Laboratório 9

SEL0359 - Controle Digital - 2° Semestre de 2024

Prof. Marcos R. Fernandes

Espaço de estados - parte 2

Entrega:[1] entregue um arquivo PDF com os gráficos e código fonte utilizado para cada questão além da resposta para as perguntas dos enunciados. Não esqueça de colocar título na figura para identificar o que cada figura representa, nome dos eixos, e legenda.

O objetivo dessa aula de laboratório é explorar recursos computacionais do Matlab para projeto de controladores de realimentação de estados em tempo discreto.

Dica: Confira sempre o help do matlab para cada comando antes de utiliza-lo!

- poly: obtem coeficientes do polinômio para dado conjunto de raízes;
- polyvalm: avalia polinômio matricial;
- acker: calcula ganho de alocação de pólos usando equação de Ackermann para sistemas SISO;
- place: calcula ganho de alocação de pólos (sem multiplicidade) para sistemas MIMO;
- ctrb: constrói matriz de controlabilidade;
- obsv: constrói matriz de observabilidade;

1 Atividade 1

Nessa atividade, o objetivo explorar os recursos computacionais do Matlab para avaliar controlabilidade e observabilidade de modelos em espaço de estados em tempo discreto.

Para isso, considere o modelo em espaço de estados a seguir

$$x_{k+1} = \begin{bmatrix} 3 & -1 \\ 0 & 2 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 2 \end{bmatrix} u_k$$
$$y_k = \begin{bmatrix} 0 & 1 \end{bmatrix} x_k$$

O código Matlab a seguir avalia o rank da matriz de controlabilidade e observabilidade do sistema.

```
close all
clear all
```

[1] Última atualização: 31/10/2024

```
clc
%%
A=[3 -1;0 2];
B=[1;2];
H=[0 1];
%% controlabilidade
C=ctrb(A,B)
disp(['rank(C)=' num2str(rank(C))])
if rank(C)==size(A,1)
    disp('Sistema controlavel')
else
    disp('Sistema parcialmente controlavel')
end
%% observabilidade
O=obsv(A,H)
disp(['rank(O)=' num2str(rank(O))])
if rank(O)==size(A,1)
    disp('Sistema observavel')
else
    disp('Sistema parcialmente observavel')
end
```

Usando a mesma lógica, avalie a controlabilidade e observabilidade do sistema:

$$x_{k+1} = \begin{bmatrix} 3 & 0 & 0 \\ 5 & 4 & 0 \\ 1 & 2 & 3 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 2 \\ 5 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} x_k$$

Caso seja não controlável, proponha uma modificação estrutural no sistema para torna-lo controlável. Da mesma forma, caso seja não-observável, proponha uma modificação estrutural do sistema para torná-lo observável.

2 Atividade 2

Nessa atividade, o objetivo explorar os recursos computacionais do Matlab para projetar controladores por realimentação de estados em tempo discreto.

Para isso, considere o seguinte sistema como exemplo:

$$x_{k+1} = \begin{bmatrix} 1 & -2 \\ 2 & 1 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u_k$$

Suponha uma lei de controle na forma de realimentação de estados:

$$u_k = -Kx_k.$$

Deseja-se que o sistema em malha fechada tenha os pólos em posições especificadas. Ou seja, o polinômio característico do sistema em malha-fechada deve satisfazer

$$\det(zI - (A - BK)) = (z - p_1)(z - p_2)\cdots(z - p_n)$$

em que $p_1, p_2, \ldots, p_n$ são os pólos desejados em malha fechada.

Se o sistema for completamente controlável então existe um ganho K que faz com que os pólos em malha fechada fiquem nas posições desejadas!

Para sistemas SISO, pode-se usar a equação de Ackermann para obter o ganho de realimentação:

$$K = \begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} \mathcal{C}^{-1}\gamma(A),$$

em que $\gamma(A)$ é a equação característica desejada em malha fechada avaliada na própria matriz de dinâmica.

No Matlab, pode-se utilizar o comando **acker**. Já para sistemas MIMO, sem pólos com multiplicidade, pode-se usar o comando **place**.

