Datenfusion
Praktische Laborübungen

MATLAB-VERSION 14B

Hochschule Esslingen
Fakultät Informationstechnik

Inhaltsverzeichnis

1 Klassisches Kalman-Filter **2**

1.1 Problemstellung 2

1.2 Aufgabenstellung 2

1.3 Matlab-Code 2

2 Kalman-Filter mit Messaussetzer **3**

2.1 Problemstellung 3

2.2 Aufgabenstellung 3

2.3 Matlab-Code zum Einlesen der Daten 3

3 Kalman-Filter mit asynchronen Messdaten **4**

3.1 Problemstellung 4

3.2 Aufgabe: Kalman-Filter 4

3.3 Matlab-Code zum Einlesen der Daten 4

4 Vergleich: Kalman-, ROSE- und Partikel-Filter **5**

4.1 Problemstellung 5

4.2 Aufgabe: Kalman-Filter 5

4.3 Aufgabe: ROSE-Filter 6

4.4 Aufgabe: Partikel-Filter 8

5 Literaturverzeichnis **9**

Literatur **9**

1 Klassisches Kalman-Filter

1.1 Problemstellung

Aktienkurse unterliegen einem mehr oder wenig starken Rauschen. Beispielhaft ist der Aktienkurs des Unternehmens „Leoni“ dargestellt.

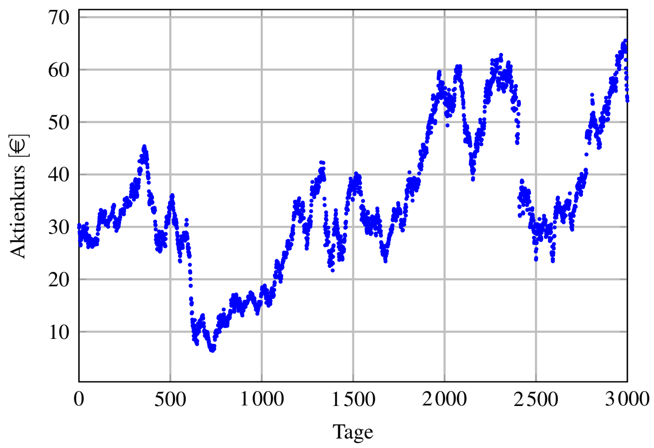

Abbildung 1: Eingangsdaten

Eintrag	Datum	Schlusskurs €
66	31.08.2006	27.29
67	01.09.2006	27.33
68	04.09.2006	27.69
69	05.09.2006	27.49
70	06.09.2006	27.10
71	07.09.2006	26.50
72	08.09.2006	26.41
73	11.09.2006	26.19
74	12.09.2006	26.64
75	13.09.2006	26.98

Tabelle 1: Auszug aus der Datei *„Leoni.dat“*

1.2 Aufgabenstellung

Entwerfen Sie ein Kalman-Filter (Beta-Filter) um das Rauschen im Aktienkurs zu minimieren.

Hinweis zur Bearbeitung:
Bestimmen Sie zuerst für den Zeitraum die Varianz R des Messrauchens. Gehen Sie davon aus, dass sich die Varianz des Rauschens zeitinvariant ist.

Variieren Sie dann das Systemrauchen Q und schauen sich den Verlauf der beiden Zustandsgrößen an.

Optimieren Sie diesen Kalman-Filter so, dass er eine minimale Laufzeit besitzt.

