Versuch II: Linearisierung, Steuerbarkeit und Beobachtbarkeit

Andreas Jentsch, Ali Kerem Sacakli

Praktikumsbericht – Praktikum Matlab/Simulink II

TECHNISCHE UNIVERSITÄT DARMSTADT

REGELUNGSTECHNIK UND MECHATRONIK rtm

2.1 Linearisierung

Im folgenden Abschnitt wird die Funktion zur Linearisierung des Doppelpendel-Systems um einen Arbeitspunkt, sowie ihre Rückgabewerte an bestimmten Arbeitspunkten dokumentiert. Die Implementierung der Funktion ist in Listing 2.2 aufgeführt.

Bevor das System linearisiert wird sind zwei Fragen zu klären:

1. Welche Arbeitspunkte sind sinnvoll?
2. Was bedeutet es physikalisch, wenn M_{AP} ungleich null ist?

Die Antworten lauten wie folgt:

1. Es ist nur sinnvoll das System in Arbeitspunkten zu linearisieren, in denen es sowohl vollständig beobachtbar, als auch steuerbar ist.
2. Bei der Größe M_{AP} handelt es sich um den statischen Wert der Stellgröße M im Arbeitspunkt. Ist diese ungleich null muss der Motor das Moment M

In dieser Aufgabe soll eine Funktion `[A,B,C,D]=linearisierung(f,h,AP)` implementiert werden. Unter Vorgabe der Funktionen `f`, `h` und des Arbeitspunktes `AP` sollen die Matrizen der linearisierten Gleichungen in Zustandsraumdarstellung ausgegeben werden.

Listing 2.1: Das vorgegebene nichtlineare Modell

```
function [f, h] = nonlinear_model()
 syms phi1 phi2 dphi1 dphi2 ddphi1 ddphi2 M
    f = [dphi1;
        (375*((9*cos(phi1 - phi2)*sin(phi1 - phi2)*dphi1^2)/1250 ...
        + (3*sin(phi1 - phi2)*dphi2^2)/625 - (4*M)/5 + (8829*sin(phi1))/12500 ...
        - (8829*cos(phi1 - phi2)*sin(phi2))/25000))/((27*cos(phi1 - phi2)^2)/10 - 24/5);
        dphi2;
        -(1250*((18*sin(phi1 - phi2)*dphi1^2)/3125 ...
        + (27*cos(phi1 - phi2)*sin(phi1 - phi2)*dphi2^2)/12500 ...
        - (8829*sin(phi2))/31250 + (79461*cos(phi1 - phi2)*sin(phi1))/250000 ...
        - (9*M*cos(phi1 - phi2))/25))/((27*cos(phi1 - phi2)^2)/10 - 24/5)];

     h = [phi1;
          phi2];
end
```

Listing 2.2: Code der Linearisierungsfunktion

```
function [ A, B, C, D] = linearisierung( f, h, AP )

syms phi1 phi2 dphi1 dphi2 ddphi1 ddphi2 M;

x = [phi1;dphi1;phi2;dphi2];
u = M;

f_M_AP = subs(f(2),x,AP);

M_AP = solve(f_M_AP == 0 , M);

A = jacobian(f,x);
B = jacobian(f,u);
C = jacobian(h,x);
D = jacobian(h,u);

A = subs(A,[x,u],[AP,M_AP]);
B = subs(B,[x,u],[AP,M_AP]);
C = subs(C,[x,u],[AP,M_AP]);
D = subs(D,[x,u],[AP,M_AP]);



A = double(A);
B = double(B);
C = double(C);
D = double(D);

end
```

Die Linearisierung um die Arbeitspunkte

$$\mathbf{x}_{\mathrm{AP}_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$$
$$\mathbf{x}_{\mathrm{AP}_2} = \begin{bmatrix} \pi & 0 & \pi & 0 \end{bmatrix}$$
$$\mathbf{x}_{\mathrm{AP}_3} = \begin{bmatrix} \pi/2 & 0 & \pi & 0 \end{bmatrix}$$

ergibt für die allgemeine Zustandsraumdarstellung:

