

## Chapter 11

# ADAPTIVE ESTIMATION AND MANEUVERING TARGETS

### 11.1 INTRODUCTION

#### 11.1.1 Adaptive Estimation — Outline

In the models considered previously, the only uncertainties consisted of additive white noises (process and measurement) with known statistical properties. In other words, the system model, consisting of the state transition matrix, the input gain (and input, if any), the measurement matrix, and noise covariances, were all assumed to be known. In the case of “colored” noise with known autocorrelation, it has been shown that it can be reduced to the standard situation in Chapter 8.

In many practical situations the above listed “parameters of the problem” are partially unknown and possibly time-varying. There are numerous techniques of *system identification* (e.g., [Ljung87]) that deal with the identification (estimation) of the *structural parameters* of a system.

The purpose of this chapter is to present state estimation techniques that can “adapt” themselves to certain types of uncertainties beyond those treated earlier — *adaptive estimation algorithms*.

One type of uncertainty to be considered is the case of unknown inputs into the system, which typifies maneuvering targets. The other type will be a combination of system parameter uncertainties with unknown inputs where the system parameters (are assumed to) take values in a discrete set.

Maneuvering targets are characterized by an equation of the same form as (4.3.1-9), namely,

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k) \quad (11.1.1-1)$$

but the input  $u(k)$ , which enters the system in addition to the process noise  $v(k)$ , is *unknown*.

For simplicity, linear models are considered — in the case of nonlinear models the same techniques that are discussed in the sequel can be used with linearization.

The approaches that can be used in such a situation fall into two broad categories:

1. The unknown input (the maneuver command) is modeled as a random process.
2. The unknown input is estimated in real time.

The random process type models can be classified according to the statistical properties of the process modeling the maneuver as

- White noise or
- Autocorrelated noise

Note that this *unknown input as a random process* approach amounts to treating the unknown input as (an additional) process noise, and, possibly, requiring augmentation of the system's state.

The use of fixed-level white noise to model maneuvers has been discussed in the context of kinematic models. However, maneuvers, by their nature, are of different magnitudes at different times. One way to accommodate this is by adjusting the level of the process noise. One option is a *continuous noise level adjustment* technique, discussed in Section 11.2. Alternatively, several discrete levels of noise can be assumed in the filter, with a *noise level switching* according to a certain rule, also discussed in Section 11.2.

The use of autocorrelated (colored) noise to model maneuvers has been presented in Section 8.2 — the approach is to “prewhiten” the noise and augment the state with the prewhitening subsystem. The noise level adjustment techniques can be used with the augmented system, which is driven by white noise.

All these methods relying on *modeling maneuvers as random processes* are approximations because maneuvers are, in general, not stochastic processes. Nevertheless, such approaches are simple and can be quite effective.

The second type of approach, *input estimation*, is implemented assuming the input to be constant over a certain period of time. The estimation can be done based on the least squares criterion, and the result can be used in the following ways:

- The state estimate is corrected with the estimated input, or
- The state is augmented and the input becomes an extra state component that is reestimated sequentially within the augmented state.

The input estimation with state estimate correction technique is presented in Section 11.3. The technique of estimating the input and, when “statistically significant,” augmenting the state with it (which leads to *variable state dimension*), is the topic of Section 11.4. These two algorithms and the noise level switching technique of Section 11.2 are compared in Section 11.5.

The so-called ***multiple model (MM)*** algorithms are the topic of Section 11.6. These algorithms assume that the system behaves according to one of a finite number of models — it is in one of several ***modes*** (operating regimes). The models can differ in *noise levels* or their *structure* — different state dimensions and unknown inputs can be accommodated as well. Such systems are called ***hybrid systems*** — they have both discrete (structure/parameters) and continuous uncertainties (additive noises).

First the *static MM* algorithm — static from the point of view of the assumed evolution of the models, i.e., for fixed (nonswitching) models — is considered. Then the optimal *dynamic* algorithm — for switching models, according to a Markov chain — is presented. Since the optimal dynamic algorithm is not practical for implementation, two suboptimal approaches, one called ***generalized pseudo-Bayesian (GPB)*** and the other the ***interacting multiple model (IMM)***, are also presented.

The design of an IMM estimator for ***air traffic control (ATC)*** is discussed in detail. Guidelines are also developed for when an adaptive estimator is really needed, i.e., when a (single model based) Kalman filter is not adequate.

The use of the extended Kalman filter for state and system parameter estimation is briefly presented in Section 11.9.

### 11.1.2 Adaptive Estimation — Summary of Objectives

Show modeling of maneuvers as

- A random process (white or autocorrelated process noise)
  - continuously variable level;
  - several discrete levels with switching between them.
- A fixed input, estimated as an unknown constant, with
  - state estimate compensation;
  - variable state dimension.

Compare some of these algorithms.

Illustrate the optimal method of comparison of algorithms.

Multiple-model (hybrid system) approach:

Several models are used in parallel with a probabilistic weighting:

- Static — for fixed (nonswitching) models
- Dynamic — for switching models
  - Optimal
  - GPB1
  - GPB2
  - IMM

Present the design of an IMM estimator for air traffic control.

Answer the question: When is an adaptive estimator really needed?

Show the use of the EKF for state and system parameter estimation.

## 11.2 ADJUSTABLE LEVEL PROCESS NOISE

### 11.2.1 Continuous Noise Level Adjustment

In this **continuous noise level adjustment** approach, the target is tracked with a filter in which a certain low level of process noise is assumed. The level of the noise is determined by its variance (or covariance matrix).

A maneuver manifests itself as a “large” innovation. A simple detection procedure for such an occurrence is based on the **normalized innovation squared**

$$\epsilon_\nu(k) = \nu(k)' S(k)^{-1} \nu(k) \quad (11.2.1-1)$$

A threshold is set up such that based on the target model (for the nonmaneuvering situation)

$$P\{\epsilon_\nu(k) < \epsilon_{max}\} = 1 - \mu \quad (11.2.1-2)$$

with, say, tail probability  $\mu = 0.01$ .

If the threshold is exceeded, then the process noise variance  $Q(k-1)$  is scaled up until  $\epsilon_\nu$  is reduced to the threshold  $\epsilon_{max}$ .

Using a **scaling factor**  $\phi(k)$  (i.e., a **fudge factor**), the covariance of the innovations becomes

$$S(k) = H(k)[F(k-1)P(k-1|k-1)F(k-1)' + \phi(k)Q(k-1)]H(k)' + R(k) \quad (11.2.1-3)$$

In place of the single-time test statistic (11.2.1-1) one can also use a time average over a “sliding window.” This is illustrated in the next subsection.

A similar technique can be used to lower the process noise level after the maneuver.

### 11.2.2 Process Noise with Several Discrete Levels

Another approach is to assume two or more levels of process noise and use a rule for **noise level switching**.

Under normal conditions the filter is operating with the low level noise  $Q_1$ .

The normalized innovation squared

$$\epsilon_\nu(k) = \nu(k)' S(k)^{-1} \nu(k) \quad (11.2.2-1)$$

is monitored; and if it *exceeds* a certain threshold, the filter switches to a prechosen higher level of process noise,  $Q_2$ .

Under the linear-Gaussian assumptions, the pdf of  $\epsilon_\nu$  is chi-square distributed with  $n_z$ , the dimension of the measurement, degrees of freedom:

$$\epsilon_\nu \sim \chi_{n_z}^2 \quad (11.2.2-2)$$

The switching threshold is chosen based on (11.2.2-2) such that the probability of being exceeded under normal conditions is small.

The decision statistic (11.2.2-1), which is based on a single sampling time, can be replaced by a ***moving average*** (or ***moving sum***) of the normalized innovations squared over a ***sliding window*** of  $s$  sampling times

$$\boxed{\epsilon_\nu^s(k) = \sum_{j=k-s+1}^k \epsilon_\nu(j)} \quad (11.2.2-3)$$

The above is chi-square distributed with  $sn_z$  degrees of freedom

$$\epsilon_\nu^s \sim \chi_{sn_z}^2 \quad (11.2.2-4)$$

since (11.2.2-3) is the sum of  $s$  independent terms with distribution (11.2.2-2).

Alternatively, a ***fading memory average*** (also called ***exponentially discounted average***)

$$\boxed{\epsilon_\nu^\alpha(k) = \alpha\epsilon_\nu^\alpha(k-1) + \epsilon_\nu(k)} \quad (11.2.2-5)$$

where

$$0 < \alpha < 1 \quad (11.2.2-6)$$

and with initial condition  $\epsilon_\nu^\alpha(0) = 0$ , can be used.

The expected value of (11.2.2-5) in steady state is

$$E[\epsilon_\nu^\alpha(k)] = \frac{n_z}{1-\alpha} \quad (11.2.2-7)$$

The ***effective window length*** of (11.2.2-5) can be considered as the sum of the weights multiplying  $\epsilon_\nu^\alpha(k)$ ,  $k = 1, 2, \dots$ , i.e.,

$$s_\alpha = 1 + \alpha + \alpha^2 + \dots = \frac{1}{1-\alpha} \quad (11.2.2-8)$$

For example, for  $\alpha = 0.8$  one has  $s_\alpha = 5$ .

Based on (11.2.2-7), one can assume, as a first approximation (matching the first moment only), that  $\epsilon_\nu^\alpha(k)$  is chi-square distributed with number of degrees of freedom given by (11.2.2-7).

Using the first and second moment-matching approximation described in Subsection 1.4.18, one obtains

$$\epsilon_\nu^\alpha \sim \frac{1}{1+\alpha} \chi_{n_\alpha}^2 \quad (11.2.2-9)$$

where the number of degrees of freedom is

$$n_\alpha = n_z \frac{1 + \alpha}{1 - \alpha} \quad (11.2.2-10)$$

For example, for  $\alpha = 0.8$  one obtains  $n_\alpha = 18$ . Using the first approximation for the same  $\alpha$  with  $n_z = 2$  leads to 10 degrees of freedom. In other words, the moment-matching approximation (with the first two moments) indicates a (relatively) narrower pdf. The “width” of a chi-square pdf can be taken as the ratio of its standard deviation to its mean. For  $\chi_n^2$ , this is (see Subsection 1.4.17)  $\sqrt{2n}/n$ , which decreases as  $n$  increases.

### Estimator Operation

Thus, the filter switches from the lower process noise covariance  $Q_1$  to the higher  $Q_2$  if the average (11.2.2-3) or (11.2.2-5) exceeds an **upcrossing threshold**, determined based on (11.2.2-4) or (11.2.2-9), respectively, with a small tail probability.

After the filter starts running with the higher-level process noise the innovations are monitored again to see whether there is reason to switch back.

The change to the model with lower-level process noise is done when the normalized innovation, or a certain average of it, falls *below* another threshold — the **downcrossing threshold**.

There is no exact way to choose these thresholds — even if the distributions were known exactly, the tail probabilities are still subjective. They can be chosen, to begin with, based on tail probabilities and then set following experimentation (Monte Carlo runs) and subjective evaluation of the results.

### Remarks

This technique of noise level switching can be extended to more than two levels of process noise. It can also be used with white process noise or autocorrelated process noise; the latter has to be prewhitened, as discussed in Subsection 8.2.1.

In general,  $n_\alpha$ , given in (11.2.2-10), is not an integer. In this case one has a gamma rather than chi-square distribution; for the sake of simplicity, however, one can use the chi-square tables with an interpolation.

### 11.2.3 Adjustable Level Process Noise — Summary

A maneuver can be detected by monitoring the *normalized innovation*; if it exceeds a threshold, then it can be assumed (in the absence of other factors) that the target has deviated from its previous “pattern.”

A *fudge factor* can be used to scale up (continuously) the process noise (at the previous time) such that the modified prediction covariance is sufficiently

large for the normalized innovation to be below the set threshold; the increased innovation covariance should “cover” the observed deviation.

Alternatively, a number of levels of process noise can be assumed. In this approach, at any given time a *single filter* operates with the “*current*” noise level.

A simple rule of switching from the current noise level to the one above or below is followed.

The innovations are monitored via

- a moving average over a sliding window or
- a fading-memory average

of their normalized squared value.

This average is compared to a threshold based on the chi-square density for a small upper tail probability. If this *upcrossing threshold* is exceeded, then the filter switches to a higher level of process noise.

The downswitch is done using a *downcrossing threshold*.

## 11.3 INPUT ESTIMATION

### 11.3.1 The Model

Consider the system with state equation

$$x(k+1) = Fx(k) + Gu(k) + v(k) \quad (11.3.1-1)$$

where  $u$  is an **unknown input** modeling the target maneuvers and  $v$  is the process noise, zero mean white with known covariance  $Q$ .

The observations are

$$z(k) = Hx(k) + w(k) \quad (11.3.1-2)$$

with the observation noise  $w$  zero mean, white, with covariance  $R$ , and independent of the process noise.

The estimation of the state is done using the model without input (nonmaneuvering)

$$x(k+1) = Fx(k) + v(k) \quad (11.3.1-3)$$

Two Kalman filters are considered:

1. The *actual one* based on the nonmaneuvering model (11.3.1-3).
2. A *hypothetical one* based on the maneuvering model (11.3.1-1) with known  $u$ .

From the innovations of the **nonmaneuvering filter** based on (11.3.1-3), the input  $u$  is to be

- detected,
- estimated, and
- used to correct the state estimate.

This will be done using a sliding window of the latest  $s$  measurements, and during this window period the input will be assumed *constant*.

### 11.3.2 The Innovations as a Linear Measurement of the Unknown Input

Denote the present time by  $k$  and assume that the target starts maneuvering at time  $k - s$ ; that is, the *maneuver onset time* is  $k - s$ . The unknown inputs during the interval  $[k - s, \dots, k]$  are  $u(i)$ ,  $i = k - s, \dots, k - 1$ .

The state estimates from the *mismatched nonmaneuvering filter* based on (11.3.1-3) will be denoted by an asterisk. The recursion for these estimates is

$$\begin{aligned}\hat{x}^*(i+1|i) &= F[I - W(i)H]\hat{x}^*(i|i-1) + FW(i)z(i) \\ &\triangleq \Phi(i)\hat{x}^*(i|i-1) + FW(i)z(i) \\ &\quad i = k - s, \dots, k - 1\end{aligned}\quad (11.3.2-1)$$

with the initial condition

$$\hat{x}^*(k-s|k-s-1) = \hat{x}(k-s|k-s-1) \quad (11.3.2-2)$$

being the correct estimate (one-step prediction) before the maneuver started. This follows from the assumption that the maneuver onset time is  $k - s$ .

Similarly to (4.3.3-1), recursion (11.3.2-1) yields, in terms of (11.3.2-2),

$$\begin{aligned}\hat{x}^*(i+1|i) &= \left[ \prod_{j=0}^{i-k+s} \Phi(i-j) \right] \hat{x}(k-s|k-s-1) \\ &\quad + \sum_{j=k-s}^i \left[ \prod_{m=0}^{i-j-1} \Phi(i-m) \right] FW(j)z(j) \quad i = k - s, \dots, k - 1\end{aligned}\quad (11.3.2-3)$$

If the inputs were known, the *hypothetical correct filter* based on (11.3.1-1) would yield estimates according to the recursion

$$\begin{aligned}\hat{x}(i+1|i) &= \Phi(i)\hat{x}(i|i-1) + FW(i)z(i) + Gu(i) \\ &= \left[ \prod_{j=0}^{i-k+s} \Phi(i-j) \right] \hat{x}(k-s|k-s-1) \\ &\quad + \sum_{j=k-s}^i \left[ \prod_{m=0}^{i-j-1} \Phi(i-m) \right] [FW(j)z(j) + Gu(j)] \\ &\quad i = k - s, \dots, k - 1\end{aligned}\quad (11.3.2-4)$$

which is the same as (11.3.2-3) except for the last term containing the inputs.

The innovations

$$\nu(i+1) = z(i+1) - H\hat{x}(i+1|i) \quad (11.3.2-5)$$

corresponding to the correct (but *hypothetical*) filter (11.3.2-4) are a zero-mean white sequence with covariance  $S(i+1)$  given as in (5.2.3-9).

The innovations corresponding to the *nonmaneuvering filter* (11.3.2-3) are

$$\nu^*(i+1) = z(i+1) - H\hat{x}^*(i+1|i) \quad (11.3.2-6)$$

From (11.3.2-3) and (11.3.2-4) it follows that the innovations (11.3.2-6) of the nonmaneuvering filter — the *only real filter* — are the same as the “white noise sequence” (11.3.2-5) plus a “bias term” related to the inputs

$$\nu^*(i+1) = \nu(i+1) + H \sum_{j=k-s}^i \left[ \prod_{m=0}^{i-j-1} \Phi(i-m) \right] Gu(j) \quad (11.3.2-7)$$

Assuming the input to be constant over the interval  $[k-s, \dots, k-1]$ , that is,

$$u(j) = u \quad j = k-s, \dots, k-1 \quad (11.3.2-8)$$

yields

$$\boxed{\nu^*(i+1) = \Psi(i+1)u + \nu(i+1) \quad i = k-s, \dots, k-1} \quad (11.3.2-9)$$

where

$$\Psi(i+1) \triangleq H \sum_{j=k-s}^i \left[ \prod_{m=0}^{i-j-1} \Phi(i-m) \right] G \quad (11.3.2-10)$$

Equation (11.3.2-9) shows that

the *nonmaneuvering filter innovation*  $\nu^*$

is

a *linear measurement of the input*  $u$

in the presence of the additive “white noise”  $\nu$  given by (11.3.2-5) above.

### 11.3.3 Estimation of the Unknown Input

Based on (11.3.2-9), the input can be estimated via LS from

$$y = \Psi u + \epsilon \quad (11.3.3-1)$$

where

$$y = \begin{bmatrix} \nu^*(k) \\ \vdots \\ \nu^*(k-s+1) \end{bmatrix} \quad (11.3.3-2)$$

is the stacked “measurement” vector,

$$\Psi = \begin{bmatrix} \Psi(k) \\ \vdots \\ \Psi(k-s+1) \end{bmatrix} \quad (11.3.3-3)$$

is the measurement matrix, and the “noise”  $\epsilon$ , whose components are the innovations (11.3.2-5), is zero mean with block-diagonal covariance matrix

$$S = \text{diag}[S(i)] \quad (11.3.3-4)$$

The **input estimate** in batch form is

$$\hat{u} = (\Psi' S^{-1} \Psi)^{-1} \Psi' S^{-1} y \quad (11.3.3-5)$$

with the resulting covariance matrix

$$L = (\Psi' S^{-1} \Psi)^{-1} \quad (11.3.3-6)$$

An estimate is accepted, that is, a **maneuver detection** is declared if and only if (11.3.3-5) is *statistically significant* (see Subsection 1.5.2). The significance test for the estimate  $\hat{u}$ , which is an  $n_u$ -dimensional vector, is

$$\epsilon_{\hat{u}} = \hat{u}' L^{-1} \hat{u} \geq c \quad (11.3.3-7)$$

where  $c$  is a threshold chosen as follows. If the input is zero, then

$$\hat{u} \sim \mathcal{N}(0, L) \quad (11.3.3-8)$$

The statistic  $\epsilon_{\hat{u}}$  from (11.3.3-7) is then chi-square distributed with  $n_u$  degrees of freedom, and  $c$  is such that the probability of false alarm is

$$P\{\epsilon_{\hat{u}} \geq c\} = \alpha \quad (11.3.3-9)$$

with, e.g.,  $\alpha \leq 0.01$ .

