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

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^{At}
调用格式：$eAt = \mathrm{expm}(A*t)$
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$(sI-A)^{-1}$，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$\boldsymbol{A}=\begin{bmatrix}-3 & 1\\ 1 & -3\end{bmatrix}$，应用Matlab求解状态转移矩阵$e^{\boldsymbol{A}t}$。

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：

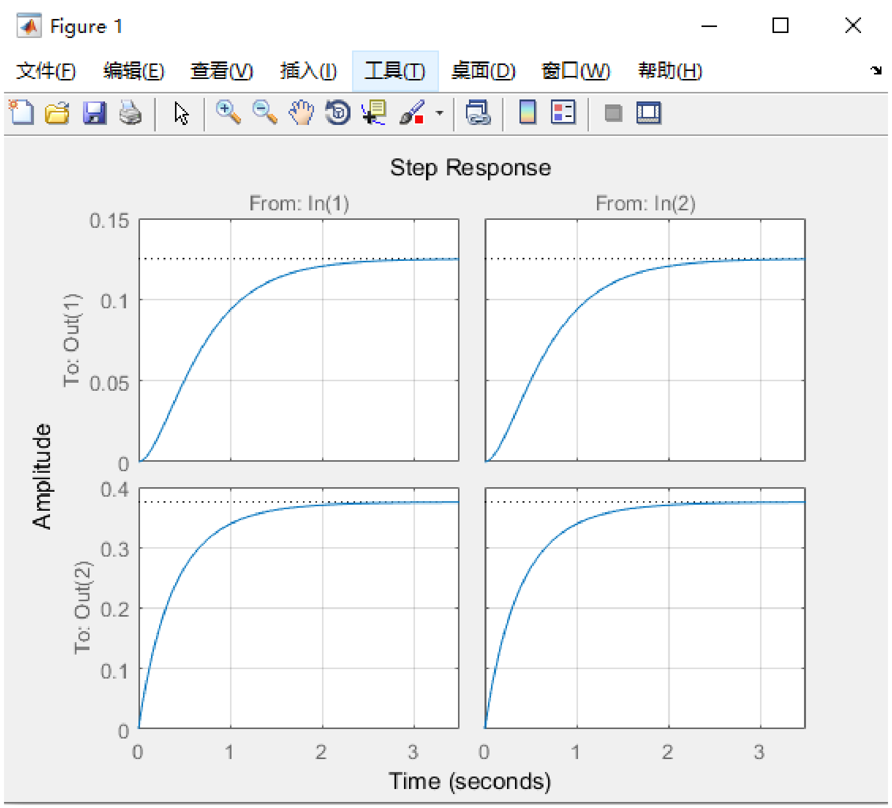
（1）输入$u_1(t)=1(t)$, $u_2(t)=1(t)$单独作用下的系统输出响应；

