返回一个关于决策变量 x 的可行或者最优解，相应的矩阵变量函数 dec2mat 得到。

4.结果验证

所得解 x 很容易由函数 evallmi 和 showlmi 进行验证，从而实现检查分析结果。LMI 系统中的所有变量都可以通过 evallmi 有决策变量 x 的值来估计，此时，每个 LMI 左边和右边的部分都变味了常数矩阵，可以由 showlmi 显示。

例1、 下面给出一个例子看看，寻找两个对称矩阵 $X \in R^{6\times6}$ 和 $S = D^T D \in R^{4\times4}$

其中

$$D = \begin{bmatrix} d_1 & & & \\ & d_1 & & \\ & & d_2 & d_3 \\ & & d_4 & d_5 \end{bmatrix}$$

使得

$$\begin{bmatrix} A^T X + XA + C^T SC & XB \\ B^T X & -S \end{bmatrix} < 0$$

$$X > 0$$

$$S > I$$

使用 Matlab 的 LMI 工具箱函数编程如下，现在看不懂不要紧：

```
%具体函数的意义和用法在后面会详细介绍
%%初始化LMI系统
setlmi([])
%%定义决策变量
X=lmivar(1,[6,1]);
S=lmivar(1,[2 0;2 1]);
%%添加LMI内因子的项
%注意只要给出右上角(或左下角)的即可
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)
%%
lmiterm([-2 1 1 X],1,1);
%%
lmiterm([-3 1 1 S],1,1);
lmiterm([3 1 1 0],1);
%%
lmisys=getlmis;
```

二、LMI 工具箱函数列表

1.确定 LMI 系统的函数

setlmis	初始化 LMI 系统
lmivar	定义矩阵变量
lmiterm	确定 LMI 系统中每一项的内容
newlmi	多 LMI 系统中添加新的 LMI
getlmis	获得 LMI 系统的内部描述
lmiedit	通过 GUI 界面确定 LMI 系统

2.对 LMI 变量的操作

dec2mat	将求解器的输出转化为矩阵变量值
mat2dec	通过给定的矩阵变量值返回决策向量

3.LMI 解算器

feasp	验证 LMI 的可行性
mincx	LMI 限制下线性目标的极小值
defcx	在 mincx 命令中第一 C^Tx 目标
gevp	LMI 限制下的广义特征值最小化

4.LMI 结果验证与修改

evallmi	由决策变量的给定值来验证所有的变量项
showlmi	返回一个已经评估的 LMI 的左右边
dellmi	从系统中删除一个 LMI
delmvar	从问题中移除一个矩阵变量
setmvar	将一个矩阵变量赋予指定值

5.LMI 系统信息的提取

decinfo	以决策变量的形式表示每个输入的矩阵变量
decnbr	得到决策变量的个数
lmiinfo	查询现存 LMI 系统的信息
lminbr	得到问题中 LMI 的个数
mntnbr	得到问题中矩阵变量的个数

三、LMI 工具箱函数详解

1.确定 LMI 系统的函数

(1)setlmis([])或者 setlmis(lmi0)

在通过 lmivar 以及 lmiterm 描述一个 LMI 系统之前，利用 setlmis 初始化其内部表示。为一个已经存在的 LMI 系统中添加新描述，使用后者，其中 lmi0 表示已存在的 LMI 系统，之后的 lmivar 和 lmiterm 被用来在 lmi0 中添加新的变量和项。

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

为 LMI 问题定义矩阵变量，其中 type 和 struct 是描述该矩阵变量的必须参数。type 确定变量 X 的类型，struct 描述变量 X 的内容。

type=1： 此时 X 是具有块对角化对称矩阵。X 对角线上的每一个矩阵 D_i 必须是方阵，注意标量也 1×1 的方阵。此时的矩阵变量 X 大体结果如下：

$$X=\begin{bmatrix} D_1 & & & \\ & D_2 & & \\ & & \mathrm{O} & \\ & & & D_n \end{bmatrix} \quad (D_i\text{必须是方阵})$$

如果 X 具有 n 个对角块，那么 struct 是一个 n×2 的矩阵：
m=struct(i,1)表示第 i 个方阵(即 D_i)的大小，比如 D_i 是 5×5 的方阵，那么 struct(i,1)=5，
n=struct(i,2) 表示 D_i 的内容，n=-1 表示 D_i 是零矩阵，n=0 表示标量，n=1 表示满矩阵。

type=2： 表示 X 是一个 m×n 的长方形矩阵，此时 struct=[m,n]，很简单。

type=3： 表示其他结构，一般用于复杂的 LMI 系统，正常情况使用的比较少，此时若 X(i,j)=0，struct(i,j)=0，若 $X(i,j)=x_k$ 则 struct(i,j)=k，若 $X(i,j)=-x_k$ 则 struct(i,j)=-k，其中 x_k 表示第 k 个决策变量。

type	struct=[m n]			说明
1	m		第 i 个矩阵的大小	X 必须是对角化的块对称矩阵，每个块也必须是方阵
	n	-1	零矩阵	
		0	标量	
		1	满矩阵	
2	[m n]			X 为 m×n 的长方形矩阵，不能包含子块矩阵
3				