### 11.3.4 Correction of the State Estimate

If a maneuver is detected, then the state estimate has to be corrected. This is done, as before, assuming that the maneuver onset time was  $k-s$ . The term reflecting the effect of the input in (11.3.2-4) is used to correct the predicted state as follows

$$\hat{x}^u(k|k-1) = \hat{x}^*(k|k-1) + M(k)\hat{u} \quad (11.3.4-1)$$

where

$$M(k) \triangleq \sum_{j=k-s}^{k-1} \left[ \prod_{m=0}^{k-j-2} \Phi(k-1-m) \right] G \quad (11.3.4-2)$$

The covariance associated with the prediction (11.3.4-1) is

$$P^u(k|k-1) = P(k|k-1) + M(k)L M(k)' \quad (11.3.4-3)$$

where  $L$  is given in (11.3.3-6). Note that the covariance increases — the last term above is positive semidefinite or positive definite — to account for the uncertainty in the correction term used in (11.3.4-1).

The assumption about the maneuver onset time being  $k-s$  at every  $k$  makes the algorithm very elegant, but not optimal.

## Maneuver Onset Time Estimation

It is possible to estimate the onset time using a maximum likelihood criterion — this requires running in parallel a number of such algorithms. Each algorithm evaluates the likelihood function for its assumed maneuver onset time and the most likely effective window length  $s$  is chosen.

It is clear that this becomes quite expensive.

### 11.3.5 Input Estimation — Summary

The input estimation method is based on the following:

Given a linear system driven by a constant input, a filter based on the state model of the system *without input (input equal to zero)* estimates the state.

Then the innovations of this *mismatched filter* are

- a linear measurement of the input
- with additive zero-mean white noise with covariance equal to the filter's innovation covariance.

This allows the estimation of the input, assuming that

- it started at the beginning of the assumed sliding window, and
- it is approximately constant over this window in time.

The estimation is done via least squares.

The estimated input (e.g., acceleration) is tested for *statistical significance* (i.e., if it is “large enough” compared to its standard deviation).

If the estimated input is statistically significant, then the state estimate is corrected with the effect of the input from the time it was assumed to have started (at the beginning of the window).

The covariance of the state is adjusted (increased) because the correction was done with an estimated input, rather than the exact input, which is not available.

## 11.4 THE VARIABLE STATE DIMENSION APPROACH

### 11.4.1 The Approach

The target maneuver is considered here an inherent part of its dynamics, rather than noise. In the absence of maneuvers the filter operates using a “quiescent state model.” Once a maneuver is detected, new state components are added — thus the name **variable state dimension (VSD)** approach.

The extent of the maneuver as detected is then used to yield an estimate for the extra state components, and corrections are made on the other state

components. The tracking is then done with the augmented state model until it will be reverted to the quiescent model by another decision.

The rationale for using a lower-order quiescent model and a higher-order maneuvering model is the following: This will allow good tracking performance in both situations rather than a compromise. For example, if the target does not have acceleration, using a third-order model increases the estimation errors for both position and velocity.

The two models used here, described in Section 6.3, are a (nearly) constant velocity model for the quiescent situation and a (nearly) constant acceleration model for the maneuvering situation.

The state vector, for a planar motion, is in the *quiescent model*

$$x = [\xi \ \dot{\xi} \ \eta \ \dot{\eta}]' \quad (11.4.1-1)$$

In the *maneuvering model*, the state is

$$x^m = [\xi \ \dot{\xi} \ \eta \ \dot{\eta} \ \ddot{\xi} \ \ddot{\eta}]' \quad (11.4.1-2)$$

The method is not limited to these models. For example, the maneuvering situation could be modeled using autocorrelated acceleration, as in Section 8.2.

### 11.4.2 The Maneuver Detection and Model Switching

The target tracking filter is initialized under the constant velocity model assumption as in Section 5.5, using a two-point differencing procedure.

The maneuver detection is done as follows:

A *fading memory average* of the innovations from the estimator based on the quiescent model is computed

$$\epsilon_\nu^\alpha(k) = \alpha\epsilon_\nu^\alpha(k-1) + \epsilon_\nu(k) \quad (11.4.2-1)$$

with

$$\epsilon_\nu(k) \triangleq \nu(k)'S(k)^{-1}\nu(k) \quad (11.4.2-2)$$

where  $0 < \alpha < 1$ ,  $\nu(k)$  is the innovation and  $S(k)$  its covariance.

Since  $\epsilon_\nu(k)$  is, under the Gaussian assumptions, chi-square distributed with  $n_z$  (dimension of the measurement) degrees of freedom, one has

$$\lim_{k \rightarrow \infty} E[\epsilon_\nu^\alpha(k)] = \frac{n_z}{1 - \alpha} \quad (11.4.2-3)$$

As in Section 11.2, one can look at

$$s_\alpha = \frac{1}{1 - \alpha} \quad (11.4.2-4)$$

as the *effective window length* over which the presence of a maneuver is tested.

The hypothesis that a maneuver is taking place is accepted if  $\epsilon_{\nu}^{\alpha}(k)$  exceeds a certain threshold that can be determined based on the chi-square distribution as in Section 11.2. Then the estimator switches from the quiescent model to the maneuvering model.

The scheme for reverting to the quiescent model is as follows: The estimated accelerations are compared to their variances, and if they are not statistically significant the maneuver hypothesis is rejected.

### Maneuver Termination vs. Maneuver Onset Detection

It is more difficult to detect *maneuver termination* than *maneuver onset* because a maneuvering model has a larger innovation covariance than a nonmaneuvering model. This is due to the fact that the latter has a larger state vector and assumes more motion uncertainty than the former.

### Significance Test for the Acceleration Estimate

The test statistic for significance of the acceleration estimates is

$$\epsilon_{\hat{a}}(k) = \hat{a}(k|k)' P_a^m(k|k)^{-1} \hat{a}(k|k) \quad (11.4.2-5)$$

where  $\hat{a}$  is the estimate of the acceleration components and  $P_a^m$  is the corresponding block from the covariance matrix of the maneuvering model.

When the *moving average* over a window of length  $p$

$$\epsilon_{\hat{a}}^p(k) = \sum_{j=k-p+1}^k \epsilon_{\hat{a}}(j) \quad (11.4.2-6)$$

falls below a threshold, the acceleration is deemed *insignificant*. A fading memory average can also be used.

The situation where the acceleration drops suddenly to zero can lead to large innovations for the maneuvering model. This can be taken care of by allowing a switch to the lower-order model also when the maneuvering model's innovation exceeds, say, the 99% confidence region.

### 11.4.3 Initialization of the Augmented Model

When a maneuver is detected at time  $k$ , the filter assumes that the target had a constant acceleration starting at  $k-s-1$ , where  $s$  is the effective window length. The state estimates for time  $k-s$  are then modified as follows. It is assumed that only position measurements are available.

First, the estimates at  $k-s$  for the acceleration are

$$\hat{x}_{4+i}^m(k-s|k-s) = \frac{2}{T^2} [z_i(k-s) - \hat{z}_i(k-s|k-s-1)] \quad i = 1, 2 \quad (11.4.3-1)$$

The position components of the estimate at  $k - s$  are taken as the corresponding measurements, that is,

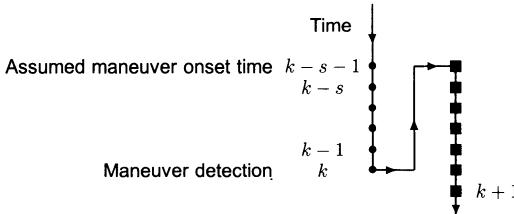
$$\hat{x}_{2i-1}^m(k-s|k-s) = z_i(k-s) \quad i = 1, 2 \quad (11.4.3-2)$$

while the velocity components are corrected with the acceleration estimates as follows:

$$\hat{x}_{2i}^m(k-s|k-s) = \hat{x}_{2i}(k-s-1|k-s-1) + T\hat{x}_{4+i}^m(k-s|k-s) \quad i = 1, 2 \quad (11.4.3-3)$$

The covariance matrix associated with the modified state estimate as given in (11.4.3-1) to (11.4.3-3) is  $P^m(k-s|k-s)$ , and its elements are presented in [Bar-Shalom82] together with their derivation. These equations specify in full the initialization of the filter based on the maneuvering model for time  $k - s$ .

Then, a recursive estimation algorithm (Kalman filter) based on this model reprocesses the measurements sequentially up to time  $k$ . Following this, the measurements are processed sequentially as they arrive. This is depicted in Fig. 11.4.3-1.



- Sequential estimation with quiescent model;   ■ with maneuvering model

**Figure 11.4.3-1:** Switching from quiescent to maneuvering model.

#### 11.4.4 VSD Approach — Summary

In the absence of maneuvers, the filter is based on a *quiescent* (low-order) model of the state.

If a maneuver is detected, the estimator switches to a higher dimension (augmented) *maneuvering* model that incorporates additional state components (acceleration).

The use of the lower-order model yields maximum estimation accuracy in the case where the target undergoes no accelerations — no information is “wasted” on estimating state components that are zero.

A maneuver is declared detected when a *fading memory average* of the normalized innovations exceeds a threshold. The fading memory has an *effective*

*window length*, and the onset of the maneuver is then taken as the beginning of this sliding window.

The augmented filter is initialized at the beginning of the maneuver detection window.

A maneuver is declared terminated when the extra state component's (acceleration) estimates become statistically insignificant.

## 11.5 A COMPARISON OF ADAPTIVE ESTIMATION METHODS FOR MANEUVERING TARGETS

### 11.5.1 The Problem

In this section the following methods of maneuver detection are illustrated and compared:

1. White process noise with two levels (Section 11.2)
2. Input estimation (Section 11.3)
3. Variable dimension filtering (Section 11.4)

The example considers a target whose position is sampled every  $T = 10$  s. The target is moving in a plane with constant course and speed until  $k = 40$  when it starts a slow  $90^\circ$  turn that is completed in 20 sampling periods. A second fast turn of  $90^\circ$  starts at  $k = 61$  and is completed in 5 sampling times.

The initial condition of the target, with state

$$x = [\xi \ \dot{\xi} \ \eta \ \dot{\eta}]' \quad (11.5.1-1)$$

is, with position and velocity units m and m/s, respectively

$$x(0) = [2000 \ 0 \ 10000 \ -15]' \quad (11.5.1-2)$$

The slow turn is the result of acceleration inputs

$$u_\xi = u_\eta = 0.075 \text{ m/s}^2 \quad 400 \text{ s} \leq t \leq 600 \text{ s} \quad (11.5.1-3)$$

and the fast turn has accelerations

$$u^\xi = u^\eta = -0.3 \text{ m/s}^2 \quad 610 \text{ s} \leq t \leq 660 \text{ s} \quad (11.5.1-4)$$

Note that the changes in velocity are 0.75 and 3 m/s per sampling period for the slow and fast turn, respectively.

The measurements are position only according to the equation

$$z(k) = [1 \ 0 \ 1 \ 0] x(k) + w(k) \quad (11.5.1-5)$$

with  $w(k)$  zero mean, white, independent of the process noise, and with variance

$$E[w(k)^2] = rI \quad (11.5.1-6)$$

where  $r = 10^4 \text{ m}^2$ .

### 11.5.2 The White Noise Model with Two Levels

The adaptive algorithm based on white process noise with switching between two levels for maneuver modeling used the following state representation per coordinate (same for each coordinate and decoupled between them)

$$x(k+1) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} T/2 \\ 1 \end{bmatrix} v(k) \quad (11.5.2-1)$$

with  $v(k)$  zero mean white with variance

$$E[v(k)^2] = q \quad (11.5.2-2)$$

Note that here  $v(k)$  represents the *velocity increment over a sampling period*.

The two levels of process noise were  $q_0$  for the quiescent period and  $q_1$  after a maneuver was detected.

A maneuver was declared detected if the normalized innovation squared (11.2.2-1) exceeded a threshold  $\epsilon_m$ . Then  $q_0$  was replaced by  $q_1$  in the filter. A maneuver was considered terminated when the normalized innovation squared, obtained from the filter with the higher  $q_1$ , fell below the same threshold  $\epsilon_m$  — for simplicity, the upcrossing and downcrossing thresholds were taken the same.

The value for the quiescent process noise was  $q_0 = 0$ . Several values for the variance  $q_1$  of the process noise modeling the maneuver and the threshold  $\epsilon_m$  were tried. The best combination was determined based on the MSE in the position estimate of the target from 50 Monte Carlo runs. The resulting values were  $q_1 = 10$  and  $\epsilon_m = 3$ .

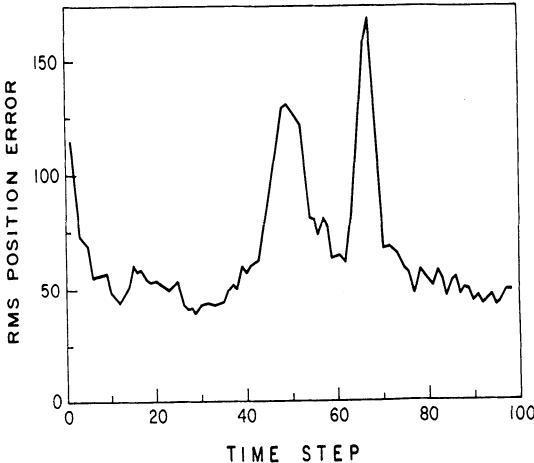
The RMS position error (coordinate  $\xi$ ) corresponding to these values is shown in Fig. 11.5.2-1. Note that the value  $\sqrt{q_1} \approx 3$  matches approximately the *maximum change in velocity per sampling period* during the fast maneuver, indicated in Subsection 11.5.1.

### 11.5.3 The IE and VSD Methods

The same problem was simulated using the *input estimation (IE)* and *variable state dimension (VSD)* methods.

The IE method was run with a window of  $s = 5$  samples to estimate the input. The state equation was the constant velocity model (6.3.2-1) for each coordinate and no process noise.

The VSD method was based on a quiescent model with state (11.4.1-1) and a maneuvering model with state (11.4.1-2). The quiescent model was, for each coordinate, the constant velocity state equation (6.3.2-1) with no process noise. The maneuvering model was a nearly constant acceleration state equation, given by (6.3.3-1) with process noise standard deviation  $\sigma_v$  in (6.3.3-4) taken equal to 5% of the estimated acceleration.



**Figure 11.5.2-1:** Position estimation error in coordinate  $\xi$  for the adaptive filter with two process noise levels (from 50 runs).

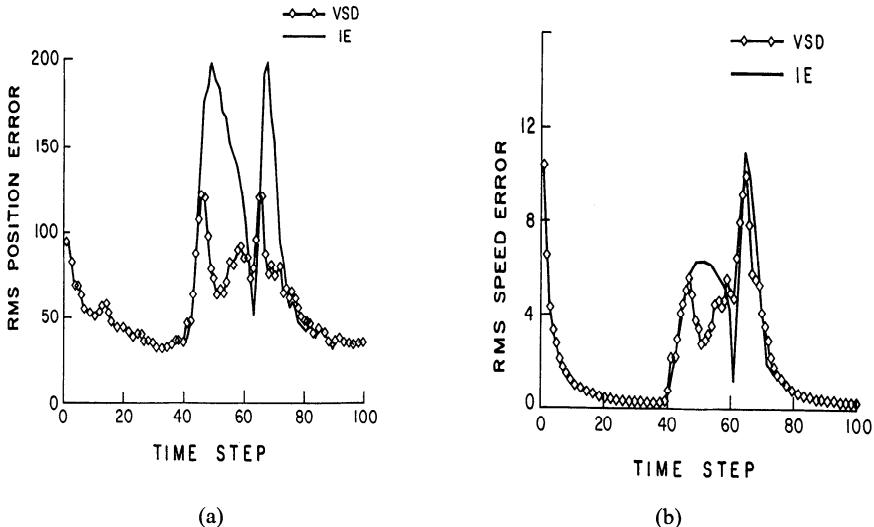
The following maneuver detection parameters were used in the VSD method:

1. The fading memory parameter from (11.4.2-1) was  $\alpha = 0.8$ , which corresponds to an effective window length (11.4.2-4) of  $s = 5$ .
2. The threshold for the fading memory average (11.4.2-1) was  $\chi_{10}^2(95\%) = 18.3$ . The maneuver detection was done together in the two coordinates based on both measurements (i.e.,  $n_z = 2$ ), which, with (11.4.2-3), yields that  $\epsilon_\nu^\alpha$  is approximately chi-square distributed with 10 degrees of freedom.
3. The window for the calculation of the average normalized acceleration (11.4.2-6) for the significance test was  $p = 2$ .
4. The threshold for the acceleration significance test was  $\chi_4^2(95\%) = 9.49$ , based on a 4-degree-of-freedom chi-square random variable (the dimension of the acceleration, 2, multiplied by  $p = 2$ ).

Figure 11.5.3-1a presents the position RMS errors (in coordinate  $\xi$ ) for the IE and VSD filters based on a 50-run Monte Carlo average. These two filters were run with the same random numbers for further comparison.

The RMS errors are smaller in the VSD filter compared to the IE filter. Comparing with Fig. 11.5.2-1, it is seen that the simple approach of two-level white process noise modeling of the maneuver is quite effective — it is slightly worse than the VSD filter but better than the IE filter.

Figure 11.5.3-1b presents the velocity RMS errors for the IE and VSD filters based on the same 50 Monte Carlo runs. For velocity, the VSD filter is only slightly superior compared to the IE algorithm.



**Figure 11.5.3-1:** Estimation errors in coordinate  $\xi$  for the VSD and IE filters (from 50 runs).

### Remark

It should be noted that none of these three adaptive algorithms considered here manage to keep the RMS position error in the estimate below the position measurement noise RMS value (which is, in one coordinate, 100 m) *during the entire maneuver period*. In other words, these algorithms have difficulty providing **noise reduction** during the critical time of a maneuver.

This is a general problem with adaptive estimation algorithms: The adaptation might not be rapid enough. In the present example, the errors reach their peak shortly after the start of the maneuver.

This issue will be considered again in a later example.

### 11.5.4 Statistical Test for Comparison of the IE and VSD Methods

Next, we shall examine more closely the results of the IE and VSD filters and carry out the systematic comparison of algorithms presented in Subsection 11.5.3. The question is whether, in view of the limited sample size, one can state that the VSD algorithm is superior to the IE algorithm.

The performance of interest will be the mean square error in the estimate of one state component, not indicated explicitly, for simplicity

$$\epsilon^{\text{VSD}}(k) \triangleq E[[\hat{x}^{\text{VSD}}(k|k)]^2] \quad (11.5.4-1)$$

$$\epsilon^{\text{IE}}(k) \triangleq E[[\hat{x}^{\text{IE}}(k|k)]^2] \quad (11.5.4-2)$$

The *sample mean* of the performance from  $N$  Monte Carlo runs is

$$\bar{\epsilon}^{\text{VSD}}(k) = \frac{1}{N} \sum_{i=1}^N [\tilde{x}^{\text{VSD}i}(k|k)]^2 \quad (11.5.4-3)$$

$$\bar{\epsilon}^{\text{IE}}(k) = \frac{1}{N} \sum_{i=1}^N [\tilde{x}^{\text{IE}i}(k|k)]^2 \quad (11.5.4-4)$$

where  $\tilde{x}^{\text{VSD}i}(k|k)$  is the estimation error of the state component under consideration at time  $k$  in run  $i$  with algorithm VSD, and similarly for IE. These are the quantities shown in Figs. 11.5.3-1a and 11.5.3-1b.

