02471 Machine Learning for Signal Processing

Week 11: State-space models – Kalman filtering

This exercise is based on S. Theodoridis: Machine Learning, A Bayesian and Optimization Perspective 2nd edition, section(s) 4.10.

The objective of this exercise is to get a view on the usage of state-space models in the linear dynamical system setting. As application we choose the most often used approach and use the Kalman filter, where some parameters are assumed to be known. These parameters can for example be learned using the EM algorithm (just like the HMM).

Overview

The exercise have the following structure:

11.1 will derive the Kalman filter equations.

11.2 will experiment with the derived formulas, and get acquainted on how Kalman filter works in a 1-d setting of an AR(1) process.

11.3 is a demo where we use Kalman filtering and a RTS smoother to track an moving object using noisy observations.

Notation

- $J(\boldsymbol{\theta})$ is a cost function that we are seeking to minimize with respect to $\boldsymbol{\theta}$.
- $\mathcal{N}(\mu, \sigma^2)$ denotes a normal (Gaussian) distribution with mean value μ and variance σ^2.
- $\hat{a}_{n|n-1}$ denotes the estimation of signal a at time n given data observed until time $n-1$.

Code

The code can be found in the .m and .py files named in the same way as exercises, ie. the code for exercise 11.x.y is in the file 11_x_y.m (or .py).

For coding exercises that requires implementation we will usually write `complete this line` where the implementing should be done.

Solutions

The solution is provided for all derivation exercises, and often hints are provided at the end of the document. If you get stuck, take a look at the hints, and if you are still stuck, take a look in the solution to see the approach being taken. Then try to do it on your own.

Solutions are also provided for some coding exercises. If you get stuck, take a look at the solution, and then try to implement it on your own.

11.1 Derivation of the Kalman filter

We consider the space-state model

$$\mathbf{x}_n = F_n \mathbf{x}_{n-1} + \boldsymbol{\eta}_n$$
$$\mathbf{y}_n = H_n \mathbf{x}_n + \mathbf{v}_n$$

The goal is to predict $\mathbf{x}_n$ but we have only observed $\mathbf{y}_n$. If we have observed up to $n-1$ points, we call the estimator for $\mathbf{x}_n$ the prior estimator since we estimate without having observed $\mathbf{y}_n$. We denote this estimator $\hat{\mathbf{x}}_{n|n-1}$.

If we have observed up to n points, we call the estimator for $\mathbf{x}_n$ the posterior estimator since we now estimate with having observed $\mathbf{y}_n$. We denote this estimator $\hat{\mathbf{x}}_{n|n}$.

To goal of the Kalman filter is to provide expressions for these two estimators in a recursive manner. Then we can, as we acquire a new point $\mathbf{y}_n$ do a re-estimation and not solve the problem anew. This nature of this approach is very similar to LMS and RLS, and for that reason, Kalman filtering is often talked about as an RLS algorithm.

In Kalman filtering, we make the following assumptions:

- The distribution of the noise terms are known, and have the following properties
  - $\mathbb{E}\left[\boldsymbol{\eta}_n \boldsymbol{\eta}_n^T\right] := Q_n$
  - $\mathbb{E}\left[\boldsymbol{\eta}_n \boldsymbol{\eta}_m^T\right] = 0, n \neq m$
  - $\mathbb{E}[\boldsymbol{\eta}_n] = \mathbf{0}$
  - $\mathbb{E}\left[\mathbf{v}_n \mathbf{v}_n^T\right] := R_n$
  - $\mathbb{E}\left[\mathbf{v}_n \mathbf{v}_m^T\right] = 0, n \neq m$
  - $\mathbb{E}[\mathbf{v}_n] = \mathbf{0}$
  - $\mathbb{E}\left[\boldsymbol{\eta}_n \mathbf{v}_m^T\right] = 0, \forall n, \forall m$
- The matrices F_n and H_n are known.

