用ode45求解微分方程数值解

朱芳来

考虑SISO线性系统：

$$\begin{aligned}\dot{x}(t) &= Ax(t) + bu(t) \\ y(t) &= cx(t)\end{aligned} \tag{1}$$

其中

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 4 & 0 \end{bmatrix},\ b = \begin{bmatrix} 0 \\ 1 \\ 0 \\ -1 \end{bmatrix},\ c = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

可以验证，该系统是完全可控和完全可观测器的系统。设计基于观测器的状态反馈控制器，使得闭环系统在平衡状态渐近收敛稳定。

第一步：设计观测器

观测器设计为

$$\dot{\hat{x}}(t) = A\hat{x}(t) + bu(t) + l\big(y(t) - c\hat{x}(t)\big) \tag{2}$$

观测器误差动态方程为

$$\dot{e}(t) = (A - lc)e(t) \tag{3}$$

其中 $e(t) = x(t) - \hat{x}(t)$. 要求选取观测器增益矩阵 $l = [l_1 \quad l_2 \quad l_3 \quad l_4]^T$ 使得观测器(3)的系统矩阵A-lc的极点配置到 $a_1 = [-1 \quad -2 \quad -3 \quad -4]^T$.

Matlab 代码为

```
a1=[-1  -2  -3  -4]';
l = (place(A',c', a1))';
```

给出的输出是 $l=[10 \quad 39 \quad -45 \quad -90]^T$

第二步：状态反馈控制器

整体反馈控制器为$u(t)$=-$kx(t)$,要求设计状态反馈增益矩阵$k=[k_1\ k_2\ k_3\ k_4]^T$ 使得闭环习题(既将$u(t)$=-$kx(t)$代入(1)后的系统)的系统矩阵A-bk的极大配置到 $a_2=[-1 \quad -1-i \quad -1+i \quad -2]^T$. Matlab代码为

```
a2=[-1  -1-i  -1+i  -2]';
k = place(A,b, a1));
```

给出的答案是 $k=\begin{bmatrix}-2 & -5 & -16 & -10\end{bmatrix}$.

第三步：设计基于观测器的状态反馈：

$$u(t)=-k\hat{x}(t) \tag{4}$$

闭环系统为

$$\dot{x}(t)=Ax(t)-bk\hat{x}(t) \tag{5}$$

则有 $\lim\limits_{t\to\infty}x(t)=0.$

所以，为了给出仿真结果，我们需要求解8阶的微分方程：系统(1)的原系统(4维，在控制器(4)下)和观测器(2)(4维，在控制器(4)下)，系统(1)和观测器(2)是一个整体，由输出$y(t)$联系，所以总系统是8维系统。

定义8阶系统的m文件

```
function dx=linearobcontr(t,xx)
A=[0 1 0 0;0 0 -2 0;0 0 0 1;0 0 4 0];
b=[0 1 0 -1]';
c=[1 0 0 0];
a1=[-1 -2 -3 -4]';
a2=[-1 -1-i -1+i -2]';
l = (place(A',c',a1))';
k = place(A,b,a2);

x=xx(1:4);      % define the original state vector
ex=xx(5:8);     % define the observer state vector
y=c*x;          % define the output
u=-k*ex;        % state feedback using the estimated state
dx1=A*x+b*u;    % define the equation of closed-loop system under the controller of u=k*ex
dx2=A*ex+b*u+l*(y-c*ex);    % define the equation of observer under the controller of u=k*ex
dx=[dx1;dx2];   % define the equation of state feedback based on the observer.
```

用ode45求解8阶微分方程

```
function ODElinearobcontr
h=0.01;
N=20;
t=0:h:N;
xx0=[0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8]';
A=[0 1 0 0;0 0 -2 0;0 0 0 1;0 0 4 0];
b=[0 1 0 -1]';
c=[1 0 0 0];
[t, xx]=ode45(@linearobcontr,t,xx0);
t=t';
xx=xx';
x=xx(1:4,:);     % the numerical solution of the original state x(t).
ex=xx(5:8,:);    % the numerical solution of the etimated state xhat(t) given by the observer.
y=c*x;           % the numerical solution of the output y(t).
ey=c*ex;         % the numerical solution of the extimated output y(t).

figure;
plot(t,x(1,:),'-',t,ex(1,:),'--');        % plot the curvs of x1(t) and xhat1(t).
legend('x1(t)','estimated x1(t)');
```

```
figure;
plot(t,x(2,:),'-',t,ex(2,:),'--');          % plot the curvs of x2(t) and xhat2(t).
legend('x2(t)','estimated x2(t)');

figure;
plot(t,x(3,:),'-',t,ex(3,:),'--');          % plot the curvs of x3(t) and xhat3(t).
legend('x3(t)','estimated x3(t)');

figure;
plot(t,x(4,:),'-',t,ex(4,:),'--');          % plot the curvs of x4(t) and xhat4(t).
legend('x4(t)','estimated x4(t)');

figure;
plot(t,y,'-',t,ey,'--');                    % plot the curvs of y(t) and yhat4(t).legend('y(t)','estimated y(t)');
legend('y(t)','estimated y(t)');
```