We want to test if one can accept the hypothesis

$$H_1 : T^{\text{IV}}(k) \triangleq \bar{\epsilon}^{\text{IE}}(k) - \bar{\epsilon}^{\text{VSD}}(k) > 0 \quad (11.5.4-5)$$

that is, that algorithm VSD is superior to algorithm IE. Note that “superiority” is based on the *true mean square errors* defined in (11.5.4-1) and (11.5.4-2). A comparison of the *sample mean square errors* (11.5.4-3) and (11.5.4-4) corresponding to the two algorithms is not sufficient (actually is rather naive) since the inaccuracy in these sample means is not considered at all.

The correct statistical test is, as discussed in Subsection 1.5.3, based on the *sample performance differences*

$$T^{\text{IV}i}(k) \triangleq [\tilde{x}^{\text{IE}i}(k|k)]^2 - [\tilde{x}^{\text{VSD}i}(k|k)]^2 \quad (11.5.4-6)$$

The test uses the sample mean  $\bar{T}$  of the above differences and its standard error  $\sigma_{\bar{T}}$  given in (1.5.3-9) and (1.5.3-10), respectively.

If the ratio  $\bar{T}/\sigma_{\bar{T}}$  exceeds a threshold, then the *difference of the true means* in (11.5.4-5) is accepted as positive ( $H_1$  is accepted) and the null hypothesis

$$H_0 : T^{\text{IV}} \leq 0 \quad (11.5.4-7)$$

is rejected because it has a “low level of significance.” This is equivalent to saying that the estimated mean  $\bar{T}$  is positive and statistically significant (i.e.,  $H_1$  can be accepted because it is “significant”).

Hypothesis  $H_1$  is usually accepted only if the significance level of  $H_0$  is less than 5%, in which case the test is

$$\frac{\bar{T}}{\sigma_{\bar{T}}} > \mathcal{G}(95\%) = 1.65 \quad (11.5.4-8)$$

where  $\mathcal{G}(1 - \alpha)$  represents the point on the standard Gaussian distribution corresponding to upper tail probability of  $\alpha$ .

The above statistical test was applied for comparing the performance over intervals  $[k, l]$ , rather than single points in time, by replacing (11.5.4-6) with

$$T^{\text{IV}i}(k, l) = \frac{1}{l - k + 1} \sum_{m=k}^l \{[\tilde{x}^{\text{IE}i}(m|m)]^2 - [\tilde{x}^{\text{VSD}i}(m|m)]^2\} \quad (11.5.4-9)$$

Table 11.5.4-1 shows the test for the difference of the MSE between the two algorithms over several time intervals. Note that the slow maneuver took place during the time interval [40, 60] and that the fast one took place during the interval [61, 66]. Statistically significant improvements of the VSD algorithm over the IE algorithm are observed in position over all the intervals considered and in velocity over two intervals. In the remaining two intervals, while there was improvement, it was not sufficient to reject  $H_0$  at the 5% level. At the 8% level, however,  $H_0$  would have been rejected in all cases.

**Table 11.5.4-1:** Test of means for comparison of VSD and IE algorithms from 50 runs.

Time interval	Velocity			Position		
	$\bar{T}$	$\sigma_{\bar{T}}$	Test statistic	$\bar{T}$	$\sigma_{\bar{T}}$	Test statistic
40-60	8.06	1.19	6.77	13,560	735	18.45
60-70	4.99	3.28	1.52	10,336	1,013	10.20
40-70	7.72	1.43	5.40	12,677	619	20.48
60-80	2.45	1.75	1.40	6,091	626	9.73

## 11.5.5 Comparison of Several Algorithms — Summary

The optimal statistical method of comparing two algorithms based on Monte Carlo runs has been illustrated in the evaluation of the variable state dimension and input estimation filters.

The VSD filter turned out to be superior (based on statistical significance analysis) and less demanding computationally than the IE filter.

The much simpler two-level white noise filter appears to have performance close to the VSD filter.

A multiple-level white or autocorrelated noise model seems to be the best compromise from the point of view of performance versus cost of implementation.

The computational requirements of the IE filter, the VSD filter, and the two-level process noise filter were, approximately, 8 : 2 : 1.

Neither of these three filters considered here managed to keep the *peak position estimation error* (during the maneuver) below the *raw position measurement RMS error* — the measurement noise standard deviation. This is a major shortcoming of most adaptive estimation schemes.

As will be shown in the next section, the multiple model approach can accomplish this (almost<sup>1</sup>) while yielding good noise reduction during the constant velocity portions of the trajectory.

<sup>1</sup>In the spirit of “Our algorithms can almost do almost everything.”

## 11.6 THE MULTIPLE MODEL APPROACH

### 11.6.1 Formulation of the Approach

In the *multiple model (MM) approach* it is assumed that the system obeys one of a finite number of models. Such systems are called *hybrid*: They have both *continuous* (noise) uncertainties and *discrete* uncertainties — *model or mode*, or *operating regime* uncertainties.

A *Bayesian framework* is used: Starting with prior probabilities of each model being correct (i.e., the system is in a particular mode), the corresponding posterior probabilities are obtained.

First the static case in which the model the system obeys is *fixed*, that is, no switching from one mode to another occurs during the estimation process (time-invariant mode) is considered. This will result in the *static MM estimator*. While the model that is in effect stays fixed, each model has its own dynamics, so *the overall estimator is dynamic*.

The model, assumed to be in effect throughout the entire process, is one of  $r$  possible models (the system is in one of  $r$  modes)

$$M \in \{M_j\}_{j=1}^r \quad (11.6.1-1)$$

The prior probability that  $M_j$  is correct (the system is in mode  $j$ ) is

$$P\{M_j|Z^0\} = \mu_j(0) \quad j = 1, \dots, r \quad (11.6.1-2)$$

where  $Z^0$  is the prior information and

$$\sum_{j=1}^r \mu_j(0) = 1 \quad (11.6.1-3)$$

since the correct model is among the assumed  $r$  possible models.

It will be assumed that all models are linear-Gaussian. This approach can be used for nonlinear systems as well via linearization.

Subsequently, the dynamic situation of *switching models* or *mode jumping* is considered. In the latter case, the system undergoes transitions from one mode to another. The resulting estimator is a *dynamic MM estimator*.

### 11.6.2 The Static Multiple Model Estimator

The *static MM estimator* — for fixed models — is obtained as follows.

Using Bayes' formula, the posterior probability of model  $j$  being correct, given the measurement data up to  $k$ , is given by the recursion

$$\begin{aligned} \mu_j(k) &\triangleq P\{M_j|Z^k\} = P\{M_j|z(k), Z^{k-1}\} = \frac{p[z(k)|Z^{k-1}, M_j]P\{M_j|Z^{k-1}\}}{p[z(k)|Z^{k-1}]} \\ &= \frac{p[z(k)|Z^{k-1}, M_j]P\{M_j|Z^{k-1}\}}{\sum_{i=1}^r p[z(k)|Z^{k-1}, M_i]P\{M_i|Z^{k-1}\}} \end{aligned} \quad (11.6.2-1)$$

or

$$\mu_j(k) = \frac{p[z(k)|Z^{k-1}, M_j]\mu_j(k-1)}{\sum_{i=1}^r p[z(k)|Z^{k-1}, M_i]\mu_i(k-1)} \quad j = 1, \dots, r \quad (11.6.2-2)$$

starting with the given prior probabilities (11.6.1-2).

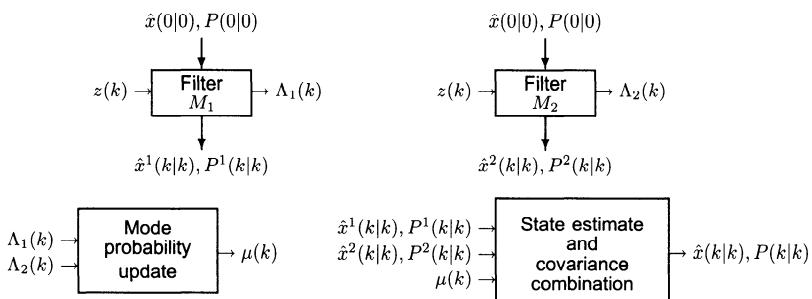
The first term on the right-hand side above is the *likelihood function of mode  $j$*  at time  $k$ , which, under the linear-Gaussian assumptions, is given by the expression (5.2.6-7)

$$\Lambda_j(k) \triangleq p[z(k)|Z^{k-1}, M_j] = p[\nu_j(k)] = \mathcal{N}[\nu_j(k); 0, S_j(k)] \quad (11.6.2-3)$$

where  $\nu_j$  and  $S_j$  are the innovation and its covariance from the *mode-matched filter* corresponding to mode  $j$ . In a nonlinear and/or non-Gaussian situation the same Gaussian likelihood functions are used, even though they are clearly approximations.

Thus a Kalman filter matched to each mode is set up yielding *mode-conditioned state estimates* and the associated *mode-conditioned covariances*. The probability of each mode being correct — the *mode estimates* — is obtained according to (11.6.2-2) based on its likelihood function (11.6.2-3) relative to the other filters' likelihood functions. In a nonlinear situation the filters are EKF instead of KF.

This modular estimator, which is a *bank of filters*, is illustrated in Fig. 11.6.2-1.



**Figure 11.6.2-1:** The static multiple model estimator for  $r = 2$  fixed models.

The output of each mode-matched filter is the *mode-conditioned state estimate*  $\hat{x}^j$ , the associated covariance  $P^j$  and the *mode likelihood function*  $\Lambda_j$ .

After the filters are initialized, they run recursively *on their own estimates*. Their likelihood functions are used to update the mode probabilities. The latest mode probabilities are used to combine the mode-conditioned estimates and covariances.

Under the above assumptions the pdf of the state of the system is a Gaussian mixture with  $r$  terms

$$p[x(k)|Z^k] = \sum_{j=1}^r \mu_j(k) \mathcal{N}[x(k); \hat{x}^j(k|k), P^j(k|k)] \quad (11.6.2-4)$$

The combination of the mode-conditioned estimates is done therefore as follows

$$\hat{x}(k|k) = \sum_{j=1}^r \mu_j(k) \hat{x}^j(k|k) \quad (11.6.2-5)$$

and the covariance of the combined estimate is (see Subsection 1.4.16)

$$P(k|k) = \sum_{j=1}^r \mu_j(k) \{ P^j(k|k) + [\hat{x}^j(k|k) - \hat{x}(k|k)][\hat{x}^j(k|k) - \hat{x}(k|k)]' \} \quad (11.6.2-6)$$

where the last term above is the *spread of the means* term.

The above is exact under the following assumptions:

1. The correct model is among the set of models considered,
2. The same model has been in effect from the initial time.

Assumption 1 can be considered a reasonable approximation; however, 2 is obviously not true if a maneuver has started at some time within the interval  $[1, k]$ , in which case a model change — *mode jump* — occurred.

### Convergence of the Mode Estimates

If the mode set includes the correct one and no mode jump occurs, then the probability of the true mode will converge to unity, that is, this approach yields consistent estimates of the system parameters. Otherwise the probability of the model “nearest” to the correct one will converge to unity (this is discussed in detail in [Baram78]).

### Ad Hoc Modifications for the Case of Switching Models

The following ad hoc modification can be made to the static MM estimator for estimating the state in the case of switching models: An artificial lower bound is imposed on the model probabilities (with a suitable renormalization of the remaining probabilities).

A shortcoming of the static MM estimator when used with switching models is that, in spite of the above ad hoc modification, the mismatched filters’ errors can grow to unacceptable levels. Thus, reinitialization of the filters that are mismatched is, in general, needed. This is accomplished by using the estimate from filter corresponding to the best matched model in the other filters.

It should be pointed out that the above “fixes” are automatically (and rigorously) built into the dynamic MM estimation algorithms to be discussed next.

### 11.6.3 The Dynamic Multiple Model Estimator

In this case the mode the system is in can undergo switching in time. The system is modeled by the equations

$$x(k) = F[M(k)]x(k-1) + v[k-1, M(k)] \quad (11.6.3-1)$$

$$z(k) = H[M(k)]x(k) + w[k, M(k)] \quad (11.6.3-2)$$

where  $M(k)$  denotes the mode or model “at time  $k$ ” — in effect *during the sampling period ending at  $k$* . Such systems are also called **jump-linear systems**. The mode jump process is assumed **left-continuous** (i.e., the impact of the new model starts at  $t_k^+$ ).

The mode at time  $k$  is assumed to be among the possible  $r$  modes

$$M(k) \in \{M_j\}_{j=1}^r \quad (11.6.3-3)$$

The continuous-valued vector  $x(k)$  and the discrete variable  $M(k)$  are sometimes referred to as the **base state** and the **modal state**, respectively.

For example,

$$F[M_j] = F_j \quad (11.6.3-4)$$

$$v(k-1, M_j) \sim \mathcal{N}(u_j, Q_j) \quad (11.6.3-5)$$

that is, the structure of the system and/or the statistics of the noises might be different from model to model. The mean  $u_j$  of the noise can model a maneuver as a deterministic input.

The  $l$ th **mode history** — or **sequence of models** — through time  $k$  is denoted as

$$M^{k,l} = \{M_{i_{1,l}}, \dots, M_{i_{k,l}}\} \quad l = 1, \dots, r^k \quad (11.6.3-6)$$

where  $i_{\kappa,l}$  is the model index at time  $\kappa$  from history  $l$  and

$$1 \leq i_{\kappa,l} \leq r \quad \kappa = 1, \dots, k \quad (11.6.3-7)$$

Note that the number of histories increases *exponentially with time*.

For example, with  $r = 2$  one has at time  $k = 2$  the following  $r^k = 4$  possible sequences (histories) as shown below:

$l$	$i_{1,l}$	$i_{2,l}$
1	1	1
2	1	2
3	2	1
4	2	2

It will be assumed that the **mode (model) switching** — that is, the **mode jump process** — is a Markov process (Markov chain) with known mode transition probabilities

$$\boxed{p_{ij} \triangleq P\{M(k) = M_j | M(k-1) = M_i\}} \quad (11.6.3-8)$$

These **mode transition probabilities** will be assumed time-invariant and independent of the base state. In other words, this is a **homogeneous Markov chain**.

The system (11.6.3-1), (11.6.3-2), and (11.6.3-8) is a generalized version of a **hidden Markov model**.

The event that model  $j$  is in effect at time  $k$  is denoted as

$$M_j(k) \triangleq \{M(k) = M_j\} \quad (11.6.3-9)$$

The conditional probability of the  $l$ th history

$$\mu^{k,l} \triangleq P\{M^{k,l}|Z^k\} \quad (11.6.3-10)$$

will be evaluated next.

The  $l$ th sequence of models through time  $k$  can be written as

$$M^{k,l} = \{M^{k-1,s}, M_j(k)\} \quad (11.6.3-11)$$

where sequence  $s$  through  $k-1$  is its *parent sequence* and  $M_j$  is its last element.

Then, in view of the Markov property,

$$P\{M_j(k)|M^{k-1,s}\} = P\{M_j(k)|M_i(k-1)\} \triangleq p_{ij} \quad (11.6.3-12)$$

where  $i = s_{k-1}$ , the index of the last model in the parent sequence  $s$  through  $k-1$ .

The conditional pdf of the state at  $k$  is obtained using the total probability theorem with respect to the mutually exclusive and exhaustive set of events (11.6.3-6), as a **Gaussian mixture** with an *exponentially increasing number of terms*

$$p[x(k)|Z^k] = \sum_{l=1}^{r^k} p[x(k)|M^{k,l}, Z^k] P\{M^{k,l}|Z^k\} \quad (11.6.3-13)$$

Since *to each mode sequence one has to match a filter*, it can be seen that an exponentially increasing number of filters are needed to estimate the (base) state, which makes the optimal approach impractical.

The probability of a mode history is obtained using Bayes' formula as

$$\begin{aligned} \mu^{k,l} &= P\{M^{k,l}|Z^k\} \\ &= P\{M^{k,l}|z(k), Z^{k-1}\} \\ &= \frac{1}{c} p[z(k)|M^{k,l}, Z^{k-1}] P\{M^{k,l}|Z^{k-1}\} \\ &= \frac{1}{c} p[z(k)|M^{k,l}, Z^{k-1}] P\{M_j(k), M^{k-1,s}|Z^{k-1}\} \\ &= \frac{1}{c} p[z(k)|M^{k,l}, Z^{k-1}] P\{M_j(k)|M^{k-1,s}, Z^{k-1}\} \mu^{k-1,s} \\ &= \frac{1}{c} p[z(k)|M^{k,l}, Z^{k-1}] P\{M_j(k)|M^{k-1,s}\} \mu^{k-1,s} \end{aligned} \quad (11.6.3-14)$$

where  $c$  is the normalization constant.

If the current mode depends only on the previous one (i.e., it is a Markov chain), then

$$\mu^{k,l} = \frac{1}{c} p[z(k)|M^{k,l}, Z^{k-1}] P\{M_j(k)|M_i(k-1)\} \mu^{k-1,s} \quad (11.6.3-15)$$

or

$$\boxed{\mu^{k,l} = \frac{1}{c} p[z(k)|M^{k,l}, Z^{k-1}] p_{ij} \mu^{k-1,s}} \quad (11.6.3-16)$$

where  $i = s_{k-1}$  is the last model of the parent sequence  $s$ .

The above equation shows that *conditioning on the entire past history* is needed even if the random parameters are Markov.

## Practical Algorithms

The only way to avoid the exponentially increasing number of histories, which have to be accounted for, is by going to suboptimal techniques.

A simple-minded suboptimal technique is to keep, say, the  $N$  histories with the largest probabilities, discard the rest, and renormalize the probabilities such that they sum up to unity.

The *generalized pseudo-Bayesian (GPB)* approaches combine histories of models that differ in “older” models. The first-order GPB, denoted as GPB1, considers only the possible models in the last sampling period. The second-order version, GPB2, considers all the possible models in the last two sampling periods. These algorithms require  $r$  and  $r^2$  filters to operate in parallel, respectively.

Finally, the *interacting multiple model (IMM)* estimation algorithm will be presented. This algorithm is conceptually similar to GPB2, but requires only  $r$  filters to operate in parallel.

## The Mode Transition Probabilities

The *mode transition probabilities* (11.6.3-8), indicated as assumed to be known, are actually *estimator design parameters* to be selected in the design process of the algorithm. This will be discussed in detail in Subsections 11.6.7 and 11.7.3.

## Note

The GPB1 and IMM algorithms have approximately the same computational requirements as the static (fixed model) MM algorithm, but do not require ad hoc modifications as the latter, which is actually obsolete for switching models.

### 11.6.4 The GPB1 Multiple Model Estimator for Switching Models

In the *generalized pseudo-Bayesian estimator of first order (GPB1)*, at time  $k$  the state estimate is computed under each possible current model — a total of  $r$  possibilities (hypotheses) are considered. All histories that differ in “older” models are combined together.

The total probability theorem is thus used as follows:

$$\begin{aligned} p[x(k)|Z^k] &= \sum_{j=1}^r p[x(k)|M_j(k), Z^k] P\{M_j(k)|Z^k\} \\ &= \sum_{j=1}^r p[x(k)|M_j(k), z(k), Z^{k-1}] \mu_j(k) \\ &\approx \sum_{j=1}^r p[x(k)|M_j(k), z(k), \hat{x}(k-1|k-1), P(k-1|k-1)] \mu_j(k) \end{aligned} \quad (11.6.4-1)$$

Thus at time  $k-1$  there is a single *lumped estimate*  $\hat{x}(k-1|k-1)$  and the associated covariance that summarize (approximately) the past  $Z^{k-1}$ . From this, one carries out the prediction to time  $k$  and the update at time  $k$  under  $r$  hypotheses, namely,

$$\hat{x}^j(k|k) = \hat{x}[k|k; M_j(k), \hat{x}(k-1|k-1), P(k-1|k-1)] \quad j = 1, \dots, r \quad (11.6.4-2)$$

$$P^j(k|k) = P[k|k; M_j(k), P(k-1|k-1)] \quad j = 1, \dots, r \quad (11.6.4-3)$$

After the update, the estimates are combined with the weightings  $\mu_j(k)$  (detailed later), resulting in the new combined estimate  $\hat{x}(k|k)$ . In other words, *the  $r$  hypotheses are merged into a single hypothesis at the end of each cycle*.

