Assignment 6

Analysis of accuracy decrease of tracking in conditions of biased state noise

By

Group 19

Ahmed Baza, Skoltech, 2020

Ahmed Elgohary, Skoltech, 2020

Mohammed Sayed, Skoltech, 2020

Ziad Baraka, Skoltech, 2020

1) Generate a true trajectory $X_i$of an object motion disturbed by normally distributed biased random acceleration. Bias (mathematical expectation) of random noise q = 0.2

2) Generate measurements Z_i of the coordinate X_i

$$a_{i-1}^{biased} = a_{i-1} + q$$

$$X_i = X_{i-1} + V_{i-1}T + a_{i-1(\text{biased})}\frac{T^2}{2}$$

$$V_i = V_{i-1} + a_{i-1_{(\text{biased})}}T$$

$$Z_i = X_i + \eta$$

Size of trajectory is ***200*** points.

Initial conditions: $X_1 = 5;\ V_1 = 1;\ T = 1;$

Variance of noise a_i : $\sigma_a^2 = 0.2^2$

Variance of noise Z_i : $\sigma_\eta^2 = 20^2$

```
clc; clear; close all;

normaldist  =  makedist('Normal',0,0.2);    % noramally distributed random acceleration with va
q  =  0.2;     % bias (mathematical expectation) of random noise
a = random(normaldist,200,1)+q;  % biased random acceleration.
```

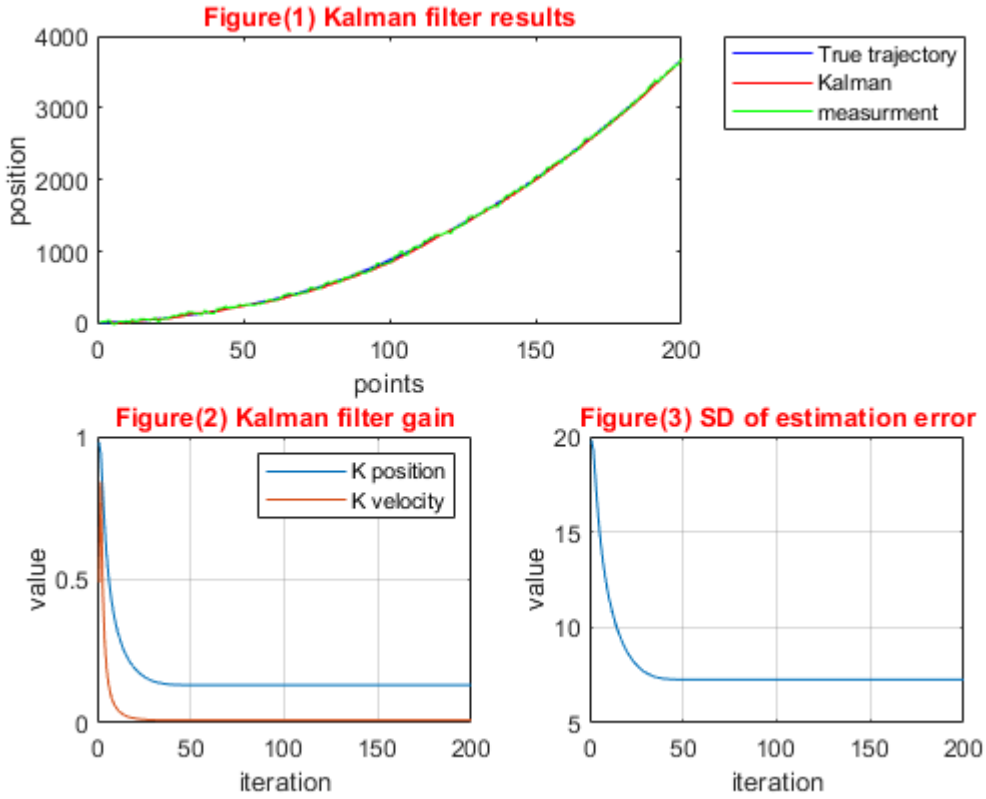
```
X = zeros(1,200);     % true position data
X(1) = 5;             % initial position
V = zeros(1,200);     % true velocity data
V(1) = 1;             % initial velocity
T = 1;                % time step
N = length(X);        % size of trajectory

% generation of true trajectory
for i = 2:N
    X(i) = X(i-1)+V(i-1)*T+a(i-1)*T^2/2;   % position
    V(i) = V(i-1)+a(i-1)*T;                % Velocity
end

normaldist2 = makedist('Normal',0,20);
eta = random(normaldist2,200,1);    % normally distributed random noise with varianec of noise
Z = zeros(1,200);                   % measurements

% generation of measurments
for i = 1:N
    Z(i) = X(i)+eta(i);            % Measurments
end
```