O código Matlab a seguir obtêm o ganho de realimentação de estados para alocar os pólos do sistema em $p_1 = 0.8$ e $p_2 = 0.9$ usando a equação de Ackermann, o comando **acker** e o comando **place**.

```
close all
clear all
clc
%%
Ad=[1 -2;2 1];
Bd=[1;1];
rank(ctrb(Ad,Bd))
beta=poly([0.8 0.9]) %coef. da eq. caracteristica
%% Eq. Ackermann (SISO)
K=[0 1]*inv(ctrb(Ad,Bd))*polyvalm(beta,Ad)
%% Comando do matlab acker
K=acker(Ad,Bd,[0.8 0.9])
%% Comando do matlab place (MIMO)
K=place(Ad,Bd,[0.8 0.9])
%% verifica autovalores da dinamica em malha fechada
eig(Ad-Bd*K)
```

Usando a mesma lógica, obtenha o ganho de realimentação de estados que aloca os

pólos de malha fechada em $p_1 = 0.5, p_2 = 0.8, p_3 = 0.7$ para o sistema a seguir:

$$x_{k+1} = \begin{bmatrix} 0.75 & -0.25 & 4 \\ 2.25 & -0.75 & 0 \\ -0.375 & 0.125 & 3.5 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_k$$

3 Atividade 3

Nessa atividade, o objetivo explorar os recursos computacionais do Matlab para obter e simular observadores de estados em tempo discreto.

Considere o sistema dado por

$$x_{k+1} = \begin{bmatrix} 0.7143 & 0.3571 \\ -1.0714 & -0.3571 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u_k$$
$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k$$

Uma estratégia para obter o ganho de realimentação para um observador de estados é tratar o problema de observação como dual ao problema de controle!

Assim, pode-se usar o mesmo processo de cálculo do ganho de realimentação de estados para calcular o ganho do observador.

Para isso, adota-se o modelo dual do sistema fazendo as seguintes equivalências:

$$A_d \rightarrow A_d^{\mathsf{T}} \tag{1}$$
$$B_d \rightarrow H^{\mathsf{T}} \tag{2}$$

O seguinte código em Matlab obtêm um observador de estados para esse sistema e faz a simulação com uma entrada senoidal.

```
close all
clear all
clc
%%
Ad=[0.7143    0.3571;...
   -1.0714   -0.3571];
Bd=[1;1];
H=[1 0];
rank(obsv(Ad,H))
L=place(Ad',H',[0.5 0.6])'
eig(Ad-L*H)
%%
N=100;
x=zeros(2,N);
```

```
hx=zeros(2,N);
y=zeros(1,N);
u=zeros(1,N);
%% cond. inicial real
x(:,1)=[10 10]';
%% cond. inicial do observador
hx(:,1)=[20 -20]';
for k=1:N
 %% sistema real
 u(k)=sin(k);
 x(:,k+1)=Ad*x(:,k)+Bd*u(k);
 y(k)=H*x(:,k);
 %% observador
 hx(:,k+1)=Ad*hx(:,k)+Bd*u(k)+L*(y(k)-H*hx(:,k));
end
%%
figure
subplot(2,1,1)
stairs(x(1,:),'LineWidth',1.5)
hold on
stairs(hx(1,:),'-','LineWidth',1)
xlim([0 50])
grid on
legend('x_1','hx_1')
subplot(2,1,2)
stairs(x(2,:),'LineWidth',1.5)
hold on
stairs(hx(2,:),'-','LineWidth',1)
xlim([0 50])
grid on
legend('x_2','hx_2')
suptitle('Observador de Estados')
```

Seguindo a mesma lógica, obtenha um observador de estados e simule o sistema a seguir com entrada senoidal. Considere $x_0 = [10\ 15\ \ -5]^\intercal$ e $\hat{x}_0 = [0\ 0\ 0]^\intercal$. Escolha os pólos do observador em malha fechada de forma que ele seja mais rápido que o sistema original.

$$x_{k+1} = \begin{bmatrix} -0.5 & 0.25 & 0 \\ 0 & -0.2500 & 0 \\ 0 & 0 & -0.7500 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} u_k$$

$$y_k = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} x_k$$

4 Atividade 4

Nessa atividade, o objetivo é projetar um controlador de realimentação de estados usando um observador de estados.

