Dynamical Inverse Problems and

Kalman Filtering

Erkki Somersalo
Institute of Mathematics
Helsinki University of Technology

INTRODUCTORY EXAMPLES

Problem: Given a sequence of direct or indirect noisy observations of a time dependent system, try to predict and estimate the current state.

Example 1: Time varying heating. Estimate the temperature distribution within the body.

Example 2: Prediction of stock prices from market indeces.

Example 3: Noisy sampling of a time varying signal.

MODELLING EQUATIONS

Assume that the system can be described in terms of a Markov model + noisy obsevation.

State equations:

$$\begin{aligned} x_{k+1} &= A_k x_k + w_{k+1}, \quad k = 1, 2, \ldots \\ y_k &= B_k x_k + v_k, \quad k = 1, 2, \ldots \end{aligned}$$

Here,

$$\begin{aligned} x_k &= \text{state vector at } t = t_k,\ x_k \in \mathbb{R}^n \\ y_k &= \text{observation vector at } t = t_k,\ y_k \in \mathbb{R}^m \\ w_k &= \text{state noise vector at } t = t_k,\ w_k \in \mathbb{R}^n \\ v_k &= \text{observation noise vector at } t = t_k,\ v_k \in \mathbb{R}^m. \end{aligned}$$

DERIVING STATE EQUATIONS

Finding an appropriate state evolution equation is a mathematical modelling problem. Typically, it can be

- a discretized differential equation
- a time series model
- a random walk model

Example 1: Heat equation:

$$\begin{aligned}
\kappa \frac{\partial^2 T}{\partial s^2} - \frac{\partial T}{\partial t} &= 0, \\
\left(\alpha \frac{\partial T}{\partial s} - T\right)(0,t) &= T_{\text{flame}}(t), \\
\left(\alpha \frac{\partial T}{\partial s} - T\right)(L,t) &= T_0(t) = \text{observation.}
\end{aligned}$$

Discretization:

$$\begin{aligned}
x_k &= (T(s_1,t_k), T(s_2,t_k), \ldots, t(s_n,t_k))^{\mathrm{T}} \\
&= (x_k^1, x_k^2, \ldots, x_k^n)^{\mathrm{T}} \in \mathbb{R}^n, \\
y_k &= T_0(t_k) + \text{measurement noise} \in \mathbb{R}
\end{aligned}$$

The unknown flame temperature is hidden in the state noise.

Example 2: Select n most important companies. Let $P_j(t_k)$ denote the price of a stock of company number j at time t_k. Use an AR(1)–model

$$P_j(t_k) = \sum_{\ell=1}^{n} \alpha_\ell^j P_\ell(t_{k-1}) + Z_j(t_k).$$

Observation: Assume that the market index depends mostly on the prices of the stocks of the n major companies:

$$y(t_k) = \sum_{j=1}^{n} \beta_j P_j(t_k) + n(t_k).$$

State vector consists of the current stock prices of the selected companies:

$$x_k = (P_1(t_k), P_2(t_k), \ldots, P_n(t_k))^{\mathrm{T}}.$$

State noise describes the insecurity of the AR-model, observation noise describes the insecurity of the linear dependence of the market index of the selected stock prices.

Problem 3: Denote by $x_k = (x_k^0, \ldots, x_k^{n-1})^{\mathrm{T}} \in \mathbb{R}^n$ the discretized signal at $t = t_k$. Observe every P th sample point with noise:

$$y_k = (x_k^0, x_k^P, x_k^{2P}, \ldots, x_k^{mP})^{\mathrm{T}} + \text{ noise } \in \mathbb{R}^m.$$

Random walk model for time evolution:

$$x_{k+1} = x_k + w_k,$$

where w_k is *random walk process.*
Observation equation:

$$y_k = Bx_k + v_k,$$

where $B \in \mathbb{R}^{m \times n}$ is a matrix whose k th row is

$$\begin{array}{ccccc} & & (kP) & & \\ (0 & \ldots & 1 & \ldots & 0) \end{array}$$

ESTIMATION PROBLEMS

Prediction problem:
Based on the past observations

$$y_1, y_2, \ldots, y_{k-1},$$

find a *prediction* $\hat{x}_k$ for x_k that is optimal in some sense.