3) Obtain estimates of state vector $X = [x\,; V]$ by Kalman filter in assumption of unbiased acceleration ($q = 0$).

```
phi = [1 T;0 1];        % transition matrix that relates Xi to Xi-1
G = [T^2; T];           % input matrix, that determines how random acceleration affects state v
H = [1 0];              % observation matrix
xi = zeros(1,200);      % filtered position vector

Xi = [2;0];               % intial filtered estimate of the state vector X = [x; v].
P = [10000 0; 0 10000];   % initial filtration error covariance matrix
R = 20^2;                 % covariance R.
sigma_a = 0.2^2;
K = P*H'/(H*P*H'+R);      % initial value for the Kalman gain.
Ki = zeros(N,2);          % Matrix for holding the kalman gain.
Pi = zeros(N,1);

for i = 1:N
    Xi = phi*Xi;
    Q = G*G'*sigma_a;        % intial value for covariance Q.
    P = phi*P*phi'+Q;
    Xi = Xi+K*(Z(i)-H*Xi);
    xi(i) = Xi(1);
    K = P*H'/(H*P*H'+R);
    Ki(i,:) = K';
    P = (eye(2)-K*H)*P;
    Pi(i) = sqrt(P(1,1));
end
```

4) Plot results including true trajectory, measurements, filtered estimates of state vector X_i

```
% plotting results.
figure()
subplot(2,2,[1,2])
plot(1:200,X,'b',1:200,xi,'r',1:200,Z,'g')
title('Figure(1) Kalman filter results','color','r')
xlabel('points')
ylabel('position')
legend('True trajectory','Kalman','measurment','location','northeastoutside')
subplot(2,2,3)
plot(1:200,Ki(:,1),1:200,Ki(:,2))
title('Figure(2) Kalman filter gain','color','r')
xlabel('iteration')
ylabel('value')
legend('K position','K velocity')
grid on
subplot(2,2,4)
plot(1:200,Pi);
title('Figure(3) SD of estimation error','color','r')
xlabel('iteration')
ylabel('value')
grid on
```



Comments:

1- In the first graph are the true trajectory, the measurements and the Kalman filter estimation graphed against each other, it can be seen that the estimation values are kind of following the trajectory with some disturbances. However, if we zoom in a little bit, we will clearly see that still there is a nearly constant shift between both trajectories. the filtered data follows the true trajectory almost in parallel, keeping visually almost the same distance. And plotting the filter with different random trajectories leads to the same results.

2- Plotting the Kalman filter gain for the X vector, it can be seen that the filter gain for the velocity approaches zero faster than the position gain and that's because our measurements account only for the position, which leads to the observation matrix equal to H=[1 0] thus affects the value of the filtration gain for velocity. However, the filter gain related to the position of the trajectory approaches a constant value relatively fast.

3- Plotting the standard deviation resulting from the error covariance matrix, it also settles and reaches a constant value almost at the same rate of the filter gain.

Overall, both the standard deviation and filter gain settled for a constant value at a fast rate, thus verifying that given our conditions we cannot estimate more than the established limit of accuracy due to uncertainty.

5) Make $M = 500$ runs of filter and estimate dynamics of mean-squared error of estimation over observation interval.calculate this error only for filtered estimate of coordinate $x_{i,i}$.