（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
```

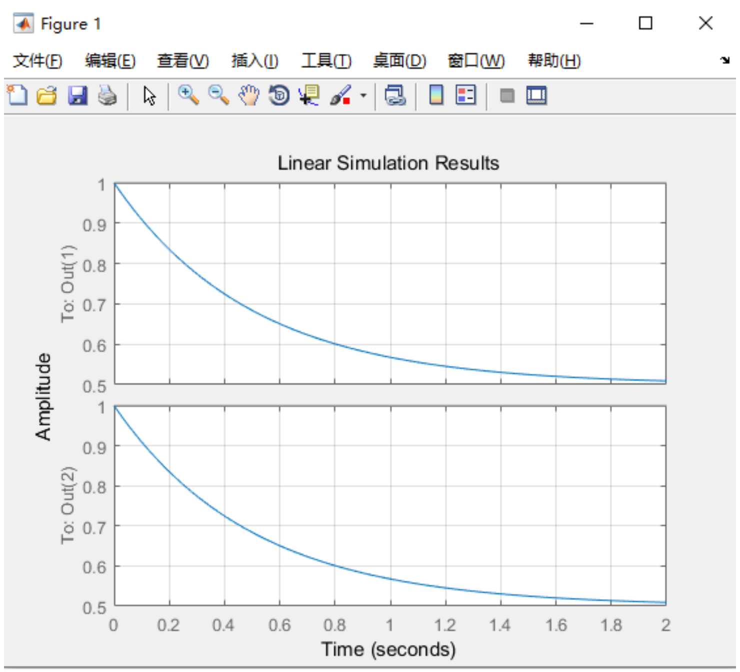


运行结果

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

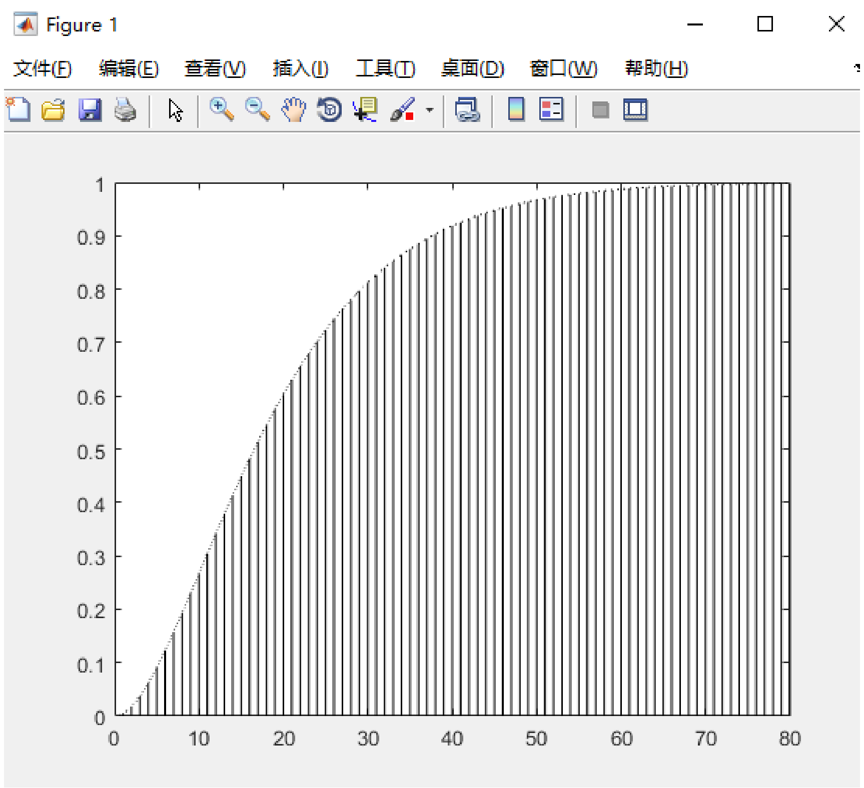
【例5】 对于例2-13所示的离散状态方程，试用Matlab求解当T=0.1s，输入为单位阶跃函数，且初始状态为零状态时的离散输出y(kT)。

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

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

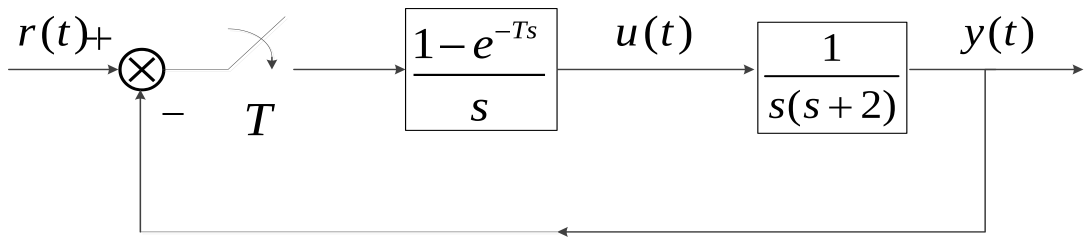
功能：进行线性定常连续系统状态方程的离散化求解

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

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)