中国地质大学 自动化学院
School of Automation, China University of Geosciences

上机实验

实例讲解

Matlab在系统数学模型中的应用

实例讲解

一、线性系统的数学模型

1 tf()函数

功能：建立系统的传递函数模型TF。

调用格式：sys=tf(num,den)

设单输入单输出连续系统的传递函数为

$$W(s)=\frac{b_m s^m + b_{m-1}s^{m-1}+\cdots+b_1 s+b_0}{s^n + a_{n-1}s^{n-1}+\cdots+a_1 s+a_0}$$

在Matlab中，可用传递函数分子、分母多项式按 s 的降幂系数排列的行向量，即：

$$num=[b_m, b_{m-1}, \cdots, b_1, b_0];$$

$$den=[1, a_{n-1}, a_{n-2}, \cdots, a_1, a_0];$$

实例讲解

对于单输入单输出离散系统的脉冲传递函数为：

$$W(z)=\frac{b_m z^m + b_{m-1} z^{m-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0}$$

在Matlab中，调用tf()函数建立系统的传递函数模型TF：

$$num = \left[b_m, b_{m-1}, \cdots, b_1, b_0\right];$$

$$den = \left[1, a_{n-1}, a_{n-2}, \cdots, a_1, a_0\right];$$

$$sys = \mathrm{tf}\left(num, den, T\right)$$

式中， T 为系统采样周期。

实例讲解

【例1】 已知系统的传递函数为：

$$W(s)=\frac{s^2+3s+1}{s^3+2s^2+4s+6}$$

试用Matlab描述其系统模型。

实例讲解

【例1】

解： Matlab仿真代码如下：

```
num=[1 3 1];      % 分子多项式
den=[1 2 4 6];    % 分母多项式
G=tf(num,den)     %建立传递函数模型
```

运行结果如下：

Transfer function:

```
          s^2 + 3 s + 1
     ---------------------
     s^3 + 2 s^2 + 4 s + 6
```

实例讲解

2 zpk()函数

功能：建立系统的零级点形式的传递函数模型ZPK

调用格式：

$$sys = \mathrm{zpk}(\boldsymbol{z}, \boldsymbol{p}, k)$$

设系统的传递函数表示为零极点的形式：

$$W(s) = k\frac{(s-z_1)(s-z_2)\cdots(s-z_m)}{(s-p_1)(s-p_2)\cdots(s-p_n)}$$

则：

$$\boldsymbol{z} = [z_1, z_2, \cdots, z_m];$$

$$\boldsymbol{p} = [p_1, p_2, \cdots, p_n];$$

$$k = k$$

实例讲解

【例2】 已知系统的零极点传递函数形式为：

$$W(s)=\frac{7(s-3)}{(s-1)(s+2)(s+4)}$$

试用Matlab描述其系统模型。

实例讲解

【例2】

解：Matlab仿真代码如下：

```
z=[3];    %W(s)零点
p=[1,-2,-4];%W(s)极点
k=7;
sys=zpk(z,p,k)
```

运行结果如下：

sys =

$$\frac{7\ (s-3)}{(s-1)\ (s+2)\ (s+4)}$$

实例讲解

3. ss()函数

功能：建立系统的状态空间模型SS

调用格式：$sys = \mathrm{ss}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D})$ $sys = \mathrm{ss}(\boldsymbol{G}, \boldsymbol{H}, \boldsymbol{C}, \boldsymbol{D}, T)$

对于线性定常连续系统：$\begin{cases} \dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t) \end{cases}$

在Matlab中，可调用ss()函数建立连续系统的状态空间模型，其中

$$\boldsymbol{A} = [a_{11}, a_{12}, \cdots, a_{1n}; a_{21}, a_{22}, \cdots, a_{2n}; \cdots; a_{n1}, a_{n2}, \cdots, a_{nn}];$$

$$\boldsymbol{B} = [b_{11}, b_{12}, \cdots, b_{1n}; b_{21}, b_{22}, \cdots, b_{2n}; \cdots; b_{n1}, b_{n2}, \cdots, b_{nr}];$$

$$\boldsymbol{C} = [c_{11}, c_{12}, \cdots, c_{1n}; c_{21}, c_{22}, \cdots, c_{2n}; \cdots; c_{m1}, c_{m2}, \cdots, c_{mn}];$$

$$\boldsymbol{D} = [d_{11}, d_{12}, \cdots, d_{1r}; d_{21}, d_{22}, \cdots, d_{2r}; \cdots; d_{m1}, d_{m2}, \cdots, d_{mr}];$$

实例讲解

对于线性定常离散系统

$$\begin{cases} \boldsymbol{x}(k+1) = \boldsymbol{G}\boldsymbol{x}(k) + \boldsymbol{H}\boldsymbol{u}(k) \\ \boldsymbol{y}(k) = \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{u}(k) \end{cases}$$

在建立系数矩阵 G、H 、C 、D 后，同样可以调用ss()函数建立系统的状态空间模型：

$$sys = \mathrm{ss}(\boldsymbol{G}, \boldsymbol{H}, \boldsymbol{C}, \boldsymbol{D}, T)$$

式中，T 为系统采样周期。

实例讲解

【例3】 已知系统的状态空间表达式为：

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 2 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u \end{cases}$$

试用Matlab描述其系统模型。

实例讲解

【例3】

解： Matlab仿真代码如下：

```
A=[0 1;-2 -3];
B=[1 0;1 1];
C=[1 0;1 1;0 2];
D=[0 0;1 0;0 1];
sys=ss(A,B,C,D)
```

运行结果如下：

sys =

A =

	x1	x2
x1	0	1
x2	-2	-3

B =

	u1	u2
x1	1	0
x2	1	1

C =

	x1	x2
y1	1	0
y2	1	1
y3	0	2

D =

	u1	u2
y1	0	0
y2	1	0
y3	0	1

实例讲解

4. series()函数，parallel()函数，feedback()函数

功能：分别实现两个子系统W_1和W_2的串联、并联和反馈连接

调用格式：$sys = \mathrm{series}(sys\_1, sys\_2)$

$$sys = \mathrm{parallel}(sys\_1, sys\_2)$$

$$sys = \mathrm{feedback}(sys\_1, sys\_2, sign)$$

式中，sys，sys_1 和sys_2 可以是状态空间模型，也可是是传递函数模型；$sign$ 表示反馈极性，正反馈取1，负反馈取-1或默认。

实例讲解

当sys、sys_1 和 sys_2 是状态空间模型时，调用格式为：

$$[A,B,C,D]=\text{series}(A_1,B_1,C_1,D_1,A_2,B_2,C_2,D_2)$$
$$[A,B,C,D]=\text{parallel}(A_1,B_1,C_1,D_1,A_2,B_2,C_2,D_2)$$
$$[A,B,C,D]=\text{feedback}(A_1,B_1,C_1,D_1,A_2,B_2,C_2,D_2)$$

当sys、sys_1 和 sys_2 是状态空间模型时，调用格式为：

$$[num,den]=\text{series}(num\_1,den\_1,num\_2,den\_2)$$
$$[num,den]=\text{parallel}(num\_1,den\_1,num\_2,den\_2)$$
$$[num,den]=\text{feedback}(num\_1,den\_1,num\_2,den\_2)$$

实例讲解

【例4】 已知两个系统的传递函数分别为：

$$W_1(s)=\frac{3s+1}{s^2+3s+2},W_2(s)=\frac{s+4}{s+2}$$

试用Matlab求出它们并联组合系统的传递函数。

实例讲解

【例4】

解：Matlab仿真代码如下：

```
num_1=[3 1];
den_1=[1 3 2];
num_2=[1 4];
den_2=[1 2];
[num,den]=parallel(num_1,den_1,num_2,den_2)
```

运行结果如下：

num =
 1   10   21   10
den =
1   5   8   4

实例讲解

【例5】 已知两个系统$\Sigma_1(A_1,B_1,C_1,D_1)$ 和$\Sigma_2(A_2,B_2,C_2,D_2)$状态空间模型分别为

$$\begin{cases} \dot{x}_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -4 & -8 & -5 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_1 \\ y_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x_1 \end{cases}$$

和

$$\begin{cases} \dot{x}_2 = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x_2 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_2 \\ y_2 = \begin{bmatrix} 1 & 0 \end{bmatrix} x_2 \end{cases}$$

试用Matlab求出$\Sigma_1(A_1,B_1,C_1,D_1)$ 为前向通道和$\Sigma_2(A_2,B_2,C_2,D_2)$ 为负反馈通道的组合系统的状态空间模型。

实例讲解

【例5】

解： Matlab仿真代码如下：

```
A_1=[0 1 0;0 0 1;-4 -8 -5];
B_1=[0;0;1];
C_1=[1 0 0];
D_1=0;
A_2=[0 1;-2 -3];
B_2=[0;1];
C_2=[1 0];
D_2=0;
sign=-1;
[A,B,C,D]=feedback(A_1,B_1,C_1,D_1,A_2,B_2,C_2,D_2,sign)
```