Finally, the mode (or model) probabilities are updated.

Figure 11.6.4-1 describes this estimator, which requires  $r$  filters in parallel.

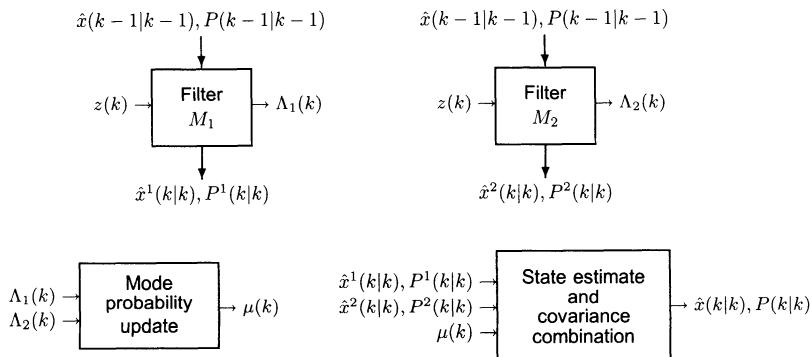
The output of each model-matched filter is the *mode-conditioned state estimate*  $\hat{x}^j$ , the associated covariance  $P^j$  and the *mode likelihood function*  $\Lambda_j$ .

After the filters are initialized, they run recursively using *the previous combined estimate*. Their likelihood functions are used to update the mode probabilities. The latest mode probabilities are used to combine the model-conditional estimates and covariances.

The structure of this algorithm is

$$(N_e; N_f) = (1; r) \quad (11.6.4-4)$$

where  $N_e$  is the *number of estimates* at the start of the cycle of the algorithm and  $N_f$  is the *number of filters* in the algorithm.



**Figure 11.6.4-1:** The GPB1 MM estimator for  $r = 2$  switching models (one cycle).

### The Algorithm

One cycle of the algorithm consists of the following:

1. **Mode-matched filtering** ( $j = 1, \dots, r$ ). Starting with  $\hat{x}(k-1|k-1)$ , one computes  $\hat{x}^j(k|k)$  and the associated covariance  $P^j(k|k)$  through a filter matched to  $M_j(k)$ . The likelihood functions

$$\Lambda_j(k) = p[z(k)|M_j(k), Z^{k-1}] \quad (11.6.4-5)$$

corresponding to these  $r$  filters are evaluated as

$$\boxed{\Lambda_j(k) = p[z(k)|M_j(k), \hat{x}(k-1|k-1), P(k-1|k-1)]} \quad (11.6.4-6)$$

2. **Mode probability update** ( $j = 1, \dots, r$ ). This is done as follows:

$$\begin{aligned}
\mu_j(k) &\triangleq P\{M_j(k)|Z^k\} \\
&= P\{M_j(k)|z(k), Z^{k-1}\} \\
&= \frac{1}{c} p[z(k)|M_j(k), Z^{k-1}] P\{M_j(k)|Z^{k-1}\} \\
&= \frac{1}{c} \Lambda_j(k) \sum_{i=1}^r P\{M_j(k)|M_i(k-1), Z^{k-1}\} \\
&\quad \cdot P\{M_i(k-1)|Z^{k-1}\}
\end{aligned} \quad (11.6.4-7)$$

which yields with  $p_{ij}$  the known *mode transition probabilities*,

$$\boxed{\mu_j(k) = \frac{1}{c} \Lambda_j(k) \sum_{i=1}^r p_{ij} \mu_i(k-1)} \quad (11.6.4-8)$$

where  $c$  is the normalization constant

$$c = \sum_{j=1}^r \Lambda_j(k) \sum_{i=1}^r p_{ij} \mu_i(k-1) \quad (11.6.4-9)$$

**3. State estimate and covariance combination.** The latest combined state estimate and covariance are obtained according to the summation (11.6.4-1) as

$$\hat{x}(k|k) = \sum_{j=1}^r \hat{x}^j(k|k) \mu_j(k) \quad (11.6.4-10)$$

$$P(k|k) = \sum_{j=1}^r \mu_j(k) \{ P^j(k|k) + [\hat{x}^j(k|k) - \hat{x}(k|k)][\hat{x}^j(k|k) - \hat{x}(k|k)]' \} \quad (11.6.4-11)$$

### 11.6.5 The GPB2 Multiple Model Estimator for Switching Models

In the **generalized pseudo-Bayesian estimator of second order** (or **GPB2**), at time  $k$  the state estimate is computed under *each possible current and previous model* — a total of  $r^2$  hypotheses (histories) are considered. All histories that differ only in “older” models are merged.

The total probability theorem is thus used as follows:

$$\begin{aligned} p[x(k)|Z^k] &= \sum_{j=1}^r \sum_{i=1}^r p[x(k)|M_j(k), M_i(k-1), Z^k] P\{M_i(k-1)|M_j(k), Z^k\} \\ &\quad \cdot P\{M_j(k)|Z^k\} \\ &= \sum_{j=1}^r \sum_{i=1}^r p[x(k)|M_j(k), z(k), M_i(k-1), Z^{k-1}] \mu_{i|j}(k-1|k) \mu_j(k) \\ &\approx \sum_{j=1}^r \sum_{i=1}^r p[x(k)|M_j(k), z(k), \hat{x}^i(k-1|k-1), P^i(k-1|k-1)] \\ &\quad \cdot \mu_{i|j}(k-1|k) \mu_j(k) \end{aligned} \quad (11.6.5-1)$$

that is, the past  $\{M_i(k-1), Z^{k-1}\}$  is approximated by the **mode-conditioned estimate**  $\hat{x}^i(k-1|k-1)$  and associated covariance.

Thus at time  $k-1$  there are  $r$  estimates and covariances, each to be predicted to time  $k$  and updated at time  $k$  under  $r$  hypotheses, namely,

$$\hat{x}^{ij}(k|k) \triangleq \hat{x}[k|k; M_j(k), \hat{x}^i(k-1|k-1), P^i(k-1|k-1)] \quad i, j = 1, \dots, r \quad (11.6.5-2)$$

$$P^{ij}(k|k) \triangleq P[k|k; M_j(k), P^i(k-1|k-1)] \quad i, j = 1, \dots, r \quad (11.6.5-3)$$

After the update, the estimates corresponding to the same latest model hypothesis are combined with the weightings  $\mu_{i|j}(k-1|k)$ , detailed later, resulting

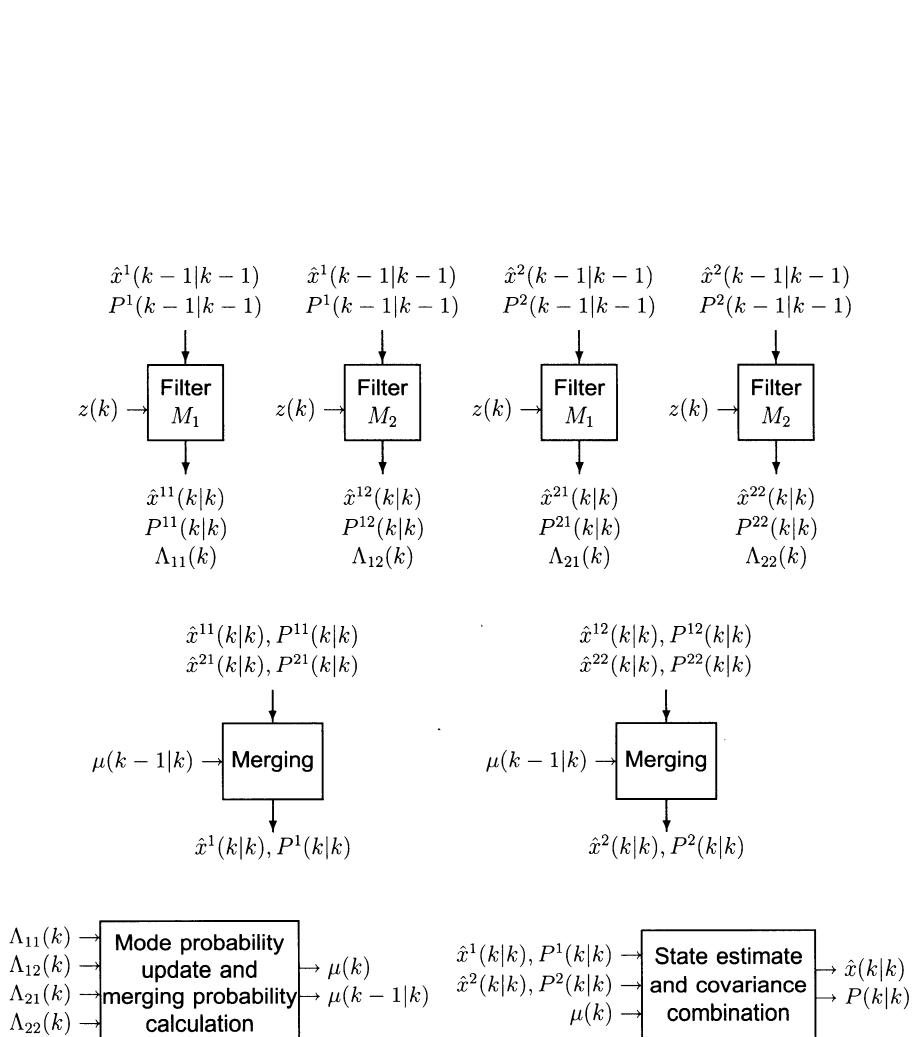


Figure 11.6.5-1: The GPB2 MM estimator for  $r = 2$  models (one cycle).

in  $r$  estimates  $\hat{x}^j(k|k)$ . In other words, the  $r^2$  hypotheses are *merged* into  $r$  at the end of each estimation cycle.

Figure 11.6.5-1 describes this algorithm, which requires  $r^2$  parallel filters.

The structure of the GPB2 algorithm is

$$(N_e; N_f) = (r; r^2) \quad (11.6.5-4)$$

where  $N_e$  is the *number of estimates* at the start of the cycle of the algorithm and  $N_f$  is the *number of filters* in the algorithm.

## The Algorithm

One cycle of the algorithm consists of the following:

1. **Mode-matched filtering** ( $i, j = 1, \dots, r$ ). Starting with  $\hat{x}^i(k-1|k-1)$ , one computes  $\hat{x}^{ij}(k|k)$  and the associated covariance  $P^{ij}(k|k)$  through a filter matched to  $M_j(k)$ . The likelihood functions corresponding to these  $r^2$  filters

$$\Lambda_{ij}(k) = p[z(k)|M_j(k), M_i(k-1), Z^{k-1}] \quad (11.6.5-5)$$

are evaluated as

$$\boxed{\Lambda_{ij}(k) = p[z(k)|M_j(k), \hat{x}^i(k-1|k-1), P^i(k-1|k-1)] \quad i, j = 1, \dots, r} \quad (11.6.5-6)$$

2. **Calculation of the merging probabilities** ( $i, j = 1, \dots, r$ ). The probability that mode  $i$  was in effect at  $k-1$  if mode  $j$  is in effect at  $k$  is, conditioned on  $Z^k$ ,

$$\begin{aligned} \mu_{i|j}(k-1|k) &\triangleq P\{M_i(k-1)|M_j(k), Z^k\} \\ &= P\{M_i(k-1)|z(k), M_j(k), Z^{k-1}\} \\ &= \frac{1}{c_j} P[z(k), M_j(k)|M_i(k-1), Z^{k-1}] P\{M_i(k-1)|Z^{k-1}\} \\ &= \frac{1}{c_j} p[z(k)|M_j(k), M_i(k-1), Z^{k-1}] \\ &\quad \cdot P\{M_j(k)|M_i(k-1), Z^{k-1}\} P\{M_i(k-1)|Z^{k-1}\} \end{aligned} \quad (11.6.5-7)$$

where  $P[\cdot]$  denotes a mixed pdf-probability. Thus the *merging probabilities* are

$$\boxed{\mu_{i|j}(k-1|k) = \frac{1}{c_j} \Lambda_{ij}(k) p_{ij} \mu_i(k-1) \quad i, j = 1, \dots, r} \quad (11.6.5-8)$$

where

$$c_j = \sum_{i=1}^r \Lambda_{ij}(k) p_{ij} \mu_i(k-1) \quad (11.6.5-9)$$

The **mode transition probabilities**  $p_{ij}$  are assumed to be known — their selection is part of the algorithm design process.

3. **Merging** ( $j = 1, \dots, r$ ). The state estimate corresponding to  $M_j(k)$  is obtained by combining the estimates (11.6.5-2) according to the inner summation in (11.6.5-1) as follows

$$\hat{x}^j(k|k) = \sum_{i=1}^r \hat{x}^{ij}(k|k) \mu_{ij}(k-1|k) \quad j = 1, \dots, r \quad (11.6.5-10)$$

The covariance corresponding to the above is

$$P^j(k|k) = \sum_{i=1}^r \mu_{ij}(k-1|k) \{ P^{ij}(k|k) \\ + [\hat{x}^{ij}(k|k) - \hat{x}^j(k|k)][\hat{x}^{ij}(k|k) - \hat{x}^j(k|k)]' \} \quad (11.6.5-11)$$

4. **Mode probability updating** ( $j = 1, \dots, r$ ). This is done as follows

$$\begin{aligned} \mu_j(k) &\triangleq P\{M_j(k)|z(k), Z^{k-1}\} \\ &= \frac{1}{c} P[z(k), M_j(k)|Z^{k-1}] \\ &= \frac{1}{c} \sum_{i=1}^r P[z(k), M_j(k)|M_i(k-1), Z^{k-1}] P\{M_i(k-1)|Z^{k-1}\} \\ &= \frac{1}{c} \sum_{i=1}^r p(z(k)|M_j(k), M_i(k-1), Z^{k-1}) \\ &\quad \cdot P\{M_j(k)|M_i(k-1), Z^{k-1}\} \mu_i(k-1) \end{aligned} \quad (11.6.5-12)$$

Thus the updated **mode probabilities** are

$$\mu_j(k) = \frac{1}{c} \sum_{i=1}^r \Lambda_{ij}(k) p_{ij} \mu_i(k-1) = \frac{c_j}{c} \quad j = 1, \dots, r \quad (11.6.5-13)$$

where  $c_j$  is the expression from (11.6.5-9) and  $c$  is the normalization constant

$$c = \sum_{j=1}^r c_j \quad (11.6.5-14)$$

5. **State estimate and covariance combination**. The latest state estimate and covariance for output only are

$$\hat{x}(k|k) = \sum_{j=1}^r \hat{x}^j(k|k) \mu_j(k) \quad (11.6.5-15)$$

$$P(k|k) = \sum_{j=1}^r \mu_j(k) \{ P^j(k|k) + [\hat{x}^j(k|k) - \hat{x}(k|k)][\hat{x}^j(k|k) - \hat{x}(k|k)]' \} \quad (11.6.5-16)$$

### 11.6.6 The Interacting Multiple Model Estimator

In the *interacting multiple model (IMM) estimator*, at time  $k$  the state estimate is computed under *each possible current model* using  $r$  filters, with each filter using a different combination of the previous model-conditioned estimates — *mixed initial condition*.

The total probability theorem is used as follows to yield  $r$  filters running in parallel:

$$\begin{aligned} p[x(k)|Z^k] &= \sum_{j=1}^r p[x(k)|M_j(k), Z^k] P\{M_j(k)|Z^k\} \\ &= \sum_{j=1}^r p[x(k)|M_j(k), z(k), Z^{k-1}] \mu_j(k) \end{aligned} \quad (11.6.6-1)$$

The model-conditioned posterior pdf of the state, given by

$$p[x(k)|M_j(k), z(k), Z^{k-1}] = \frac{p[z(k)|M_j(k), x(k)]}{p[z(k)|M_j(k), Z^{k-1}]} p[x(k)|M_j(k), Z^{k-1}] \quad (11.6.6-2)$$

reflects one cycle of the state estimation filter matched to model  $M_j(k)$  starting with the prior, which is the last term above.

The total probability theorem is now applied to the last term above (the prior), yielding

$$\begin{aligned} p[x(k)|M_j(k), Z^{k-1}] &= \sum_{i=1}^r p[x(k)|M_j(k), M_i(k-1), Z^{k-1}] \\ &\quad \cdot P\{M_i(k-1)|M_j(k), Z^{k-1}\} \\ &\approx \sum_{i=1}^r p[x(k)|M_j(k), M_i(k-1), \{\hat{x}^l(k-1|k-1), P^l(k-1|k-1)\}_{l=1}^r] \\ &\quad \cdot \mu_{i|j}(k-1|k-1) \\ &= \sum_{i=1}^r p[x(k)|M_j(k), M_i(k-1), \hat{x}^i(k-1|k-1), P^i(k-1|k-1)] \\ &\quad \cdot \mu_{i|j}(k-1|k-1) \end{aligned} \quad (11.6.6-3)$$

The second line above reflects the approximation that the past through  $k-1$  is summarized by  $r$  model-conditioned estimates and covariances. The last line of (11.6.6-3) is a mixture with weightings, denoted as  $\mu_{i|j}(k-1|k-1)$ , different for each current model  $M_j(k)$ . This mixture is assumed to be a mixture of Gaussian pdfs (a Gaussian sum) and then approximated via moment matching

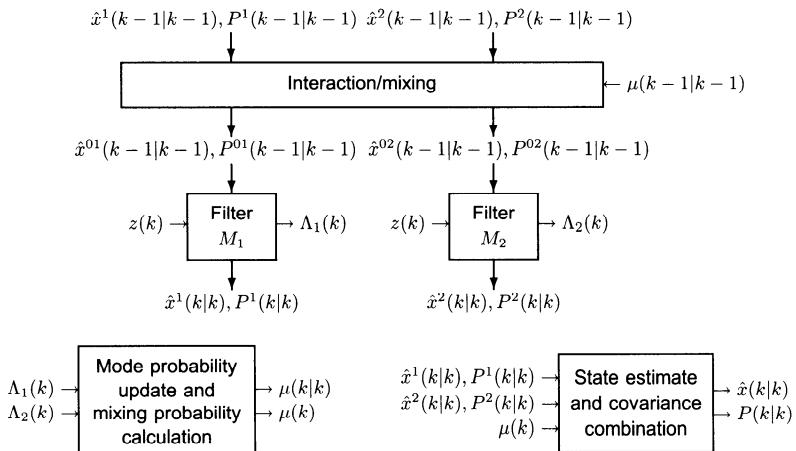
by a single Gaussian (details given later):

$$\begin{aligned}
 p[x(k)|M_j(k), Z^{k-1}] &= \sum_{i=1}^r \mathcal{N}[x(k); E[x(k)|M_j(k), \hat{x}^i(k-1|k-1)], \text{cov}[\cdot]] \\
 &\quad \cdot \mu_{ij}(k-1|k-1) \\
 &\approx \mathcal{N}\left[x(k); \sum_{i=1}^r E[x(k)|M_j(k), \hat{x}^i(k-1|k-1)] \mu_{ij}(k-1|k-1), \text{cov}[\cdot]\right] \\
 &= \mathcal{N}\left[x(k); E[x(k)|M_j(k), \sum_{i=1}^r \hat{x}^i(k-1|k-1) \mu_{ij}(k-1|k-1)], \text{cov}[\cdot]\right]
 \end{aligned} \tag{11.6.6-4}$$

The last line above follows from the linearity of the Kalman filter and amounts to the following:

The input to the filter matched to model  $j$  is obtained from an **interaction** of the  $r$  filters, which consists of the **mixing** of the estimates  $\hat{x}^i(k-1|k-1)$  with the weightings (probabilities)  $\mu_{ij}(k-1|k-1)$ , called the **mixing probabilities**.