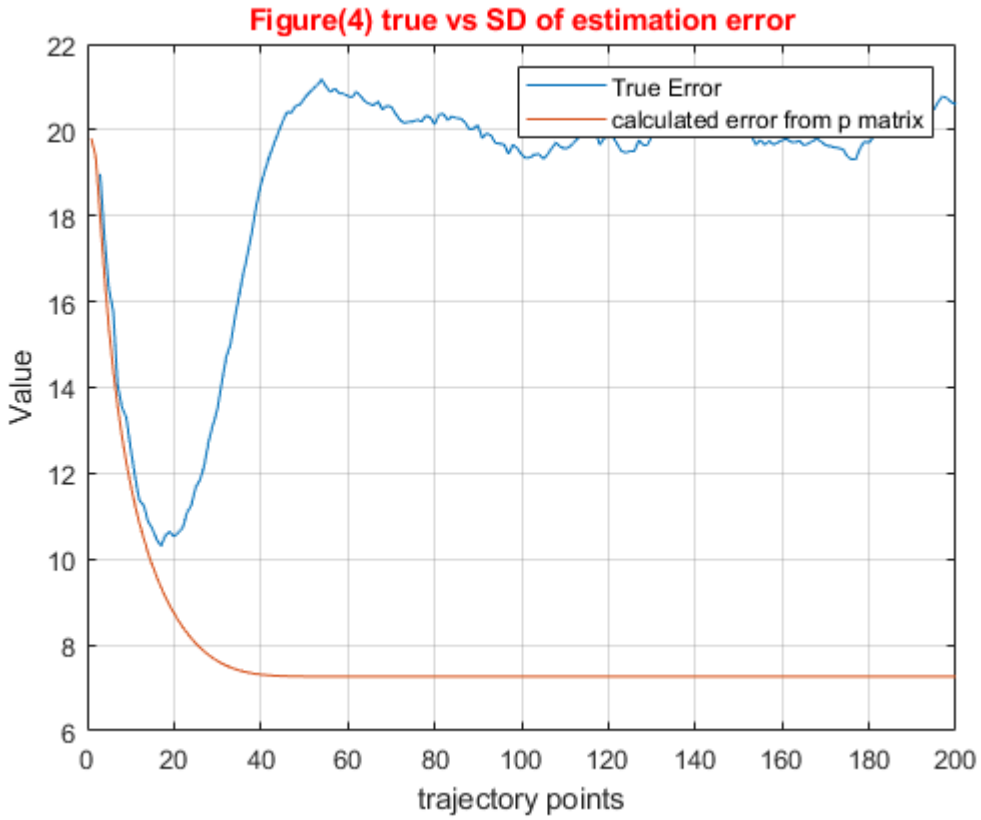
```
Err = zeros(500,200);

for M = 1:500
    % generating true trajectory
    normaldist = makedist('Normal',0,0.2);
    q = 0.2;
    a = random(normaldist,200,1)+q;     % random acceleration.
    X = zeros(1,200);
    X(1) = 5;
    V = zeros(1,200);
    V(1) = 1;
    T = 1;
    N = length(X);

    for i = 2:N
        X(i) = X(i-1)+V(i-1)*T+a(i-1)*T^2/2;
        V(i) = V(i-1)+a(i-1)*T;
    end

    % generating noise
    normaldist2 = makedist('Normal',0,20);
    eta = random(normaldist2,200,1);
    Z = zeros(1,200);

    for i=1:N
        Z(i)=X(i)+eta(i);
    end

    % Kalman filter
```

```
    phi = [1 T;0 1];
    G = [T^2; T];
    H = [1 0];
    xi = zeros(1,200);
    Xi = [2;0];
    P = [10000 0; 0 10000];
    R = 20^2;
    sigma_a = 0.2^2;
    K = P*H'/(H*P*H'+R);
    Ki = zeros(N,2);
    Pi = zeros(N,1);

    for i = 1:2
        Xi = phi*Xi;
        Q = G*G'*sigma_a;
        P = phi*P*phi'+Q;
        Xi = Xi+K*(Z(i)-H*Xi);
        xi(i) = Xi(1);
        K = P*H'/(H*P*H'+R);
        Ki(i,:) = K';
        P = (eye(2)-K*H)*P;
        Pi(i) = sqrt(P(1,1));
    end
    for i = 3:N
        Xi = phi*Xi;
        Q = G*G'*sigma_a;
        P = phi*P*phi'+Q;
        Xi = Xi+K*(Z(i)-H*Xi);
        xi(i) = Xi(1);
        K = P*H'/(H*P*H'+R);
        Ki(i,:) = K';
        P = (eye(2)-K*H)*P;
        Pi(i) = sqrt(P(1,1));
        Err(M,i) = (Xi(1)-X(i))^2;
     end
end

ErrAvg = zeros(1,N-2);
for j = 1:N-2
    ErrAvg(j) = sqrt((1/(M-1))*sum(Err(:,j+2)));
end
```