运行结果如下：

A =

0	1	0	0	0
0	0	1	0	0
-4	-8	-5	-1	0
0	0	0	0	1
1	0	0	-2	-3

B =

0
0
1
0
0

C =

1	0	0	0	0

D =

0

实例讲解

二、传递函数模型与状态空间模型的相互转换

1 tf2ss()函数，zp2ss()函数

功能：分别将多项式形式、零极点形式的传递函数转换为状态空间的形式

调用格式：$[\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}]=\mathrm{tf2ss}(num,den)$，$[\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}]=\mathrm{zp2ss}(\boldsymbol{z},\boldsymbol{p},k)$

注意：ss()函数不仅可用于建立系统的状态空间模型SS，而且可以将任意LTI系数模型sys(传递函数模型TF、零极点模型ZPK)转换为状态空间模型，其调用格式为：$SYS=ss(sys)$

实例讲解

【例6】 已知系统的传递函数为：

$$W(s)=\frac{s^2+3s+1}{s^3+2s^2+4s+6}$$

试用Matlab求其状态空间表达式。

实例讲解

【例6】解： Matlab仿真代码如下：

```
num=[1 3 1];
den=[1 2 4 6];
[A,B,C,D]=tf2ss(num,den)
```

运行结果如下：

A =

-2 -4 -6
1 0 0
0 1 0

B =

1
0
0

C =

1 3 1

D =

0

实例讲解

2. ss2tf() 函数，ss2zp()函数

功能：实现从状态空间表达式到传递函数阵的转换

调用格式：对于单输入单输出系统，调用格式如下：

$$[num, den] = \mathrm{ss2tf}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}), [\boldsymbol{z}, \boldsymbol{p}, k] = \mathrm{ss2zp}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D})$$

对于多输入多输出系统，调用格式如下：

$$[\boldsymbol{z}, \boldsymbol{p}, k] = ss2zp(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, iu)$$

实例讲解

【例7】 已知系统的状态空间表达式为:

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 2 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u \end{cases}$$

试用Matlab求系统的传递函数阵。

实例讲解

【例7】解：Matlab仿真代码如下：

```
A=[0,1;-2,-3];
B=[1,0;1,1];
C=[1,0;1,1;0,2];
D=[0,0;1,0;0,1];
[num1,den1]=ss2tf(A,B,C,D,1)
[num2,den2]=ss2tf(A,B,C,D,2)
```

运行结果如下：

```
num1 =
0        1.0000    4.0000
1.0000   5.0000    4.0000
0        2.0000   -4.0000

den1 =
1    3    2

num2 =
0        0         1.0000
0        1.0000    1.0000
1.0000   5.0000    2.0000

den2 =
1    3    2
```

可得系统传递函数为：

$$W(s)=\frac{1}{s^2+3s+2}\begin{bmatrix} s+4 & 1 \\ s^2+5s+4 & s+1 \\ 2s-4 & s^2+5s+2 \end{bmatrix}$$

实例讲解

三、 线性系统的线性变换

1. eig()函数

功能：直接计算矩阵特征值和特征向量

调用格式：$\lambda = eig(\boldsymbol{A})$, $[\boldsymbol{P}, \boldsymbol{\Lambda}] = eig(\boldsymbol{A})$

其中，λ 为矩阵$\boldsymbol{A}$的所有特征值排列而成的向量；P 是由矩阵$\boldsymbol{A}$的所有特征向量组成的矩阵；Λ 是由矩阵$\boldsymbol{A}$的所有特征值为对角元素组成的对角线矩阵。

实例讲解

【例8】试用 Matlab求出下面矩阵A的特征值和特征向量。

$$\boldsymbol{A}=\begin{bmatrix}2 & -1 & -1\\ 0 & -1 & 0\\ 0 & 2 & 1\end{bmatrix}$$

实例讲解

【例8】解：(1) 单独求取特征值，Matlab仿真代码如下：

```
A=[2 -1 -1; 0 -1 0; 0 2 1];
Larmda = eig(A)
```

运行结果如下

Larmda =

2

1

-1

实例讲解

（2）同时求取特征值和特征向量，Matlab仿真代码如下：

```
A = [2 -1 -1; 0 -1 0; 0 2 1];
[V, Jianj] = eig(A)
```

运行结果如下

V =

1.0000	0.7071	0
0	0	0.7071
0	0.7071	-0.7071

实例讲解

Jianj =

2	0	0
0	1	0
0	0	-1

同样可得矩阵的3个特征值分别为2, 1和-1，它们对应的3个特征向量分别为

$$\boldsymbol{p}_1=\begin{bmatrix}1\\0\\0\end{bmatrix},\quad \boldsymbol{p}_2=\begin{bmatrix}0.7071\\0\\0.7071\end{bmatrix},\quad \boldsymbol{p}_3=\begin{bmatrix}0\\0.7071\\-0.7071\end{bmatrix}$$

实例讲解

2. jordan()函数

功能：当矩阵***A***具有重特征根时，函数eig()不具有直接计算广义特征向量的功能，可以借助符号计算工具箱的jordan()函数计算所有特征向量。

调用格式：$[\boldsymbol{P},\boldsymbol{J}]=\mathrm{jordan}(\boldsymbol{A})$

其中，*P*是由矩阵***A***的所有特征向量（包括广义特征向量）组成的矩阵；*J* 是与矩阵***A***对应的约当阵。

如果仅求取矩阵A对应的约当阵***J***，函数的调用格式可为

$$\boldsymbol{J}=\mathrm{jordan}(\boldsymbol{A})$$

实例讲解

【例9】试用 Matlab求出下面矩阵A的特征值和特征向量。

$$\boldsymbol{A}=\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 8 & -12 & 6 \end{bmatrix}$$

实例讲解

【例9】解： 同时求取特征值和特征向量，Matlab仿真代码如下：

```
A = [0 1 0;0 0 1;8 -12 6];
[P,J] = jordan(A)
```

运行结果如下

P =

4	-2	1
8	0	0
16	8	0

J =

2	1	0
0	2	1
0	0	2

可得A的三个特征值都是2，对应的3个特征向量（包括两个广义特征向量）分别为

$$\boldsymbol{p}_1=\begin{bmatrix}4\\8\\16\end{bmatrix},\quad \boldsymbol{p}_2=\begin{bmatrix}-2\\0\\8\end{bmatrix},\quad \boldsymbol{p}_3=\begin{bmatrix}1\\0\\0\end{bmatrix}$$

实例讲解

3. ss2ss()函数

功能：实现系统的线性非奇异变换

调用格式：

$$\boldsymbol{GP}=\mathrm{ss2ss}(\boldsymbol{G},\boldsymbol{P}),[\boldsymbol{At},\boldsymbol{Bt},\boldsymbol{Ct},\boldsymbol{Dt},\boldsymbol{P}]=\mathrm{ss2ss}(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D},\boldsymbol{P})$$

其中，对于前者，G 、GP 分别为变换前和变换后的系统状态空间模型， P为线性非奇异变换矩阵。

对于后者，$(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D})$、$[\boldsymbol{At},\boldsymbol{Bt},\boldsymbol{Ct},\boldsymbol{Dt}]$分别为变换前和变换后系统的状态空间模型的系数矩阵，P为线性非奇异变换矩阵。

实例讲解

注意：但是，Matlab中没有可将一般状态空间表达式变换为约当标准型(对角标准型)的函数，只能先调用jordan()函数求出化为约当标准型的变换矩阵，然后再利用ss2ss()函数将状态空间表达式变换为约当标准型(对角标准型)。

实例讲解

【例10】已知状态空间表达式为

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 8 & -12 & 6 \end{bmatrix} x + \begin{bmatrix} 5 \\ 1 \\ 5 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} x \end{cases}$$

试用Matlab将其变换为约当标准型（对角标准型）

实例讲解

【例10】解：Matlab仿真代码如下：

```
A = [0 1 0;0 0 1;8 -12 6];
B=[5;1;5];
C=[1,0,1];
D=0;
[P,J]=jordan(A)
[Ap,Bp,Cp,Dp]=ss2ss(A,B,C,D,inv(P))
```

运行结果如下

P =

4	-2	1
8	0	0
16	8	0

J =

2	1	0
0	2	1
0	0	2

Ap =

2	1	0
0	2	1
0	0	2

Bp =

0.1250
0.3750
5.2500

Cp =

20 6 1

Dp =

0

实例讲解

线性变换后得到的状态空间表达式为:

$$
\begin{cases}
\dot{x}=\begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}x+\begin{bmatrix} 0.1250 \\ 0.3750 \\ 5.2500 \end{bmatrix}u \\
y=\begin{bmatrix} 20 & 6 & 1 \end{bmatrix}x
\end{cases}
$$