The above is equivalent to hypothesis merging *at the beginning* of each estimation cycle [Blom88]. More specifically, the  $r$  hypotheses, instead of “fanning out” into  $r^2$  hypotheses (as in the GPB2 — see Fig. 11.6.5-1), are “mixed” into a new set of  $r$  hypotheses as shown in Fig. 11.6.6-1. This is the key feature that yields  $r$  hypotheses with  $r$  filters, rather than with  $r^2$  filters as in the GPB2 algorithm.



**Figure 11.6.6-1:** The IMM estimator (one cycle).

Figure 11.6.6-1 describes this algorithm, which consists of  $r$  interacting filters operating in parallel. The mixing is done at the input of the filters with

the probabilities, detailed later in (11.6.6-6), conditioned on  $Z^{k-1}$ . In contrast to this, the GPB2 algorithm has  $r^2$  filters and a somewhat similar mixing is done, but at their outputs, with the probabilities (11.6.5-7), conditioned on  $Z^k$ .

The structure of the IMM algorithm is

$$(N_e; N_f) = (r; r) \quad (11.6.6-5)$$

where  $N_e$  is the *number of estimates* at the start of the cycle of the algorithm and  $N_f$  is the *number of filters* in the algorithm.

## The Algorithm

One cycle of the algorithm consists of the following:

1. **Calculation of the mixing probabilities** ( $i, j = 1, \dots, r$ ). The probability that mode  $M_i$  was in effect at  $k-1$  given that  $M_j$  is in effect at  $k$  conditioned on  $Z^{k-1}$  is

$$\begin{aligned} \mu_{i|j}(k-1|k-1) &\triangleq P\{M_i(k-1)|M_j(k), Z^{k-1}\} \\ &= \frac{1}{\bar{c}_j} P\{M_j(k)|M_i(k-1), Z^{k-1}\} P\{M_i(k-1)|Z^{k-1}\} \end{aligned} \quad (11.6.6-6)$$

The above are the **mixing probabilities**, which can be written as

$$\mu_{i|j}(k-1|k-1) = \frac{1}{\bar{c}_j} p_{ij} \mu_i(k-1) \quad i, j = 1, \dots, r \quad (11.6.6-7)$$

where the normalizing constants are

$$\bar{c}_j = \sum_{i=1}^r p_{ij} \mu_i(k-1) \quad j = 1, \dots, r \quad (11.6.6-8)$$

Note the difference between (11.6.6-6), where the conditioning is  $Z^{k-1}$ , and (11.6.5-7), where the conditioning is  $Z^k$ . This is what makes it possible to carry out the mixing at the *beginning* of the cycle, rather than the standard merging at the *end* of the cycle.

2. **Mixing** ( $j = 1, \dots, r$ ). Starting with  $\hat{x}^i(k-1|k-1)$ , one computes the mixed initial condition for the filter matched to  $M_j(k)$  according to (11.6.6-4) as

$$\hat{x}^{0j}(k-1|k-1) = \sum_{i=1}^r \hat{x}^i(k-1|k-1) \mu_{i|j}(k-1|k-1) \quad j = 1, \dots, r \quad (11.6.6-9)$$

The covariance corresponding to the above is

$$\boxed{P^{0j}(k-1|k-1) = \sum_{i=1}^r \mu_{i|j}(k-1|k-1) \left\{ P^i(k-1|k-1) + [\hat{x}^i(k-1|k-1) - \hat{x}^{0j}(k-1|k-1)] \cdot [\hat{x}^i(k-1|k-1) - \hat{x}^{0j}(k-1|k-1)]^T \right\}} \quad j = 1, \dots, r \quad (11.6.6-10)$$

3. **Mode-matched filtering** ( $j = 1, \dots, r$ ). The estimate (11.6.6-9) and covariance (11.6.6-10) are used as input to the filter matched to  $M_j(k)$ , which uses  $z(k)$  to yield  $\hat{x}^j(k|k)$  and  $P^j(k|k)$ .

The likelihood functions corresponding to the  $r$  filters

$$\Lambda_j(k) = p[z(k)|M_j(k), Z^{k-1}] \quad (11.6.6-11)$$

are computed using the mixed initial condition (11.6.6-9) and the associated covariance (11.6.6-10) as

$$\boxed{\Lambda_j(k) = p[z(k)|M_j(k), \hat{x}^{0j}(k-1|k-1), P^{0j}(k-1|k-1)]} \quad j = 1, \dots, r \quad (11.6.6-12)$$

that is,

$$\boxed{\Lambda_j(k) = \mathcal{N}[z(k); \hat{z}^j[k|k-1; \hat{x}^{0j}(k-1|k-1)], S^j[k; P^{0j}(k-1|k-1)]]} \quad j = 1, \dots, r \quad (11.6.6-13)$$

4. **Mode probability update** ( $j = 1, \dots, r$ ). This is done as follows:

$$\begin{aligned} \mu_j(k) &\triangleq P\{M_j(k)|Z^k\} \\ &= \frac{1}{c} p[z(k)|M_j(k), Z^{k-1}] P\{M_j(k)|Z^{k-1}\} \\ &= \frac{1}{c} \Lambda_j(k) \sum_{i=1}^r P\{M_j(k)|M_i(k-1), Z^{k-1}\} P\{M_i(k-1)|Z^{k-1}\} \\ &= \frac{1}{c} \Lambda_j(k) \sum_{i=1}^r p_{ij} \mu_i(k-1) \quad j = 1, \dots, r \end{aligned} \quad (11.6.6-14)$$

or

$$\boxed{\mu_j(k) = \frac{1}{c} \Lambda_j(k) \bar{c}_j} \quad j = 1, \dots, r \quad (11.6.6-15)$$

where  $\bar{c}_j$  is the expression from (11.6.6-8) and

$$c = \sum_{j=1}^r \Lambda_j(k) \bar{c}_j \quad (11.6.6-16)$$

is the normalization constant for (11.6.6-15).

**5. Estimate and covariance combination.** Combination of the model-conditioned estimates and covariances is done according to the mixture equations

$$\hat{x}(k|k) = \sum_{j=1}^r \hat{x}^j(k|k) \mu_j(k) \quad (11.6.6-17)$$

$$P(k|k) = \sum_{j=1}^r \mu_j(k) \{ P^j(k|k) + [\hat{x}^j(k|k) - \hat{x}(k|k)][\hat{x}^j(k|k) - \hat{x}(k|k)]' \}$$

(11.6.6-18)

This combination is *only* for output purposes — it is not part of the algorithm recursions.

### Note

One possible generalization of the IMM estimator is the “second-order IMM” with an extra period depth. While the derivations are rather lengthy, it has been reported that this algorithm is *identical* to the GPB2 [Barret90].

### 11.6.7 An Example with the IMM Estimator

The use of the IMM estimator is illustrated on the example simulated in Section 11.5 where several of the earlier techniques were compared. The results presented in the sequel, which were obtained with DynaEst™, deal with the turn of 90° over 20 sampling periods.

A two-model IMM, designated as IMM2, was first used. This algorithm consisted of

1. A constant velocity model (second-order, with no process noise); and
2. A Wiener process acceleration model (third-order model) with process noise (acceleration increment over a sampling period)  $q = 10^{-3} = (0.0316 \text{ m/s}^2)^2$ .

Note that the acceleration in this case ( $0.075 \text{ m/s}^2$ ) corresponds to about  $2.4\sqrt{q}$ .

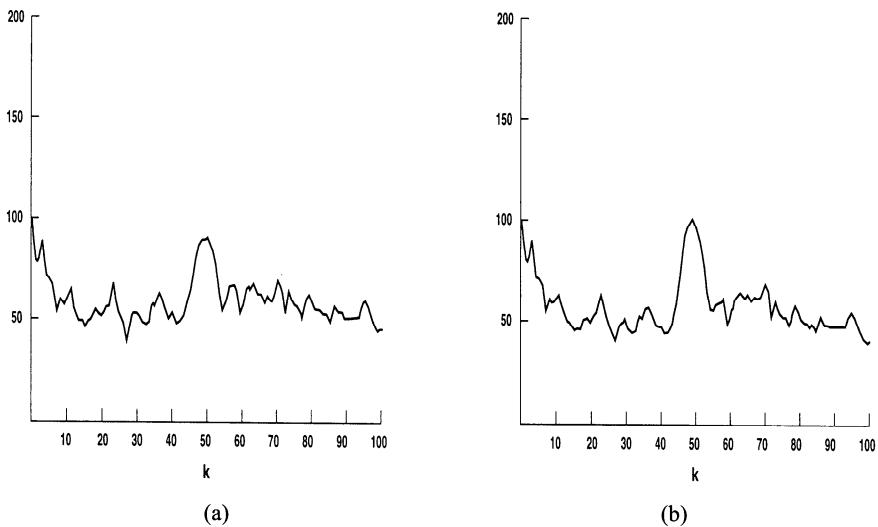
The Markov chain transition matrix between these models was taken as

$$[p_{ij}] = \begin{bmatrix} 0.95 & 0.05 \\ 0.05 & 0.95 \end{bmatrix} \quad (11.6.7-1)$$

The final results were not very sensitive to these values (e.g.,  $p_{11}$  can be between 0.8 and 0.98). The lower (higher) value will yield less (more) peak error during maneuver but higher (lower) RMS error during the quiescent period; that is, it has a higher (lower) bandwidth.

A three-model IMM, designated as IMM3, was also used. This consisted of the above two models plus another one:

3. A constant acceleration (third-order) model without process noise.



**Figure 11.6.7-1:** Position RMS error. (a) IMM2, (b) IMM3.

The Markov chain transition matrix was

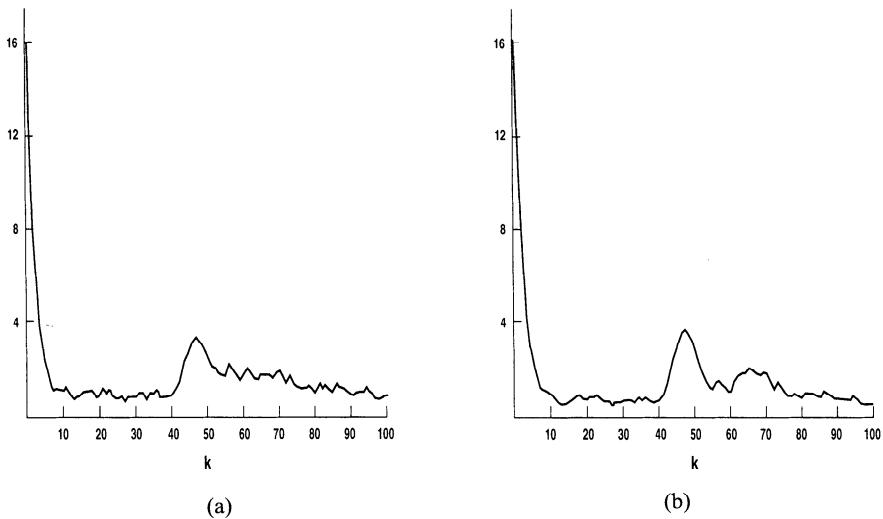
$$[p_{ij}] = \begin{bmatrix} 0.95 & 0.05 & 0 \\ 0.33 & 0.34 & 0.33 \\ 0 & 0.05 & 0.95 \end{bmatrix} \quad (11.6.7-2)$$

Figure 11.6.7-1 shows the position (coordinate  $\xi$ ) RMS error from 50 Monte Carlo runs for the IMM2 and IMM3, with the estimator design parameters as indicated above. Comparing with Fig. 11.5.3-1a, it can be seen that the peak errors are approximately equal to the measurement noise standard deviation (which is 100 m), that is, substantially smaller than with the IE or VSD. During the nonaccelerating period the errors are somewhat larger, but still there is an **RMS noise reduction factor** of 2; this corresponds to a **variance noise reduction factor** of 4.

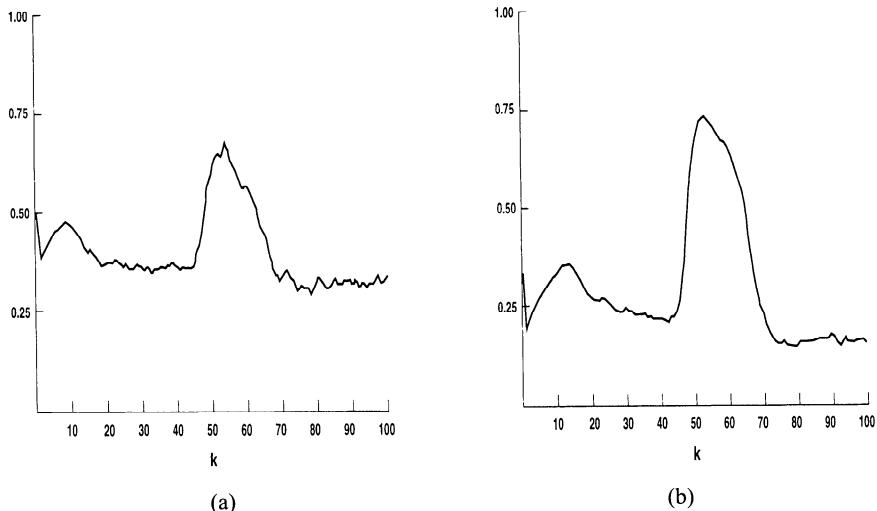
Similar results for the velocity (in coordinate  $\xi$ ) can be observed by comparing Figs. 11.6.7-2 with Fig. 11.5.3-1b.

Overall, the IMM2 and the IMM3 perform similarly in the position estimation; the IMM3 has a somewhat better velocity estimation capability.

Figure 11.6.7-3 shows the evolution in time of the maneuvering mode probabilities in IMM2 and IMM3. The figure illustrates the *soft switching* that takes place when a maneuver is initiated: It takes a few samples in this case to “detect” the maneuver. The “detection” of a maneuver manifests itself here as a sharp increase in the probability of the maneuvering mode — mode 2 in IMM2 and mode 3 in IMM3.



**Figure 11.6.7-2:** Velocity RMS error. (a) IMM2, (b) IMM3.



**Figure 11.6.7-3:** Maneuvering mode probability. (a) IMM2, (b) IMM3.

## Remark

These results, based on [Bar-Shalom89], are not only superior to the IE technique of Section 11.3 (illustrated in Subsection 11.5.3), but also superior to the augmented version of the IE algorithm that estimates the maneuver onset time [Bogler87].

This shows that the IMM, at the cost of about three Kalman filters, is preferable to the augmented IE technique, which has a much larger implementation complexity — of 30 to 100 Kalman filters.

### 11.6.8 Use of DynaEst<sup>TM</sup> to Design an IMM Estimator

In Subsection 5.3.3 the use of the companion software DynaEst<sup>TM</sup> for specifying a simulation scenario and designing a Kalman filter for estimation was demonstrated. In this section, the design of an IMM estimator using DynaEst is illustrated. One of the IMM estimator configurations presented in Subsection 11.6.7, IMM2, which consisted of two filter modules, is used for this. Before proceeding to design the IMM estimator, DynaEst has to be used to define the simulation scenario in Section 11.5 as shown in Figs. 5.3.3-3–5.3.3-11.

Following the design of the simulation scenario, the IMM estimator design steps are as follows.

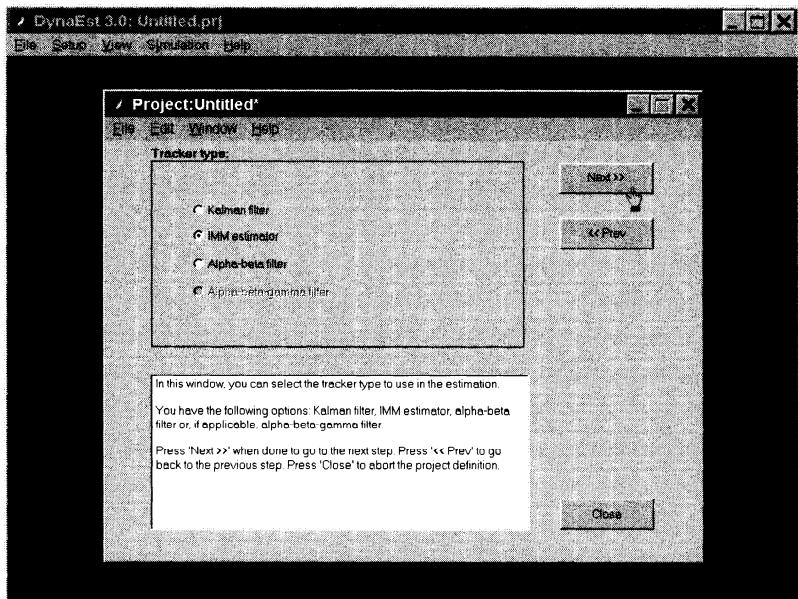
**Selection of a Tracker** The selection of a Kalman filter as the estimator was shown in Fig. 5.3.3-13. Now, the IMM estimator is selected as the tracker as shown in Fig. 11.6.8-1.

**Structure Selection of an IMM Estimator** First, the number of filter modules in the IMM estimator has to be specified. In addition, the nature of the mode transition probability matrix  $[p_{ij}]$  has to be selected. DynaEst offers a choice between using a fixed mode transition probability matrix and using an adaptive one whose elements are modified at each sampling time based on the revisit interval and the **mean sojourn time** of each mode.

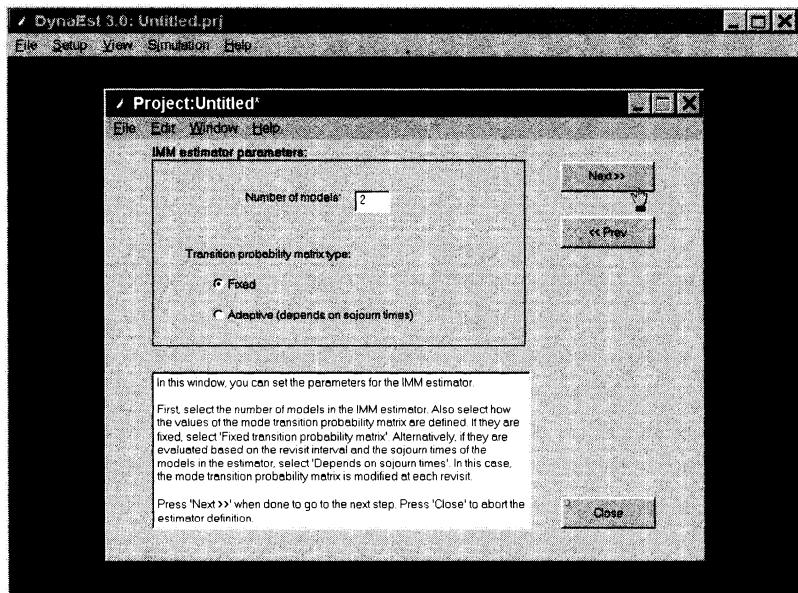
The simulation scenario defined in Section 11.5 uses a constant sampling interval  $T = 10\text{ s}$ . Hence, using a fixed mode transition probability matrix, as shown in Fig. 11.6.8-2, is appropriate.

**The Mode Transition Probability Matrix** The fixed mode transition probability matrix of the IMM2 estimator, given in (11.6.7-1), is specified in DynaEst as shown in Fig. 11.6.8-3.