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}$$
$$\mathbf{y} = \mathbf{Cx} + \mathbf{Du}$$

die folgenden Systemmatrizen:

- $\vec{x}_{AP1} = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$
  - $\mathbf{A_{AP1}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -126{,}1286 & 0 & 63{,}0643 & 0 \\ 0 & 0 & 0 & 1 \\ 189{,}1929 & 0 & -168{,}1714 & 0 \end{pmatrix}$
  - $\mathbf{B_{AP1}} = \begin{pmatrix} 0 \\ 142{,}8571 \\ 0 \\ -214{,}2857 \end{pmatrix}$
  - $\mathbf{C_{AP1}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
  - $\mathbf{D_{AP1}} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$
- $\vec{x}_{AP2} = \begin{pmatrix} \pi & 0 & \pi & 0 \end{pmatrix}$
  - $\mathbf{A_{AP2}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 126.1286 & 0 & -63{,}0643 & 0 \\ 0 & 0 & 0 & 1 \\ -189{,}1929 & 0 & 168{,}1714 & 0 \end{pmatrix}$
  - $\mathbf{B_{AP2}} = \begin{pmatrix} 0 \\ 142{,}8571 \\ 0 \\ -214{,}2857 \end{pmatrix}$
  - $\mathbf{C_{AP2}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
  - $\mathbf{D_{AP2}} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$
- $\vec{x}_{AP3} = \begin{pmatrix} \pi/2 & 0 & \pi & 0 \end{pmatrix}$
  - $\mathbf{A_{AP3}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 73{,}575 & 0 \end{pmatrix}$
  - $\mathbf{B_{AP3}} = \begin{pmatrix} 0 \\ 62{,}5 \\ 0 \\ 0 \end{pmatrix}$

- $\mathbf{C_{AP3}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
- $\mathbf{D_{AP3}} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Was bedeutet es physikalisch, wenn M_{AP} ungleich null ist?

(TEXT)

2.2 Vergleich der linearisierten Modelle

Die Eigenwerte der Zustandsraummodelle um die drei Arbeitspunkte lauten wie folgt:

$\vec{x}_{AP1} = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$:

$$\lambda_1 = \begin{pmatrix} 16{,}0744i \\ -16{,}0744i \\ 5{,}9929i \\ -5{,}9929i \end{pmatrix}$$

$\vec{x}_{AP2} = \begin{pmatrix} \pi & 0 & \pi & 0 \end{pmatrix}$:

$$\lambda_2 = \begin{pmatrix} -16{,}0744 \\ 16{,}0744 \\ 5{,}9929 \\ -5{,}9929 \end{pmatrix}$$

$\vec{x}_{AP_3} = \begin{pmatrix} \pi/2 & 0 & \pi & 0 \end{pmatrix}$:

$$\lambda_3 = \begin{pmatrix} -8{,}5776 \\ 8{,}5776 \\ 0 \\ 0 \end{pmatrix}$$

Welche Unterschiede zwischen den Eigenwerte der Zustandsraummodelle liegen vor und worauf sind diese zurückzuführen? Können Sie sich vorstellen, was sich ändern würde, wenn die Reibung berücksichtigt wäre?

(TEXT)

2.3 Normalformen des Zustandsraummodelles

Es soll eine Funktion `[AD,BD,CD,DD]=diagonalForm(A,B,C,D)` implementiert werden, die ein gegebenes System in Diagonalform transformiert.