上机练习题

1.1系统的微分方程为

$$y^{(3)}(t)+3y^{(2)}(t)+3\dot{y}(t)+y(t)=\dot{u}(t)+2u(t)$$

试用MATLAB求其状态空间表达式

1.2系统的传递函数为

$$W(s)=\frac{s^2+3s+1}{s^2+2s+1}$$

试用MATLAB求其状态空间表达式

1.3系统的传递函数为

$$W(s)=\frac{s+1}{s^3+7s^2+14s+5}$$

试用MATLAB求其约当标准型状态空间表达式

上机练习题

1.4系统的状态空间表达式为

$$\begin{cases} \dot{x}=\begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & -1 & 1 & 1 \\ 1 & 0 & -3 & 0 \\ -50 & 0 & 0 & -2 \end{bmatrix}x+\begin{bmatrix} 0 \\ 0 \\ 0 \\ 10 \end{bmatrix}u \\ y=\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}x \end{cases}$$

试用MATLAB求系统的传递函数

上机练习题

1.5子系统$\Sigma_1(A_1,B_1,C_1)$：

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}2 & 4\\ 4 & 2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

$$y=\begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

子系统$\Sigma_2(A_2,B_2,C_2)$

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}1 & 2\\ 4 & 3\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}1 & -1\\ 0 & 1\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

$$y=\begin{bmatrix}1 & 0\\ 0 & 2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

试用MATLAB求这两个系统分别的传递函数，以及串联，并联和负反馈连接的组合系统的状态空间表达式和传递函数

上机实验

实例讲解

一、MATLAB在线性系统动态分析中的应用

Matlab求解线性定常系统的状态转移矩阵

1. expm()函数
功能：求解状态转移矩阵$e^{\boldsymbol{A}t}$
调用格式：$eAt = \mathrm{expm}(\boldsymbol{A} * t)$
式中，A为系统矩阵，t为定义的符号标量。

2. ilaplace()函数
功能：对于线性定常系统，求解矩阵的拉普拉斯反变换
调用格式：$eAt = \mathrm{ilaplace}(FS, s, t)$
式中，FS 为进行拉普拉斯反变换的矩阵，s, t 为定义的符号标量
调用该函数求解线性定常系统的状态转移矩阵，需首先计算出$(s\boldsymbol{I} - \boldsymbol{A})^{-1}$，进而对其进行拉普拉斯反变换即可求得状态转移矩阵$e^{At}$。

实例讲解

【例1】 对于矩阵$A=\begin{bmatrix}-3 & 1\\ 1 & -3\end{bmatrix}$，应用Matlab求解状态转移矩阵$e^{At}$。

解：方法1：利用拉普拉斯反变换法求解，其Matlab仿真程序如下：

```
clear all
syms s t                %定义基本符号标量 s 和 t
A=[-3,1;1,-3];
FS=inv(s*eye(2)-A);     %求预解矩阵 FS=(sI-A)^-1，eye(2)为 2x2 单位矩阵
eAt=ilaplace(FS,s,t);   %求拉普拉斯反变换
eAt=simplify(eAt)       %化简表达式
```

其运行结果如下：

```
eAt =
[ (exp(-4*t)*(exp(2*t) + 1))/2, (exp(-4*t)*(exp(2*t) - 1))/2]
[ (exp(-4*t)*(exp(2*t) - 1))/2, (exp(-4*t)*(exp(2*t) + 1))/2]
```

说明：

（1）inv()为Matlab中符号矩阵的求逆函数；

（2）Matlab中，时域函数ft的拉普拉斯变换函数为Fs=laplace(Ft,t,s)；相应的，频域函数FS的拉普拉斯反变换函数为Ft=ilaplace(Fs,s,t)。需注意的是，在调用函数之前，必须正确定义符号变量s，t以及符号表达式Fs，Ft。

（3）Matlab中，simplify()函数的作用是化简符号计算结果表达式。

实例讲解

方法2：调用expm()函数，其Matlab程序如下：

```
syms t                %定义基本符号标量t
A=[-3,1;1,-3];
eAt=expm(A*t)
eAt=simplify(eAt)     %化简表达式
```

其运行结果如下：

eAt =
[(exp(-4*t)*(exp(2*t) + 1))/2, (exp(-4*t)*(exp(2*t) - 1))/2]
[(exp(-4*t)*(exp(2*t) - 1))/2, (exp(-4*t)*(exp(2*t) + 1))/2]

实例讲解

说明：
expm()函数还可以求解 e^{At} 对于于某一时刻t (t为某一常数)的值。如可求解上例中当 t=0.2时e^{At}的值，其Matlab仿真程序如下：

```
A=[-3 1;1 -3];
t=0.2;
eAt=expm(A*t)
```

其运行结果如下：

eAt =

0.5598   0.1105
0.1105   0.5598

实例讲解

3. iztrans()函数
功能：对于线性定常离散系统，求解矩阵的 反变换
调用格式：Fk=iztrans(Fz,z,k)
式中，Fz 为进行Z反变换的矩阵，z, k为定义的符号标量。
应用iztrans()函数求解线性定常离散系统的状态转移矩阵时，需首先计算出$(z\boldsymbol{I}-\boldsymbol{G})^{-1}z$，进而对其进行Z反变换即可求得状态转移矩阵$\boldsymbol{\Phi}(k)$。

实例讲解

【例2】 对于矩阵$A=\begin{bmatrix}0 & 1\\ -0.2 & -0.9\end{bmatrix}$，应用Matlab求解状态转移矩阵$\boldsymbol{\Phi}(k)$。

解：利用Z反变换法求解，其Matlab程序如下：

```
clear all
syms z k                      %定义基本符号标量z和k
G=[0,1;-0.2,-0.9];
Fz=(inv(z*eye(2)-G))*z;
Fk=iztrans(Fz,z,k);           %求Z反变换
Fk=simplify(Fk)               %化简表达式
```

其运算结果：

Fk =

[5*(-2/5)^k - 4*(-1/2)^k, 10*(-2/5)^k - 10*(-1/2)^k]

[2*(-1/2)^k - 2*(-2/5)^k,   5*(-1/2)^k - 4*(-2/5)^k]

实例讲解

二、Matlab求解定常系统时间响应

1. dsolve()函数

功能：求解线性定常齐次状态方程的解

调用格式：$r = \mathrm{dsolve}('eq1,\ eq2',\ \ldots 'cond1,\ cond2',\ \ldots,\ 'v')$

式中，‘eq1, eq2’, …为输入参数，描述常微分方程，这些常微分方程以‘v’作为自变量，如‘v’不指定，则默认t为自变量。‘cond1, cond2’, …用以指定方程的边界条件或初始条件，同样以‘v’作为自变量，r为返回的存放符号微分方程解的架构数组。在方程中，常用大写字母D表示一次微分，D2、D3表示二次、三次微分运算。以此类推，符号D2y表示$\dfrac{\mathrm{d}^2 y}{\mathrm{d}t^2}$。

实例讲解

【例3】 应用Matlab求解下函数中，无输入作用时状态方程的解。

$$\begin{cases} \dot{x} = \begin{bmatrix} -3 & 1 \\ 1 & -3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ x(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{cases}$$

解：调用dsolve()函数求解，其Matlab仿真程序如下：

```
clear all
r=dsolve('Dv=-3*v+w,Dw=v-3*w','v(0)=1,w(0)=0'); %默认t为自变量
x1=r.v                    %返回x1的求解结果
x2=r.w                    %返回x2的求解结果
```

其运行结果如下：

x1 =exp(-4*t)*(exp(2*t)/2 + 1/2)
x2 =exp(-4*t)*(exp(2*t)/2 - 1/2)

实例讲解

2. step()函数，impulse()函数，initial ()函数，lsim()函数

功能：分别计算单位阶跃响应、单位脉冲响应、零输入响应以及任意输入(包括系统初始状态)响应。

调用格式：$\text{step}(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D},iu,t,x0)$ $\text{impulse}(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D},iu,t,x0)$

$\text{initial}(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D},iu,t,x0)$ $\text{lsim}(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D},iu,t,x0)$

式中， A,B,C,D为系统状态空间模型矩阵；iu表示从第iu个输入到所有输出的单位阶跃响应数据； t为用户指定时间向量，缺省时Matlab自动设定；x0为系统初始状态，缺省时默认为零。

相应的，对于线性定常离散系统，Matlab Symbolic Math Toolbox提供了dstep()、dimpulse()、dinitial()、dlsim()函数来计算其单位阶跃响应、单位脉冲响应、零输入响应和任意输入响应。

Matlab中的时域响应分析函数功能非常强大，此处仅做简单的举例说明，其详情和查阅Matlab帮助文档。

实例讲解

【例4】 对于如下状态空间表达式：

$$\begin{cases} \dot{x} = \begin{bmatrix} -3 & 1 \\ 1 & -3 \end{bmatrix} x + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x \\ x(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \end{cases}$$

