Table of Contents

1a 1
1b 2
1c 2
1d 3
1 f 4
2A 4
2B 5
2C 5

1a

Tolerance

```
tol = 1e-1;
N = 20;

% Define prior
x_0     = 2;
n       = length(x_0);
P_0     = 6;

% Define process model
A       = diag(ones(n,1));
Q       = 1.5;

% generate state sequence
X   = genLinearStateSequence(x_0, P_0, A, Q, N);

% measurement sequence
absTol = 1e-1;
relTol = 5e-2;

%n = randi(1,1);
m = randi(n,1);

% Define measurement model
H = 1;
R = 2.5;

% Generate measurements
Y = genLinearMeasurementSequence(X, H, R);

% Plot results
figure(1);
clf;
hold on;
grid on;
plot(0:20,X(1,1:21), '--k');
plot(1:20, Y(1,1:20), '*r');
```

```
legend('State sequence', 'Measurements')
title('Your solution');
xlabel('k');
ylabel('position');
```

1b

```
% Filter
%[stateSequence, covarianceSequence] =
 kalmanFilter(measuremetnSequence, xPrior, PPrior, A, Q, H, R);
[xHat, P] = kalmanFilter(Y, x_0, P_0, A, Q, H, R);
% plot results
figure(2); clf;
hold on;
grid on;
plot(X);
plot([1:N], Y, '*r');
plot([0:N], [x_0 xHat], 'b');
plot([0:N], [x_0 xHat] + 3*sqrt([P_0 P(:)']), '--b');
plot([0:N], [x_0 xHat] - 3*sqrt([P_0 P(:)']), '--b');

title('Your solution')
xlabel('k');
ylabel('x');
legend('true','measurements', 'state estimate', '+3-sigma level', '-3-
sigma level','Location','southeast');
timeframe = [5,10,15];
for i=1:length(timeframe)
    xmeanest  = Xfiltered(timeframe(i));
    xvar = P(timeframe(i));
    xnormpdf = xmeanest-(4*xvar):0.01:xmeanest+(4*xvar);
    ynormpdf = normpdf(xnormpdf,xmeanest,sqrtm(xvar));
    xtrue = X(timeframe(i)+1);
    figure(3)
    subplot(3,1,i);
    plot(xnormpdf,ynormpdf, 'LineWidth',2);
    hold on;
    plot([xtrue,xtrue], [0,max(ynormpdf)],'--', 'LineWidth',3);
    ylim([0,max(ynormpdf)]);
    title(['posterior distribution for t=',num2str(timeframe(i))]);
    legend('PDF','true')
end
```

1c

CHOICE OF K

```
k = 5;
[xHat, P] = kalmanFilter(Y, x_0, P_0, A, Q, H, R);
% PRIOR FROM K-1
PRIOR_X = xHat(:,k-1);
PRIOR_P = P(:,:,k-1);
```

```
% PREDICTION USING PRIOR
[PREDICTED_X,PREDICTED_P] = linearPrediction(PRIOR_X, PRIOR_P, A, Q);

% MEASUREMENT AT K
MEASUREMENT = Y(:,k);

% UPDATE USING THE PREDICTION AND MEASUREMENT
[UPDATED_X,UPDATED_P] = linearUpdate(PREDICTED_X, PREDICTED_P,
 MEASUREMENT, H, R);

% APPLYING FIGURE SETTINGS
figure(4);
hold on;
grid on;
x = PREDICTED_X-4*sqrt(PREDICTED_P):0.01:PREDICTED_X
+4*sqrt(PREDICTED_P);
plot(x, normpdf(x, PRIOR_X, sqrt(PRIOR_P)),'LineWidth', 2);
plot(x, normpdf(x, PREDICTED_X, sqrt(PREDICTED_P)),'LineWidth', 2);
plot(x, normpdf(x, UPDATED_X, sqrt(UPDATED_P)),'LineWidth', 2);
plot(MEASUREMENT, 0, 'sk', 'LineWidth', 2, 'MarkerSize', 10);
title('Single Kalman Filter Iteration','FontSize',14);
xlabel('State','FontSize',12);
ylabel('Probability density','FontSize',12);
leg = legend('Prior density', 'Predicted density', 'Updated
 density', 'Measurement', 'location','eastoutside','FontSize',8);
htitle = get(leg,'Title');
set(htitle,'String',['k = ' num2str(k)]);
```

1d

```
close all;
clear all;
A=1;
Q=1.5;
R=2.5;
x_0=2;
P_0=6;

H=1;
N = 5000;
X = genLinearStateSequence(x_0,P_0,A,Q,N);
Y = genLinearMeasurementSequence(X, H, R);


[Xfiltered, P,xp,Pp,l] = kalmanFilterextract(Y, x_0, P_0, A, Q, H, R);

Xfilteredm=mean(Xfiltered);
vm=mean([l.innov]);
%
  xnormpdf = 0-(4*P(:,:,end)):0.01:0+(4*P(:,:,end));
  ynormpdf = normpdf(xnormpdf,0,sqrtm(P(:,:,end)));
 figure(5);
 hold on, grid on;
```

```
histogram( (Xfiltered-X(:,2:end)), 100 ,'Normalization','pdf');
hold on,
plot(xnormpdf,ynormpdf, 'LineWidth',2 );
xlabel('$x_k-\hat{x}_{k|k}$','Interpreter','Latex')
ylabel('normalized frequency','Interpreter','Latex')
title 'Histogram of normalized estimation error'
legend('histogram','Probablity density function')

figure(6);
hold on, grid on;
autocorr([l.innov]);
xlabel 'Lag',
ylabel 'Autocorrelation',
title ' Autocorrelation of innovation'
```