例 2、 举个例子说明下吧，考虑有 3 个矩阵变量的 LMI 系统，其中 X1 是 3×3 的对称矩阵，X2 是 2×4 的矩阵，X3 具有下面的形式

$$\begin{bmatrix} \Delta & 0 & 0 \\ 0 & \delta_1 & 0 \\ 0 & 0 & \delta_2 I_2 \end{bmatrix}$$

其中 Δ 是一个任意的 5×5 的对称矩阵，δ1 和 δ2 是标量，I2 是 2×2 的单位矩阵，利用 lmivar 定义上述三个矩阵变量如下：

```
%%定义X1
X1=lmivar(1,[3,1]);
%%定义X2
X2=lmivar(2,[2,4]);
%%定义X3
%注意此时的X3(3,3)是算标量的，原因有二，一是它不满也不全为零，二是只需要δ2一个数据即可确定
X3=lmivar(1,[5 1;1,0,2,0]);
```

例 3、 下面考虑更复杂的矩阵变量，使用 lmivar 定义矩阵变量 X

$X = \begin{bmatrix} X_1 & \\ & X_2 \end{bmatrix}$ 其中X_1是2×3，X_2是3×2，

当然我们可以通过两个 type=2 和一个 type=3 来构成 X，即

```
%由于Di不符合是方阵，故不能使用type=1来构造X
[X1,n,sX1]=lmivar(2,[2 3]);
[X2,n,sX2=lmivar(2,[3,2]);
%有X1和X1组合成X
X=lmivar(3,[sX1,zeros(2);zeros(3),sX2]);
```

(3)lmiterm(termID,A,B,flag)

确定 LMI 中每一项的内容，包括内外因子、常数项以及变量项。再次强调，在描述一个具有分块对称的 LMI 时，只需要确定右上角或者左下角即可。

termID：四元向量，确定该项的位置以及包含的矩阵变量。

termID(1)表述所描述的项属于哪个 LMI，它可取值为 p 或者-p，p 代表该项位于第 p 个 LMI 的左边，而-p 则代表该项位于第 p 个 LMI 的右边。我们再起强调，左边是指 LMI 较小的那一边。
termID(2:3)表示所描述的项所在 LMI 中块的位置，如果该项位于内因子的第(i,j)块，那么 termID(2:3)=[i,j]；如果

该项在外因子中，那么 termID=[0 0]。
termID(4)表示所描述的项是常数项还是变量项。0 代表常数项，k 代表该项中包含第 k 个矩阵变量 X_k，-k 代表该项中包含第 k 个矩阵变量 X_k 的转置 X_k^T。

termID	取值	所描述的项
termID(1)	p	位于第 p 个 LMI 的左边
	-p	位于第 p 个 LMI 的右边
termID(2:3)	[0 0]	外因子
	[i j]	在内因子中的位置(i,j)
termID(4)	0	常数
	Xk	项中包含第 k 个矩阵变量 X_k
	-Xk	项中包含第 k 个矩阵变量的转置 X_k^T

参数 A、B 是描述该项包含的数据信息，具体可以看下面的表格：

所描述项	A	B
外因子 N	N	省略
常数项 C 或者 Q*Q'	C/ Q*Q'	省略
变量项 AXB 或者 AXB^T	A/A	B/ B'

flag：是个可选参数，并且只能是's'，它只对 $AXB+B^TX^TA^T$ 这种项有效，即可以一次性同时添加这两项，命令如下：

```
lmiterm([1 1 1 X],A,B,'s');
```

当然可以分开写了

```
lmiterm([1 1 1 X],A,B);
lmiterm([1 1 1 -X],B',A');
```

例 4、 看个例子吧，考虑 LMI，给出这个 LMI 的描述

$$\begin{bmatrix} 2AX_2A^T - x_3E + DD^T & B^TX_1 \\ X_1^TB & -I \end{bmatrix} < M^T \begin{bmatrix} CX_1C^T + CX_1^TC^T & 0 \\ 0 & -fX_2 \end{bmatrix} M$$