Filtering problem:
Based on the observations up to the present,

$$y_1, y_2, \ldots, y_{k-1}, y_k,$$

find a *filtered estimator* $x_k^{\#}$ for x_k that is optimal in some sense.

Smoothing problem:
Based on the observations of the whole data set

$$y_1, y_2, \dots, y_{k-1}, y_k, \dots y_N$$

find a *smoothed estimator* x_k^* for x_k that is optimal in some sense.

OPTIMALITY

A predictor $\hat{x}_k$ of the vector is *optimal in the mean square (MS) sense* if the expected mean square error

$$MSE = E|\hat{x}_k - x_k|^2$$

is minimized. In general, the MS–optimal predictor is the conditional expectation,

$$\hat{x}_k = E(x_k \mid y_1, \dots, y_{k-1}).$$

Similarly, the MS–optimal estimator is

$$x_k^{\#} = E(x_k \mid y_1, \dots, y_{k-1}, y_k),$$

and

$$x_k^* = E(x_k \mid y_1, \dots, y_{k-1}, y_k, \dots, y_N),$$

Under general assumptions, these vectors do not depend linearly on the observation. Yields a computationally heavy algorithm.

OPTIMALITY + LINEARITY

In order to obtain computationally effective prediction and filtering algorithms, we seek a predictor and estimator that are

1. Linear with respect to the observation:

$$\begin{aligned} \hat{x}_k &= M_1 y_1 + M_2 y_2 + \dots + M_{k-1} y_{k-1}, \\ x_k^{\#} &= N_1 y_1 + N_2 y_2 + \dots + N_{k-1} y_{k-1} + N_k y_k, \end{aligned}$$

2. Optimal in the MS–sense among all *linear* predictors and estimators.

SIMPLIFYING ASSUMPTIONS

- Matrices $A_k \in \mathbb{R}^{n\times n}$ and $B_k \in \mathbb{R}^{m\times n}$ are known for all $k = 1, 2, \ldots$.
- The noise vectors are zero mean with known kovariances:

$$Ew_k = 0, \qquad Ew_k^{\mathrm{T}} w_{k'} = \delta_{k,k'} Q_k \in \mathbb{R}^{n\times n},$$
$$Ev_k = 0, \qquad Ev_k^{\mathrm{T}} v_{k'} = \delta_{k,k'} R_k \in \mathbb{R}^{m\times m},$$

- w_k is uncorrelated with x_1.
- The state noises and measurement noises are uncorrelated.

RECURSIVE LINEAR ESTIMATION

Linear predictor and estimator are *recursive*, if

$$\begin{aligned} \hat{x}_k &= L_k \hat{x}_{k-1} + M_k y_{k-1}, \\ x_k^{\#} &= P_k x_{k-1}^{\#} + N_k y_k, \end{aligned}$$

ANALOG FROM LINEAR ALGEBRA

Given vectors

$$v_1, v_2, \ldots, v_N \in \mathbb{R}^n.$$

Let $u \in \mathbb{R}^n$. Denote

$$\begin{aligned} P(u \mid v_1, \ldots, v_k) &= \text{orthogonal projection of } u \\ &\ \text{on the subspace spanned by } v_1, \ldots, v_k \end{aligned}$$

Write

$$\begin{aligned} v_k &= (v_k - P(v_k \mid v_1, \ldots, v_{k-1})) + P(v_k \mid v_1, \ldots, v_{k-1}) \\ &= v_k^{\perp} + v_k^{\parallel}. \end{aligned}$$

$$
\begin{aligned}
P(u \mid v_1, \ldots, v_k) &= P(u \mid v_1, \ldots, v_{k-1}, v_k^{\perp} + v_k^{\parallel}) \\
&= P(u \mid v_1, \ldots, v_{k-1}, v_k^{\perp}) \\
&= P(u \mid v_1, \ldots, v_{k-1}) + P(u \mid v_k^{\perp})
\end{aligned}
$$