Nesse exemplo, será utilizado o modelo do *self-balanced robot* conforme ilustrado na Figura 1. As EDO que descrevem o comportamento dessa planta são dadas a seguir:

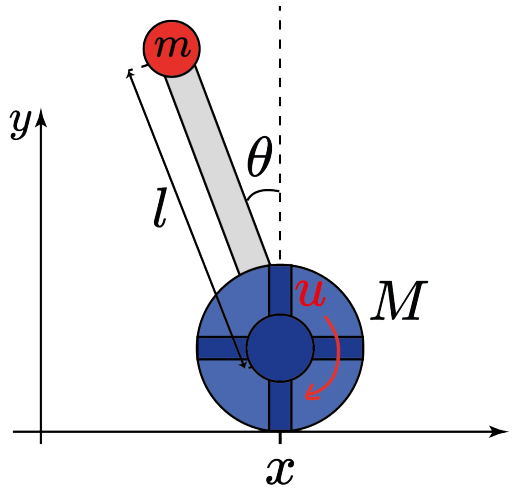

Figura 1: Self-balanced robot.

$$(M+m)\ddot{x} - ml\cos(\theta)\ddot{\theta} + ml\dot{\theta}^2\sin(\theta) + \alpha_1\dot{x} = u, \tag{3}$$

$$l^2 m\ddot{\theta} - ml\cos(\theta)\ddot{x} - gml\sin(\theta) + \alpha_2\dot{\theta} = 0. \tag{4}$$

o vetor de estados é dado por

$$x = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix}, \quad \begin{cases} x \rightarrow \text{posição} \\ \dot{x} \rightarrow \text{velocidade} \\ \theta \rightarrow \text{posição angular} \\ \dot{\theta} \rightarrow \text{velocidade angular} \end{cases}$$

e o sinal de entrada é o torque que atua na roda. Suponha que esteja disponível medições da posição x e da posição angular θ. Assim, a equação de saída desse sistema é dada por

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

Adotando,

$$x_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad u_0 = 0,$$

como ponto de operação do sistema (posição vertical com velocidade zero), pode-se obter um modelo linearizado na forma

$$\dot{x} = A(x_0, u_0)x + B(x_0, u_0)u,$$

válido para regiões próximas do ponto (x_0, u_0). Após obter o modelo linearizado, pode-se utilizar o comando **c2d** para obter o equivalente em tempo discreto.

O código Matlab a seguir obtém as matrizes $A(x_0, u_0)$ e $B(x_0, u_0)$ do modelo linearizado para esse sistema e as respectivas matrizes em tempo discreto para $T_s = 0.1s$.

```
close all
clear all
clc
%%
syms m M l u g alpha1 alpha2
syms th th_d th_dd
syms x x_d x_dd

eq1=(M+m)*x_dd-m*l*th_dd*cos(th)+m*l*th_d^2*sin(th)+alpha1*x_d-u;
eq2=l^2*m*th_dd-x_dd*m*l*cos(th)-g*m*l*sin(th)+alpha2*th_d;
S=solve(eq1==0, eq2==0, x_dd, th_dd)
%%
x_vet=[x;x_d;th;th_d];
x_vet_dot=[x_d; S.x_dd; th_d; S.th_dd];

A=simplify(jacobian(x_vet_dot,x_vet))
B=simplify(jacobian(x_vet_dot,u))

%% ponto de operacao
x=0;
x_d=0;
th=0;
th_d=0;
u=0;
%%
A0=simplify(subs(A)) %matriz dinamica
B0=simplify(subs(B)) %matriz de entrada
```

```
%%
M=1.5;
m=0.5;
l=1;
g=9.81;
alpha1=0.01;
alpha2=0.01;
A0=double(simplify(subs(A))) %matriz dinamica
B0=double(simplify(subs(B))) %matriz de entrada
%%
Ts=0.1; %tempo de amostragem
[Ad,Bd]=c2d(A0,B0,Ts)
```

Considere a condição inicial do sistema real como $x_0 = [2\ 0\ \frac{10\pi}{180}\ 0]^\intercal$.

1. Avalie a controlabilidade e a observabilidade do sistema. Ele é completamente controlável e observável no ponto de operação escolhido?

2. Projete um controle de realimentação de estados de forma a trazer o sistema para a origem ($x_k = [0\ 0\ 0\ 0]^\intercal$) em menos de 5s. Assuma que os estados são acessíveis diretamente (não tem necessidade de um observador).

3. Considere agora que apenas medições de posição x e de posição angular θ estão disponíveis e então utilize um observador de estados para fazer o controle do item anterior.

4. Ao final, apresente um comparativo dos estados verdadeiros do sistemas e os estados estimados pelo observador e também um comparativo do controle com acesso direto aos estados e do controle usando os estados do observador. Comente com suas palavras as diferenças que são percebidas quando utiliza-se o observador de estado para fazer o controle.