应用Matlab求解：

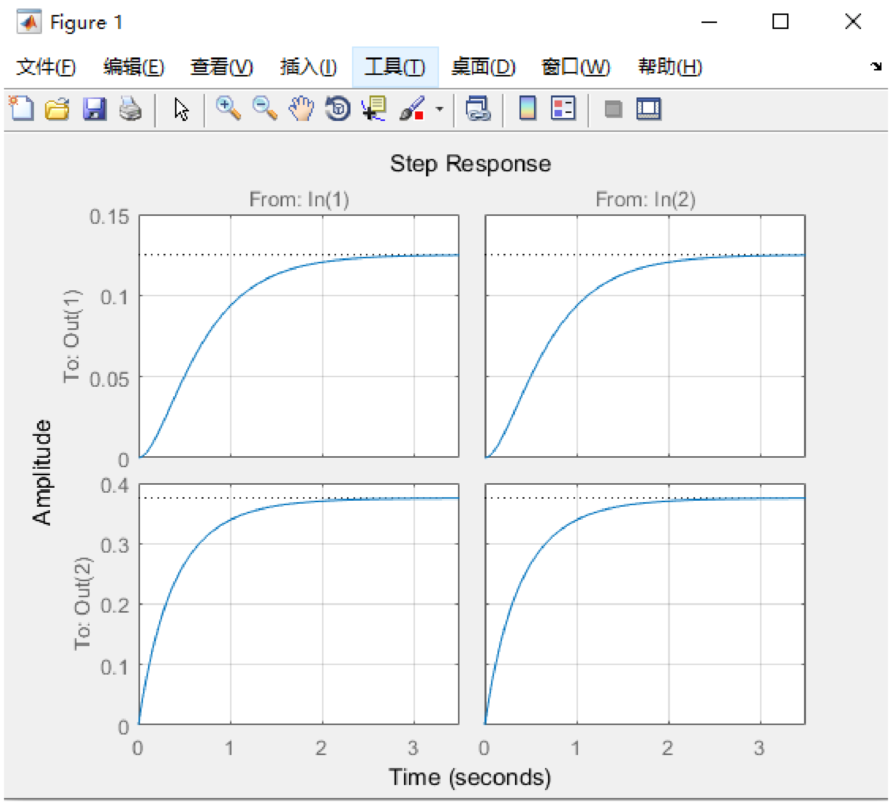
（1）输入$u_1(t)=1(t)$ ，$u_2(t)=1(t)$单独作用下的系统输出响应；

（2）输入$u_1(t)=1(t)$ ，$u_2(t)=1(t)$ 共同作用下系统的输出响应。

实例讲解

解：(1) 调用step()函数求解，其Matlab仿真程序如下：

```
clear all
A=[-3,1;1,-3];
B=[0,1;1,0];
C=[1,0;0,1];
D=[0,0;0,0];
x0=[1;1];
step(A,B,C,D,x0)
grid
```

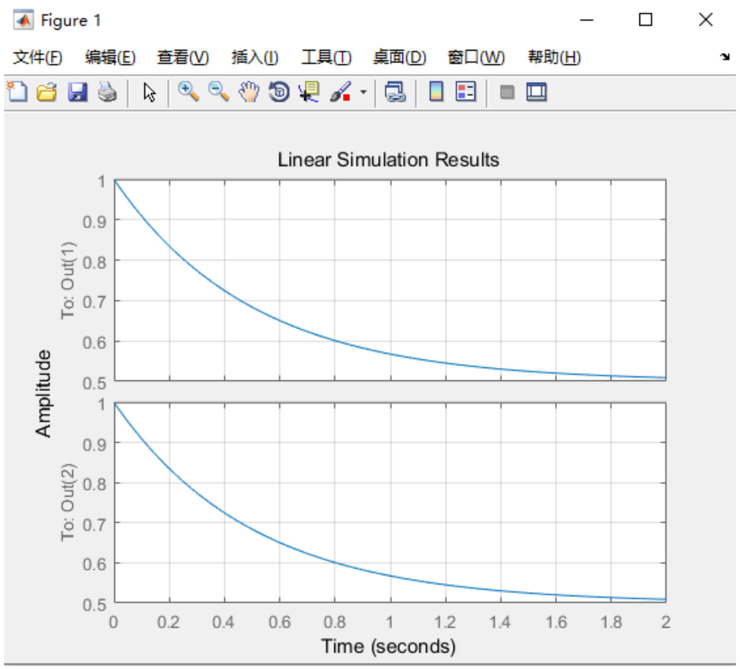


运行结果

实例讲解

(2) 调用lsim()函数求解，其Matlab仿真程序如下：

```
clear all
A=[-3,1;1,-3];
B=[0,1;1,0];
C=[1,0;0,1];
D=[0,0;0,0];
x0=[1;1];
t=0:0.01:2;            %设置时间向量
LT=length(t);          %求时间向量的长度
u1=ones(1,LT);         %生成单位阶跃信号对应于向量t的离   散序列，且与t同维
u2=ones(1,LT);
u=[u1;u2];
lsim(A,B,C,D,u,t,x0)
 grid
```



运行结果

实例讲解

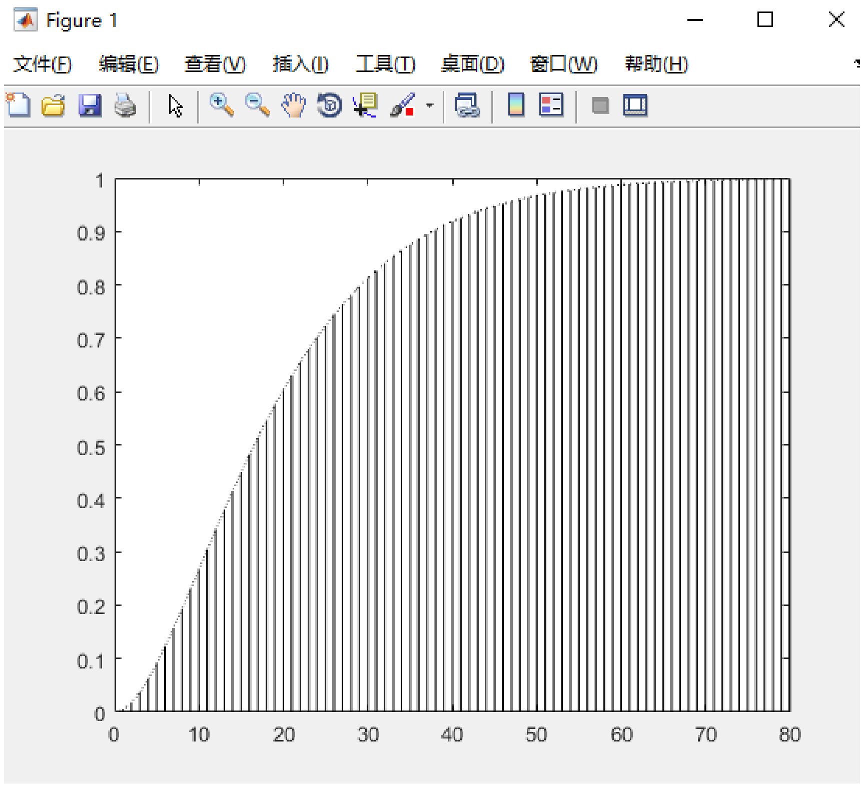
【例5】 对于例2-13所示的离散状态方程，试用Matlab求解当T=0.1s，输入为单位阶跃函数，且初始状态为零状态时的离散输出y(kT)。

$$\begin{cases} \begin{bmatrix} x_1((k+1)T) \\ x_2((k+1)T) \end{bmatrix} = \begin{bmatrix} 1.0187 & 0.0906 \\ -0.1 & 1 \end{bmatrix} \begin{bmatrix} x_1(kT) \\ x_2(kT) \end{bmatrix} + \begin{bmatrix} 0.0047 \\ 0.1 \end{bmatrix} r(kT) \\ y(kT) = [1,0] \begin{bmatrix} x_1(kT) \\ x_2(kT) \end{bmatrix} = x_1(kT) \end{cases}$$

实例讲解

解：Matlab仿真程序如下：

```
clear all;
T=0.1;
G=[0.9953,0.0906;-0.0906,0.8187];
H=[0.0047;0.0906];
C=[1,0];
D=0;
[yd,x,n]=dstep(G,H,C,D);
for k=1:n
    plot([k-1,k-1],[0,yd(k)],'k')
    hold on
end
e=1-yd;
for k=1:n
    for j=1:100
        u(j+(k-1)*100)=e(k);
    end
end
t=(0:0.01:n-0.01)*T;
[yc]=lsim([0,1;0,-2],[0;1],[1,0],[0],u,t);
plot(t/T,yc,':k')
axis([0 80 0 1])
hold off
```