其中，X_1 和 X_2 分别是 type=2 和 type=1 的矩阵变量，x_3 是一个标量(再次说明，Matlab 将标量视为 1×1 的对称矩阵)

```
%确定LMI左边的项(即不等号较小的那边，再次强调)，还有记住只要描述一半即可
lmiterm([1 1 1 X2],2*A,A');
lmiterm([1 1 1 x3],-1,E);
lmiterm([1 1 1,0],D*D');
lmiterm([1 2 1 -X],1,B);
lmiterm([1 2 2 0],-I);
%同理描述右边的项，注意零矩阵可以不描述，当然你描述了也不会错
```

```
lmiterm([-1 0 0 0],M);%描述LMI右边的外因子
lmiterm([-1 1 1 X1],C,C','s');
lmiterm([-1 2 2 X2],-f,1)
```

(4)tag=newlmi

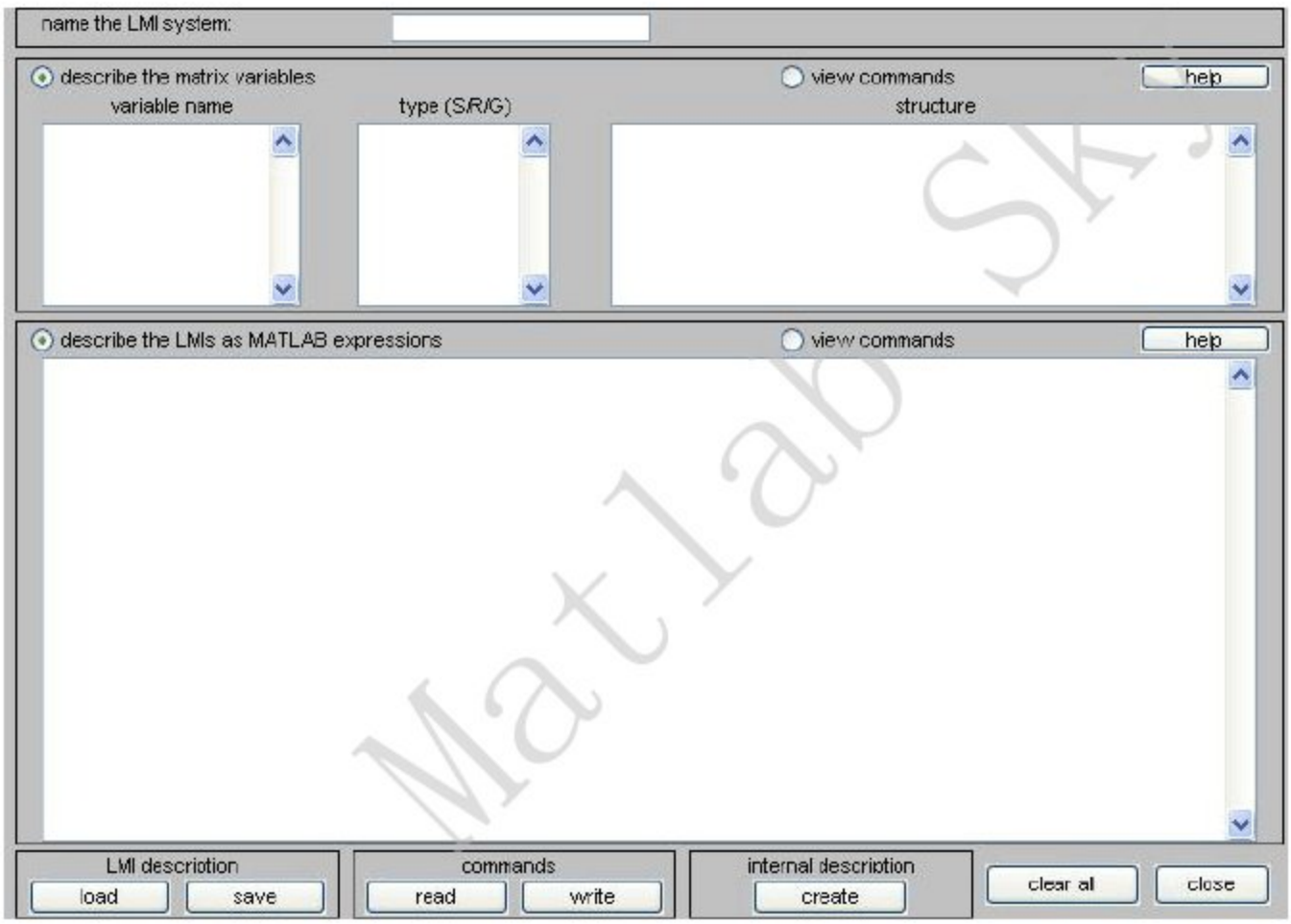
在当前多描述的 LMI 系统中添加一个新的 LMI，并返回它的句柄 tag。

(5)lmisys=getlmis

在利用 lmivar 和 lmiterm 描述给定 LMI 后，通过 getlmis 获取当前 LMI 系统的内部描述，注意这个命令是必须的。

(6)lmiedit

打开 LMI 的 GUI 界面，它可以直观的编辑 LMI，在后面的实例中我们会详细介绍如何使用该界面的。



其实这个界面对与大部分人来说没有太大的意义，由于它只是免去你书写和记忆 lmiterm 函数中的参数，但是 lmivar 参数，比如 type 和 struct 还是要理解的，当然对初学这可能可以方便些。

2.对 LMI 变量的操作

(1)valX=dec2mat(lmisys,decvars,X)
给定决策变量的值返回相应矩阵变量的值。

(2)decvec=mat2dec(lmisys,X1,X2...)
根据矩阵变量的特殊值来返回决策变量组成的决策向量。给定一个 LMI 系统 lmisys 和矩阵变量 X1,X2...，得到决策向量 decvec。这个函数在初始化 LMI 求解器 mincx 和 gevp 的时候很重要，通过给定一个关于矩阵变量 X1,X2... 的初始值，mat2vec 将会组成相应的关于决策变量 xinit 的向量。

例 5、考虑一个 LMI 系统具有两个矩阵变量 X 和 Y，其中 X 具有两个分块对角块，一个是 2×2 的满矩阵，另一个是 2×2 的标量矩阵，Y 是一个 2×3 的长方形矩阵，例如当

$$X_0=\begin{bmatrix}1 & 3 & 0 & 0\\ 3 & -1 & 0 & 0\\ 0 & 0 & 5 & 0\\ 0 & 0 & 0 & 5\end{bmatrix} \qquad Y_0=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix}$$