KALMAN FILTERS: DERIVATION

Let $P(u \mid y_1, \ldots y_k)$ denote an orthonormal projection in the probabilistic sense:

$$E(u - P(u \mid y_1, \ldots y_k))y_j^{\mathrm{T}} = 0 \text{ for } j = 1, \ldots, k.$$

Denote

$$\hat{y}_k = P(y_k \mid y_1, \ldots y_{k-1}).$$

Innovation process

$$\mu_k = y_k - \hat{y}_k.$$

Define the Kalman predictor

$$\hat{x}_k = P(x_k \mid y_1, \ldots, y_{k-1})$$

and Kalman filter

$$x_k^{\#} = P(x_k \mid y_1, \ldots, y_k).$$

Then

$$
\begin{aligned}
\hat{x}_k &= P(x_k \mid y_1, \ldots, y_{k-1}) \\
&= P(A_k x_{k-1} + w_k \mid y_1, \ldots, y_{k-1}) \\
&= A_k P(x_{k-1} \mid y_1, \ldots, y_{k-1}) \\
&= A_k P(x_{k-1} \mid y_1, \ldots, y_{k-2}) + P(A_k x_{k-1} \mid \mu_{k-1}) \\
&= A_k \hat{x}_{k-1} + K_k \mu_{k-1},
\end{aligned}
$$

where the matrix K_k is the *Kalman gain* matrix. Similarly,

$$
\begin{aligned}
x_k^{\#} &= P(x_k \mid y_1, \ldots, y_k) \\
&= P(x_k \mid y_1, \ldots, y_{k-1}) + P(x_k \mid \mu_k) \\
&= \hat{x}_k + \tilde{K}_k \mu_k.
\end{aligned}
$$

KALMAN PREDICTOR AND FILTER

The Kalman predictor $\hat{x}_k$ and the covariance of the prediction error, $\hat{C}_k$, are given by the formulas

$$\begin{aligned}
\hat{x}_{k+1} &= A_k\hat{x}_k + K_k(y - B_k\hat{x}_k), \\
\hat{C}_{k+1} &= (A_k - K_kB_k)\hat{C}_k(A_k - K_kB_k)^{\mathrm{T}} + Q_{k+1} + K_kR_kK_k,
\end{aligned}$$

where the matrix K_k is the *Kalman gain*, given by

$$K_k = A_k\hat{C}_kB_k^{\mathrm{T}}(B_k\hat{C}_kB_k^{\mathrm{T}} + R_k)^{-1}.$$

The Kalman filter $x_k^{\#}$ is given by the formula

$$x_k^{\#} = \hat{x}_k + \tilde{K}_k(y_k - B_k\hat{x}_k),$$

where

$$\tilde{K}_k = \hat{C}_kB^{\mathrm{T}}(B_k\hat{C}_kB^{\mathrm{T}} + R_k)^{-1}$$

PROGRAMMING KALMAN FILTERS

Main program:

`x_pred = x_0;` (Initial prediction, no observations)

`C_pred = C_0;`

```
for j = 1:N
    New observation:
    y = y(j);
    Update filter of x_j:
    x_filt = Filter(x_pred,C_pred ,y);
    Update prediction of x_{j+1}:
    [x_pred,C_pred] = Predict(x_pred,C_pred,y);
end
```

Kalman predicting:

`[x_pred,C_pred] = Predict(x_pred,C_pred,y)`

```
A = A_k; B = B_k; Q=Q_(k+1); R = R_k;
K = A*C_pred*B'/(B*C_pred*B' + R);
x_pred = A*x_pred + K*(y-B*x_pred);
C_pred = (A - K*B)*C_pred*(A - K*B)'+Q + K*R*K';
```

Kalman filtering:

```
x_filt = Filter(x_pred,C_pred ,y)
```

```
A = A_k; B = B_k; R = R_k;
K = C_pred*B'/(B*C_pred*B'+R);
x_filt =x_pred + K*(y-B*x_pred);
```