1.3 Matlab-Code

```
clearvars;

in = readtable('Leoni.dat','Delimiter','semi');
y = in.Schlusskurs;
u = zeros(length(y),1);
t = in.Eintrag;
d = datenum(in.Datum,'dd.mm.yyyy');

Ts = t(2)-t(1);
R = ...;  Q = ...;

Ad=[1 Ts; 0 1]; Bd=[0 0]'; C=[1 0]; D=0; Gd=[Ts 1]';

%%% Init %%%
x_dach = [y(1); 0];
P_dach = 50*[1 0; 0 1];

%%% Kalman %%%
for k=1:length(y)
  d_y(k) = y(k) - ( C*x_dach + D*u(k));
  K = P_dach*C'*pinv(C*P_dach*C' + R);
  x_tilde = x_dach + K*d_y(:,k);
  P_tilde = (eye(length(Bd)) - K*C)*P_dach;
  x_dach = Ad*x_tilde + Bd*u(k);
  P_dach = Ad*P_tilde*Ad' + Gd*Q*Gd';

  s(k) = x_tilde(1);  v(k) = x_tilde(2);
  K1(k) = K(1); K2(k) = K(2);
  P_tilde1(k)=P_tilde(1); P_tilde2(k)=P_tilde(2);
  P_tilde3(k)=P_tilde(3); P_tilde4(k)=P_tilde(4);
end

figure(1); clf;
subplot(2,1,1);
plot(t,y,'k*',t,s,'b--',...
     t,s+1.5*sqrt(P_tilde1),'b-',...
     t,s-1.5*sqrt(P_tilde1),'b-');
grid on; xlabel('Zeit / Tag'); ylabel('Kurs / Euro');
subplot(2,1,2);
plot(t,v,'r-*');
grid on; xlabel('Zeit / Tag');
ylabel('Kursänderungen / Euro/Tag');
```

2 Kalman-Filter mit Messaussetzer

2.1 Problemstellung

Für ein Fahrzeug soll die zurückgelegte Strecke abgeschätzt werden. Die Position des Fahrzeugs wird durch einen GPS-Sensor in äquidistant Zeitabständen erfasst. Ab und zu fährt das Fahrzeug durch ein Tunnel, sodass die Position des Fahrzeugs zeitweise nicht erfassbar werden kann.

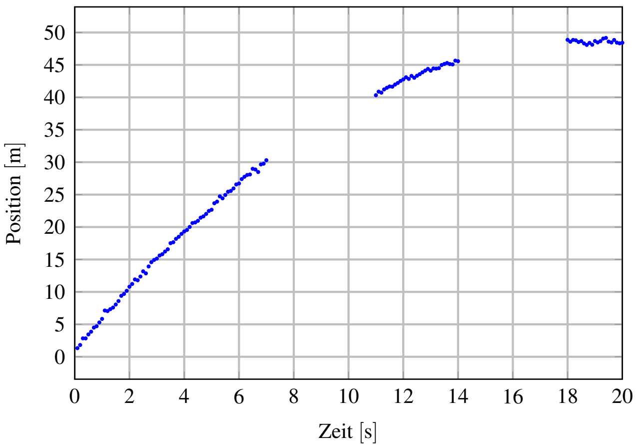

Abbildung 2: Eingangsdaten

time s	position m
6.600	28.872
6.700	28.485
6.800	29.656
6.900	29.771
7.000	30.303
7.100	NaN
7.200	NaN
7.300	NaN
7.400	NaN
7.500	NaN

Tabelle 2: Auszug aus der Datei *„Messaussetzer_data_in.dat“*

2.2 Aufgabenstellung

Entwerfen Sie ein geeignetes Kalman-Filter um zum einem die Position zu filtern und auf der anderen Seite, an den Stellen an denen das Fahrzeug im Tunnel ist, die Postion zu prädizieren.

Hinweis zur Bearbeitung:
Untersuchen Sie verschiedene Modellansätze (Alpha-,Beta-,Gamma-Filter).

2.3 Matlab-Code zum Einlesen der Daten

```
clearvars;
in= readtable('Messaussetzer_data_in.dat',...
            'Delimiter','space');
t = in.time;
y = in.position;
u = zeros(1,length(y));
```

3 Kalman-Filter mit asynchronen Messdaten

3.1 Problemstellung

In einem Fahrzeug wird mittels eines GPS-Sensors die Position und über einen Inertialsensor die Beschleunigung erfasst. Die Datenerfassung findet asynchron und in nicht äquidistanten Zeitabständen statt.