图像输出

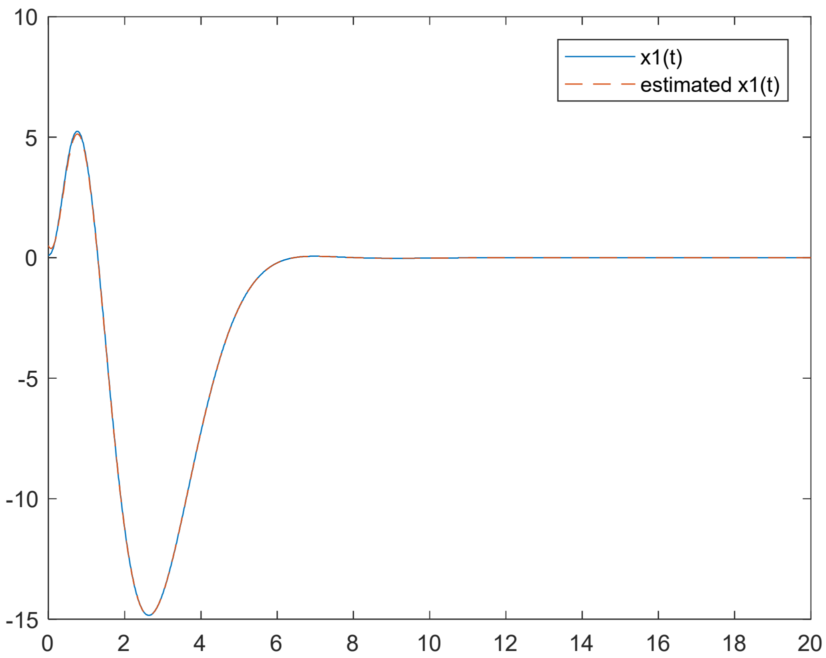

图1：状态$x_1(t)$及其估计曲线

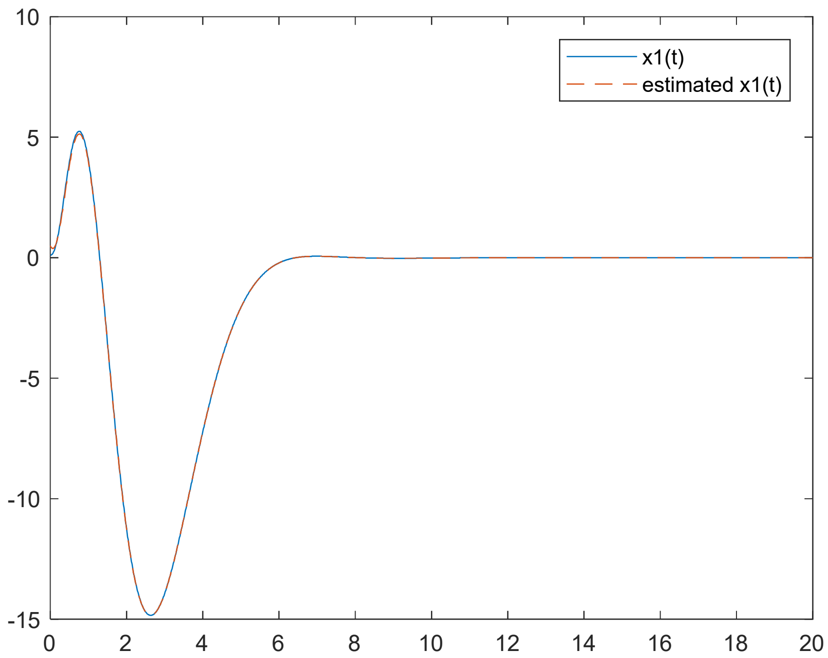

图2：状态$x_2(t)$及其估计曲线

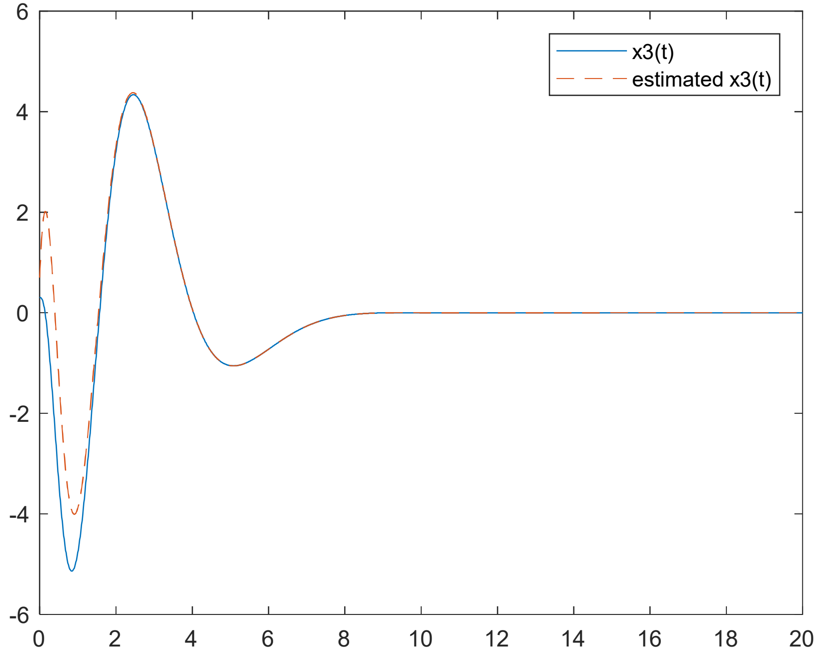