Listing 2.3: Code der Diagonalisierungsfunktion

```
function [ AD, BD, CD, DD ] = diagonalForm( A, B, C, D )

[V, Deig] = eig(A);

if rank(V)==length(V)
    V_inv = inv(V);

    AD = Deig;
    BD = V_inv*B;
    CD = C*V;
    DD = D;
else
    disp('Matrix␣A␣ist␣nicht␣diagonalähnlich!')
end

end
```

Zusätzlich soll das System im zweiten Arbeitspunkt anhand dieser Funktion und mit der Funktion canon auf Diagonalform transformiert werden.

Die Ergebnisse der Funktion `diagonalForm` sind hier wie folgt:

- $\mathbf{A2D} = \begin{pmatrix} -16{,}0744 & 0 & 0 & 0 \\ 0 & 16{,}0744 & 0 & 0 \\ 0 & 0 & -5{,}9929 & 0 \\ 0 & 0 & 0 & 5{,}9929 \end{pmatrix}$
- $\mathbf{B_{AP3}} = \begin{pmatrix} 138{,}1293 \\ 138{,}1293 \\ -21{,}3960 \\ 21{,}3960 \end{pmatrix}$
- $\mathbf{C_{AP3}} = \begin{pmatrix} -0{,}0267 & 0{,}0267 & 0{,}0943 & 0{,}0943 \\ 0{,}0560 & -0{,}0560 & 0{,}1349 & 0{,}1349 \end{pmatrix}$
- $\mathbf{D_{AP3}} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Die Diagonalisierung mittels der Funktion canon führt zu folgendem Ergebnis:

- $\mathbf{A2D} = \begin{pmatrix} -16{,}07 & 0 & 0 & 0 \\ 0 & -5{,}993 & 0 & 0 \\ 0 & 0 & 16{,}07 & 0 \\ 0 & 0 & 0 & 5{,}993 \end{pmatrix}$

- **B2D** $= \begin{pmatrix} 2{,}535 \\ -0{,}7024 \\ 2{,}535 \\ 0{,}7024 \end{pmatrix}$
- **C2D** $= \begin{pmatrix} -1{,}456 & 2{,}873 & 1{,}456 & 2{,}873 \\ 3{,}053 & 4{,}109 & -3{,}053 & 4{,}109 \end{pmatrix}$
- **D2D** $= \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Welche Unterschiede sind zu erkennen? Wie lassen sie sich erklären?

(TEXT)

Als nächstes soll das System im ersten Arbeitspunkt auf Modalform transformiert werden. Das in Modalform transormierte System im ersten Arbeitspunkt sieht wie folgt aus:

- **A2D** $= \begin{pmatrix} 0 & 16{,}07 & 0 & 0 \\ -16{,}07 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5{,}993 \\ 0 & 0 & -5{,}993 & 0 \end{pmatrix}$
- **B2D** $= \begin{pmatrix} 5{,}374 \\ 0 \\ 0 \\ -2{,}143 \end{pmatrix}$
- **C2D** $= \begin{pmatrix} 0 & -1{,}374 & -1{,}883 & 0 \\ 0 & 2{,}881 & -2{,}694 & 0 \end{pmatrix}$
- **D2D** $= \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

2.4 Untersuchung von Steuerbarkeit und Beobachtbarkeit

Hier sollen die im Skript vorgestellten Überprüfungsverfahren unter Verwendung der im Rahmen des Versuchs erstellten Funktionen implementiert werden. Die Funktionen für die Überprüfungsverfahren sollen möglichst allgemein verwendbar sein.

Die Überprüfungsverfahren nach Kalman:

Listing 2.4: Code der Steuerbarkeitsfunktion nach Kalman

```
function checkCtrbKalman(A,B)

%Berechnung der Anzahl der Zustände
```

```
n = size(A,1);
%Anfangswert der Steuerbarkeitsmatrix ist gleich der Eingangmatrix →
   ←B
S_s_Kalman = B;

for i = 1:1:n-1
    %Berechnung der Steuerbarkeitsmatrix
    S_s_Kalman = [B A*S_s_Kalman];
end

disp('S=')
disp(S_s_Kalman)

%Überprüfung der Steuerbarkeit nach KALMAN
if (rank(S_s_Kalman)==n)
    disp('steuerbar');
else
    disp('nicht␣steuerbar');
end
end

%%Überprüfung der Steuerbarkeit auf die drei Zustandsraummodelle
%S:Steuerbarkeitsmatrix

%1. Zustandsraummodell mit AP_1
S=
   1.0e+04 *

         0    0.0143         0   -3.1532
    0.0143         0   -3.1532         0
         0   -0.0214         0    6.3064
   -0.0214         0    6.3064         0

steuerbar

%2. Zustandsraummodell mit AP_2

S=
   1.0e+04 *
```

```
             0    0.0143         0    3.1532
        0.0143         0    3.1532         0
             0   -0.0214         0   -6.3064
       -0.0214         0   -6.3064         0

steuerbar

%3. Zustandsraummodell mit AP_3

S=
             0   62.5000         0         0
       62.5000         0         0         0
             0         0         0         0
             0         0         0         0

nicht steuerbar
```

Listing 2.5: Code der Beobachtbarbarkeitsfunktion nach Kalman

```
function [ S_b_Kalman ] = checkObsvKalman( A, C )

n = length(A);

if rank(A) == n
    S_b_Kalman = C;
    for count = 1:n-1
        S_b_Kalman = [S_b_Kalman ; C * A^count];
    end
    if rank(S_B_Kalman == n
        disp('System␣ist␣beobachtbar␣nach␣Kalman');
    end
else
    disp('Matrix␣A␣hat␣nicht␣vollen␣Rang');
end
end
```

Im folgenden Listing sind die Vergleiche zwischen den Ergebnissen der selbsterstellten Funktionen mit den Ergebnissen der Funktionen ctrb und obsv:

Listing 2.6: Vergleich zwischen den Ergebnisse der Kalman-Verfahere und den Überprüfungen mit ctrb und obsv

```
%% Kalman: eigene Funktion und Überprüfung mit ctrb und obsv in AP1
%Steuerbarkeit:
ans =

   1.0e+04 *

         0    0.0143         0   -3.1532
    0.0143         0   -3.1532         0
         0   -0.0214         0    6.3064
   -0.0214         0    6.3064         0


ans =

   1.0e+04 *

         0    0.0143         0   -3.1532
    0.0143         0   -3.1532         0
         0   -0.0214         0    6.3064
   -0.0214         0    6.3064         0

%Beobachtbarkeit:

ans =

    1.0000         0         0         0
         0         0    1.0000         0
         0    1.0000         0         0
         0         0         0    1.0000
 -126.1286         0   63.0643         0
  189.1929         0 -168.1714         0
         0 -126.1286         0   63.0643
         0  189.1929         0 -168.1714


ans =

    1.0000         0         0         0
         0         0    1.0000         0
         0    1.0000         0         0
```

```
         0         0         0    1.0000
 -126.1286         0   63.0643         0
  189.1929         0 -168.1714         0
         0 -126.1286         0   63.0643
         0  189.1929         0 -168.1714

%Kalman eigene Funktion und Überprüfung mit ctrb und obsv in AP2

%Steuerbarkeit:
ans =

   1.0e+04 *

         0    0.0143         0    3.1532
    0.0143         0    3.1532         0
         0   -0.0214         0   -6.3064
   -0.0214         0   -6.3064         0


ans =

   1.0e+04 *

         0    0.0143         0    3.1532
    0.0143         0    3.1532         0
         0   -0.0214         0   -6.3064
   -0.0214         0   -6.3064         0

%Beobachtbarkeit:

ans =

    1.0000         0         0         0
         0         0    1.0000         0
         0    1.0000         0         0
         0         0         0    1.0000
  126.1286         0  -63.0643         0
 -189.1929         0  168.1714         0
         0  126.1286         0  -63.0643
         0 -189.1929         0  168.1714
```

```
ans =

    1.0000         0         0         0
         0         0    1.0000         0
         0    1.0000         0         0
         0         0         0    1.0000
  126.1286         0  -63.0643         0
 -189.1929         0  168.1714         0
         0  126.1286         0  -63.0643
         0 -189.1929         0  168.1714

%Kalman eigene Funktion und Überprüfung mit ctrb und obsv in AP3
%Steuerbarkeit:
Matrix A hat nicht vollen Rang

%Beobachtbarkeit
Matrix A hat nicht vollen Rang
```