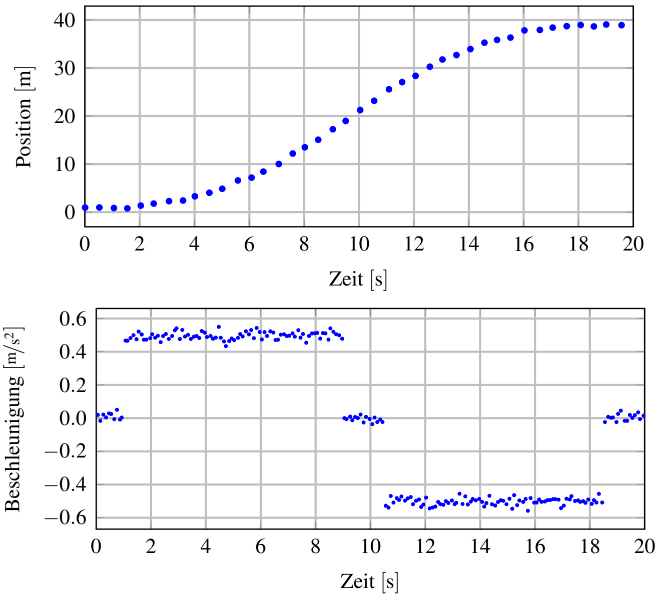

Abbildung 3: Eingangsdaten

time	position	acceleration
s	m	$^{m}/_{s^2}$
0.011	0.939	NaN
0.060	NaN	0.019
0.150	NaN	−0.016
0.260	NaN	0.021
0.360	NaN	0.004
0.470	NaN	0.029
0.531	0.962	NaN
0.550	NaN	0.025
0.660	NaN	−0.006
0.760	NaN	0.050

Tabelle 3: Auszug aus der Datei *„Asynchron_data_in.dat“*

3.2 Aufgabe: Kalman-Filter

Entwerfen Sie ein geeignetes Kalman-Filter, welches in der Lage ist die nicht zeitäquidistanten Größen zu fusionieren.

Hinweis zur Bearbeitung:
Überlegen sich welches klassisches Kalman-Filter für die Aufgabe geeignet ist und was man an dem in der Vorlesung behandelten Kalman-Filter ändern müsste, sodass man den nicht äquidistant Abtastintervallen gerecht wird.

3.3 Matlab-Code zum Einlesen der Daten

```
clearvars;

in = readtable('Asynchron_data_in.dat',...
              'Delimiter','space');
t = in.time;
y(:,1)=in.position;
y(:,2)=in.acceleration;
u = zeros(1,length(y));
```

4 Vergleich: Kalman-, ROSE- und Partikel-Filter

4.1 Problemstellung

Gegeben ist die unten beschriebene zeitliche Folge. Diese kann z.B. als Position eines Fahrzeugs interpretiert werden.

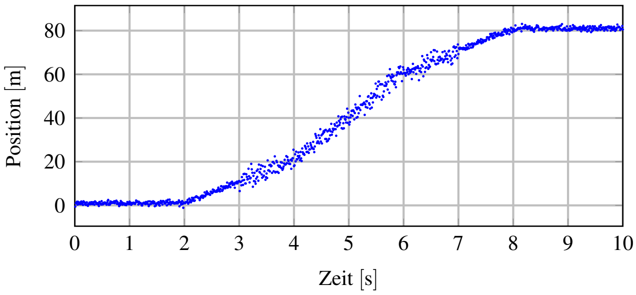

Abbildung 4: Eingangsdaten

time s	position m
0.010	−0.557
0.020	0.952
0.030	1.096
0.040	1.646
0.050	0.372
0.060	1.638
0.070	0.661
0.080	0.659
0.090	1.047
0.100	1.409

