9 Multi-Model State Estimation

9.1 Dealing with Model Uncertainty

Kalman filter design is based on some *design model*, which represents the actual system. In reality, there is often considerable uncertainty about the system model. In some cases the systems is too complex to model exactly, and various parameters (such as noise levels) are not know a-priori. In other cases the system may change over time, in an unpredictable manner.

When the state is not observed, the problem of estimating the model parameters is not trivial. For off-line operation, we can use similar schemes to the HMM case (ML estimation of the parameters, usually via the EM algorithm).

One can distinguish two basic approaches to handle model uncertainty.

a. Robust estimation: The goal here is to find a fixed filter that provides "good-enough" performance for a wide range of model parameters. Major approaches here include:

1. *Robust Kalman (or H_2) filtering:* Design filters that give small MSE for all systems whose parameters belong to a given intervals or set.

2. *H_∞ filtering:* Here the basic approach departs from the Kalman filtering, and is more robust relative to the noise assumptions. Rather than random noise, the model considers the 'noise" processes as deterministic, and the goal is to minimize the energy of the estimation error signals relative to the noise energy,

e.g.:

$$J = \frac{\sum_{k=0}^{N}(x_k - \hat{x}_k)' S_k^{-1}(x_k - \hat{x}_k)}{(x_0 - \overline{x}_0)' P_0^{-1}(x_0 - \overline{x}_0) + \sum_{k=0}^{N}(w_k' Q_k^{-1} w_k + v_k' R_k^{-1} v_k)}$$

The resulting filter still has a form similar to the KF, but with different gains.

b. Adaptive state estimation: In this case the filter essentially needs to estimate both the unknown state x_k and the unknown system parameters, namely the system matrices and/or noise parameters. Many approaches have been developed, among them:

1. *Joint filtering of state and parameters:* Here the parameters are appended to the state vector to create a nonlinear (actually bilinear) system with this augmented state. Standard on-linear filters (such as the Extended KF) may may now be used to jointly estimate both. Such procedures should be used with care, as they often fail to converge. Recently, advanced nonlinear filters such as the particle filters have been applied to this problem, but their efficacy is yet to be established in practice.

2. *On-line noise tuning:* Relatively simple rules can be used to tune model noise levels when the filter is "diverging" (not behaving correctly). Divergence is often detected by innovations $\tilde{z}_k$ that do not match their calculated covariance M_k, or are not white.

3. *Batch estimation of parameters:* System and noise parameters are estimated off-line, based on a batch of measurements. For example, approximate Maximum Likelihood estimates may be computed directly or using EM iterations (see the chapter on HMMs). At the other end, heuristic procedures may be used to estimate noise covariances (see the Appendix for an example to such a procedure).

4. *Multi-model estimation:* This is a simpler alternative to adaptive estimation, which has shown good on-line performance in practical applications such as tracking. We discuss this below.

9.2 Multi-Model Estimation: Static Case

In this approach, we essentially implement J Kalman Filters in parallel, with each corresponding to a possible model of the actual system. The state estimation is computed as a Bayes-optimal combination of the individual estimates.

In more detail, we start here with a finite set of possible system models:

$$\{m^j\ ,\ j = 1, \ldots, J\}$$

We make the following assumptions for filter derivation:

- The true system is fixed and coincides with exactly one of these models.
- Each model is linear and Gaussian.
- The prior probability that the true model is m^j is given and denoted by $p_0(m^j)$

We implement J Kalman Filters in parallel, with filter j designed according to model m^j. Filter j calculates the estimates $\hat{x}_k^j = E(x_k|Z_k, m^j)$ and covariances P_k^j.

The optimal (MMSE) estimate of x_k is then given by

$$\begin{aligned}\hat{x}_k &\stackrel{\triangle}{=} E(x_k|Z_k) \\ &= \sum_j p(m^j|Z_k) E(x_k|Z_k, m^j) \stackrel{\triangle}{=} \sum_j \mu_k^j \hat{x}_k^j\,.\end{aligned}$$

The posterior probabilities $\mu_k^j = p(m^j|Z_k)$ can be computed recursively. From Bayes' rule:

$$p(m^j|Z_k) = \frac{p(z_k|m^j, Z_{k-1}) p(m^j|Z_{k-1})}{\sum_j \{\qquad\qquad \text{-"-} \qquad\qquad\}}$$

But $p(z_k|m^j, Z_{k-1}) = p(\tilde{z}_k^j|m^j)$ with $(\tilde{z}_k^j|m^j) \sim N(0, S_k^j)$.

To calculate the covariance, note that

$$p(x_k|Z_k) = \sum_j \mu_k^j p(x_k|Z_k, m^j)$$

(this is a *mixture of Gaussians* distribution). It follows that

$$P_{k|k} = \sum_j \mu_k^j \{P_{k|k}^j + (\hat{x}_k^j - \hat{x}_k)(\hat{x}_k^j - \hat{x}_k)^T\}$$

where the last term accounts for the "spread of the means".