6) Compare true estimation error obtained in item 5 with errors of estimation $P_{i,i}$ provided by Kalman filter algorithm

```
% plot error
figure()
plot(3:N,ErrAvg); hold on
plot(Pi);
title('Figure(4) true vs SD of estimation error','color','r');
xlabel('trajectory points');
```

```
ylabel('Value');
legend('True estimation error','calculated error from p matrix');
grid on
```



In figure(4) above, Comparing the true estimation error and the calculation error obtained from the P matrix, it can be seen that they both settle for a certain value, but not the same value unlike what was obtained in the lab 5. The true estimation error is biased from the Kalman error with true error settling for the value ≈20 and the calculation error settling at 7.2644. Obviously, this is due to the fact that we haven't accounted for the bias in the acceleration.

7) Develop optimal Kalman filter algorithm that takes into account bias of acceleration (state noise).

```
Err = zeros(500,200);

for M = 1:500
    normaldist = makedist('Normal',0,0.2);
    q = 0.2;
    a = random(normaldist,200,1)+q;
    X = zeros(1,200);
    X(1) = 5;
    V = zeros(1,200);
    V(1) = 1;
    T = 1;
    N = length(X);
```

```
    for i = 2:N
        X(i) = X(i-1)+V(i-1)*T+a(i-1)*T^2/2;
        V(i) = V(i-1)+a(i-1)*T;
    end

    normaldist2 = makedist('Normal',0,20);
    eta = random(normaldist2,200,1);
    Z = zeros(1,200);

    for i = 1:N
        Z(i) = X(i)+eta(i);
    end

    phi = [1 T;0 1];
    G = [T^2; T];
    H = [1 0];
    xi = zeros(1,200);
    Xi = [2;0];
    P = [10000 0; 0 10000];
    R = 20^2;
    sigma_a = 0.2^2;
    K = P*H'/(H*P*H'+R);
    Ki = zeros(N,2);
    Pi = zeros(N,1);

    for i = 1:2
        Xi = phi*Xi+G*q;
        Q = G*G'*sigma_a;
        P = phi*P*phi'+Q;
        Xi = Xi+K*(Z(i)-H*Xi);
        xi(i) = Xi(1);
        K = P*H'/(H*P*H'+R);
        Ki(i,:) = K';
        P = (eye(2)-K*H)*P;
        Pi(i) = sqrt(P(1,1));
    end
    for i = 3:N
        Xi = phi*Xi+G*q;
        Q = G*G'*sigma_a;
        P = phi*P*phi'+Q;
        Xi = Xi+K*(Z(i)-H*Xi);
        xi(i) = Xi(1);
        K = P*H'/(H*P*H'+R);
        Ki(i,:) = K';
        P = (eye(2)-K*H)*P;
        Pi(i) = sqrt(P(1,1));
        Err(M,i) = (Xi(1)-X(i))^2;
     end
end
ErrAvg = zeros(1,N-2);
for j = 1:N-2
    ErrAvg(j) = sqrt((1/(M-1))*sum(Err(:,j+2)));
```

```
end

% plot error
figure()
plot(3:N,ErrAvg); hold on
plot(Pi);
title('Figure(5) true vs SD of estimation error in optimal conditions','color','r');
xlabel('trajectory points');
ylabel('Value');
legend('True estimation error','calculated error from p matrix');
grid on
```

It is shown in figure(5) above that after developing an optimal Kalman filter that accounts for the noise bias, the true estimation value decreased and was aligned with the calculation error from the P matrix with the value ≈7. Now that this filter modification was applied a better estimation of our true trajectory can be used.

Learning Log:

In this assignment we continued on applying the Kalman filter and its modification, and we visualized how a small difference in our model can affect the result obtained by the filter, so a good knowledge of our experimental environment and physics affects the end results. And in few points:

1- We have learnt how and when to apply the Kalman filter with noise bias and how the end result is affected.

2- The Kalman filter sensitivity to input parameters, the quality of filtration and that the Kalman filter accuracy doesn't increase or decrease given a certain statistical properties, and that's because either when we used the optimal Kalman filter with or without accounting for bias, the error reached a certain value and settled on it.

3- We wanted to experiment like in assignment 5 what if we use the Kalman filter accounting for bias with a deterministic model and the result was similar to when we didn't account for bias when it existed with an error shifted from the calculation error.