Tabelle 4: Auszug aus der Datei *„BetaFilter_data_in.dat“*

4.2 Aufgabe: Kalman-Filter

Realisieren Sie hierfür ein Beta-Filter, welches die Daten filtert und die Geschwindigkeit schätzt. Basis ist das folgende Beta-Filter. Hierzu müssen Sie noch die Varianz des Mess- und Systemrauschens abschätzen.

```
clearvars;

in = readtable('BetaFilter_data_in.dat',...
               'Delimiter','space');
y = in.position;
u = zeros(length(y),1);
t = in.time;

Ts = t(2)-t(1);
R = ...;
Q = ...;

Ad=[1 Ts; 0 1]; Bd=[0 0]'; C=[1 0]; D=0; Gd=[Ts 1]';

%%% Init %%%
x_dach = [y(1); 0];
P_dach = 50*[1 0; 0 1];

%%% Kalman %%%
for k=1:length(y)
  d_y(k) = y(k) - ( C*x_dach + D*u(k));
  K = P_dach*C'*pinv(C*P_dach*C' + R);
  x_tilde = x_dach + K*d_y(:,k);
  P_tilde = (eye(length(Bd)) - K*C)*P_dach;
  x_dach = Ad*x_tilde + Bd*u(k);
  P_dach = Ad*P_tilde*Ad' + Gd*Q*Gd';

  s(k) = x_tilde(1);  v(k) = x_tilde(2);
  K1(k) = K(1); K2(k) = K(2);
  P_tilde1(k)=P_tilde(1); P_tilde2(k)=P_tilde(2);
  P_tilde3(k)=P_tilde(3); P_tilde4(k)=P_tilde(4);
end

figure(1); clf;
subplot(2,1,1);
plot(t,y,'k*',t,s,'r--');
grid on; xlabel('Zeit / s'); ylabel('Position / m');
subplot(2,1,2);
plot(t,v,'r-');
grid on; xlabel('Zeit / s');
ylabel('Geschwindigkeit / s');
```

4.3 Aufgabe: ROSE-Filter

Nun soll für das selbe Problem ein ROSE-Filter entwickelt werden. Basis ist wieder ein Beta-Filter.

Messrauschen

Die Schätzung des Messrauschens lässt sich über den folgenden Zusammenhang bestimmen:

$$\begin{aligned}
\underline{R}(k) &= \mathrm{E}\Big(\big(\underline{y}(k) - \mathrm{E}(\underline{y}(k))\big) \cdot \big(\underline{y}(k) - \mathrm{E}(\underline{y}(k))\big)^T\Big) && (1)\\
&\approx \mathrm{E}\Big(\big(\underline{y}(k) - \underline{\hat{y}}_R(k)\big) \cdot \big(\underline{y}(k) - \underline{\hat{y}}_R(k)\big)^T\Big) && (2)\\
&\approx \mathrm{E}\Big(\Delta\underline{\hat{y}}_R(k) \cdot \Delta\underline{\hat{y}}_R(k)^T\Big) && (3)
\end{aligned}$$

Die Berechnung von $\mathrm{E}(\underline{y}(k))$ geschieht über ein Beta-Filter mit fester Verstärkung und die Berechnung von $\mathrm{E}(\Delta\underline{\hat{y}}_R(k) \cdot \Delta\underline{\hat{y}}_R(k)^T)$ mit einem Alpha-Filter auch mit fester Verstärkung.

Da für das Beta-Filter die die Größen R und Q zeitinvariant sind, konvergiert die Kalman-Verstärkung auf einen festen Wert, der sich mit der Gleichung:

$$\begin{aligned}
K &= \frac{0.125}{T_s} \cdot \begin{bmatrix} T_s \cdot (-\lambda^2 - 8 \cdot \lambda + (\lambda + 4) \cdot \sqrt{\lambda^2 + 8 \cdot \lambda}) \\ 2 \cdot (\lambda^2 + 4 \cdot \lambda - \lambda \cdot \sqrt{\lambda^2 + 8 \cdot \lambda}) \end{bmatrix}\\
\text{mit:} \quad \lambda &= T_s \cdot \sqrt{\frac{Q}{R}} = T_s \cdot \sqrt{\frac{\mathrm{Var}(z(k))}{\mathrm{Var}(v(k))}} && (4)
\end{aligned}$$

bestimmen lässt [1].