运行结果

实例讲解

1. c2d()函数

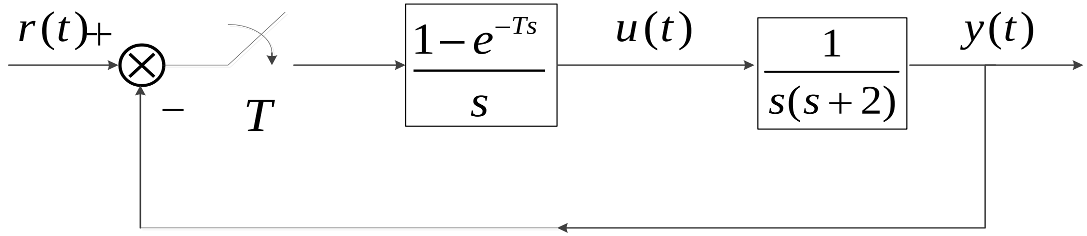
功能：进行线性定常连续系统状态方程的离散化求解

调用格式：$\left[\boldsymbol{G},\boldsymbol{H}\right]=\mathrm{c2d}\left(\boldsymbol{A},\boldsymbol{B},T\right)$

式中， A，B为连续系统的系统矩阵和输入矩阵；G,H分别对应离散化后的系统矩阵和输入矩阵；当输入端采用零阶保持器，T为采样周期时。

实例讲解

【例6】 应用Matlab，将下例中连续被控对象进行离散化。



解：针对例6中的连续被控对象，设计Matlab仿真程序为：

```
clear all
syms T
A=[0,1;0,-2];
B=[0;1];
[G,H]=c2d(A,B,T)
```

其运行结果如下：

```
G =
   [ 1,    1/2 - exp(-2*T)/2;
     0,        exp(-2*T)]
H =
   T/2 + exp(-2*T)/4 - 1/4
     1/2 - exp(-2*T)/2
```

实例讲解

Matlab还可以求解指定采样周期的离散化状态方程，例T=0.1s 时：

```
clear all
A=[0,1;0,-2];
B=[0;1];
T=0.1;
[G,H]=c2d(A,B,T)
```

其运行结果如下：

```
G =
    1.0000    0.0906
         0    0.8187
H =
    0.0047
    0.0906
```

实例讲解

2. c2dm()函数

功能：允许用户可以指定不同的离散变换方式，将连续状态空间模型变换为离散状态空间模型，以提高离散化的精度

调用格式：$[\boldsymbol{G},\boldsymbol{H}]=\mathrm{c2d}(\boldsymbol{A},\boldsymbol{B},T,\text{'method'})$

当’method’=’zoh’时，变换时输入端采用零阶保持器；

当’method’=’foh’时，变换时输入端采用一阶保持器；

当’method’=’tustin’时，变换时输入端采用双线性逼近导数等。

Matlab Symbolic Math Toolbox还提供了d2c()、d2cm()函数分别对应于c2d()、c2dm()的逆过程，完成从离散时间系统到连续时间系统的变换。关于这些函数的具体使用，用户可通过Matlab联机帮助查阅，此处不再详述。

上机练习题

2.1已知系统状态空间表达式为

$$\begin{cases} \dot{x}(t)=\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 2 & -5 & 4 \end{bmatrix}x(t)+\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}u(t) \\ y(t)=\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}x(t) \end{cases}$$

试用MATLAB求：1）系统状态转移矩阵
2）系统的零输入响应，并绘制响应曲线
3）系统的阶跃响应，并绘制阶跃响应曲线

上机练习题

2.2已知系统状态空间表达式为

$$\begin{cases} \dot{x}(t)=\begin{bmatrix} 1 & -2 & 2 \\ -2 & -2 & 4 \\ 2 & 4 & -2 \end{bmatrix}x(t)+\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}u(t) \\ y(t)=\begin{bmatrix} 1 & 2 & 1 \\ 0 & -1 & 2 \end{bmatrix}x(t) \\ x(0)=\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \end{cases}$$

试用MATLAB求：1）系统零输入响应

2）输入$u_1(t)=1(t), u_2(t)=1(t), u_3(t)=1(t)$ 单独作用下的系统输出响应

3）输入$u_1(t)=1(t), u_2(t)=1(t), u_3(t)=1(t)$ 共同作用下的系统输出响应

上机练习题

2.3已知系统状态空间表达式为

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 2 & -5 & 4 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x(t) \end{cases}$$

试用MATLAB求：1）系统离散化后的状态空间表达式
2）当采样周期T=0.1S时的状态转移矩阵
3）当采样周期T=0.1S，初始状态为零状态时的离散输出y(kT)

上机实验

实例讲解

Matlab在系统能控性与能观性分析中的应用

Matlab控制工具箱为系统能控性、能观性分析提供了专用函数：

ctrb()函数
功能：根据动态系统 生成能控性判别矩阵
调用格式：$\boldsymbol{Q}_c = \mathrm{ctrb}(\boldsymbol{A}, \boldsymbol{B})$

obsv()函数
功能：根据动态系统 生成能观性判别矩阵
调用格式：$\boldsymbol{Q}_o = \mathrm{obsv}(\boldsymbol{A}, \boldsymbol{C})$

实例讲解

ctrbf()函数

功能：将不能控子系统 按能控性分解。

调用格式：$\left[\boldsymbol{A}_{hat} \ \boldsymbol{B}_{hat} \ \boldsymbol{C}_{hat} \ \boldsymbol{P} \ \boldsymbol{K} \right] = \mathrm{ctrbf}\left(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \right)$

obsvf()函数

功能：将不能观子系统 按能观性分解。

调用格式：$\left[\boldsymbol{A}_{hat} \ \boldsymbol{B}_{hat} \ \boldsymbol{C}_{hat} \ \boldsymbol{P} \ \boldsymbol{K} \right] = obsvf\left(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \right)$

实例讲解

【例3-34】 系统的状态空间表达式为：

$$\begin{cases} \dot{\boldsymbol{x}} = \begin{bmatrix} 4 & 1 & 0 & 0 \\ 0 & 4 & 1 & 0 \\ 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 4 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 & 0 \\ 1 & 2 \\ 0 & 0 \\ 2 & 1 \end{bmatrix} \boldsymbol{u} \\ \boldsymbol{y} = \begin{bmatrix} 1 & 0 & 2 & 0 \\ 2 & 0 & 4 & 2 \end{bmatrix} \boldsymbol{x} \end{cases}$$

判断系统的能控性与能观性。

实例讲解

解：应用Matlab秩判据求解，Matlab程序为

```
clc;
clear all;
A=[4,1,0,0;0,4,1,0;0,0,4,1;0,0,0,4];
B=[0,0;1,2;0,0;2,1];
C=[1,0,2,0;2,0,4,2];
Qc=ctrb(A,B);
Qo=obsv(A,C);
rc=rank(Qc);
ro=rank(Qo);
L=size(A);
if rc==L
    str='系统能控'
else
    str='系统不能控'
end
if ro==L
    str='系统能观'
else
    str='系统不能观'
end
```

运行结果如下：

str =

系统能控

str =

系统能观

上机练习题

3.1已知系统状态空间表达式为：

$$\begin{cases} \dot{x}=\begin{bmatrix} -2 & 2 & -1 \\ 0 & -2 & 0 \\ 1 & -4 & 0 \end{bmatrix}x+\begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}u \\ y=\begin{bmatrix} 1 & -1 & 2 \\ -2 & 0 & 1 \end{bmatrix}x \end{cases}$$

试用MATLAB：

1）求系统传递函数

2）判断系统能控性

3）若不能控，对系统进行能控性分解

上机练习题

3.2已知系统状态空间表达式为：

$$\begin{cases} \dot{x} = \begin{bmatrix} 4 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 4 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & -1 \\ -2 & 0 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 1 & 2 & 1 \\ 1 & 2 & 2 & 0 \end{bmatrix} x \end{cases}$$

试用MATLAB：

1）求系统传递函数

2）判断系统能观性

3）若不能观，对系统进行能观性分解

上机练习题

3.3已知系统状态空间表达式为:

$$\begin{cases} \dot{x} = \begin{bmatrix} -4 & 1 & 0 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 1 & 3 \\ 5 & 7 \\ 4 & 3 \\ 0 & 0 \\ 1 & 6 \\ 0 & 0 \end{bmatrix} u \\ y = \begin{bmatrix} 3 & 1 & 0 & 5 & 0 & 0 \\ 1 & 4 & 0 & 2 & 0 & 0 \end{bmatrix} x \end{cases}$$

试用MATLAB对系统进行能控能观性结构分解

上机实验

中国地质大学 自动化学院
School of Automation, China University of Geosciences

实例讲解

Matlab在系统稳定性分析中的应用

1. Poly()函数， roots()函数

功能： **Poly()**函数用来求矩阵特征多项式系数，

roots()函数用来求取特征值。

调用格式：P=poly($\boldsymbol{A}$),V=roots(P)

实例讲解

【例1】 已知线性时不变系统的状态方程为

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} -3 & -6 & -2 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

试用特征值判据判断系统的稳定性。

实例讲解

解： MATLAB程序如下：

```
A=[-3 -6 -2 -1; 1 0 0 0; 0 1 0 0; 0 0 1 0];
P=poly(A),V=roots(P)
```

运行以上程序得：

```
P =

   1.0000    3.0000    6.0000    2.0000    1.0000
V =

  -1.3544 + 1.7825i
  -1.3544 - 1.7825i
  -0.1456 + 0.4223i
  -0.1456 - 0.4223i
```