Listing 2.7: Code der Steuerbarkeitsfunktion nach Gilbert

```
function [ isCtrb_Gilbert ] = checkCtrbGilbert( A, B )

[T, lam] = eig(A);

if length(unique(diag(lam))) ~= length(A)
    disp('A␣hat␣mehrfache␣Eigenwerte');
end

Bd = T/B;
isCtrb_Gilbert = all(any(Bd,2));

if isCtrb_Gilbert
    disp('System␣ist␣steuerbar␣nach␣Gilbert');
else
    disp('System␣ist␣NICHT␣steuerbar␣nach␣Gilbert');
end
end
```

Listing 2.8: Code der Beobachtbarbarkeitsfunktion nach Gilbert

```
function [ isObsv_Gilbert ] = checkObsvGilbert( A, C )

[T, lam] = eig(A);

if length(unique(diag(lam))) ~= length(A)
    disp('A␣hat␣mehrfache␣Eigenwerte');
end

Cd = C*T;
isObsv_Gilbert = all(any(Cd,1));

if isObsv_Gilbert
    disp('System␣ist␣beobachtbar␣nach␣Gilbert');
else
    disp('System␣ist␣NICHT␣beobachtbar␣nach␣Gilbert');
end
end
```

Listing 2.9: Code der Steuerbarkeitsfunktion nach Hautus

```
function checkCtrbKalman(A,B)

%Berechnung der Anzahl der Zustände
n = size(A,1);
%Anfangswert der Steuerbarkeitsmatrix ist gleich der Eingangmatrix →
   ←B
S_s_Kalman = B;

for i = 1:1:n-1
    %Berechnung der Steuerbarkeitsmatrix
    S_s_Kalman = [B A*S_s_Kalman];
end

disp('S=')
disp(S_s_Kalman)

%Überprüfung der Steuerbarkeit nach KALMAN
if (rank(S_s_Kalman)==n)
    disp('steuerbar');
```

```
else
    disp('nicht␣steuerbar');
end
end

%%Überprüfung der Steuerbarkeit auf die drei Zustandsraummodelle
%S:Steuerbarkeitsmatrix

%1. Zustandsraummodell mit AP_1
S=
   1.0e+04 *

         0    0.0143         0   -3.1532
    0.0143         0   -3.1532         0
         0   -0.0214         0    6.3064
   -0.0214         0    6.3064         0

steuerbar

%2. Zustandsraummodell mit AP_2

S=
   1.0e+04 *

         0    0.0143         0    3.1532
    0.0143         0    3.1532         0
         0   -0.0214         0   -6.3064
   -0.0214         0   -6.3064         0

steuerbar

%3. Zustandsraummodell mit AP_3

S=
         0   62.5000         0         0
   62.5000         0         0         0
         0         0         0         0
         0         0         0         0

nicht steuerbar
```

Listing 2.10: Code der Beobachtbarbarkeitsfunktion nach Hautus

```
function [ S_b_Kalman ] = checkObsvKalman( A, C )

n = length(A);

if rank(A) == n
    S_b_Kalman = C;
    for count = 1:n-1
        S_b_Kalman = [S_b_Kalman ; C * A^count];
    end
    if rank(S_B_Kalman == n
        disp('System ist beobachtbar nach Kalman');
    end
else
    disp('Matrix A hat nicht vollen Rang');
end
end
```