Bei der Wahl des Verhältnisses von Q/R ist man frei. Aufgrund der großen Dynamik sollte man das Verhältnis jedoch nicht zu klein wählen. Sinnvoll ist z.B. eine Wahl von $Q/R = 1$

Da bei den Kalman-Filtern mit fester Kalman-Verstärkung nur die geschätzte Zustandsgröße für die weitere Verarbeitung relevant ist, müssen nur die folgenden beiden Gleichungen betrachtet werden:

$$\begin{aligned}
\underline{\hat{x}}(k) &= \underline{A}_d \cdot \underline{\tilde{x}}(k-1) + \underline{B}_d \cdot \underline{u}(k-1) && (5)\\
\underline{\tilde{x}}(k) &= \underline{\hat{x}}(k) + \underline{K} \cdot \big(\underline{y}(k) - \underline{C} \cdot \underline{\hat{x}}(k) - \underline{D} \cdot \underline{u}(k)\big) && (6)
\end{aligned}$$

Berücksichtigt man, dass $\underline{B}_d = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$ und $\underline{D} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$ und setzt Gleichung (5) in Gleichung (6) ein, folgt:

$$\begin{aligned}
\underline{\tilde{x}}(k) &= \underline{\hat{x}}(k) + \underline{K} \cdot \underline{y}(k) - \underline{K} \cdot \underline{C} \cdot \underline{\hat{x}}(k)\\
&= \underline{K} \cdot \underline{y}(k) + \big(\underline{I} - \underline{K} \cdot \underline{C}\big) \cdot \underline{\hat{x}}(k)\\
&= \underline{K} \cdot \underline{y}(k) + \underbrace{\big(\underline{I} - \underline{K} \cdot \underline{C}\big) \cdot \underline{A}_d}_{\underline{H}} \cdot \underline{\tilde{x}}(k-1)
\end{aligned}$$

Mit diesem Filter wird der Erwartungswert $\hat{y}_R(k) = \mathrm{E}(y(k))$ für die gemessene Position abgeschätzt.

$$\begin{aligned}
\underline{\tilde{x}}(k) &= \begin{bmatrix} \tilde{x}_1(k) \\ \tilde{x}_2(k) \end{bmatrix} = \underline{K} \cdot y(k) + \underline{H} \cdot \underline{\tilde{x}}(k-1)\\
\hat{y}_R(k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \underline{\tilde{x}}(k) = \tilde{x}_1(k) && (7)
\end{aligned}$$

Zur Schätzung der Kovarianz des Messrauschens $\underline{R}(k)$ ist es aufgrund der geringen Dynamik der Größe $\underline{\hat{y}}_R(k)$ ausreichend, den Erwartungswert in Gleichung (3) über einen Alpha-Filter abzuschätzen:

$$\begin{aligned}
\underline{R}(k) = \gamma \cdot \alpha_R \cdot \big(\underline{\hat{y}}_R(k) - \underline{y}(k)\big) \cdot \big(\underline{\hat{y}}_R(k) - \underline{y}(k)\big)^T&\\
+ (1 - \alpha_R) \cdot \underline{R}(k-1)& && (8)
\end{aligned}$$

Der frei wählbare Verstärkungsfaktor γ dient als Korrekturfaktor, da die Kovarianz des Messrauschens oft zu klein geschätzt wird. Aufgrund dessen sind Werte von $\gamma > 1$ sinnvoll.

Systemrauschen

Zur Schätzung der Kovarianz des Systemrauschens ist es notwendig, die Hilfsgröße $\underline{M}(k)$ zu bestimmen. Diese wird definiert durch:

$$\begin{aligned}
&\underline{M}(k) = \mathrm{E}\big(\Delta\underline{y}(k) \cdot \Delta\underline{y}(k)^T\big) && (9)\\
&\text{mit:} \quad \Delta\underline{y}(k) = \underline{y}(k) - \underline{C} \cdot \underline{\hat{x}}(k) - \underline{D} \cdot \underline{u}(k) && (10)
\end{aligned}$$