**Selection of an IMM Filter Module to Design** Since the IMM estimator consists of a number of filter modules, possibly with different state-space models, the parameters for them have to be specified individually for each module. The first filter module is selected for design as shown in Fig. 11.6.8-4.



**Figure 11.6.8-1:** Selection of an IMM estimator as the tracker.



**Figure 11.6.8-2:** Structure selection of an IMM estimator.

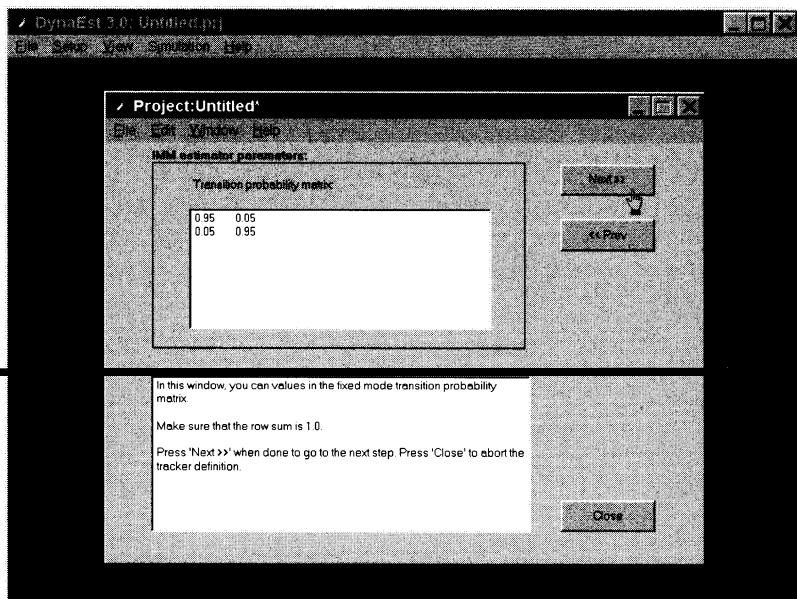


Figure 11.6.8-3: Specification of the mode transition probability matrix.

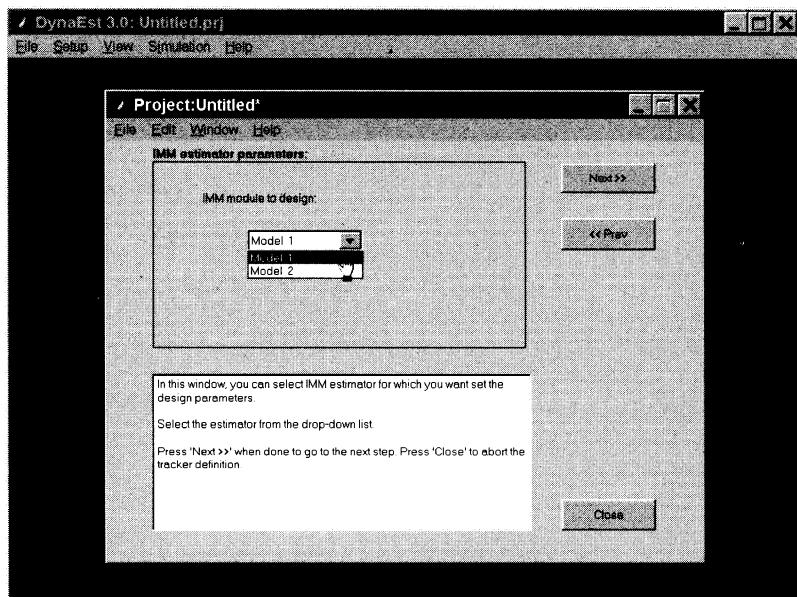
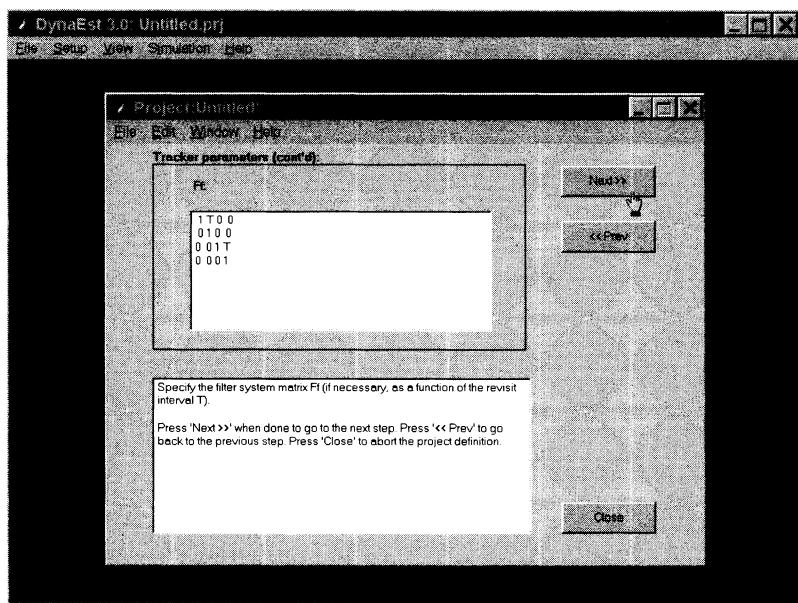


Figure 11.6.8-4: Selection of an IMM filter module to design.



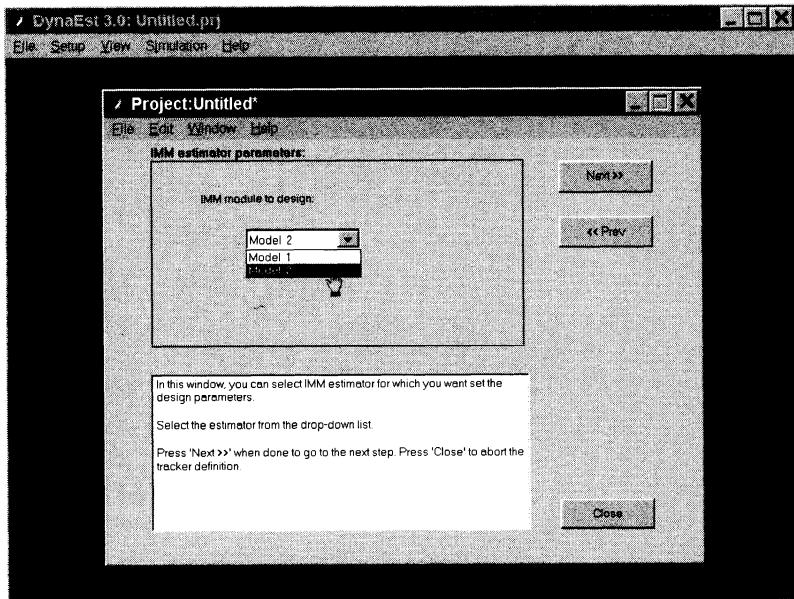
**Figure 11.6.8-5:** Design of a filter module.

**Design of a Filter Module** Selecting the parameters in a module is the crucial step in the IMM estimator design. This step involves selecting the assumed motion model and setting the corresponding process noise parameters in the module. DynaEst offers a sequence of windows similar to those in Figs. 5.3.3-6–5.3.3-11 for setting the motion model, noise parameters, initial estimates, and the corresponding covariance matrix. The first window in the IMM filter module design process is shown in Fig. 11.6.8-5.

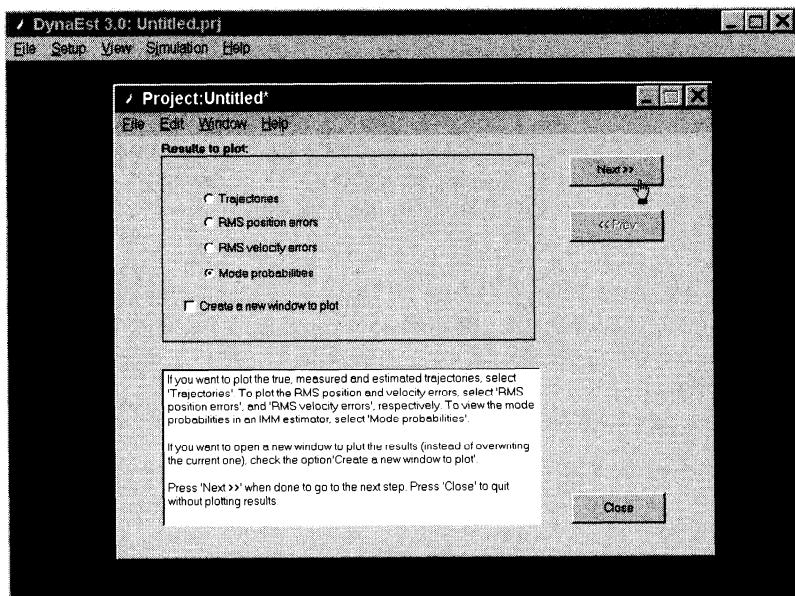
**Selection of the Next Filter Module to Design** After defining the target motion model, process noise parameters, initial estimate, and the corresponding covariance matrix for the first IMM filter module, the next step is to select another filter module for designing as shown in Fig. 11.6.8-6. After selecting this filter module, its parameters are defined as described earlier.

**Viewing Estimation Results** After the IMM estimator is designed, DynaEst executes the specified number of Monte Carlo runs and offers choices to plot the estimation results as shown in Fig. 11.6.8-7. Note that the mode probability plot option is enabled only when the IMM estimator is used for estimation.

The design of the other IMM estimator, IMM3, can be carried out in a similar



**Figure 11.6.8-6:** Selection of the next filter module to design.



**Figure 11.6.8-7:** Viewing estimation results.

manner, but with the additional constant acceleration (third-order) model. This is left as an exercise to the reader.

### 11.6.9 The Multiple Model Approach — Summary

The *multiple model* or *hybrid system* approach assumes the system to be in one of a finite number of modes (i.e., that it is described by one out of a finite number of models).

Each model is characterized by its parameters — the models can differ, for instance, in the level of the process noise (its variance), a deterministic input, and/or any other parameter (different dimension state vectors are also possible).

For the *fixed model* case the estimation algorithm consists of the following:

- For each model a filter “matched” to its parameters is yielding *model-conditioned estimates and covariances*.
- A *mode probability calculator* — a *Bayesian model comparator* — updates the probability of each mode using
  - the likelihood function (innovations) of each filter
  - the prior probability of each model
- An *Estimate combiner* computes the overall estimate and the associated covariance as the weighted sum of the model-conditioned estimates and the corresponding covariance — via the (Gaussian) mixture equations.

The typical ad hoc modification of the fixed model (static) approach to handle switching models is to impose a lower bound on the probability of each model.

For systems that undergo changes in their mode during their operation — *mode jumping (model switching)* — one can obtain the *optimal multiple model estimator* which, however, consists of an exponentially increasing number of filters.

This is because the optimal approach requires conditioning on each *mode history*, and their number is increasing exponentially. Thus, suboptimal algorithms are necessary for the (realistic) mode transition situation.

The *first-order generalized pseudo-Bayesian* (GPB1) MM approach computes the state estimate accounting for each possible current model.

The *second-order generalized pseudo-Bayesian* (GPB2) MM approach computes the state estimate accounting for

- Each possible current model
- Each possible model at the previous time

The *interacting multiple model* (IMM) approach computes the state estimate that accounts for *each possible current model* using a suitable mixing of the previous model-conditioned estimates depending on the current model.

These algorithms are *decision-free* — no maneuver detection decision is needed — the algorithms undergo a *soft switching* according to the latest updated mode probabilities.

Table 11.6.9-1 presents a comparison of the complexities (in terms of the functions to be performed) of the static algorithm and the three algorithms presented for switching models.

**Table 11.6.9-1:** Comparison of complexities of the MM algorithms.

	<i>Static</i>	<i>GPB1</i>	<i>GPB2</i>	<i>IMM</i>
Number of filters	$r$	$r$	$r^2$	$r$
Number of combinations of $r$ estimates and covariances	1	1	$r + 1$	$r + 1$
Number of probability calculations	$r$	$r$	$r^2 + r$	$r^2 + r$

As can be seen from the above, the static algorithm has the same requirements as the GPB1. The IMM has only slightly higher requirements than the GPB1, but clearly significantly lower than GPB2.

In view of this, the modifications of the static algorithm for the switching situation are considered obsolete.

In view of the fact that the IMM performs significantly better than GPB1 and almost as well as GPB2 [Blom88], the IMM is considered to be the best compromise between complexity and performance.

Furthermore, the IMM has been shown to be able to keep the position estimation error not worse than the raw measurement error during the critical maneuver periods (onset and termination) and provide significant improvement (noise reduction) at other times.

## 11.7 DESIGN OF AN IMM ESTIMATOR FOR ATC TRACKING

### 11.7.1 ATC Motion Models

In *air traffic control* (ATC), civilian aircraft have two basic modes of flight:

- **Uniform motion (UM)** — the straight and level flight with a constant speed and course
- **Maneuver** — turning or climbing/descending

The horizontal and vertical motion models can be, typically, assumed to be decoupled [Wang99]. The design and evaluation of several estimators for the horizontal motion will be discussed next. The flight modes in the *horizontal plane* can be modeled by

- A nearly constant velocity model for the uniform motion, implemented as a WNA (white noise acceleration, or second-order kinematic, discussed in Section 6.5.3) model with low-level process noise
- A maneuvering model, which can be implemented as
  - a WNA model with significant process noise, commensurate with the expected maneuvers, or
  - a *nearly “coordinated turn” model*

The nearly constant velocity model is given by

$$x(k+1) = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} \frac{1}{2}T^2 & 0 \\ T & 0 \\ 0 & \frac{1}{2}T^2 \\ 0 & T \end{bmatrix} v(k) \quad (11.7.1-1)$$

where  $T$  is the sampling interval;  $x$  is the state of the aircraft, defined as

$$x = [\xi \dot{\xi} \eta \dot{\eta}]' \quad (11.7.1-2)$$

with  $\xi$  and  $\eta$  denoting the Cartesian coordinates of the horizontal plane; and  $v$  is a zero-mean Gaussian white noise used to model (“cover”) small accelerations, the turbulence, wind change, and so on, with an appropriate covariance  $Q$ , which is a design parameter.

The turn of a civilian aircraft usually follows a pattern known as *coordinated turn (CT)* (Subsection 4.2.2) — characterized by constant turn rate and constant speed. Although the actual turns are not exactly “coordinated” since the ground speed is the airspeed plus the wind speed, it can be suitably described by the “coordinated turn” model plus a fairly small noise representing the modeling error, resulting in the nearly coordinated turn model. The *CT model* is necessarily a nonlinear one if the turn rate is not a known constant. Augmenting the state vector (11.7.1-2) by one more component — the turn rate  $\Omega$ , that is,

$$x = [\xi \dot{\xi} \eta \dot{\eta} \Omega]' \quad (11.7.1-3)$$

the *nearly coordinated turn model* is then given by

$$x(k+1) = \begin{bmatrix} 1 & \frac{\sin \Omega(k)T}{\Omega(k)} & 0 & -\frac{1-\cos \Omega(k)T}{\Omega(k)} & 0 \\ 0 & \cos \Omega(k)T & 0 & -\sin \Omega(k)T & 0 \\ 0 & \frac{1-\cos \Omega(k)T}{\Omega(k)} & 1 & \frac{\sin \Omega(k)T}{\Omega(k)} & 0 \\ 0 & \sin \Omega(k)T & 0 & \cos \Omega(k)T & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} x(k)$$

$$+ \begin{bmatrix} \frac{1}{2}T^2 & 0 & 0 \\ T & 0 & 0 \\ 0 & \frac{1}{2}T^2 & 0 \\ 0 & T & 0 \\ 0 & 0 & T \end{bmatrix} v(k) \quad (11.7.1-4)$$

Note that the process noise  $v$  in (11.7.1-1) has different dimension from the one in (11.7.1-4).

Assuming only position measurements are available, this yields the following observation equation

$$z(k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} x(k) + w(k) \quad (11.7.1-5)$$

where  $w$  is the measurement noise. Although the original radar measurements are in polar coordinates, they can be converted to Cartesian coordinates, as discussed in Subsection 10.4.3.

### 11.7.2 The EKF for the Coordinated Turn Model

Since the model in (11.7.1-4) is nonlinear, the estimation of the state (11.7.1-3) will be done via the EKF.

The equations for state prediction and the corresponding covariance, the only ones affected by the nonlinearity of (11.7.1-4), are given below. The dynamic equation (11.7.1-4) can be rewritten compactly as

$$x(k+1) = f[k, x(k)] + \Gamma_{CT}(k)v(k) \quad (11.7.2-1)$$

To obtain the predicted state  $\hat{x}(k+1|k)$ , the nonlinear function in (11.7.2-1) is expanded in Taylor series around the latest estimate  $\hat{x}(k|k)$  with terms up to first order to yield the (first-order) EKF<sup>2</sup>. The vector Taylor series expansion of (11.7.2-1) up to first order is

$$\begin{aligned} x(k+1) = & f[k, \hat{x}(k|k)] + f_x(k)[x(k) - \hat{x}(k|k)] \\ & + \text{HOT} + \Gamma_{CT}(k)v(k) \end{aligned} \quad (11.7.2-2)$$

where HOT represents the higher-order terms and

$$f_x(k) = [\nabla_x f(k, x)']' |_{x=\hat{x}(k|k)}$$

---

<sup>2</sup>The second-order EKF was found not to provide any benefit in this problem.

$$= \begin{bmatrix} 1 & \frac{\sin \hat{\Omega}(k)T}{\hat{\Omega}(k)} & 0 & -\frac{1-\cos \hat{\Omega}(k)T}{\hat{\Omega}(k)} & f_{\Omega,1}(k) \\ 0 & \cos \hat{\Omega}(k)T & 0 & -\sin \hat{\Omega}(k)T & f_{\Omega,2}(k) \\ 0 & \frac{1-\cos \hat{\Omega}(k)T}{\hat{\Omega}(k)} & 1 & \frac{\sin \hat{\Omega}(k)T}{\hat{\Omega}(k)} & f_{\Omega,3}(k) \\ 0 & \sin \hat{\Omega}(k)T & 0 & \cos \hat{\Omega}(k)T & f_{\Omega,4}(k) \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (11.7.2-3)$$

is the Jacobian of the vector  $f$  evaluated at the latest estimate of the state. The partial derivatives with respect to  $\Omega$  are given by

$$= \begin{bmatrix} f_{\Omega,1}(k) \\ f_{\Omega,2}(k) \\ f_{\Omega,3}(k) \\ f_{\Omega,4}(k) \\ \frac{(\cos \hat{\Omega}(k)T) T \hat{\xi}(k)}{\hat{\Omega}(k)} - \frac{(\sin \hat{\Omega}(k)T) \hat{\xi}(k)}{\hat{\Omega}(k)^2} - \frac{(\sin \hat{\Omega}(k)T) T \hat{\eta}(k)}{\hat{\Omega}(k)} - \frac{(-1+\cos \hat{\Omega}(k)T) \hat{\eta}(k)}{\hat{\Omega}(k)^2} \\ -(\sin \hat{\Omega}(k)T) T \hat{\xi}(k) - (\cos \hat{\Omega}(k)T) T \hat{\eta}(k) \\ \frac{(\sin \hat{\Omega}(k)T) T \hat{\xi}(k)}{\hat{\Omega}(k)} - \frac{(1-\cos \hat{\Omega}(k)T) \hat{\xi}(k)}{\hat{\Omega}(k)^2} + \frac{(\cos \hat{\Omega}(k)T) T \hat{\eta}(k)}{\hat{\Omega}(k)} - \frac{(\sin \hat{\Omega}(k)T) \hat{\eta}(k)}{\hat{\Omega}(k)^2} \\ (\cos \hat{\Omega}(k)T) T \hat{\xi}(k) - (\sin \hat{\Omega}(k)T) T \hat{\eta}(k) \end{bmatrix} \quad (11.7.2-4)$$

Based on the above expansion, the state prediction and state prediction covariance in the EKF are

$$\hat{x}(k+1|k) = f[k, \hat{x}(k|k)] \quad (11.7.2-5)$$

$$P(k+1|k) = f_x(k)P(k|k)f_x(k)' + \Gamma_{CT}(k)Q(k)\Gamma_{CT}(k)' \quad (11.7.2-6)$$

where  $Q$  is the covariance of the process noise in (11.7.2-1).