Exercise 11.1.1

Describe in words what these properties mean. Are these assumptions reasonable?

Exercise 11.1.2

We make the following definitions

$$\hat{\mathbf{y}}_n := H_n \hat{\mathbf{x}}_{n|n-1}$$
$$\mathbf{e}_n := \mathbf{y}_n - \hat{\mathbf{y}}_n$$
$$\mathbf{e}_{n|n} := \mathbf{x}_n - \hat{\mathbf{x}}_{n|n}$$
$$\mathbf{e}_{n|n-1} := \mathbf{x}_n - \hat{\mathbf{x}}_{n|n-1}$$
$$P_{n|n} := \mathbb{E}\left[\mathbf{e}_{n|n} \mathbf{e}_{n|n}^T\right]$$
$$P_{n|n-1} := \mathbb{E}\left[\mathbf{e}_{n|n-1} \mathbf{e}_{n|n-1}^T\right]$$

Discuss these definition and describe what they mean in words. Why e.g. do we define $\hat{\mathbf{y}}_n$ with respect to $\hat{\mathbf{x}}_{n|n-1}$ and not $\hat{\mathbf{x}}_{n|n}$?

Exercise 11.1.3

We first seek expressions for the predictions $\hat{\mathbf{x}}_{n|n-1}$ and $P_{n|n-1}$. For predicting $\hat{\mathbf{x}}_{n|n-1}$ we simply use the model and have a noise-free prediction, so that

$$\hat{\mathbf{x}}_{n|n-1} = F_n \hat{\mathbf{x}}_{n-1|n-1}$$

To get an expression for $P_{n|n-1}$ we have to use substitutions of the previous definitions. Show that

$$P_{n|n-1} = F_n P_{n-1|n-1} F_n^T + Q_n$$

Exercise 11.1.4

The next step is to derive the updates.

In Kalman filtering we define the following recursion

$$\hat{\mathbf{x}}_{n|n} := \hat{\mathbf{x}}_{n|n-1} + K_n \mathbf{e}_n$$

This is simply a choice that is being made, stating that we aim to update our prediction in an adaptive manner as points become available. We have seen this update form many times before in the course, e.g. relate this to the LMS/RLS algorithm update. It simply means, at iteration n, we have to make a correction to our previous point. This correction is based on the prediction error $\mathbf{e}_n$ and some correction K_n. We aim to derive an expression for the correction factor.

To this end, we choose to minimize the mean squared error of the prediction performance of $\hat{\mathbf{x}}_{n|n}$, that is, we minimize the following with respect to K_n:

$$J(K_n) = \mathbb{E}\big[\mathbf{e}_{n|n}^T \mathbf{e}_{n|n}\big]$$

Using the previously stated definitions we have (show this)

$$J(K_n) = \text{trace}(P_{n|n})$$

In order to take the derivate of $J(K_n)$ w.r.t. K_n we need to rewrite $P_{n|n}$. By using the previous definitions, show

$$P_{n|n} = \mathbb{E}\big[\mathbf{e}_{n|n-1}\mathbf{e}_{n|n-1}^T\big] - K_n \mathbb{E}\big[\mathbf{e}_n \mathbf{e}_{n|n-1}^T\big] - \mathbb{E}\big[\mathbf{e}_{n|n-1}\mathbf{e}_n^T\big] K_n^T + K_n \mathbb{E}\big[\mathbf{e}_n \mathbf{e}_n^T\big] K_n^T$$

Next, show that these individual expectations can be written as

$$\begin{aligned}
\mathbb{E}\big[\mathbf{e}_{n|n-1}\mathbf{e}_{n|n-1}^T\big] &= P_{n|n-1} \\
\mathbb{E}\big[\mathbf{e}_n \mathbf{e}_{n|n-1}^T\big] &= H_n P_{n|n-1} \\
\mathbb{E}\big[\mathbf{e}_{n|n-1}\mathbf{e}_n^T\big] &= P_{n|n-1} H_n^T \\
\mathbb{E}\big[\mathbf{e}_n \mathbf{e}_n^T\big] &= H_n P_{n|n-1} H_n^T + R_n
\end{aligned}$$

