Stochastic Processes, Sessions 18, 19 & 20 — Group Work
Kalman Filtering

Troels Pedersen and Carles Navarro, Aalborg University

November, 2017

Go through the exercises below. Allow yourself the time to reflect over your results and discuss them with other students! Use the book and lecture notes for inspiration and for further information.

1 Scalar LMMSEE for First-order AR Process

In this problem we consider the estimation of a random process $\{Y(n)\}$ from observations of a random process $\{X(n)\}$. We model the (unobservable) random process $\{Y(n)\}$ as a first-order autoregressive (AR) process

$$Y(n) = h \cdot Y(n-1) + Z(n), \qquad n = 1, 2, \ldots, \tag{1}$$

where $\{Z(n)\} \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_Z^2)$ and h a known deterministic constant. The random variable $Y(0) \sim \mathcal{N}(0, \sigma_Y^2)$ is assumed independent of $\{Z(n)\}$. To ensure that $\{Y(n)\}$ is WSS, we set $\sigma_Y^2 = \sigma_Z^2/(1-h^2)$. The observable process $\{X(n)\}$ is modeled as

$$X(n) = Y(n) + W(n), \qquad n = 1, 2, \ldots, \tag{2}$$

with $\{W(n)\} \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_W^2)$.

1. Implement the model and plot realizations of $\{Y(n)\}$ and $\{X(n)\}$ for different settings of h, σ_Z^2 and σ_W^2. Compare and discuss the results. What happens if h is negative?

For each n we now use a linear minimum mean square error estimator (LMMSEE) to estimate $Y(n)$ based on the observations of $X(1), \ldots, X(n)$. From now on, we set $h = 0.95$, $\sigma_Z^2 = 0.1$ and $\sigma_W^2 = 1$.

2. Derive the LMMSEE for $Y(n)$ given the data $X(1), \ldots, X(n)$. (*Hint: recall that the ACF of the AR process in* (1) *is given by* $R_Y(k) = \sigma_Z^2 \frac{h^{|k|}}{1-h^2}$).

3. Implement the LMMSEE in a flexible way such that you can test it for different settings of n. Plot and compare the simulated mean square error (MSE) to the theoretical MSE for $n = 1, \ldots, 30$. (*Hint: The* `toeplitz` *command in MATLAB may be useful to generate covariance matrices.*)

4. Push your simulator to the extreme — how many observations can you use before the simulator breaks down or gets too slow? What causes this effect?

2 Scalar Kalman Filter for First-order AR Process

We consider the same models as given in (1) and (2) in the previous problem. We set the parameters

$$h = 0.95, \qquad \sigma_Z^2 = 0.1 \qquad \sigma_W^2 = 1 \tag{3}$$

For each n we now use a Kalman filter to estimate $Y(n)$ based on the observation of $X(1), \ldots, X(n)$.

1. Draw a block diagram of the state and observation model. State the equations for the prediction step and the updating step of the Kalman filter.
2. Next, write up (by hand) the prediction and updating steps, including the Kalman gain and the MSEs, for times $n = 1$ and $n = 2$ assuming that the first two observations received have values $X(1)$ and $X(2)$. Avoid inserting the numerical values of the parameters given in (3) and use their notations instead. Choose as well the initial values $\hat{Y}(0|0)$ and $R(0|0)$.
3. Implement the Kalman filter in Matlab and investigate the behavior of the simulated mean squared estimation error as a function of n by running multiple simulations. Compare to the theoretical MSE $R(n|n)$ and to the results for the LMMSEE in Problem 1. (*Hint: Refer to the lecture notes for initialization of the Kalman filter*).
4. Run the Kalman filter for large n values. Compare to the LMMSEE case.
5. Vary the parameters h, σ_Z^2 and σ_W^2 and investigate how it affects $R(n|n)$.
6. Initialize the filter differently i.e. with $\hat{Y}(0|0) = 100$ and $R(0|0) = 0.0001$. Investigate how it affects $R(n|n)$.
7. Calculate the steady state solutions for the MSE and compare to the simulated MSE for your Kalman filter.

3 Vector Kalman Filter — Target Tracking

In this problem we consider the same situation as described in Section 5.4 of the lecture notes. We encourage you to carefully read all the provided information on this example.