的时候，求相应的决策向量。

```
setlmis([])
X=lmivar(1,[2,1;2,0]);
Y=lmivar(2,[2 3]);
lmisys=getlmis;
X0=[1 3 0 0;3 -1 0 0;0 0 5 0;0 0 0 5]
Y0=[1 2 3;4 5 6]
dec=mat2dec(lmisys,X0,Y0);
dec'

X0 =

     1     3     0     0
     3    -1     0     0
     0     0     5     0
     0     0     0     5

Y0 =

     1     2     3
     4     5     6

ans =
```

```
    1     3    -1     5     1     2     3     4     5     6
```

注意到 dec 向量的长度是 10，这是由于 Y 就有 6 个自由变量，而 X 只有 4 独立变量，我们可以利用 decinfo 获取关于 X 和 Y 中决策变量分布的详细信息。

3.LMI 求解器命令

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

计算以 lmisys 描述的 LMI 系统的一个可行解，向量 xfeas 是一个 guanyu 决策变量的特殊值，它满足任意一个 LMI。即：

给定 LMI 系统

$$N^T L(x)N \le M^T R(x)M$$

xfeas 通过求解辅助凸优化问题

$$\min t$$

$$s.t.\ N^T L(x)N - M^T R(x)M \le tI \qquad (式5)$$

计算得到

其中，(式 5)的全局最优解由标量值 tmin 表示，如果 tmin<0，那么系统是严格可行的。如果问题是可行但非严格可行的话，tmin 是一个非常小的整数。

options：是一个长度为 5 的向量，用来描述优化算法中的某些控制参数

options	说明	
options(1)	不使用	
options(2)=n	设置优化算法的过程中允许的最大迭代次数 n，默认 n=100	
options(3)=R	设定可行域半径 R，表示决策变量被限制在球体 $\sum_{i=1}^{N} x_i^2 < R^2$ 内，默认 $R=10^9$	
options(4)=J	加快优化迭代过程，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%，迭代即终止，默认 J=10	
options(5)	0	显示迭代过程，默认 0
	1	不显示迭代过程