特征值的实部都小于0，故系统稳定。

实例讲解

2. lyap()函数

功能：函数lyap()用来求解系统的李雅普诺夫方程。

调用格式：$P = lyap(A,Q)$

【例2】已知线性定常系统如图所示，

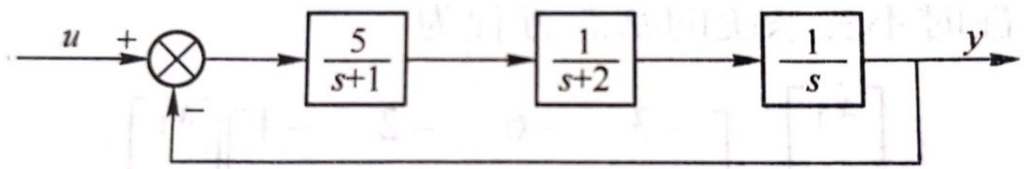


图 4. 10　系统框图

试求系统的状态方程；选择正定的实对称矩阵Q后计算李雅普诺夫方程的解，并利用李雅普诺夫函数确定系统的稳定性。

实例讲解

解： 讨论系统的稳定性时可令给定输入u=0。根据题目要求，因为需要调用函数lyap()，故首先将系统转换成状态空间模型。选定半正定矩阵Q为

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

为了确定系统的稳定性，需验证矩阵P的正定性，这可以对各主子行列式进行校验。综合以上考虑，给出调用函数lyap()的程序：

实例讲解

```
n1 = 5;d1 = [1 1];s1 = tf(n1,d1);
n2 = 1;d2 = [1 2];s2 = tf(n2,d2);
n3 = 1;d3 = [1 0];s3 = tf(n3,d3);
s123 = s1 * s2 * s3;
sb - = feedback(s123,1);
a=tf2ss(sb,num{1},sb.den{1});
q=[0 0 0;0 0 0;0 0 1];
if det(a)~=0
    P=lyap(a,q)
    det1=det(P(1,1))
    det2=det(P(2,2))
    detp=det(P)
end
```

运行结果如下：

P=

12.5000 0.0000 -7.5000

0.0000 7.5000 -0.5000

-7.5000 -0.5000 4.7000

det1=

12.5000

det2=

7.5000

detp=

15.6250

实例讲解

即系统的状态方程为：

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -3 & -2 & -5 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

李雅普诺夫函数为：

$$P = \begin{bmatrix} 12.5 & 0 & -7.5 \\ 0 & 7.5 & -0.5 \\ -7.5 & -0.5 & 4.7 \end{bmatrix}$$

中国地质大学 自动化学院
School of Automation, China University of Geosciences

实例讲解

因为

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

是半正定矩阵，由式

$$\dot{V}(x)=-x^TQx=-x_3{}^2$$

可知，$\dot{V}(x)$是半正定的。最后，对各主子行列式(det1,det2,detp)进行检验，说明矩阵P是正定阵，所以本系统在坐标原点的平衡状态是稳定的，而且是大范围渐近稳定的。

实例讲解

【例3】 已知线性系统动态方程为

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

试计算李雅普诺夫方程的解，并利用李雅普诺夫函数确定系统的稳定性并求李雅普诺夫函数。

实例讲解

解 首先选择正定实对称Q为单位矩阵，即

$$\boldsymbol{Q}=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

根据题意，给出调用函数lyap()的程序。

MATLAB程序如下：

```
a = [0 1;  -1 -1];q=[1 0;  0 1];
if det (a) ~ = 0
    P = lyap(a,q)
    det1 = det(P(1,1))
    detp = det(P)
end
```

实例讲解

运行程序可得

```
P =

    1.5000   -0.5000
   -0.5000    1.0000

det1 =

    1.5000

detp =

    1.2500
```

即李雅普诺夫方程的解为

$$\boldsymbol{P}=\begin{bmatrix}1.5 & -0.5\\ -0.5 & 1\end{bmatrix}$$

程序对各主子行列式(det1,detp)进行计算，计算结果说明矩阵P确是正定阵。李雅普诺夫函数为

$$\boldsymbol{V}(\boldsymbol{x})=\boldsymbol{x}^T\boldsymbol{P}\boldsymbol{x}=\begin{bmatrix}x_1 & x_2\end{bmatrix}\begin{bmatrix}1.5 & -0.5\\ -0.5 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=\frac{1}{2}(3x_1^2-2x_1x_2+2x_2^2)$$

实例讲解

在状态空间内，V(x)是正定的，而

$$\boldsymbol{V}(\boldsymbol{x}) = \boldsymbol{x}^T(A^TP+PA)\boldsymbol{x} = x(-I)x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = -(x_1^2+x_2^2)$$

是负定的。另外，当 $\|x\| \to \infty$ 时，有$V(x) \to \infty$ ，因此系统原点处的平衡状态是大范围渐近稳定的。

对于稳定性与李雅普诺夫方法，MATLAB并不限于上面介绍的函数及方法，有兴趣的同学可以参考有关资料获得更多更方便的方法。

上机练习题

4.1设线性系统状态方程如下，试用MATLAB判断系统平衡状态的稳定性

1） $\dot{x} = \begin{bmatrix} 1 & 2 & 4 \\ 1 & 1 & 1 \\ 0 & 2 & 1 \end{bmatrix} x$

2） $\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & -3 & -3 \end{bmatrix} x$

上机练习题

4.2已知线性系统动态方程

$$\dot{x}=\begin{bmatrix}1 & 0 & -1\\ 0 & 1 & 0\\ 0 & 0 & 2\end{bmatrix}x$$

试计算李雅普诺夫方程的解，利用李雅普诺夫函数确定系统的稳定性并求李雅普诺夫函数

上机实验

实例讲解

一、利用MATLAB实现系统的综合

MATLAB控制系统工具箱为极点配置、状态观测器、系统解耦等系统综合提供了专用函数。

1 acker() 函数

功能：单输入系统$\Sigma(\boldsymbol{A},\boldsymbol{b},\boldsymbol{c})$的极点配置

调用格式：$\boldsymbol{K}=\mathrm{acker}(\boldsymbol{A},\boldsymbol{b},\boldsymbol{P})$

其中，P为配置极点， K为反馈增益矩阵。

2 place()函数

功能：单输入或多输入系统$\Sigma(\boldsymbol{A},\boldsymbol{B},\boldsymbol{c})$的极点配置

调用格式：$\boldsymbol{K}=place(\boldsymbol{A},\boldsymbol{B},\boldsymbol{P})$

其中，P为配置极点， K为反馈增益矩阵。

实例讲解

二、利用MATLAB进行闭环系统极点配置

当系统完全能控时，通过状态反馈可实现闭环系统极点的任意配置。关键是求解状态反馈矩阵K，当系统的阶数大于3，或为多输入多输出系统时，具体设计要困难的多。采用MATLAB控制系统工具箱的专用函数，具体设计问题就简单多了。

【例5-1】已知系统的状态方程为：

$$\dot{x} = \begin{bmatrix} -2 & -1 & 1 \\ 1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} u$$

试用状态反馈将闭环系统的极点配置为-1,-2,-3，求状态反馈矩阵K。

实例讲解

解：(1) MATLAB仿真程序

```
A=[-2,-1,1;1,0,1;-1,0,1];
b=[1;1;1];
Qc=ctrb(A,b);
rc=rank(Qc);
f=conv([1,1],conv([1,2],[1,3]));
K=[zeros(1,length(A)-1) 1]*inv(Qc)*polyvalm(f,A)
```

运行结果如下：

K =   -1   2   4

实例讲解

(2) 用函数进行极点配置的程序为:

```
A=[-2,-1,1;1,0,1;-1,0,1];
b=[1;1;1];
Qc=ctrb(A,b);
rc=rank(Qc);
P=[-1,-2,-3];
K=acker(A,b,P)
```

运行结果如下:

K = -1 2 4

实例讲解

(3) 用函数 进行极点配置的仿真程序

```
A=[-2,-1,1;1,0,1;-1,0,1];
b=[1;1;1];
Qc=ctrb(A,b);
rc=rank(Qc);
P=[-1,-2,-3];
K=place(A,b,P)
```

运行结果如下：

K = -1.0000   2.0000   4.0000

实例讲解

三、利用MATLAB设计全维状态观测器

极点配置要求系统的状态向量必须全部可测量，当状态向量全部或部分不可直接测量时，则应设计状态观测器进行状态重构。对于被控系统$\Sigma(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C})$，其状态空间表达式为：

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$$

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x}$$

若系统完全能观测，则可构造状态观测器。在MATLAB控制系统工具箱中，利用对偶原理，可使设计问题大为简化。首先构造被控系统$\Sigma(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C})$的对偶系统为：

$$\dot{\boldsymbol{z}} = \boldsymbol{A}^T\boldsymbol{z} + \boldsymbol{C}^T\boldsymbol{u}$$

$$\boldsymbol{y} = \boldsymbol{B}^T\boldsymbol{z}$$

实例讲解

然后，对偶系统按极点配置求状态反馈矩阵$\boldsymbol{K}$

$$\boldsymbol{K} = \mathrm{acker}(\boldsymbol{A}^T, \boldsymbol{C}^T, \boldsymbol{P})$$