The initial estimate of the turn rate is zero since one does know whether it will be a left turn ( $\Omega > 0$ ) or<sup>3</sup> a right turn ( $\Omega < 0$ ). Since, from this initial estimate, the algorithm is capable of estimating the turn rate including its sign, there is no need for left- and right-turn models, as in some other IMM versions. Note that at the initialization step and when the estimated turn rate is zero (or when it is sufficiently close to zero), the limiting form of  $f_x(k)$ , given in

<sup>3</sup>This sign convention follows the trigonometric convention (the navigation convention has the opposite sign for the turn rates.)

(11.7.2-7), should be used, namely,

$$f_x(k)|_{\hat{\Omega}(k)=0} = \begin{bmatrix} 1 & T & 0 & 0 & -\frac{1}{2}T^2\hat{\eta}(k) \\ 0 & 1 & 0 & 0 & -T\hat{\eta}(k) \\ 0 & 0 & 1 & T & \frac{1}{2}T^2\hat{\xi}(k) \\ 0 & 0 & 0 & 1 & T\hat{\xi}(k) \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (11.7.2-7)$$

### Mixing of States of Different Dimension in the IMM

Since the dimension of the state (11.7.1-3) is 5, while that of (11.7.1-2) is 4, the mixing in the IMM estimator is done by augmenting the latter with a component that is zero. This amounts to having a zero turn rate in the constant velocity motion, which can be also seen to reduce (11.7.1-4) to (11.7.1-1). Similarly, the covariance associated with (11.7.1-2) is to be augmented by a column and a row of zeros.

### 11.7.3 Selection of Models and Parameters

To obtain the best possible results, the IMM algorithm has to be properly designed to meet the following requirements of the ATC tracking simultaneously:

- Reduce as much as possible the estimation errors during the uniform motion.
- Maintain the peak estimation error during the maneuver lower than that of the unfiltered raw measurements.
- Provide correct and timely indication of the flight mode, especially rapid detection of the maneuver.

These requirements are fulfilled by means of

- Design of aircraft motion models for all modes of flight
- Selection of the model parameters, such as the noise levels
- Determination of the parameters of the underlying Markov chain, that is, the transition probabilities

*Model selection* should consider both the quality and complexity of the model. Typically, the models used in the IMM configuration for ATC tracking will include one (a nearly constant velocity model) for the uniform motion and one or more for the maneuver — the nearly coordinated turn model of (11.7.1-4) is a typical one.

*Selection of noise levels* for each model is an important part of the estimator design. Although the uniform motion is better modeled in (11.7.1-1) with a small process noise  $v$  to model the air turbulence, winds aloft changes, and

so forth, it may be legitimate to use a somewhat larger process noise for the uniform motion to cover slow turns as well as small linear accelerations so as to ease the burden of modeling a broad range of maneuvers. The right choice of the noise level of the nearly coordinated turn model depends on what turn rate range is expected and how many models are to be used for the maneuvers.

The performance of the IMM algorithm is not very sensitive to the *choice of the transition probabilities*. However, this choice provides to a certain degree the trade-off between the peak estimation errors at the onset of the maneuver and the maximum reduction of the estimation errors during the uniform motion. The guideline for a proper choice is to match roughly the transition probabilities with the system's actual ***mean sojourn time*** in each mode. See problem 11-6 for the expression of the mean sojourn time in units of the sampling interval.

#### 11.7.4 The ATC Scenario

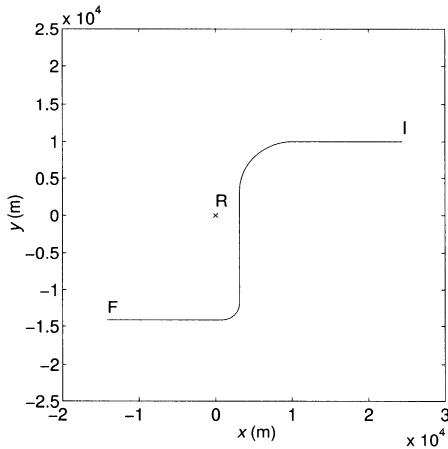
As a generic ATC tracking problem, the following scenario is considered. The radar, stationed at [0 m, 0 m], provides direct position only measurements (after the polar-to-Cartesian conversion) with RMS errors of 100 m in each of the two Cartesian coordinates.<sup>4</sup> The intervals between the samples are  $T = 5$  s. In the scenario under consideration, starting from [25,000 m, 10,000 m] at time  $t = 0$  s, the aircraft flies westward for 125 s at 120 m/s, before executing a  $1^\circ/\text{s}$  coordinated turn (which amounts to an acceleration of  $0.2g$  at this speed), for 90 s. Then it flies southward for another 125 s, followed by a  $3^\circ/\text{s}$  turn (an acceleration of  $0.6g$  at this speed) for 30 s. After the turn, it continues to fly westward at constant velocity. The target trajectory is shown in Fig. 11.7.4-1. This scenario leads to a maneuvering index (see Chapter 6) that is quite high (almost 1.5) and thus very little noise reduction can be achieved by a single model based state estimator, which will have to be, by necessity, conservative, i.e., designed for the maximum acceleration.

The following estimator configurations were considered:

- KF: A Kalman filter using a second order linear kinematic model (WNA) with process noise of standard deviation  $1 \text{ m/s}^2$  modeling the maneuver. This is a compromise between the maximum acceleration ( $6 \text{ m/s}^2$ ) and the constant velocity trajectory portions.
- IMM-L: An IMM estimator with two second-order linear kinematic models (WNA) with two noise levels. The one with the lower noise level with standard deviation  $0.1 \text{ m/s}^2$  is used to model the uniform motion and the other one with standard deviation  $2 \text{ m/s}^2$  for the maneuvers. The mode

---

<sup>4</sup>Its location under these assumptions is not relevant. In practice, the measurements are in polar coordinates and it is best to convert them into Cartesian (see Section 10.4.3). In the polar measurement case, the location of the sensor matters because the angular measurement errors are multiplied by the range from the sensor to the target.



**Figure 11.7.4-1:** The target trajectory (I — initial point, F — final point, R — radar location).

transition probability matrix  $\pi_L$  was

$$\pi_L = \begin{bmatrix} 0.95 & 0.05 \\ 0.10 & 0.90 \end{bmatrix} \quad (11.7.4-1)$$

- **IMM-CT:** An IMM estimator with one second-order kinematic model (a nearly constant velocity model with process noise standard deviation  $0.1 \text{ m/s}^2$ ) for the uniform motion and a (nearly) coordinated turn model with the turn rate (as part of the state) being estimated using an EKF. The process noise standard deviations used in the coordinated turn model were  $0.5 \text{ m/s}^2$  and  $0.2^\circ/\text{s}^2$  for the linear and turn portions of the state, respectively.

The mode transition probability matrix  $\pi_{CT}$  was

$$\pi_{CT} = \begin{bmatrix} 0.95 & 0.05 \\ 0.10 & 0.90 \end{bmatrix} \quad (11.7.4-2)$$

The initial estimates were based on one-point initializations; that is, the initial position estimate was at the first measured position and the initial velocity/turn rate was zero. The initial velocity/turn rate variances were based on their assumed maximum values. The mode set, the transition mode probability matrices, process noise standard deviations and initial estimates/variances are estimator design parameters, which should be selected based on the expected scenarios and the designer's experience.

## 11.7.5 Results and Discussion

The performances of the various estimators are shown in Table 11.7.5-1, where all the position estimation errors are for the  $\xi$  and  $\eta$  coordinates *combined*. The

*maneuver detection delay* is defined to be the latency, measured in sampling periods, from the maneuver onset time to the time that the probability of the uniform motion mode falls below 0.5. The “UM probability error” stands for the (steady-state) probability of the maneuver modes during the uniform motion. In other words, this is the estimator-calculated probability that the UM mode is not in effect while, in truth, it is.

**Table 11.7.5-1:** Estimation errors in the estimators (raw position measurement error 141m).

	Estimator		
	KF	IMM-L	IMM-CT
Peak position error (m)	143	138	109
UM position error (m)	123	83	71
Peak speed error (m/s)	22.5	9.3	4.8
UM speed error (m/s)	3.2	1.9	1.3
UM course error ( $^{\circ}$ )	3.8	2.7	1.8
$1^{\circ}/\text{s}$ Maneuver detection delay (scans)	—	4	1
$3^{\circ}/\text{s}$ Maneuver detection delay (scans)	—	3	1
UM probability error (%)	—	5.0	3.5

As presented in Table 11.7.5-1, the IMM-CT estimator yields a peak RMS position error during the mode of flight change that is almost 25% below the raw measurement error and 50% RMS error<sup>5</sup> reduction during the uniform motion. The speed estimation error during the uniform motion is 1% of the aircraft speed. The detection of the maneuver is quick: in two scans (i.e., one scan delay) for both turns in the target trajectory.

Since the coordinate-combined raw measurement error ( $1\sigma$ ) is  $100\sqrt{2} = 141$  m, the angle deviation from the straight line in one scan (sampling period) during the uniform motion due to  $1\sigma$  measurement error can be as large as

$$\tan^{-1} \frac{141 \text{ m}}{120 \text{ m/s} \cdot 5 \text{ s}} = 13.2^{\circ} \quad (11.7.5-1)$$

This is about the same as the expected course change caused by a  $3^{\circ}/\text{s}$  turn in one scan, which is  $(3^{\circ}/\text{s}) \cdot (5 \text{ s}) = 15^{\circ}$ . In view of this, the UM probability error of only 3.5% seems quite low. The rapid detection of the maneuver, together with this small probability error, verifies the good reliability of this design in terms of providing the correct and timely information of the flight mode.

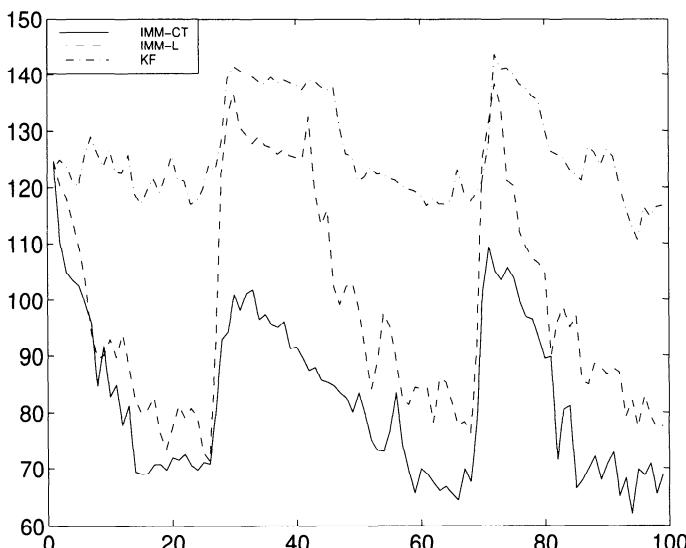
If the sampling period is shorter, one can achieve more reduction of the errors in the state estimate compared to the unfiltered radar measurements during the maneuver, and substantially more noise reduction can be achieved during the uniform motion.

<sup>5</sup>Square root of the sample average (mean) of the squared error from the Monte Carlo simulations.

Figure 11.7.5-1 presents the RMS position errors ( $\xi$  and  $\eta$  coordinates *combined*) for the IMM-L and the IMM-CT estimators as well as a baseline Kalman filter (tuned specially for this case). The initial large estimation errors are due to the fact that the initial probability of each model was set to be equal to account for the worst case of ignorance.

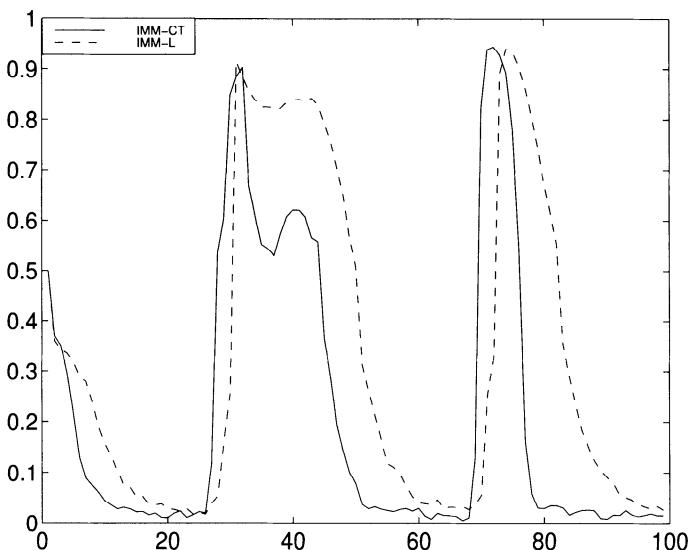
The following observations about the relative performances of the above estimators can be made:

IMM-CT has the peak RMS position error of about 110 m, superior to the single measurement position RMS error, which is  $100\sqrt{2} = 141$  m. Moreover, when the aircraft is not maneuvering, the RMS error is around 70 m — that is, a reduction of about 50% in the RMS error (70% in MSE). The Kalman filter has a peak RMS position error of 143 m. During the uniform motion, however, its RMS error stays at about 120 m. This estimation error of the specially tuned Kalman filter for the uniform motion can be reduced only slightly at a cost of a much higher peak error. This is the typical behavior of a single-model based Kalman filter. Between the IMM-CT estimator and the IMM-L, the former yields faster maneuver detection and lower peak estimation errors.



**Figure 11.7.5-1:** RMS position estimation errors.

Figure 11.7.5-2 shows the evolution of the average mode 2 (turning mode) probabilities in the IMM-L and IMM-CT estimators. The figure clearly indicates the rapid “detection” capability of IMM-CT — in two samples for both  $1^\circ/\text{s}$  and  $3^\circ/\text{s}$  turns. The less sophisticated IMM-L detects the turns in three to four samples.



**Figure 11.7.5-2:** Mode 2 probabilities in IMM-L and IMM-CT estimators.

### Sensitivity of the IMM to the Design Parameters

When the IMM uses several linear models with different process noise variances, a large ratio of these variances between models is needed; otherwise the models cannot be “distinguished” by the mode probability evaluator of the IMM (if they are the same, the IMM becomes a disguised KF).

Values of the maneuver onset probability between 0.05 and 0.1 yield similar results. Overall, the sensitivity of the performance to such changes in the transition probabilities are moderate.

Variable sampling interval, which is common in real systems, can be accounted for easily — see [Bar-Shalom95].

### Some Remarks

The main findings from this example are as follows:

- The KF is clearly inferior to any (reasonably designed) IMM estimator.
- The accuracy of the turn rate estimate is not very important as far as the quality of the position, speed, and course estimates are concerned. What is important is the correct and timely detection of the maneuver and the fast response of the filter to this detection.
- The IMM-CT estimator is the best choice for tracking maneuvering with its capability to track the linear as well as turn motion of the target.

When the aircraft is in uniform motion, the estimates given by the filter(s) based on the nearly coordinated turn model(s) have little effect on the overall (combined) estimate of the IMM algorithm (and on the interaction estimates for the filter based on the linear model at the next cycle) because their estimates have small weights. Consequently, while the aircraft is indeed in uniform motion, the filter matched to the uniform motion model is dominant and yields accurate estimates.

Once the aircraft starts to maneuver, the filter matched to the coordinated turn model takes over rapidly, leading to a significant reduction in the peak estimation errors. The IMM-CT estimator performs better than the KF and the IMM-L estimators during maneuvers.

A noise level switching version of the KF (with one model at a time: running a filter with a low variance process noise during the constant velocity portions of the trajectory and, following a maneuver detection, switching to a filter with a high level process noise) is not practical due to the very short duration of the maneuver. Also, in real systems where the origin of the measurements is uncertain, the “switching” technique is totally inadequate [Bar-Shalom95].

In view of the nature of maneuvers, which are not zero-mean white random processes, as the standard assumptions of the KF require, one cannot expect any of the filters to be perfectly consistent. The judgment as to which estimator design is best is a somewhat subjective one, and it should take into account the following:

- The performance (RMS errors) in both position and velocity (or speed and course): the maximum error and the average during the uniform motion
- Our desire for filter consistency
- The complexity of the implementation

Based on the above consideration, the IMM-CT is the best choice.

## 11.8 WHEN IS AN IMM ESTIMATOR NEEDED?

In Section 11.7.4 the design of an IMM estimator for a generic ATC problem was presented and it was shown that the IMM estimator performed significantly better than a Kalman filter. Specifically, the use of an IMM estimator with two filter modules, namely, a white-noise acceleration model and a coordinated turn model, resulted in about 50% reduction in RMS position errors vs. the Kalman filter. This improvement is due to the adaptive bandwidth (or the dynamic range) capability of the IMM estimator, that is, its ability to handle uniform motion (in which case it acts as a narrowband lowpass filter), as well as high-frequency maneuver motion (in which case it acts as wideband lowpass filter).

The above results raise an important question: When is an adaptive estimator — in particular, the IMM estimator — necessary? Intuitively, the higher the uncertainty — that is, the higher the maneuverability of the target — the more

a versatile tracker like the IMM estimator is needed. In the sequel we present an answer to the above question using simulation-based estimation results.

It is shown below that the choice between a (single model) Kalman filter and a multiple model tracker like the IMM estimator depends on the maneuvering index  $\lambda$  of the target, defined in (6.5.3-14). That is, it is shown that above a certain maneuvering index threshold, the performance of the IMM estimator, in terms of estimation errors, is significantly better than that of a Kalman filter, and therefore the former's use is recommended above that particular maneuvering index.

### 11.8.1 Kalman Filter vs. IMM Estimator

The test scenario, which is similar to the one used in Section 11.7.4, is constructed as follows:

1. The sensor measures the two-dimensional Cartesian positions with RMS errors of  $\sigma_w = 100$  m in each coordinate.
2. The revisit interval of the sensor is fixed at  $T = 5$  s.
3. Initially, the target moves at nearly a constant velocity for 60 s with process noise standard deviation  $\sigma_v = 0.2$  m/s<sup>2</sup>, which yields a very small maneuvering index of  $\lambda = 0.05$  in each coordinate.
4. Following this, the target executes a series of three maneuvers, each of which
  - lasts for 60 s
  - has a given (fixed) maneuvering index  $\lambda$  (with the corresponding white process noise)
  - is interleaved with nonmaneuvering periods lasting for 60s with  $\lambda = 0.05$ .

The level of maneuvering is varied from  $\lambda = 0.1$  to 2.5, in increments of 0.1. That is, the process noise standard deviation ranges from  $\sigma_v = 0.4$  m/s<sup>2</sup> to 10 m/s<sup>2</sup> for the above values of  $\sigma_w$  and  $T$ . The performance of the trackers based on Kalman filter and the IMM estimator are evaluated from 100 Monte Carlo runs for each of these maneuvering index values.