9.3 Dynamic Multi-Model Estimation

We next consider the case where the system is not fixed, but rather can "jump" between the models $\{m^j\}$.

This framework has proved very useful for target tracking. Here different models may correspond to different target maneuvers.

The simplest way to approach this case is by a heuristic modification to the static multi-model filter. Here each $p_k(m^j)$ is always kept above some lower bound $p_{\min}(m^j)$, which prevents it from going to 0. This keeps all models "alive" to some extent, so that they can kick-in when appropriate.

A more sophisticated approach relies on explicit *dynamic multi-model estimation*. The basic formulation here is as follows:

1. Temporal transitions between models are captured through a Markov chain dynamics:

$$p(M_{k+1} = m^j | M_k = m^i) = p_{ij} \, .$$

These transitions are assumed *independent of the system state*.

2. The system dynamics is

$$\begin{aligned} x_k &= F[M_k]x_{k-1} + w_{k-1}[M_k] \\ z_k &= H[M_k]x_k + v_k[M_k] \end{aligned}$$

with the noises (conditionally) Gaussian.

The Optimal Multi-Model Filter

Let $M^k = (M_0 \dots M_k)$ denote the model history up to time k.

Let $m^{k,l}$ denote the l-th possible value of M^k ($l = 1, \dots, J^{k+1}$).

We can now repeat the Bayesian calculation above *for each possible history*, to get an estimate of the form:

$$E(x_k|Z_k) = \sum_l p(M^k = m^{k,l}|Z_k) E(x_k|Z_k, m^{k,l}) \stackrel{\triangle}{=} \sum_l \mu^{k,l} \hat{x}_k^l \,.$$

The coefficients $\mu^{k,l}$ may be computed as follows.

$$\begin{aligned} \mu^{k,l} &\stackrel{\triangle}{=} p(M^k = m^{k,l}|Z_k) \\ &= p(m^{k,l}|Z_{k-1}, z_k) \\ &= \frac{1}{c} p(z_k|m^{k,l}, Z_{k-1}) p(m^{k,l}|Z_{k-1}) \end{aligned}$$

(model measurement update). For the model time update, denote $m^{k,l} = (\dots, m^i, m^j) \equiv (m^{k-1,s}, m^j)$. Then

$$\begin{aligned} p(m^{k,l}|Z_{k-1}) &= p(M^{k-1} = m^{k-1,s}, M_k = m^j|Z_{k-1}) \\ &= p(M_k = m^j|m^{k-1,s}, Z_{k-1}) p(m^{k-1,s}|Z_{k-1}) \\ &= p_{ij} \mu^{k-1,s} \end{aligned}$$

We thus obtain

$$\mu^{k,l} = \frac{1}{c} p(z_k|m^{k,l}, Z^{k-1}) p_{ij} \mu^{k-1,s}$$

where c is a normalization constant. Note that $p(z_k|m^{k,l}, Z^{k-1})$ may be computed as before using the (Gaussian) innovations distribution in the Kalman filter with model history $m^{k,l}$.

A major problem with the above (exact) approach is the exponential growth of possible model histories. Suboptimal solutions include:

1. *History pruning:* Keep only histories with (relatively) high probability.

2. *History merging:* Keep only last part (1 or 2 steps) of the model history, and *combine* histories that differ in older steps. Two variants of this approach are:
   - The GPB (generalized pseudo-Bayesian) algorithm
   - The IMM (interacting multiple-model) algorithm.

The **GPB1** algorithm (generalized pseudo-Bayesian of order 1) uses J filters, with filter j computing $\hat{x}_k^j \approx E(x_k|M_k = j)$.
It starts with initial conditions $(\hat{x}_0, P_0)$ and weights $\mu_0^j = p(M_0 = j)$. For convenience assume that the first measurement is at time $k = 1$. Then at stage $k \geq 1$ we start with $(\hat{x}_{k-1}, P_{k-1}, \{\mu_{k-1}^i\})$ and proceed as follows:

- $\hat{x}_k^j$ and P_k^j $(j = 1 \dots J)$ are computed using the model $M_k = j$ and "inputs" $(\hat{x}_{k-1}, P_{k-1})$. Similarly, the μ_k^j's are computed as

$$\mu_k^j = \frac{1}{c}(\sum_i \mu_{k-1}^i p_{ij}) p(\tilde{z}_k|M_k = j)\,.$$

- $\hat{x}_k$ (and its covariance P_k) is formed as the weighted sum

$$\hat{x}_k = \sum_j \mu_k^j \hat{x}_k^j\,.$$

The **GPB2** algorithm (generalized pseudo-Bayesian of order 2) requires J^2 filters, where filter ij computes $\hat{x}_k^{ij} \approx E(x_k|M_{k-1}M_k = ij)$.
At stage k, it starts with $(\hat{x}_{k-1}^i, P_{k-1}^i, \mu_{k-1}^i)$ and proceeds as follows:

- Compute $(\hat{x}_k^{ij}, P_k^{ij}, \mu_k^{ij})$ (for all model pairs ij) using the model $M_k = j$ and "inputs" $(\hat{x}_{k-1}^i, P_{k-1}^i, \mu_{k-1}^i)$.