或

$$\boldsymbol{K} = \mathrm{place}(\boldsymbol{A}^T, \boldsymbol{C}^T, \boldsymbol{P})$$

原系统的状态观测器的反馈矩阵$\boldsymbol{G}$ 就是其对偶系统的状态反馈矩阵$\boldsymbol{K}$ 的转置，即

$$\boldsymbol{G} = \boldsymbol{K}^T$$

其中，$\boldsymbol{P}$ 为状态观测器的给定期望极点；$\boldsymbol{G}$ 为状态观测器的反馈矩阵。

实例讲解

【例5-2】设系统的状态空间表达式为：

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 0 & -2 \\ 1 & 0 & 9 \\ 0 & 1 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} \boldsymbol{u}$$

$$\boldsymbol{y} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}$$

试设计全维状态观测器，使状态观测器的闭环极点为$-3,-4,-5$。

实例讲解

解：MATLAB仿真程序：

```
A=[0 0 2;1 0 9;0 1 0];
B=[3;2;1];
C=[0 0 1];
n=3
Qo=obsv(A,C);
ro=rank(Qo);
if (ro==n)
    disp('系统是可观测的')
    P=[-3 -4 -5];        %状态观测器的设计
A1=A';
    B1=C';
    K=acker(A1,B1,P);
        G=K'
    AGC=A-G*C
 elseif (ro~=n)
    disp('系统是不可观测的，不能进行观测器的设计')
 end
```

实例讲解

运行结果如下：

n=

3

系统是可观测的

G =

62

56

12

AGC =

0	0	-60
1	0	-47
0	1	-12

实例讲解

被控系统的全维状态观测器为

$$\dot{\hat{\boldsymbol{x}}}=\begin{bmatrix}0 & 0 & -60\\ 1 & 0 & -47\\ 0 & 1 & -12\end{bmatrix}\hat{\boldsymbol{x}}+\begin{bmatrix}3\\ 2\\ 1\end{bmatrix}\boldsymbol{u}+\begin{bmatrix}62\\ 56\\ 12\end{bmatrix}\boldsymbol{y}$$

实例讲解

四、利用MATLAB设计降维状态观测器

已知线性时不变系统为：

$$\begin{cases} \dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \\ \boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} \end{cases}$$

若系统完全能观测，则可将状态向量$\boldsymbol{x}$分为可量测和不可量测两部分，通过特定线性非奇异变换可导出相应的系统方程为分块矩阵形式：

$$\begin{cases} \begin{bmatrix} \dot{\bar{\boldsymbol{x}}}_1 \\ \dot{\bar{\boldsymbol{x}}}_2 \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{A}}_{11} & \bar{\boldsymbol{A}}_{12} \\ \bar{\boldsymbol{A}}_{21} & \bar{\boldsymbol{A}}_{22} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}_1 \\ \bar{\boldsymbol{x}}_2 \end{bmatrix} + \begin{bmatrix} \bar{\boldsymbol{B}}_1 \\ \bar{\boldsymbol{B}}_2 \end{bmatrix} \boldsymbol{u} \\ \bar{\boldsymbol{y}} = \begin{bmatrix} 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}_1 \\ \bar{\boldsymbol{x}}_2 \end{bmatrix} = \bar{\boldsymbol{x}}_2 \end{cases}$$

实例讲解

其中，m维状态$\bar{\boldsymbol{x}}_2$能够直接由输出量$\bar{\boldsymbol{y}}$获得，不必再通过观测器进行重构；$(n-m)$维状态变量$\bar{\boldsymbol{x}}_1$由观测器进行重构。可得关于$\bar{\boldsymbol{x}}_1$的状态方程

$$\begin{cases} \dot{\bar{\boldsymbol{x}}}_1 = \bar{\boldsymbol{A}}_{11}\bar{\boldsymbol{x}}_1 + \bar{\boldsymbol{A}}_{12}\bar{\boldsymbol{x}}_2 + \bar{\boldsymbol{B}}_1\boldsymbol{u} = \bar{\boldsymbol{A}}_{11}\bar{\boldsymbol{x}}_1 + \boldsymbol{M} \\ \dot{\bar{\boldsymbol{y}}} - \bar{\boldsymbol{A}}_{22}\bar{\boldsymbol{y}} - \bar{\boldsymbol{B}}_2\boldsymbol{u} = \bar{\boldsymbol{A}}_{21}\bar{\boldsymbol{x}}_2 \end{cases}$$

它与全维状态观测方程进行对比，可得到两者之间的对应关系，如表所示。

全维观测器与降维观测器对比

全维观测器	降维观测器	全维观测器	降维观测器
$\boldsymbol{x}$	$\bar{\boldsymbol{x}}_1$	$\boldsymbol{y}$	$\dot{\bar{\boldsymbol{y}}} - \bar{\boldsymbol{A}}_{22}\bar{\boldsymbol{y}} - \bar{\boldsymbol{B}}_2\boldsymbol{u}$
$\boldsymbol{A}$	$\bar{\boldsymbol{A}}_{11}$	$\boldsymbol{C}$	$\bar{\boldsymbol{A}}_{21}$
$\boldsymbol{B}\boldsymbol{u}$	$\bar{\boldsymbol{A}}_{12}\boldsymbol{y} + \bar{\boldsymbol{B}}_1\boldsymbol{u}$	$\boldsymbol{G}_{n\times 1}$	$\boldsymbol{G}_{(n-m)\times 1}$

实例讲解

降维观测器的方程为

$$\dot{\hat{\boldsymbol{w}}} = (\bar{\boldsymbol{A}}_{11} - \bar{\boldsymbol{G}}\,\bar{\boldsymbol{A}}_{21})\hat{\boldsymbol{x}}_1 + (\bar{\boldsymbol{A}}_{12} - \bar{\boldsymbol{G}}\,\bar{\boldsymbol{A}}_{22})\bar{\boldsymbol{y}} + (\bar{\boldsymbol{B}}_1 - \bar{\boldsymbol{G}}\,\bar{\boldsymbol{B}}_2)\boldsymbol{u}$$

$$\hat{\boldsymbol{x}}_1 = \hat{\boldsymbol{w}} + \bar{\boldsymbol{G}}\bar{\boldsymbol{y}}$$

或

$$\dot{\hat{\boldsymbol{w}}} = (\bar{\boldsymbol{A}}_{11} - \bar{\boldsymbol{G}}\,\bar{\boldsymbol{A}}_{21})\hat{\boldsymbol{w}} + [(\bar{\boldsymbol{A}}_{11} - \bar{\boldsymbol{G}}\,\bar{\boldsymbol{A}}_{21})\bar{\boldsymbol{G}} + (\bar{\boldsymbol{A}}_{12} - \bar{\boldsymbol{G}}\,\bar{\boldsymbol{A}}_{22})]\bar{\boldsymbol{y}} + (\bar{\boldsymbol{B}}_1 - \bar{\boldsymbol{G}}\,\bar{\boldsymbol{B}}_2)\boldsymbol{u}$$

$$\hat{\boldsymbol{x}}_1 = \hat{\boldsymbol{w}} + \bar{\boldsymbol{G}}\bar{\boldsymbol{y}}$$

然后，使用MATLAB的函数place() 或acker()，根据全维状态观测的设计方法求解反馈阵$\bar{\boldsymbol{G}}$。

实例讲解

【例5-3】设系统的状态空间表达式为：

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \boldsymbol{u},\ \boldsymbol{y} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \boldsymbol{x}$$

试设计一个降维状态观测器，使得观测器的期望极点为$-2,-3$。

解：由于x_1可量测，因此只需设计x_2和x_3的状态观测器，故根据原系统可得不可量测部分的状态空间表达式为

$$\begin{cases} \dot{\bar{\boldsymbol{x}}}_1 = \bar{\boldsymbol{A}}_{11}\bar{\boldsymbol{x}}_1 + \bar{\boldsymbol{A}}_{12}\bar{\boldsymbol{x}}_2 + \bar{\boldsymbol{B}}_1\boldsymbol{u} = \bar{\boldsymbol{A}}_{11}\bar{\boldsymbol{x}}_1 + \boldsymbol{M} \\ \bar{\boldsymbol{y}} - \bar{\boldsymbol{A}}_{22}\bar{\boldsymbol{y}} - \bar{\boldsymbol{B}}_2\boldsymbol{u} = \bar{\boldsymbol{A}}_{21}\bar{\boldsymbol{x}}_2 \end{cases}$$

实例讲解

其中$\bar{\boldsymbol{A}}_{11}=\begin{bmatrix}0 & 1\\ -11 & -6\end{bmatrix}$,$\bar{\boldsymbol{A}}_{12}=\begin{bmatrix}0\\ -6\end{bmatrix}$,$\bar{\boldsymbol{A}}_{21}=\begin{bmatrix}1 & 0\end{bmatrix}$,$\bar{\boldsymbol{A}}_{22}=0$,$\bar{\boldsymbol{B}}_{1}=\begin{bmatrix}0\\ 1\end{bmatrix}$,