Assume that $U(t)$ and $V(t)$ represent, respectively, the position and the velocity of a target at time t. Then we can write

$$U(t) = \int_0^t V(\tau) d\tau + U(0).$$

Using a sampling time of one second (denoted by T_s in the lecture notes), this integral equation can be approximated by the following difference equation

$$V(n-1) = U(n) - U(n-1), \qquad n \geq 1. \tag{4}$$

We assume that the sequence

$$G(n) = V(n) - V(n-1), \qquad n \geq 1 \tag{5}$$

of differences between consecutive velocity samples (acceleration) can be modeled as a white Gaussian driving sequence with variance 0.0001, i.e. we assume that

$$\mathbb{E}[G(n)] = 0,$$
$$\mathbb{E}\big[G(n)G(n+k)\big] = 0.0001\delta(k).$$

Now combining (4) and (5) in matrix notation yields the state model

$$\begin{bmatrix} U(n) \\ V(n) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} U(n-1) \\ V(n-1) \end{bmatrix} + \begin{bmatrix} 0 \\ G(n) \end{bmatrix}, \qquad n \geq 1. \tag{6}$$

Written in a compact form, (6) becomes

$$\boldsymbol{Y}(n) = \boldsymbol{H}(n)\boldsymbol{Y}(n-1) + \boldsymbol{Z}(n), \qquad n \geq 1,$$

where

$$\boldsymbol{Y}(n) := \big[U(n) \quad V(n)\big]^\top, \qquad \boldsymbol{H}(n) := \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{Z}(n) := \big[\, 0 \quad G(n)\big]^\top.$$

From the above assumptions on $\{G(n)\}$, the covariance matrix of $\boldsymbol{Z}(n)$ reads

$$\boldsymbol{Q}_Z(n) := \mathbb{E}\big[\boldsymbol{Z}(n)\boldsymbol{Z}(n)^\top\big] = \begin{bmatrix} 0 & 0 \\ 0 & 0.0001 \end{bmatrix}.$$

We model the observations as

$$X(n) = U(n) + W(n), \qquad n \geq 1, \tag{7}$$

where $X(n)$ is a scalar, representing the observed position of the target and $\{W(n)\}$ is a white Gaussian noise sequence with variance $\sigma_W^2 = 0.1$. Written in matrix-vector notation, (7) becomes

$$X(n) = \boldsymbol{A}(n)\boldsymbol{Y}(n) + W(n),$$

where $\boldsymbol{A}(n) = \big[\, 1 \quad 0 \,\big]$.

1. Extend the state and observation model to the 2-dimensional (2-D) case, i.e. movement in the plane.
2. State $\boldsymbol{H}(n)$, $\boldsymbol{Y}(n)$, $\boldsymbol{Z}(n)$, $\boldsymbol{Q}(n)$ and all the other matrices and vectors used in the Kalman filter. Implement the vector Kalman filter for the 2-D case in Matlab. For initialization purposes, we assume that the target starts at $x = 0$ and $y = 0$, $\hat{\boldsymbol{Y}}(0|0) = \big[\, 0\ 0\ 0\ 0 \,\big]^\top$, and $\boldsymbol{R}(0|0) = \boldsymbol{I}$.

The zip-file *targetTracks.zip* available in Moodle contains four different trajectory files named $Track1.mat$, $Track2.mat$, $Track3.mat$ and $Track4.mat$. Each trajectory contain two vectors forming a sequence of true 2-D coordinates of the target. The X-coordinate is in the vector named $Xdir$ and the Y-coordinate is named $Ydir$.

3. Plot Track1 by using $Xdir$ and $Ydir$ as the x axis and y axis, respectively. Generate noisy observations for both directions.

4. Apply the Kalman filter to estimate the position of the target by using the noisy observations. Investigate the behavior of the simulated mean squared estimation error (MSE) as a function of n by running multiple simulations. Compare the obtained MSE curve with the theoretical MSE $R(n|n)$.

5. Now, we want to investigate how to tune the parameters of the Kalman filter to increase the estimation accuracy by adjusting the variance of the driving process: (1) adjust the variance of the driving process on the x-direction while keeping the variance of the driving process on the y-direction to be 0.0001. (2) adjust the variance of the driving process on the y-direction while keeping the variance of the driving process on the x-direction to be 0.0001. Compare the predicted, the estimated and the true trajectory of the target. Compare the simulated MSE when you adjust the variance of the driving process for these two cases. State your conclusions.

6. Repeat the same procedures for the remaining tracks. How do you adjust your Kalman filter parameters to increase the estimation accuracy? State your conclusions.