- Prepare data for next stage: The estimates $\hat{x}_k^{ij}$ are unified to form the "best" estimate based on $M_k = j$:

$$\hat{x}_k^j = \sum_i (\frac{\mu_k^{ij}}{\mu_k^j}) \hat{x}_k^{ij} \,,$$

  where $\mu_k^j = \sum_i \mu_k^{ij}$. P_k^j is computed accordingly.

- Compute current estimate: $\hat{x}_k$ and its covariance P_k are formed according to the weighted sum:

$$\hat{x}_k = \sum_{ij} \mu_k^{ij} \hat{x}_k^{ij} \quad (= \sum_j \mu_k^j \hat{x}_k^j) \,.$$

The **IMM** algorithm (interacting multiple-model) is intermediate between GPB1 and GPB2. It uses J filters as in GPB1, but each has different "inputs" which are obtained by different mixtures of the previous stage outputs.
At stage k we start with $(\hat{x}_{k-1}^i, P_{k-1}^i, \mu_{k-1}^i)$ and proceeds as follows:

- The new input $\hat{\hat{x}}_{k-1}^j$ to model m^j (and corresponding covariance) is computed as

$$\hat{\hat{x}}_{k-1}^j = \sum_i \mu_{k-1}^{i|j} \hat{x}_{k-1}^i$$

  with

$$\mu_{k-1}^{i|j} = p(M_{k-1} = i | M_k = j, Z_{k-1}) \,.$$

  These coefficients may be computed using Bayes' rule.

- Proceed as in GPB1.

The details of all these variants are in Bar-Shalom's books.

9.4 Appendix: Adaptive Noise Estimation

Many procedures exist for both on-line and off-line (batch) estimation of noise parameters. We present here an example of simplified and heuristic off-line procedure for estimating the noise covariances.

We assume that the model is stationary, and that an (approximate) KF has been applied for $k = 1 \dots N$ to compute $\hat{x}_k$, P_k, etc. Recall the following relations:

$$\tilde{z}_k = z_k - H\hat{x}_k^- = H\tilde{x}_k + v_k \,,$$

$$\text{cov}(\tilde{z}_k) = M_k = HP_k^- H^T + R \tag{1}$$

The measurement covariance R is estimated as follows:

1. Estimate the Innovations bias and covariance:

$$\overline{Z} = \frac{1}{N}\sum_{k=1}^{N} \tilde{z}_k$$

$$\hat{M} = \frac{1}{N-1}\sum_{k=1}^{N} (\tilde{z}_k - \overline{Z})(\tilde{z}_k - \overline{Z})^T$$

2. Use (1) to estimate R:

$$\hat{R} = \hat{M} - H(\frac{1}{N}\sum_{1}^{N} P_k^-)H^T$$

To estimate $Q = \text{cov}(w)$, recall that

$$\tilde{x}_{k+1}^- \stackrel{\triangle}{=} x_{k+1} - \hat{x}_{k+1}^- = F(x_k - \hat{x}_k^+) + w_k$$

and consequently $Q = F\text{cov}(\tilde{x}_k^+)F^T - \text{cov}(\tilde{x}_{k+1}^-)$.

To approximate $\text{cov}(\tilde{x}_k^+)$ we can use P_k^+. However, we cannot approximate $\text{cov}(\tilde{x}_{k+1}^-)$ by P_{k+1}^- as computed by the filter, since since the filter simply computes it as $P_{k+1}^- =$

$FP_k^+F^T + Q_k$, with Q_k the *wrong* covariance. We therefore approximate x_{k+1} by $\hat{x}_{k+1}^+$, and use

$$\tilde{x}_k^- \approx \hat{x}_k^+ - \hat{x}_k^- \doteq d_k\,.$$

This leads to the approximate scheme:

3. Estimate $\mathrm{cov}(\tilde{x}_k^-)$ as the empirical covariance of d_k, namely

$$\overline{X} = \frac{1}{N}\sum_1^N d_k$$
$$\hat{P}^- = \frac{1}{N-1}\sum_1^N (d_k - \overline{X})(d_k - \overline{X})^T$$

4. Compute

$$\hat{Q} = \hat{P}^- - F(\frac{1}{N}\sum_1^N P_k^+)F^T$$

There are many variants to this scheme that improve the estimation accuracy, and also some schemes that modify the Kalman gain directly. More details can be found, for example, in: Stengel, *Optimal Control and Estimation*, 1986/94.