$\bar{\boldsymbol{B}}_{2}=\begin{bmatrix}0\end{bmatrix}$。

等效方程

$$\begin{cases}\dot{\boldsymbol{z}}=\boldsymbol{A}_c\boldsymbol{z}+\boldsymbol{b}_c\boldsymbol{\eta}\\ \boldsymbol{w}=\boldsymbol{C}_c\boldsymbol{z}\end{cases}$$

其中,$\boldsymbol{A}_c=\bar{\boldsymbol{A}}_{11}$,$\boldsymbol{b}_c\boldsymbol{\eta}=\bar{\boldsymbol{A}}_{12}\bar{\boldsymbol{y}}+\bar{\boldsymbol{B}}_1\boldsymbol{u}$,$\boldsymbol{C}_c=\bar{\boldsymbol{A}}_{21}$。

实例讲解

MATLAB仿真程序

```
A=[0 1 0;0 0 1;-6 -11 -6];
B=[0;0;1];
C=[1 0 0];
T_inv=[0 1 0; 0 0 1;1 0 0];
T=inv(T_inv);
A_bar=T_inv*A*T;
B_bar= T_inv *B;
C_bar=C*T;
A11_bar =[A_bar(1:2,1:2)];
A12_bar =[A_bar(1:2,3)];
A21_bar =[A_bar(3,1:2)];
A22_bar =[A_bar(3,3)];
B1_bar =B(1:2,1);
B2_bar =B(3,1);
A1= A11_bar;
C1=A21_bar;
AX=( A11_bar)';
BX=(C1)';
P=[-2 -3];
K=acker(AX, BX, P);
G=K'
AGAZ=( A11_bar -G* A21_bar)
AGAY=( A11_bar -G* A21_bar)*G+A12_bar-G*A22_bar
BGBU=B1_bar-G*B2_bar
```

运行结果为：

G=

-1

1

AGAZ =

1 1

-12 -6

AGAY =

0

0

BGBU =

0

1

实例讲解

即降维状态观测器为：

$$\dot{\hat{\boldsymbol{w}}}=\begin{bmatrix}1 & 1\\ -12 & -6\end{bmatrix}\hat{\boldsymbol{w}}+\begin{bmatrix}0\\1\end{bmatrix}u+\begin{bmatrix}0\\0\end{bmatrix}\bar{y}$$

$$\hat{\boldsymbol{x}}_1=\hat{\boldsymbol{w}}+\begin{bmatrix}-1\\1\end{bmatrix}\boldsymbol{y}$$

实例讲解

五、带状态观测器的系统极点配置

状态观测器解决了受控系统的状态重构问题，为那些状态变量不能直接量测的系统实现状态反馈创造了条件。带状态观测器的状态反馈系统由三部分组成，即被控系统、观测器和状态反馈。

设能控能观测的被控系统为

$$\begin{cases} \dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \\ \boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} \end{cases}$$

状态反馈控制律为

$$\boldsymbol{u} = \boldsymbol{v} + \boldsymbol{K}\hat{\boldsymbol{x}}$$

实例讲解

状态观测器方程为

$$\dot{\hat{x}} = (A - GC)\hat{x} + Bu + Gy$$

可得闭环系统的状态空间表达式为:

$$\begin{cases} \dot{x} = Ax + BK\hat{x} + Bv \\ \dot{\hat{x}} = GCx + (A - GC - BK)\hat{x} + Bv \\ y = Cx \end{cases}$$

根据分离原理，系统的状态反馈阵 K 和观测器反馈阵 G 可分别设计。

实例讲解

【例5-4】已知开环系统

$$\begin{cases} \dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 1 \\ 20.6 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \boldsymbol{u} \\ \boldsymbol{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x} \end{cases}$$

设计状态反馈使闭环极点为$-1.8 \pm j2.4$，设计状态观测器使其闭环极点为$-8,-8$。

解： 状态反馈和状态观测器的设计分开进行，状态观测器的设计借助于对偶原理。在设计之前，应先判别系统的能控性和能观测性。MATLAB仿真程序

实例讲解

```
A = [0 1;20.6 0];b=[0;1];C=[1 0];
% Check Controllability and Observability
disp('The rank of Controllability Matrix')
rc = rank(ctrb(A,b))
disp('The rank of Observability Matrix')
ro = rank(obsv(A,C))
% Design Regulator
P = [-1.8+2.4*j -1.8-2.4*j];
K = acker(A,b,P)
% Design State Observer
A1 = A';b1 = C';C1 = b';
P1 = [-8 -8];
K1 = acker(A1,b1,P1);
G = K1'
```

运行结果如下：

```
The rank of Controllability Matrix
rc =
     2
ro =
     2
K =
     29.6000    3.6000
G =
     16.0000
     84.6000
```

对于线性时不变系统的综合，MATLAB并不限于上面介绍的函数及方法，有兴趣的读者可以参考有关资料获得更多更方便的方法。

上机练习题

5.1用MATLAB求解以下习题

1）已知系统状态方程为

$$\begin{cases}\dot{\boldsymbol{x}}=\begin{bmatrix}0 & 2\\ -2 & 0\end{bmatrix}\boldsymbol{x}+\begin{bmatrix}1\\ 0\end{bmatrix}\boldsymbol{u}\\ \boldsymbol{y}=\begin{bmatrix}0 & 1\end{bmatrix}\boldsymbol{x}\end{cases}$$

由状态反馈实现闭环极点配置，使得闭环极点为 $\lambda_{1,2}=-3\pm j2$，试确定状态反馈矩阵K，并画出系统模拟结构图。

上机练习题

2）已知系统状态方程为

$$\dot{\boldsymbol{x}}=\begin{bmatrix}1 & -1 & 1\\ 0 & 1 & 1\\ 1 & 0 & 1\end{bmatrix}\boldsymbol{x}+\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}\boldsymbol{u}$$

试求：

(1) 能否用状态反馈任意配置闭环极点

(2) 确定状态反馈矩阵 $\boldsymbol{K}$，使得闭环系统极点为 $-5,-1\pm j$

上机练习题

3）已知系统状态方程为

$$\begin{cases} \dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 0 & -1 \\ 1 & 0 & -3 \\ 0 & 1 & -3 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \boldsymbol{u} \\ \boldsymbol{y} = \begin{bmatrix} 0 & 1 & -2 \end{bmatrix} \boldsymbol{x} \end{cases}$$

试求：

（1）系统是否稳定

（2）系统能否镇定。若能，试设计状态反馈使之稳定。

上机练习题

4）已知系统为

$$\begin{cases} \dot{x} = \begin{bmatrix} -1 & 0 & 1 \\ 1 & -2 & 0 \\ 0 & 0 & -3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} x \end{cases}$$

试求：(1) 全维观测器，观测器极点为 $-5, -5, -5$

(2) 降维观测器，观测器极点为 $-5, -5$

(3) 系统模拟结构图

上机练习题

5.2已知系统状态方程为

$$\dot{x}=\begin{bmatrix}2 & 1 & 0 & 0\\0 & 2 & 0 & 0\\0 & 0 & -1 & 0\\0 & 0 & 0 & -1\end{bmatrix}x+\begin{bmatrix}0\\1\\1\\1\end{bmatrix}u$$

试用MATLAB求解状态反馈阵，使系统闭环极点为-2，-2，-2和-1

上机练习题

5.3试用MATLAB判断以下系统状态反馈能否镇定

$$1)\dot{x}=\begin{bmatrix}-1 & -2 & -2\\ 0 & -1 & 1\\ 1 & 0 & -1\end{bmatrix}x+\begin{bmatrix}2\\0\\1\end{bmatrix}u$$

$$2)\dot{x}=\begin{bmatrix}-2 & 1 & 0 & 0 & 0\\ 0 & -2 & 1 & 0 & 0\\ 0 & 0 & -2 & 0 & 0\\ 0 & 0 & 0 & -5 & 1\\ 0 & 0 & 0 & 0 & -5\end{bmatrix}x+\begin{bmatrix}4\\5\\0\\7\\0\end{bmatrix}u$$

上机练习题

5.4已知如下系统

$$\begin{cases} \dot{\boldsymbol{x}} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 2 & -3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 4 & -1 & 2 & -4 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \end{bmatrix} \boldsymbol{u} \\ \boldsymbol{y} = \begin{bmatrix} 3 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x} \end{cases}$$

试用MATLAB求：

1）系统的极点，判断系统是否稳定；

2）判断系统的能控性和能观性，若不能，请按照能控性和能观性进行结构分解。

3）判断系统是否通过状态反馈镇定。

4）设计状态反馈阵，使得闭环系统极点为-5，-1和-2±2j

5）设计状态观测器，使得观测器极点为-3，-4和-1±j