Der Erwartungswert kann durch ein Alpha-Filter angenähert werden. Dieses wird durch folgende Gleichung beschrieben:

$$\underline{M}(k) = \alpha_M \cdot \Delta\underline{y}(k) \cdot \Delta\underline{y}(k)^T + (1 - \alpha_M) \cdot \underline{M}(k-1) \qquad (11)$$

Die Kovarianz des Messrauschens $Q(k-1)$ lässt sich allgemein durch folgende Gleichung berechnen:

$$\begin{aligned}
\underline{C} \cdot \underline{G}_d \cdot \underline{Q}(k-1) \cdot \underline{G}_d^T \cdot \underline{C}^T = \underline{M}(k) - \underline{R}(k)&\\
- \underline{C} \cdot \underline{A}_d \cdot \underline{\tilde{P}}(k-1) \cdot \underline{A}_d^T \cdot \underline{C}^T& && (12)
\end{aligned}$$

Bestimmen Sie den linken Teil der Gleichung für ein Beta-Filter mit $\underline{G}_d^T = \begin{bmatrix} T_s & 1 \end{bmatrix}$ und $\underline{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}$ und lösen Sie die Gleichung nach $Q(k-1)$ auf.

Nach kurzer Rechnung folgt:

$$Q(k-1) = \frac{\underline{M}(k) - \underline{R}(k) - \underline{C} \cdot \underline{A}_d \cdot \underline{\tilde{P}}(k-1) \cdot \underline{A}_d^T \cdot \underline{C}^T}{T_s^2} \qquad (13)$$

Ergänzen Sie nun in dem Programm die Zeile mit der Berechnung von $Q(k)$ und legen Sie die Größen Q_{min} und Q_{max} fest. Gute Werte sind eine Zehnerpotenz größer bzw. eine Zehnerpotenz größer als der Wert bei einem klassischen Kalman-Filter (siehe Aufgabe 4.2).

```
clearvars;

in = readtable('BetaFilter_data_in.dat',...
               'Delimiter','space');
y = in.position;
u = zeros(length(y),1);
t = in.time;

Ts = t(2)-t(1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Gamma   = 2.5;
Alpha_R = .05;
Alpha_M = .1;
Q_min   = ...;
Q_max   = ...;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Vorabbestimmung von K0 und H
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ad = [1 Ts; 0 1];
C  = [1 0];

R0  = 1;
Q0  = 1;

lambda = Ts*sqrt(Q0/R0);
K1 = -1/8*(lambda.^2 + 8*lambda ...
     -(lambda+4).*sqrt(lambda.^2+8*lambda));
K2 = .25*(lambda.^2 + 4*lambda ...
     -lambda.*sqrt(lambda.^2+8*lambda))/Ts;

K0= [K1;K2];
H = (eye(length(Ad)) - K0*C)*Ad;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   I N I T   K A L M A N - F I L T E R
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

R = 1;

Ad=[1 Ts; 0 1]; Bd=[0 0]'; C=[1 0]; D=0; Gd=[Ts 1]';
GG = Gd*Gd';

x_dach = [y(1); 0;];
x1= x_dach;

u = zeros(1,length(t));
p_tilde = 1*[1 0; 0 1];
M = C*p_tilde*C';

for k=1:length(y)
  %-----------------------------------------------
  % Measurement noise estatimation
  %-----------------------------------------------
  x1 = H*x1 + K0*y(k);    yr(k) = x1(1);
  R  = Gamma*Alpha_R*[y(k)-x1(1)]*[y(k)-x1(1)]' ...
     +(1-Alpha_R)*R;

  %-----------------------------------------------
  % Calculate Matrix M
  %-----------------------------------------------
  dy = y(k) - C*x_dach - D*u(k);
  M = Alpha_M.*dy*dy' + (1-Alpha_M).*M;

  %-----------------------------------------------
  % Calculate p_dach
  %-----------------------------------------------
  Q(k) = ...

  if Q(k)<Q_min  Q(k)=Q_min;  end
  if Q(k)>Q_max  Q(k)=Q_max;  end

  p_dach = Ad*p_tilde*Ad' + GG*Q(k);

  %-----------------------------------------------
  % kalman gain, correction of state x and P,
  % prediction of state x
  %-----------------------------------------------
  K = p_dach*C'*pinv(C*p_dach*C' + R);
  x_tilde = x_dach + K*dy;
  p_tilde = (eye(length(Bd)) - K*C)*p_dach;
  x_dach = Ad*x_tilde + Bd*u(k);

  s(k)=x_tilde(1); v(k)=x_tilde(2);
  M1(k)=M(1);
  K1(k)=K(1); K2(k)=K(2);
  R1(k)=R(1);
  p_dach1(k)=p_dach(1);   p_dach2(k)=p_dach(2);
  p_dach3(k)=p_dach(3);   p_dach4(k)=p_dach(4);
  p_tilde1(k)=p_tilde(1); p_tilde2(k)=p_tilde(2);
  p_tilde3(k)=p_tilde(3); p_tilde4(k)=p_tilde(4);
end

h2 = figure(2); clf;
subplot(211);
plot(t,y,'k*',t,s,'r--',t,s+3*sqrt(p_tilde1),...
     'b-',t,s-3*sqrt(p_tilde1),'b-');
grid on; ylabel('Position'); xlabel('time');
subplot(212); plot(t,v,'r-'); grid on;
ylabel('Geschwindigkeit'); xlabel('time / s');

h3 = figure(3); clf;
set(h3,'Name','variance measurement noise');
plot(t,R1); grid on;
ylabel('R'); xlabel('time / s');

h4 = figure(4); clf;
set(h4,'Name','variance procces noise');
plot(t,Q); grid on;
ylabel('Q'); xlabel('Zeit / s');
```