target：为 tmin 设定目标值，使得只要满足 tmin<target，则优化迭代过程终止，默认 target=0

例 6、 考虑下面的问题，找到满足 P>I 的对称矩阵 P，使得

$$A_1^T P + P^T A_1 < 0$$
$$A_2^T P + P^T A_2 < 0$$
$$A_3^T P + P^T A_3 < 0$$

其中

$$A_1 = \begin{bmatrix} -1 & 2 \\ 1 & -3 \end{bmatrix} \quad A_2 = \begin{bmatrix} -0.8 & 1.5 \\ 1.3 & -2.7 \end{bmatrix} \quad A_3 = \begin{bmatrix} -1.4 & 0.9 \\ 0.7 & -2.0 \end{bmatrix}$$

```
%这个问题是二次稳定性问题中提取出来的，首先确定已知的LMI，在调用函数feasp
%常数
A1=[-1 2;1 -3];
A2=[-0.8 1.5;1.3 -2.7];
A3=[-1.4 0.9;0.7 -2.0];
%初始化LMI
setlmis([]);
%定义变量
P=lmivar(1,[2,1]);
%添加项
lmiterm([1 1 1 P],1,A1,'s');
lmiterm([2 1 1 P],1,A2,'s');
lmiterm([3 1 1 P],1,A3,'s');
lmiterm([-4,1 1 P],1,1);%记住还有一个I<P
lmiterm([4 1 1 0],1)
%获取LMI系统描述
lmisys=getlmis;
[tmin,xfeas]=feasp(lmisys)%options参数可以自己设置，也可以默认
```

运行结果如下：

```
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                    0.972718
     2                    0.870460
     3                   -3.136305

 Result:  best value of t:    -3.136305
        f-radius saturation:  0.000% of R = 1.00e+009
```

```
tmin =

   -3.1363

xfeas =

  270.8553
  126.3999
  155.1336
```

从运行结果看 tmin<0，表明 LMI 系统是可行的。下面使用 dec2mat 来得到可行矩阵变量的值：

```
pmat=dec2mat(lmisys,xfeas,P)
```

运行结果如下：

```
pmat =

  270.8553  126.3999
  126.3999  155.1336
```

(2)[copt,xopt]=mincx(lmisys,c,options,xinit,target)
求解具有 LMI 约束的线性目标函数的最小化问题，返回目标最小值 copt 和对应的决策变量的值，即解决下面的凸包问题：

$$\min c^T x \qquad (x\text{为决策向量})$$
$$s.t.\ N^T L(x) N \le M^T R(x) M$$

lmisys：需要求解的 LMI 系统。

c：就是目标函数 min $c^T x$ 中的那个 c，通常情况题目不会直接给出 c 而是需要我们间接的求解。控制系统中大部分都是以变量矩阵 X 的形式给出目标函数，而不是刚才说到的决策向量的形式。此时我们一般需要通过 defcx 函数来求解。

xinit：决策变量 x 的初始猜测值，它可以加快问题求解过程，我们可以使用 mat2dec 函数从矩阵变量的给定值中导出 xinit，但是当输入的 xinit 不是一个可行解时，将被忽略
target：目标函数的设置目标，只要满足 $c^T x \leqslant$ target，求解过程就终止。

options：5 维向量，控制优化算法中的参数

options	说明
options(1)=tol	最优解 copt 的精度 tol，默认 0.01
options(2)=itermax	允许的最大迭代次数 itermax，默认 100
options(3)=R	设定可行域半径 R，默认 10e9
options(4)=J	最后 J 次迭代结果变化幅度在精度范围，将终止迭代过程，默认 5
options(5)	控制是否显示迭代过程，0 显示，1 不显示，默认 0

例 7、考虑关于对称矩阵 X 的问题：

$$\begin{aligned}\min Trace(X) &= trace(\begin{bmatrix} x_1 & x_2 & x_4 \\ x_2 & x_3 & x_5 \\ x_4 & x_5 & x_6 \end{bmatrix}) \\ &= x_1 + x_3 + x_6 \\ &= [1\ 0\ 1\ 0\ 0\ 1]^T [x_1\ \ x_2\ \ x_3\ \ x_3\ \ x_4\ \ x_5\ \ x_6] = c^T x\end{aligned}$$

$$s.t. \begin{bmatrix} A^T X + XA + Q & XB \\ B^T X & -I \end{bmatrix} < 0$$

其中

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

由于 trace(X)是 X 的一个线性函数，因此以上的优化问题就是一个具有 LMI 约束的线性目标函数的最小化问题，从而可以通过求解器 mincx 来求解

```
%给出常量
A=[-1 -2 1;3 2 1;1 -2 -1];
B=[1 0 1]';
Q=[1 -1 0;-1 -3 -12;0 -12 -36];
%初始化LMI
setlmis([]);
%定义变量
X=lmivar(1,[3,1])
%添加项
lmiterm([1 1 1 X],1,A,'s');
lmiterm([1 1 1 0],Q);
lmiterm([1 1 2 X],1,B);
lmiterm([1 2 2 0],-1);
```

```
%获取LMI系统描述
lmisys=getlmis;
%获取目标函数的决策向量的系数，这里没有使用defcx，是由于比较简单
c=mat2dec(lmisys,eye(3));%当然你可以直接看出c=[1 0 1 0 0 1]'，也是可以的
%定义options参数
options=[1e-5 0 0 0 1];
%调用mincx函数
[copt,xopt]=mincx(lmisys,c,options)
```

运行结果如下：

```
copt =

  -18.7167


xopt =

   -6.3542
   -5.8895
   -6.2855
    2.2046
    2.2201
   -6.0771
```

(3)[V1,V2...]=defcx(lmisys,n,X1,X2...)