图3：状态$x_3(t)$及其估计曲线

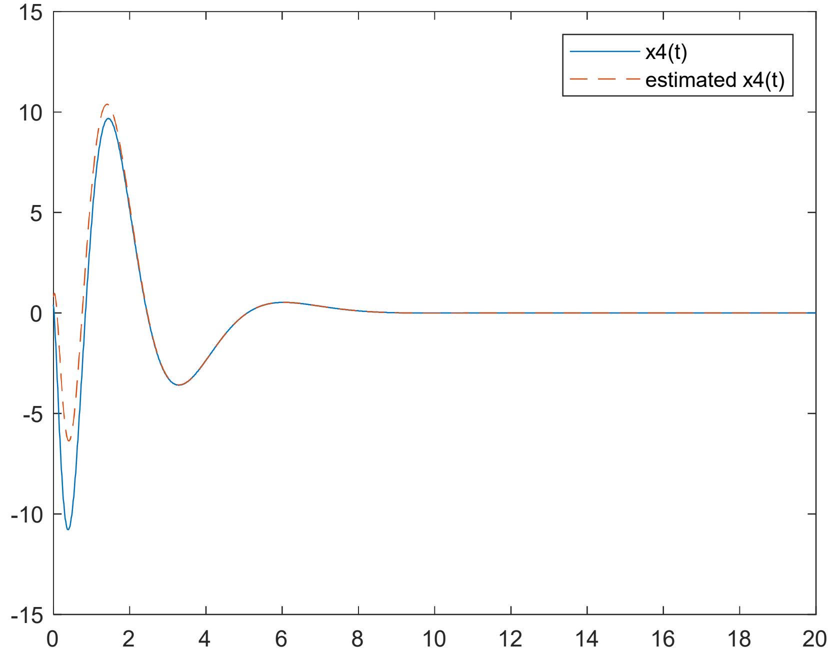

图4：状态$x_4(t)$及其估计曲线

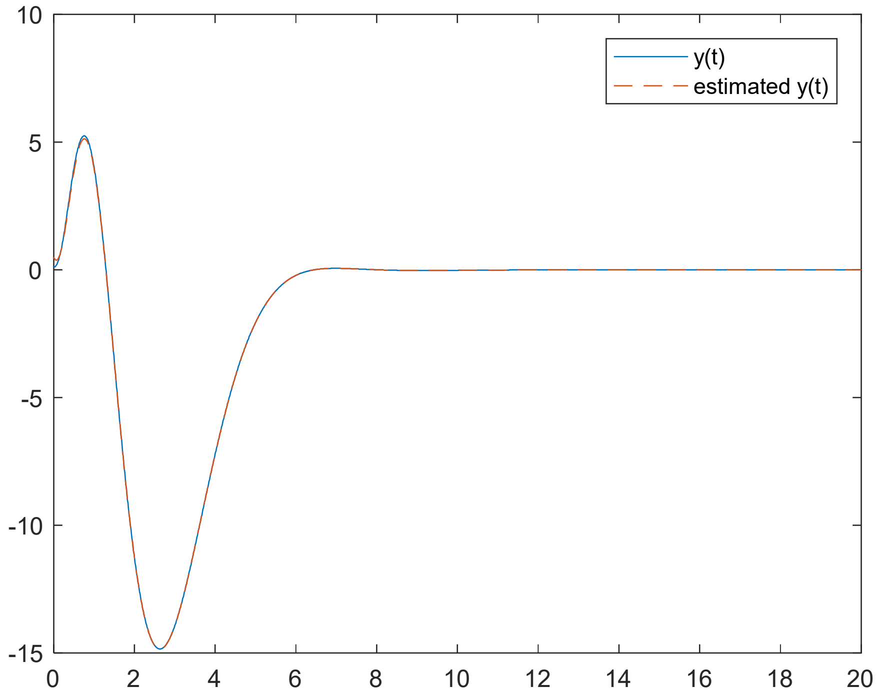

图5：输出$y(t)$及其估计曲线