4.4 Aufgabe: Partikel-Filter

Entwickeln Sie ein Partikel-Filter welches die Systembeschreibung eines klassischen Beta-Filters beinhaltet.

$$s(k+1) = s(k) + T_s \cdot \big(v(k) + z(k)\big) \tag{14}$$
$$v(k+1) = v(k) + z(k) \tag{15}$$

Basis ist der unten aufgeführte Code.

Ergänzen Sie diesen an den geeigneten Stellen und variieren Sie die Anzahl der Partikel N.

```
clearvars;

in = readtable('BetaFilter_data_in.dat',...
               'Delimiter','space');
y = in.position;
u = zeros(length(y),1);
t = in.time;
T = length(y);
Ts = t(2)-t(1);

N = ...;     % The number of particles
R = ...;
Q = ...;

s_tilde(1) = y(1,1);
v_tilde(1) = 0;
for i = 1:N
    z = sqrt(10*Q)*randn;
    s_P_tilde(1,i) = ...;
    v_P_tilde(1,i) = ...;
end

for t = 2:T
    for i = 1:N
       z = sqrt(Q)*randn;
       s_P_dach(i) = ...;
       v_P_dach(i) = ...;
       y_dach(i) = ...;
       error = sqrt((y(t)-y_dach(i))^2);
       w_P(i)= (1/sqrt(2*pi*R))*exp(-error^2/(2*R));
    end
    w_P = w_P./sum(w_P);

    r = -.001/N;
    for i = 1:N
        r = r + 1/N;
        s_P_tilde(t,i) = ...
   s_P_dach(find(r <= cumsum(w_P),1));
        v_P_tilde(t,i) = ...
   v_P_dach(find(r <= cumsum(w_P),1));
    end
    s_tilde(t) = w_P*s_P_dach';
    v_tilde(t) = w_P*v_P_dach';
end

k = 1:T;
figure(5); clf;
subplot(211); plot(k,y,k,s_tilde,'r-');grid on;
subplot(212); plot(k,v_tilde,'r-'); grid on;
```

5 Literaturverzeichnis

Literatur

[1] BAR-SHALOM, Yaakov ; LI, Xiao-Rong: Estimation and tracking: Principles Techniques and Software. Boston : Artech House, 1993. – ISBN 0 – 89006 – 643 – 4