Dica: Use o código de exemplo disponível no e-disciplina (arquivo **self-balanced-robot.zip**) como ponto de partida (script controle-self-balanced-robot2.m). Defina os pólos de malha fechada tanto para o sistema quanto para o observador de estados. Lembre-se que os pólos do observador precisam ser mais rápidos que os pólos do sistema em malha fechada. Para obter os ganhos de realimentação pode-se utilizar o comando **place**.

5 Atividade 5

Nessa atividade, o objetivo é projetar um controlador com rastreamento para modelos em espaço de estados em tempo discreto.

Nesse exemplo, será utilizado o modelo simplificado de um *quadricóptero* restrito a movimentos em um plano conforme ilustrado na Figura 2. As EDO que descrevem o compor-

Figura 2: Quadricóptero.

tamento dessa planta são dadas a seguir:

$$\ddot{z} = \frac{1}{M}(F_1 + F_2)\cos(\theta) - g \tag{5}$$

$$\ddot{y} = \frac{1}{M}(F_1 + F_2)\sin(\theta) \tag{6}$$

$$\ddot{\theta} = \frac{l}{I_{xx}}(F_1 - F_2) \tag{7}$$

o vetor de estados é dado por

$$x = \begin{bmatrix} z \\ y \\ \dot{z} \\ \dot{y} \\ \theta \\ \dot{\theta} \end{bmatrix}, \quad \begin{cases} z \rightarrow \text{posição na direção z} \\ x \rightarrow \text{posição na direção y} \\ \dot{z} \rightarrow \text{velocidade na direção z} \\ \dot{y} \rightarrow \text{velocidade na direção y} \\ \theta \rightarrow \text{posição angular (roll)} \\ \dot{\theta} \rightarrow \text{velocidade angular} \end{cases}$$

e o sinal de entrada são as forças de empuxo F_1 e F_2 geradas pelos motores:

$$u = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$$

Adotando,

$$x_0 = \begin{bmatrix} z_{\text{ref}} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad u_0 = \begin{bmatrix} \frac{1}{2}gM \\ \frac{1}{2}gM \end{bmatrix},$$

como ponto de operação do sistema (altura z_{ref} com velocidade zero e roll zero), pode-se obter um modelo linearizado na forma

$$\delta\dot{x} = A(x_0, u_0)\delta x + B(x_0, u_0)\delta u,$$

válido para regiões próximas do ponto (x_0, u_0). Após obter o modelo linearizado, pode-se utilizar o comando **c2d** para obter o equivalente em tempo discreto.

O código Matlab a seguir obtém as matrizes $A(x_0, u_0)$ e $B(x_0, u_0)$ do modelo linearizado para esse sistema e as respectivas matrizes em tempo discreto para $T_s = 0.1s$.

```
close all
clear all
clc
%%
syms M F1 F2 th g l Ixx
syms th th_d
syms z z_d y_d y
l=0.3;
M=0.5;
Ixx=0.1;
g=9.81;
z_dd=1/M*(F1+F2)*cos(th)-g
y_dd=1/M*(F1+F2)*sin(th)
th_dd=1/Ixx*(F1-F2)
%%
x_vet=[z;y;z_d;y_d;th;th_d];
x_vet_dot=[z_d;y_d;z_dd;y_dd;th_d;th_dd];
u=[F1;F2];
A=simplify(jacobian(x_vet_dot,x_vet))
B=simplify(jacobian(x_vet_dot,u))
%%
ref=[5;0;0;0;0;0];
%% ponto de operacao
z=ref(1);
y=ref(2);
```

```
th=ref(5);
F1=1/2*g*M;
F2=F1;
x0=[z;y;0;0;th;0];
u0=[F1;F2];
A0=double(subs(A)); %matriz dinamica
B0=double(subs(B)); %matriz de entrada
%%
Ts=0.1; %tempo de amostragem
[Ad,Bd]=c2d(A0,B0,Ts);
```

Considere a condição inicial do sistema real como $x_0 = [0\ 0\ 0\ 0\ 0\ 0]^\intercal$.

1. Avalie a controlabilidade e a observabilidade do sistema linearizado.

2. Verifique se o sistema atende a condição para rastreamento. Ou seja, verifique se a matriz a seguir é rank completo.

$$\begin{bmatrix} I - A_d & -B_d \\ H & 0 \end{bmatrix}.$$

3. Assuma que os estados do sistema estão disponíveis e então projete um controlador de realimentação de estados que estabilize o quadricóptero na posição $z = 5$ e $y = 0$ durante 5s e depois leve o quadricóptero para $z = 3$. Use o código exemplo disponível no site e-disciplina **controle-quadcopter2d.zip** como ponto de partida.