SSY345 – Sensor Fusion and Non-Linear Filtering
Home Assignment 2 – Implementation

Basic information

This home assignment is related to the material in lecture 3, 4 and 5. *The main focus is on understanding the properties of the Kalman filter and apply it in some concrete scenarios.*

In the analysis part we want you to use the toolbox that you have developed and apply it to a practical scenario. Associated with each scenario is a set of tasks that we would like you to perform.

It is important that you comment your code such that it is easy to follow by us and your fellow students. Poorly commented code will result in deduction of POE. Your code should be exported as a txt and uploaded as part of your submission before the deadline. The purpose is to make feedback from your peers possible and to enable plagiarism analysis (Urkund) of all your submissions.

The result of the tasks should in general be visualised and compiled together in a report (pdf-file). A template for the report can also be found on course homepage. Note that, it is sufficient to write short concise answers to the questions but they should be clearly motivate by what can be seen in the figures. Only properly referenced or captioned figures such that it is understandable what will result in POE. Also, all the technical terms and central concepts to explain an answer should be used without altering their actual meaning. The report should be uploaded on the course homepage before the deadline.

Scenario 1 – A first Kalman filter and its properties

In this scenario you will study the properties of the Kalman filter in a simple one-dimensional case (all variables are scalars).

Consider the following linear and Gaussian state space model:

$$x_k = x_{k-1} + q_{k-1}, \tag{1}$$
$$y_k = x_k + r_k \tag{2}$$

where the motion and measurement noises, q_{k-1} and r_k, are zero mean Gaussian random variables with variances $Q = 1.5$ and $R = 3$, respectively, and the initial prior is $p(x_0) = \mathcal{N}(x_0; 2, 8)$.

Task: To get a feeling for how a Kalman filter performs we would like you to do the following tasks and reflect on what you observe.

a) Generate a state sequence $x_0, x_1, \ldots$ and a measurement sequence $y_1, y_2, \ldots,$ from the above motion model and plot the result in the same figure. Suitable length of the sequence could be 35. Does the measurement behave according to the model? Motivate!

b) Filter the measurement sequence using the Kalman filter that you have implemented. Plot the sequence of estimates together with the $\hat{x}_k \pm 3\sigma$ level. In the same figure, also plot the correct states and the measurements. Are the estimates that the filter outputs reasonable, if so, in what way? Does the error covariance represent the uncertainty in the estimates well?
Additionally, in a separate figure, plot the error density around zero-mean for time instances $k = [1, 2, 4, 30]$ and motivate what you see in the figure. *Hint:* The command `normpdf` in MATLAB can be useful in plotting Gaussian densities.

c) For this task only you study what happens if you use an incorrect prior at time 0. Generate data from the models in (1) and (2), run the corresponding Kalman filter and plot $\hat{x}_{k|k}$. Now, run the filter again but this time, tell the Kalman filter that the initial mean is instead 12, such that the filter thinks that $p(x_0) = \mathcal{N}(x_0; 12, 8)$? Clearly, the filter will not perform as well but what will happen over time? Run the filter under the incorrect assumption and plot the new sequence of estimates $\hat{x}_{k|k}$ in the same graph as the estimates from the correct Kalman filter (you should not generate new data). Motivate the results that you get.

d) Plot $p(x_{k-1} \big| y_{1:k-1})$, $p(x_k \big| y_{1:k-1})$, y_k and $p(x_k \big| y_{1:k})$ for a choice of k which you think is illustrative. Which conclusions can you draw from the illustration regarding the behaviour of the prediction and update step of the Kalman filter? Is the behaviour reasonable? Give a short motivation!

e) **Consistency:** After awhile, the system matrices will become constant over time which means $P_{k|k}$, $K_{k|k}$, $P_{k|k-1}$ and S_k converge to constant values. Once the system has reached its stationary condition we can replace ensemble averages with time averages, which is convenient since we can then analyse the filter using a single time sequence. Generate a long true state sequence from which you generate a corresponding measurement sequence. Do the following tasks to investigate the properties of some of the parts of the Kalman filter.

   - Calculate the estimated mean of the sequence and plot a histogram of the estimation error $x_k - \hat{x}_{k|k}$. Compare the histogram to the pdf $\mathcal{N}(x; 0, P_{N|N})$, where N is the length of the sequence. In the comparison, it could be a good idea to normalise the histogram with the length of your sequence such that they have similar hight. Comment on the results! What conclusion can we draw from this regarding the interpretation of $P_{k|k}$?
   - Estimate mean and the auto correlation function of the innovation process, v_k[1], from the sequence of innovations and plot the results. Comment on which conclusions can be drawn from what we see. *Hint:* The command `autocorr` in MATLAB can be useful.

[1] v_k is calculated in the update step of the Kalman filter and you can, for example, modify the `linearUpdate` function to also output these to be able to store them outside the function.

Scenario 2 – Kalman filter and its tuning

Consider the following state space model

$$\mathbf{x}_k = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \mathbf{x}_{k-1} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} q_{k-1}, \tag{3}$$

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k + r_k \tag{4}$$

where the motion noise is $q_{k-1} \sim \mathcal{N}(0, 1.5)$, the measurement noise is $r_k \sim \mathcal{N}(0, 2)$, the initial prior is $p(\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0; [1\,3]^T, 4\mathbf{I})$ and the time step is $T = 0.01s$. Note that the above motion model is a one-dimensional constant velocity model where the first state is the position and the second state is the velocity. If you prefer to work with vector noise, you should note that

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} q_{k-1} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1.5 \end{bmatrix} \right) \tag{5}$$

As you can be seen from (4), we only observe the position state (plus noise).

Task:

a) Generate a state sequence $x_0, x_1, \ldots$ and a measurement sequence $y_1, y_2, \ldots$, from the above motion model and measurement model. Illustrate the position state sequence and the measurement sequence in the same figure. Illustrate the velocity state in its own separate figure. Does the result look reasonable? Give a short but concise motivation!

b) Filter the generated measurements using your Kalman filter and the process and measurement models above. Illustrate true state, measurement sequence and estimated state (together with $\pm 3\sigma$ level) in the same figure. Again, use separate figures for position and velocity. Comment concisely on your results and the behaviour of the filter.

c) **Tuning the filter**: The choice of motion noise variance Q and measurement noise variance R can affect the performance of the Kalman filter substantially. It is not always the case that the "true" values of these parameters are provided to us. In real applications, these parameters must be either estimated or tuned in order to obtain desirable performance from the filter.

   In this task, we generate the true states and the measurements by fixing the motion noise variance to $Q = 1.5$ and measurement noise variance to $R = 2$, as in task a). We assume the true value of R is known also to the filter, but pretend that the true value of the motion noise variance (i.e.,$Q = 1.5$) is unknown, which is the case in typical applications. We then attempt to tune the filter by varying the motion noise variance, for the following values of Q:
   i) Q = 0.1   ii) Q = 10   iii) Q = 1.5

For a long sequence of data, plot the autocovariance of the error of the position and motivate your results.

Note: The data must be generated using (3) and (4) using true values of Q and R. The motion noise variance used in the filter computations is what is being tuned.
Hint: The command `xcov` in MATLAB can be useful here. You may want to normalize your results regarding the number of samples.