# 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 Lienarisierung

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 2.1 aufgeführt.

Bevor das System linearisiert wird sind zwei Fragen zu klären:

1. Welche Arbeitspunkte sind sinnvoll?
2. Was bedeutet es physikalisch, wenn $M_{\mathrm{AP}}$ 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 $M_{\mathrm{AP}}$ handelt es sich um den statischen Wert der Stellgröße $M$ im Arbeitspunkt. Ist diese ungleich null muss der Motor das Moment $M$

**Listing 2.1:** 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 allgmeine Zustandsraumdarstellung:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$
$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

die folgenden Systemmatrizen:

$$\mathbf{A}_{\mathrm{AP}_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \qquad \mathbf{B}_{\mathrm{AP}_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$$

## 2.2 Vergleich der Linearisierten Modelle

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

$$\lambda_1 =$$

## 2.3 Normalformen des Zustandsraummodelles

**Listing 2.2:** 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
```

## 2.4 Untersuchung von Steuerbarkeit und Beobachtbarkeit

**Listing 2.3:** Code der Steuerbarkeitsfunktion nach Kalman

```
function [ S_s_Kalman] = checkCtrbKalman( A, B )

n = length(A);

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

**Listing 2.4:** Code der Beobachtbarkeitsfunktion 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
```

Vergleich mit ctrb und obsv

**Listing 2.5:** 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 2.6:** 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 2.7:** Code der Steuerbarkeitsfunktion nach Hautus

```
function [ S_s_Kalman] = checkCtrbKalman( A, B )

n = length(A);

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

**Listing 2.8:** Code der Beobachtbarkeitsfunktion 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
```