将矩阵变量形式的目标函数转成决策变量形式，从而得到c，也就是说LMI求解器mincx的目标函数必须转换成c^Tx的形式才可以使用，其中x是由决策变量组成的决策向量。然而在多数控制问题中，目标函数一般以矩阵变量X的形式给出，而不是决策向量x。比如trace(X)或者
A^TXA等。

lmisys：需要求解的LMI系统。
n：LMI系统中独立变量的个数，可以使用decnlmi函数获取。
Xk：第k个矩阵变量。
Vk：返回的第k个决策变量系数。

例8、 考虑线性目标函数：

$$\min Trace(X)+x_0^TPx_0$$

其中，矩阵X是3阶标量矩阵αI_3，P是一个2阶对称矩阵。x_0是一个给定向量，确定向量c

```
x0=[1;1];
setlmis([]);
```

```
X=lmivar(1,[3 0]);
P=lmivar(1,[2,1]);
.............................
lmisys=getlmis;
```

那么满足c'*x=trace(X)+x0'*P*x0的向量c可以通过下面的代码实现：

```
%这个就不是那么直接可以看出来了，必须使用defcx函数了
n=decnbr(lmisys);%获取LMI中决策变量的个数
c=zeros(n,1);
for jj=1:n
[xj,pj]=defcx(lmisys,jj,X,P);
c(jj)=trace(xj)+x0'*pj*x0;
end
```

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

求解LMI限制下的广义特征矩阵最小化问题：

$$\min \lambda$$

$$s.t.\ C(x) < D(x) \qquad \text{(普通约束)}$$

$$0 < B(x) \qquad \text{(普通约束)}$$

$$A(x) < \lambda B(x) \qquad \text{(线性比例约束)}$$

只要上面的三个约束是可行的，gevp就返回全局最小值lopt和相应的决策变量的值xopt。

由于大部分参数前面的相似，我们这里只是说明下不同的参数：

nlfc：线性比例约束的个数，即包括 λ 的式子的个数。
linit： λ 的初值猜测值 λ 0。
xinit：每个决策变量的初始猜测值x0，注意必须是以决策向量的形式给出，长度等于decnbr(lmisys)。
options：与mincx的完全一样，这里不具体介绍了。

例9、 对于下面三个系统

$$\min \lambda$$

$$s.t.\ I < P$$

$$A_1^T P + P A_1 < \lambda P$$

$$A_2^T P + P A_2 < \lambda P$$

$$A_3^T P + P A_3 < \lambda P$$

其中

$$A_1 = \begin{bmatrix} -1 & 2 \\ 1 & -3 \end{bmatrix} \quad A_2 = \begin{bmatrix} -0.8 & 1.5 \\ 1.3 & -2.7 \end{bmatrix} \quad A_3 = \begin{bmatrix} -1.4 & 0.9 \\ 0.7 & -2.0 \end{bmatrix}$$

```
A1=[-1 2;1 -3];
A2=[-0.8 1.5;1.3 -2.7];
A3=[-1.4 0.9;0.7 -2.0];
setlmis([]);
P=lmivar(1,[2,1]);
lmiterm([1 1 1 0],1);%第一个式子的左边
lmiterm([-1 1 1 P],1,1);%第一个式子的右边
lmiterm([2 1 1 P],1,A1,'s');
lmiterm([-2 1 1 P],1,1);
lmiterm([3 1 1 P],1,A2,'s');
lmiterm([-3 1 1 P],1,1);
lmiterm([4 1 1 P],1,A3,'s');
lmiterm([-4 1 1 P],1,1);
lmisys=getlmis;
[lambda,xopt]=gevp(lmisys,3,[1e-5 0 0 0 1])%注意到三个约束都是线性比例约束
```

运行结果为:

```
lambda =

   -0.1221

xopt =

    5.6204
   -8.4297
   18.8054
```

4.结果验证和修改

(1)evalsys=evallmi(lmisys,decvars)