Let $S = H_n P_{n|n-1} H_n^T + R_n$, so that $\mathbb{E}\big[\mathbf{e}_n \mathbf{e}_n^T\big] = S$, then put it all together to get

$$P_{n|n} = P_{n|n-1} - K_n H_n P_{n|n-1} - P_{n|n-1} H_n^T K_n^T + K_n S K_n^T$$

Exercise 11.1.5

We have to minimize trace($P_{n|n}$) with respect to K_n, so we take the derivate w.r.t. K_n and set to zero. Using that trace is a linear operator i.e. trace$(A+B) = $ trace$(A) +$ trace(B), and trace$(A) =$ trace(A^T) and using the rules

$$\frac{\partial \text{trace}(AB)}{\partial A} = B^T$$
$$\frac{\partial \text{trace}(ACA^T)}{\partial A} = 2AC$$

First show trace$(P_{n|n-1}H_n^T K_n^T) =$ trace$(K_n H_n P_{n|n-1}^T)$ and then use this to get

$$\text{trace}(P_{n|n}) = \text{trace}(P_{n|n-1}) - 2\text{trace}(K_n H_n P_{n|n-1}^T) + \text{trace}(K_n S K_n^T)$$

Take the derivative, set to zero to obtain

$$K_n = P_{n|n-1}H_n^T S^{-1}$$

That means we know have an expression for K_n that minimizes the MSE. We note that to calculate K_n, we need an expression for $P_{n|n-1}$ (the other terms are considered known per our assumptions), this is the next and final step.

Exercise 11.1.6

Go back to the previously derived recursion for $P_{n|n}$, show that $K_n S K_n^T = P_{n|n-1}H_n^T K_n^T$, and by substitution obtain

$$P_{n|n} = P_{n|n-1} - K_n H_n P_{n|n-1}$$

We have now derived the Kalman filtering algorithm. Relate the derived terms to algorithm 4.2, page 169 in the book.

11.2 Kalman filtering on an AR process

As we have discussed in the course, many signals can be modeled adequately as an AR process, so we can gain insight with Kalman filtering by assuming $\mathbf{x}$ follows an AR process, simulate data, and then see how reliably Kalman filter can predict $\mathbf{x}_n$ for the observations $\mathbf{y}_n$.

In this exercise we work through example 4.4 (although we use positive a_i coefficients instead of negative as the book).

Exercise 11.2.1

Given the following model for data generation

$$\mathrm{x}_n = \sum_{i=1}^{l} a_i \mathrm{x}_{n-i} + \eta_n$$
$$\mathrm{y}_n = \mathrm{x}_n + \mathrm{v}_n$$

where η_n is a white noise sequence with variance σ_η^2 and v_n is a white noise sequence with variance σ_v^2, specify the matrices F_n and H_n so that the model fits the state-space model. Are they fixed or time-varying (do we need the n subscript index?)?

Exercise 11.2.2

Inspect the code associated with the exercise and complete the code-lines. Generate some data using different parameters for Q_n, R_n, and a_i, and experiment with how well the Kalman filter performs for different noise levels and length of process.

11.3 Kalman filtering and smoothing on a moving object

This exercise serves as a demonstration on using Kalman filtering and smoothing on a moving object using noisy observations. The code only serves as a demo.

Inspect the code just to get an overview of the structure. Experiment with different noise parameters and sampling rate. Consider to change the script so that the filtering is interactive and happens on the fly. Another change to consider is the limit the smoother only to take a few points of the future into account instead of the entire trajectory, or to run the smoother as a window.

NOTE: The RTS smoother is NOT covered in the curriculum, it is only in this exercise as a demo.

HINTS