1 f

```
clc;
close all;
clear all;
A=1;
Q=1.5;
R=2.5;
x_0=2;
P_0=6;
x_w=10;
H=1;
N = 20;
X = genLinearStateSequence(x_0,P_0,A,Q,N);
Y = genLinearMeasurementSequence(X, H, R);

[Xfiltered, P,xp,Pp,l] = kalmanFilterextract(Y, x_0, P_0, A, Q, H, R);
[Xfilteredw, Pw,xpw,Ppw,lw] = kalmanFilterextract(Y, x_w, P_0, A, Q,
 H, R);

figure(7)
plot([0:N],[x_0 Xfiltered],'-',[0:N],[x_0 X(2:N+1)],'-',[1:N],Y,'*',
[0:N],[x_w Xfilteredw],'-');
hold on;
grid on;
xlabel 'time step', ylabel 'value'
legend('filtered est','true','meaesurement','wrong prior filtered
 est')
```

2A

```
clc;
clear all;
close all;
T=0.01;
A2=[1 T;0 1];
V2=[0;1];
H2=[1 0];
```

```
N2= 1;
Q_mu=[0;0];
Q_var=[0 0;0 1.5];
R=2;
x_0=[1 ;3];
P_0=4*eye(2);
N=200;
X = genLinearStateSequence(x_0,P_0,A2,Q_var,N);
Y = genLinearMeasurementSequence(X, H2, R);

figure(8)
plot([1:N],X(1,2:N+1),'--',[1:N],Y,'*');
hold on;
grid on;
xlabel 'time step', ylabel 'value'
legend('true','meaesurement')
title('plot of state and Measurement sequence')
figure(9)
plot([1:N],X(2,2:N+1),'-');
grid on;
xlabel 'time step', ylabel 'value'
legend('Velocity')
title('plot of velocity state ')
```

2B

```
[Xfiltered, P,xp,Pp,l] = kalmanFilterextract(Y, x_0, P_0, A2, Q_var,
 H2, R);

Xfilteredp3sig=[Xfiltered] + 3*sqrt([squeeze(P(1,1,:))']);
Xfilteredm3sig=[Xfiltered] - 3*sqrt([squeeze(P(1,1,:))']);
figure(10)
plot([1:N],Xfiltered(1,:),'-',[1:N],X(1,2:N+1),'-',[1:N],Y,'*',
[1:N],Xfilteredp3sig(1,:),'-',[1:N],Xfilteredm3sig(1,:),'-');
hold on, grid on;
xlabel 'time step', ylabel 'position'
legend('filtered est','true','measurement','+3sigma','-3sigma')
figure(11)
plot([1:N],Xfiltered(2,:),'-',[1:N],X(2,2:N+1),'-',
[1:N],Xfilteredp3sig(2,:),'-',[1:N],Xfilteredm3sig(2,:),'-');
hold on, grid on;
xlabel 'time step', ylabel 'Velocity'
legend('filtered est','true','+3sigma','-3sigma')
```

2C

```
clc;
clear all;
close all;
T=0.01;
A2=[1 T;0 1];
V2=[0;1];
H2=[1 0];
```

```
N2= 1;
Q_mu=[0;0];
Q_var=[0 0;0 1.5];
Q=[0.1 1 10 15];
R=2;
x_0=[1 ;3];
P_0=4*eye(2);
N=200;
X = genLinearStateSequence(x_0,P_0,A2,Q_var,N);
Y = genLinearMeasurementSequence(X, H2, R);
for i=1:4
Q_var=[0 0;0 Q(i)];

[G(i).Xfiltered, P,xp,Pp,l] = kalmanFilterextract(Y, x_0, P_0, A2,
 Q_var, H2, R);
end
figure(12)
plot([1:N],G(1).Xfiltered(1,:),'-',[1:N],G(2).Xfiltered(1,:),'-',
[1:N],G(3).Xfiltered(1,:),'-',[1:N],G(4).Xfiltered(1,:),'-');
hold on, grid on;
xlabel 'time step', ylabel 'distance/position'
legend('filtered est dist Q1','filtered est dist Q2','filtered est
 dist Q3','filtered est dist Q4')
figure(13)
plot([1:N],G(1).Xfiltered(2,:),'-',[1:N],G(2).Xfiltered(2,:),'-',
[1:N],G(3).Xfiltered(2,:),'-',[1:N],G(4).Xfiltered(2,:),'-');
hold on, grid on;
xlabel 'time step', ylabel 'Velocity'
legend('filtered est dist Q1','filtered est dist Q2','filtered est
 dist Q3','filtered est dist Q4')


options = struct('format', 'pdf', 'evalCode',false);
publish('kalmanFilter.m', options);
publish('genLinearMeasurementSequence.m', options);
publish('genLinearStateSequence.m', options);
publish('kalmanFilterextract.m', options);
publish('linearUpdate.m', options);
publish('linearPrediction.m', options);
publish('kalmanFilter.m', options);
```

Published with MATLAB® R2020a