对一个给定的决策向量的值decvars，验证LMI求解器的输出，返回给evalsys，之后每个LMI左右两边的矩阵值通过showlmi函数返回

例10、 考虑下面的系统

$A^T XA - X + I < 0$

其中

$A = \begin{bmatrix} 0.5 & -0.2 \\ 0.1 & -0.7 \end{bmatrix}$，找出矩阵$X > 0$，满足该系统要求

```
%显然该问题是可行解的问题，当然使用feasp求解
A=[0.5 -0.2;0.1 -0.7];
setlmis([]);
X=lmivar(1,[2,1]);
lmiterm([1 1 1 X],A',A);
lmiterm([1 1 1 X],-1,1);
lmiterm([1 1 1 0],1);
lmiterm([-2 1 1 X],1,1);%不要忘记还有一个X>0
lmisys=getlmis;
[tmin,xfeas]=feasp(lmisys)
```

运行结果如下：

```
tmin =

   -4.7117

xfeas =

  110.2905
  -11.5186
  119.4186
```

由于tmin<0，则LMI是可行的，与可行的决策向量xfeas对应的解X有下面的命令得出：

```
X=dec2mat(lmisys,xfeas,X);
```

为了验证xfeas确实可行的，通过下面命令来验证所有的LMI约束：

```
evalsys=evallmi(lmisys,xfeas);
[lhs1,rhs1]=showlmi(evalsys,1)
[lhs2,rhs2]=showlmi(evalsys,2)
eig(lhs1-rhs1)
eig(lhs2-rhs2)
```

运行结果如下：

```
lhs1 =

  -81.6756   -3.6079
   -3.6079  -58.7171

rhs1 =

     0     0
     0     0

lhs2 =

     0     0
     0     0

rhs2 =

  110.2905  -11.5186
  -11.5186  119.4186

ans =

  -82.2292
  -58.1635

ans =

 -127.2444
 -102.4647
```

(2)[lhs,rhs]=showlim(evalsys,n)

对于决策变量的给定值，返回每个LMI不等号的左右两边的值。

evalsys：由evallmi计算返回的参数。

n：第n个LMI。

(3)newsys=setvar(lmisys,X,Xval)

将Xval赋值给矩阵变量X，此时所有包含X的项都会变为常数项，当然那个矩阵变量X也就从地球上消失了，返回的新LMI系统是有剩下的矩阵变量和常数项构成。

setvar的作用就是可是冻结默一个矩阵变量，或者根据剩余变量进行优化，可以避免LMI约束的部分或全部重定义。

例11、 看下面的系统，$x' = Ax + Bu$，寻找一个镇定反馈控制率u=Kx，其中K是一个位置矩阵。

由Lyapunov定理，上面的问题等效为，寻找矩阵P>0和K，使得

$$(A+BK)P + P(A+BK)^T + I < 0$$

通过变量替换Y=KP，上面的系统退化为如下的LMI：

$$AP + PA^T + BY + Y^T B^T + I < 0$$

```
n=size(A,1);%决策变量的个数
ncon=size(B,1);%输入个数
setlmi([]);
P=lmivar(1,[n,1]);%满块对称矩阵
Y=lmivar(2,[ncon,n]);%长方形矩阵
lmiterm([1 1 1 P],A,1,'s');
lmiterm([1 1 1 P],B,1,'s');
lmiterm([1 1 1 0],1);
lmiterm([-2 1 1 P],1,1);
lmisys=getlmis;
```

为了确定这个问题对于特殊的Lyapunov矩阵P=I是否存在解K，利用下面的命令将P赋值为I：

```
newsys=setvar(lmisys,P,1);
```

那么所得新LMI系统只有一个矩阵变量Y=K，它的可行行即可利用下面的命令完成：

```
[tmin,xfeas]=feasp(newsys);
Y=dec2mat(newsys,xfeas,Y);
```

(4)newsys=dellmi(lmisys,n)

删除lmisys系统中的第n个LMI，返回新的LMI系统给newsys，当然在被删除的LMI中出现的矩阵变量也会被相应的删除。

(5)newsys=delvar(lmisys,X)

将矩阵变量从当前LMI系统中删除，相当于setvar(lmisys,X,0)。

5.LMI 系统信息提取

(1)nlmi=lminbr(lmisys)
返回由lmisys描述的LMI系统中LMI的个数。

(2)nmat=matnbr(lmisys)
返回由lmisys描述的LMI系统中矩阵变量的个数。

(3)ndec=decnbr(lmisys)
返回由lmisys描述的LMI系统中决策变量的个数。

(4)lmiinfo(lmisys)
返回有lmisys描述的LMI系统中矩阵变量的类型和结构，内因子中对角块的个数以及每一块中各项的内容等。

(5)decX=decinfo(lmisys,X)
指明X中每一个块，分别是哪个决策变量，返回的decx和X的维数相同
X(i,j)=0 ==> decX(i,j)=0
X(i,j)=xk ==> decX(i,j)=k
X(i,j)=-xk ==> decX(i,j)=-k
当第二参数省略的时候，会出现交互是界面列所有的决策变量，要求你选择。

LMI 系统实例分析

命令行形式解答

例 12、考虑一类由带时滞的 Lur'e 系统组成的具有 N 个节点的复杂网络(网络的每个节点都是一个具有相同参数的同类系统)，基本模型如下所示

$$x_i'(t)=Ax_i(t)+Ax_i(t-d)+Bf(Cx_i(t))+c\sum_{j=1}^{N}h_{ij}\Gamma x_j(t),i=1,2,\mathrm{L}\ ,N$$

其中

$$A=\begin{pmatrix}-2 & 0\\ -1 & -2\end{pmatrix},A_d=\begin{pmatrix}-0.2 & -0.5\\ 0.5 & -0.2\end{pmatrix},B=\begin{pmatrix}-0.2\\ -0.3\end{pmatrix},C=\begin{pmatrix}0.6 & 0.8\end{pmatrix}$$

$$H=\begin{pmatrix}-4 & 1 & 1 & 1 & 1\\ 1 & -1 & 0 & 0 & 0\\ 1 & 0 & -1 & 0 & 0\\ 1 & 0 & 0 & 0 & -1\end{pmatrix},\Gamma=\mathrm{I}(\text{单位阵}),c=1,d=2$$

对于这类网络，可以利用Lyapunov函数的方法，采用解耦的方法将一个高维的LMI化为几个低维的LMI，从而得到整个网络达到同步的充分条件

$$\begin{bmatrix}N & PA_d-Y & -PB-C^TV^T & d(A+c\lambda\Gamma)Z\\ & -Q & 0 & dA_d^TZ\\ & & -2I & -dB^TZ\\ & & & dZ\end{bmatrix}<0 \qquad (式1)$$

$$\begin{bmatrix}X_i & Y\\ Y^T & Z\end{bmatrix}\geq 0 \qquad (式2)$$

其中

$N=(A+c\lambda\Gamma)^TP+P(A+c\lambda\Gamma)+Y+Y^T+dX+Q$

λ是H的特征根，故$\lambda=\begin{pmatrix}0 & 0 & 0 & -1 & -4\end{pmatrix}$

V这里我们取1

P、Q、X、Y和Z是LMI的矩阵变量

```
c=1;
v=1;
d=2;
%lambda共有5个值，我们需要一个一个的验证，先验证lambda=0
lambda=0;
C=[0.6 0.8];
A=[-2 0;-1 -2];
Ad=[-0.2 -0.5;0.5 -0.2];
B=[-0.2;-0.3];
%A1为N的表达式中的A+c*lambda*Gamma那部分
Gamma= eye(size(A))
A1=A+c*lambda*Gamma;

setlmis([]);
%定义矩阵变量PQXYZ
P=lmivar(1,[2,1]);
Q=lmivar(1,[2,1]);
X=lmivar(1,[2,1]);
Y=lmivar(1,[2,1]);
Z=lmivar(1,[2,1]);
%添加(式1)LMI的项
lmiterm([1 1 1 P],1,A1,'s');
lmiterm([1 1 1 Y],1,1,'s');
lmiterm([1 1 1 X],d,1);
lmiterm([1 1 1 Q],1,1);
lmiterm([1 1 2 P],1,Ad);
lmiterm([1 1 2 Y],-1,1);
lmiterm([1 1 3 P],-1,B);
lmiterm([1 1 3 0],-v*C');
lmiterm([1 1 4 Z],d*A1',1);
lmiterm([1 2 2 Q],-1,1);
lmiterm([1 2 4 Z],d*Ad',1);
lmiterm([1 3 3 0],-2);
lmiterm([1 3 4 Z],-d*B',1);
lmiterm([1 4 4 Z],-d,1);
%添加(式2)LMI的项，注意不等式的右边为负，又由于是第二个LMI，故-2
lmiterm([-2 1 1 P],1,1);
%还注意到X、Y和Z都必须大于0，故-3
lmiterm([-3 1 1 X],1,1);
lmiterm([-3 1 2 Y],1,1);
lmiterm([-3 2 2 Z],1,1);

lmisys=getlmis;
[tmin,xfeas]=feasp(lmisys);
```

```
pp=dec2mat(lmisys,xfeas,P);
qq=dec2mat(lmisys,xfeas,Q);
xx=dec2mat(lmisys,xfeas,X);
yy=dec2mat(lmisys,xfeas,Y);
zz=dec2mat(lmisys,xfeas,Z);
```

GUI 形式解答

例13、 我们以例7的表达式进行说明