For the Kalman filter, a second order (WNA) linear kinematic model with  $\sigma_v$  corresponding to 0.8 of the maximum process noise standard deviation (during the maneuvering intervals) was used. This choice was made in order to retain as much as possible the ability to track maneuvering and nonmaneuvering intervals. Other values of  $\sigma_v$  yielded similar results.

The IMM estimator consisted of two WNA models designed as follows:

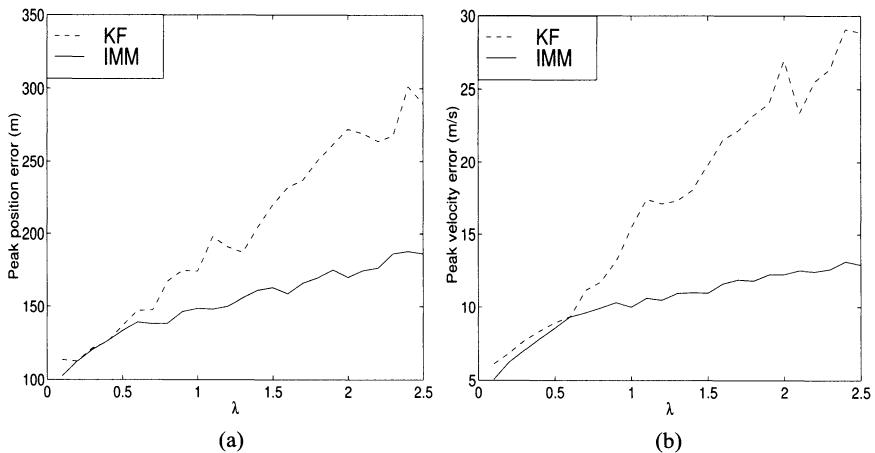


Figure 11.8.1-1: Peak RMS estimation errors. (a) Position errors, (b) velocity errors.

1. One model with low process noise with  $\sigma_v = 0.2 \text{ m/s}^2$ , the process noise standard deviation during the nonmaneuvering intervals.
2. One model with high process noise corresponding to the maneuver process noise standard deviation.
3. The mode transition probability matrix  $\pi$  was

$$\pi = \begin{bmatrix} 0.90 & 0.10 \\ 0.10 & 0.90 \end{bmatrix} \quad (11.8.1-1)$$

which corresponds to a sojourn time of about 60 s in each mode at  $T = 5 \text{ s}$ .

Figure 11.8.1-1 shows the peak position and velocity errors obtained with the two estimators. It can be seen that both estimators result in nearly the same peak estimation errors up to  $\lambda = 0.5$  while beyond that value of  $\lambda$  the IMM estimator yields significantly lower estimation errors.

Figure 11.8.1-2 shows the overall RMS position and velocity errors. Here also, one can notice that the IMM estimator results in significant error reduction over the Kalman filter for  $\lambda > 0.5$ .

Figure 11.8.1-3 shows the position and velocity RMS errors during the non-maneuvering intervals. It can be seen that the errors obtained with the IMM estimator remain almost constant, whereas those obtained with the Kalman filter increase with  $\lambda$ ; the latter is unable to provide noise reduction during non-maneuvering intervals because, in order to reduce the overall error, its parameters are tuned to the maneuvering intervals.

This is a direct result of KF's lack of adaptive bandwidth capability. In contrast, the IMM estimator is able to maintain the noise reduction during non-maneuvering intervals even with higher levels of target maneuverability — by

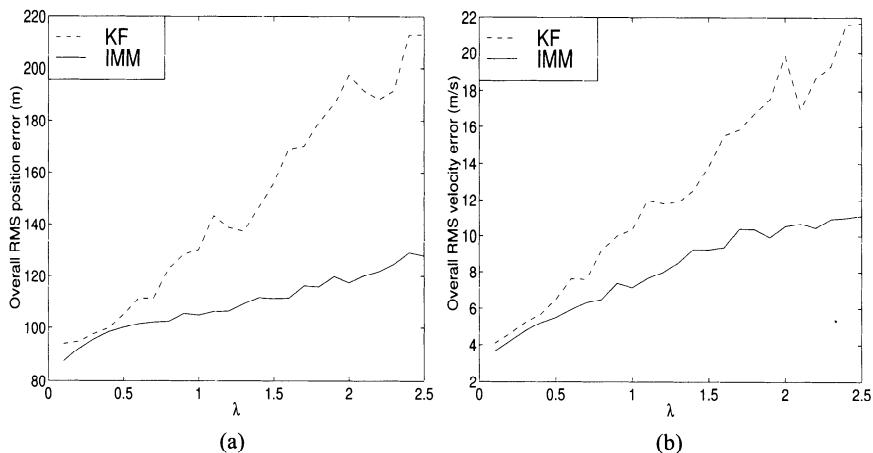


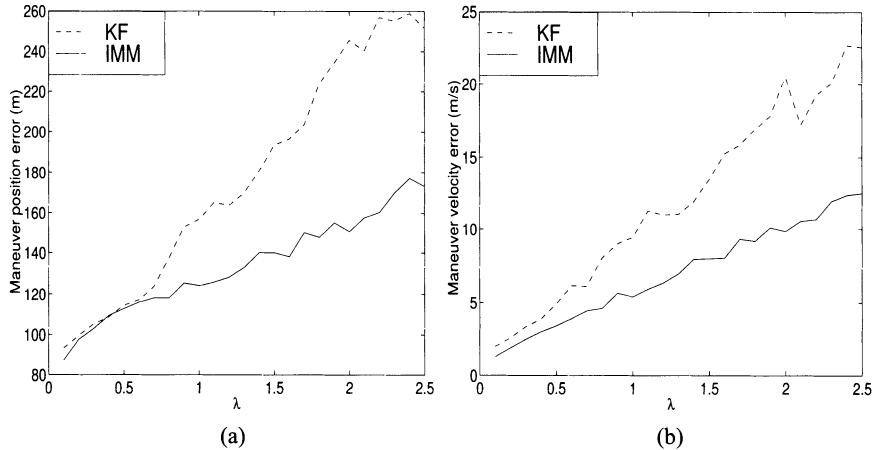
Figure 11.8.1-2: Overall RMS estimation errors. (a) Position errors, (b) velocity errors.

adapting the weights given to the two filter modules (mode probabilities), it gives consistently better results.

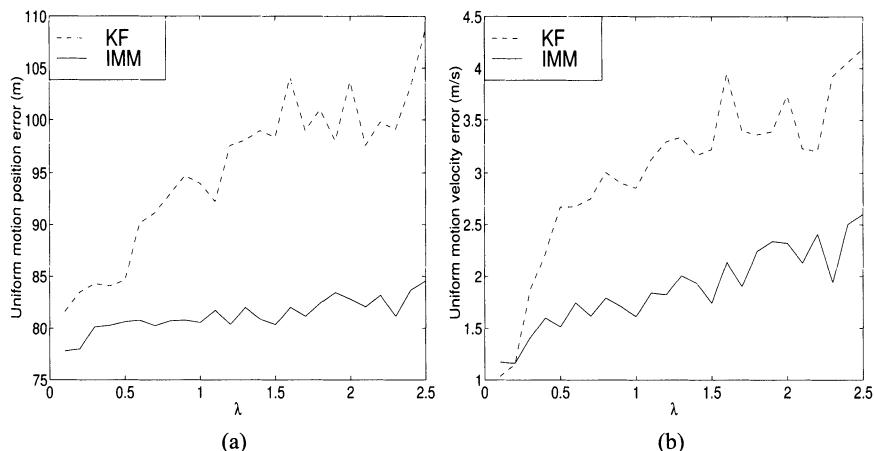
The above observation is confirmed further in Fig. 11.8.1-4, where the position and velocity RMS errors during the maneuvering intervals are shown. Notice that even with the Kalman filter's noise levels adapted to the increasing maneuverability, it is unable to match the IMM estimator's performance levels. Also notice that the breaking point for the Kalman filter is, once again, around  $\lambda = 0.5$ , up to which both estimators perform nearly the same.

These results indicate that below about  $\lambda = 0.5$ , the Kalman filter and the IMM estimator performs equally well. Other values of measurements error standard deviations and sampling intervals yielded similar results. Above this (approximate) threshold, the more sophisticated IMM estimator yields consistently better results. The **adaptive bandwidth** or **dynamic range capability** of the IMM estimator becomes a significant factor in the estimation process when the maneuverability of the target is sufficiently high. The adaptation via mode probability update helps the IMM estimator keep the estimation errors consistently low, both during maneuvers as well as during the benign motion intervals.

In summary, based on the above observations it is recommended that the **maneuvering index threshold** of  $\lambda = 0.5$  be used as a guideline in making a choice between the Kalman filter and the IMM estimator.



**Figure 11.8.1-3:** Uniform motion RMS estimation errors. (a) Position errors, (b) velocity errors.



**Figure 11.8.1-4:** Maneuver RMS estimation errors. (a) Position errors, (b) velocity errors.

## 11.9 USE OF EKF FOR SIMULTANEOUS STATE AND PARAMETER ESTIMATION

### 11.9.1 Augmentation of the State

The extended Kalman filter can be used for suboptimal state estimation in nonlinear dynamic systems.

The situation of linear systems with unknown parameters that are continuous-valued can be put in the framework of nonlinear state estimation by augmenting the base state. The **base state** is the state of the system with the parameters assumed known.

Denoting the unknown parameters as a vector  $\theta$ , the **augmented state** will be the **stacked vector** consisting of the base state  $x$  and  $\theta$

$$y(k) \triangleq \begin{bmatrix} x(k) \\ \theta \end{bmatrix} \quad (11.9.1-1)$$

The linear dynamic equation of  $x$ , with the known input  $u$  and process noise  $v$ ,

$$x(k+1) = F(\theta)x(k) + G(\theta)u(k) + v(k) \quad (11.9.1-2)$$

and the “dynamic equation” of the parameter vector (assumed time invariant)

$$\theta(k+1) = \theta(k) \quad (11.9.1-3)$$

can be rewritten as a nonlinear dynamic equation for the augmented state

$$y(k+1) = f[y(k), u(k)] + v(k) \quad (11.9.1-4)$$

The resulting nonlinear equation (11.9.1-4) is then used for an EKF to estimate the entire augmented state.

The same technique can be used if the dynamic equation of the base state is nonlinear.

The model represented by (11.9.1-3) for the parameter dynamics assumes it to be constant. Therefore, the covariance yielded by the EKF will, asymptotically, tend to zero, and consequently the filter gain for these components will tend to zero. This is because there is no process noise entering into these components of the augmented state — the **controllability condition 2** from Subsection 5.2.5, pertaining to the Riccati equation for the state estimation covariance, is not satisfied.

Since the EKF is not an optimal estimation algorithm, it will in general not yield consistent estimates for the parameters — the estimates will not converge to the true values. Thus the situation where the parameter variances tend to zero is undesirable because it will lead in practice to estimation errors much larger than the filter-calculated variances.

This can be remedied (to some extent) by assuming an ***artificial process noise*** (or ***pseudo-noise***) entering into the parameter equation. This amounts to replacing (11.9.1-3) by

$$\theta(k+1) = \theta(k) + v_\theta(k) \quad (11.9.1-5)$$

where the parameter process noise is assumed zero mean and white. Thus the parameter is modeled as a (discrete time) Wiener process.

Any nonzero variance of this process noise will prevent the filter-calculated variances of the parameter estimates from converging to zero. Furthermore, this also gives the filter the ability to estimate ***slowly varying parameters***.

The choice of the variance of the artificial process noise for the parameters — the ***tuning of the filter*** — can be done as follows:

1. Choose the standard deviation of the process noise as a few percent of the (estimated/guessed) value of the parameter.
2. Simulate the system and the estimator with random initial estimates (for the base state as well as the parameters) and monitor the normalized estimation errors.
3. Adjust the noise variances until, for the problem of interest, the filter is consistent — it yields estimation errors commensurate with the calculated augmented state covariance matrix. The criteria to be used are those from Section 5.4 — the estimation bias and the normalized estimation error squared (NEES).

The example of the coordinated turn motion with an unknown rate from (11.7.1-2) to (11.7.1-4) falls into this category.

### 11.9.2 An Example of Use of the EKF for Parameter Estimation

Consider the scalar system, that is, its base state  $x$  is a scalar, given by

$$x(k+1) = a(k)x(k) + b(k)u(k) + v_1(k) \quad (11.9.2-1)$$

where  $v_1(k)$  is the base state process noise and the two unknown parameters are  $a(k)$  and  $b(k)$ , possibly time-varying.

The observations are

$$z(k) = x(k) + w(k) \quad (11.9.2-2)$$

Following the procedure of the previous subsection, the augmented state is

$$y(k) \triangleq \begin{bmatrix} y_1(k) \\ y_2(k) \\ y_3(k) \end{bmatrix} \triangleq \begin{bmatrix} x(k) \\ a(k) \\ b(k) \end{bmatrix} \quad (11.9.2-3)$$

With this the nonlinear dynamic equation corresponding to (11.9.2-1) can be written as

$$y_1(k+1) = f^1[y(k), u(k)] + v_1(k) \triangleq y_1(k)y_2(k) + y_3(k)u(k) + v_1(k) \quad (11.9.2-4)$$

and the “dynamic equation” of the parameters is

$$y_i(k+1) = f^i[y(k), u(k)] + v_i(k) \triangleq y_i(k) + v_i(k) \quad i = 2, 3 \quad (11.9.2-5)$$

The augmented state equation is then

$$y(k+1) = f[y(k), u(k)] + v(k) \quad (11.9.2-6)$$

with the augmented process noise

$$v(k) \triangleq \begin{bmatrix} v_1(k) \\ v_2(k) \\ v_3(k) \end{bmatrix} \quad (11.9.2-7)$$

assumed zero mean and with covariance

$$Q = \text{diag}(q_1, q_2, q_3) \quad (11.9.2-8)$$

With the linear measurements (11.9.2-2), the only place in the EKF where linearizations are to be carried out is the base state prediction.

The second-order EKF will use the following augmented state prediction equations. From (11.9.2-4) the base state prediction can be obtained directly as

$$\hat{y}_1(k+1|k) = \hat{y}_2(k|k)\hat{y}_1(k|k) + \hat{y}_3(k|k)u(k) + P_{21}(k|k) \quad (11.9.2-9)$$

since

$$E[a(k)x(k)|Z^k] = E[y_2(k)y_1(k)|Z^k] = \hat{y}_2(k|k)\hat{y}_1(k|k) + \text{cov}[y_2(k), y_1(k)|Z^k] \quad (11.9.2-10)$$

Note that the last term in (11.9.2-9) is the only second-order term.

The predicted values of the remaining two components of the augmented state, which are the system’s unknown parameters, follow from (11.9.2-5) as

$$\hat{y}_i(k+1|k) = \hat{y}_i(k|k) \quad i = 2, 3 \quad (11.9.2-11)$$

Equations (11.9.2-9) and (11.9.2-11), in general, have to be obtained using the series expansion of the EKF, which requires evaluation of the Jacobian of the vector  $f$  and the Hessians of its components. In the present problem, the Jacobian is

$$F(k) = \begin{bmatrix} \hat{y}_2(k|k) & \hat{y}_1(k|k) & u(k) \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (11.9.2-12)$$

and the Hessians are

$$f_{yy}^1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad f_{yy}^2 = 0 \quad f_{yy}^3 = 0 \quad (11.9.2-13)$$

Then, with the components of  $f$  given in (11.9.2-4) and (11.9.2-5), it can be easily shown (see problem 11-3) that using (10.3.2-4) for the augmented state, that is,

$$\hat{y}(k+1|k) = f[\hat{y}(k|k), u(k)] + \frac{1}{2} \sum_{i=1}^{n_x} e_i \text{tr}[f_{yy}^i P(k|k)] \quad (11.9.2-14)$$

yields (11.9.2-9) and (11.9.2-11).

The prediction covariance of the base state can be obtained, using (10.3.2-6) as

$$\begin{aligned} P_{11}(k+1|k) &= \hat{y}_2(k|k)^2 P_{11}(k|k) + 2\hat{y}_2(k|k)\hat{y}_1(k|k)P_{21}(k|k) \\ &\quad + 2\hat{y}_2(k|k)u(k)P_{13}(k|k) + \hat{y}_1(k|k)^2 P_{22}(k|k) + 2\hat{y}_1(k|k)u(k)P_{23}(k|k) \\ &\quad + u(k)^2 P_{33}(k|k) + P_{21}(k|k)^2 + P_{22}(k|k)P_{11}(k|k) + q_1 \end{aligned} \quad (11.9.2-15)$$

### 11.9.3 EKF for Parameter Estimation — Summary

The EKF can be used to estimate simultaneously

- the base state and
- the unknown parameters

of a system.

This is accomplished by stacking them into an augmented state and carrying out the series expansions of the EKF for this augmented state.

Since the EKF is a suboptimal technique, significant care has to be exercised to avoid filter inconsistencies. The filter has to be tuned with artificial process noise so that its estimation errors are commensurate with the calculated variances.

## 11.10 NOTES, PROBLEMS, AND TERM PROJECT

### 11.10.1 Bibliographical Notes

The “adjustment” of the process noise covariance, presented in Section 11.2, has become part of the Kalman filtering folklore. It can be found in [Jazwinski69, Jazwinski70]. Among its applications, the one presented in [Chang77] deals with state and parameter estimation for maneuvering reentry vehicles. In [Tenney77b, Tenney77a] similar ideas are used for passive tracking of maneuvering targets including reinitialization of the filter upon detection of the maneuver. In [Castella80] a continuous update of the process noise covariance is presented based on a fading memory average of the residuals. An adaptation of the filter gain based on the residuals’ deviation from orthogonality is presented in [Hampton73]. In [Spingarn72] a least squares approach with exponential discount of older measurements is used for maneuvering targets. Estimation of the noise covariances has been discussed in, for example, [Li94] and [Myers76].

The switching from one model to another with different noise parameters is discussed in [McAulay73] in a manner somewhat similar to Subsection 11.2.2.

The input estimation method of Section 11.3 is based on [Chan79]. Estimation of the input and the measurement noise covariance has been discussed in [Moghaddamjoo86]. The generalized likelihood ratio technique [Willsky76b, Korn82] deals simultaneously with the estimation of the input and its onset time. Such an approach was taken by [Bogler87], who augmented the IE technique of [Chan79] to include maneuver onset time estimation; however, the resulting algorithm was very costly and performed less well than the IMM [Bar-Shalom89]. In [Demirbas87] an approach consisting of hypothesis testing via dynamic programming is proposed.

The variable dimension approach to tracking maneuvering targets discussed in Section 11.4 is from [Bar-Shalom82], which also presented the comparison of algorithms discussed in Section 11.5. An extensive comparison of various maneuver detection approaches is described in [Woolfson85].

A weighted state estimate, along the lines of Section 11.6, is presented in [Thorp73]. The multiple model (MM) approach was originally presented in [Magill65]. The Markov switching of models is discussed in [Moose73, Moose75, Moose79, Moose80, Gholsom77]. The generalized pseudo-Bayesian (GPB) algorithms are proposed by [Ackerson70, Jaffer71b, Jaffer71a]. The GPB2 MM approach is based on [Chang78]. A survey of the MM techniques and their connection with failure detection is given in [Tugnait82]. Recent work in failure detection using the static MM estimator includes [Hanlon98]. The use of the IMM for failure detection, which is substantially more effective than the static MM estimator, was presented in [Zhang98]. Related work in the area of failure detection is [Willsky76a, Caglayan80, Kerr89a]; the connection between failure detection, multiobject tracking and general modeling uncertainties is discussed in [Pattipati83]. Detection-estimation algorithms that approximate the optimal algorithm of Subsection 11.6.3 are discussed in [Hadidi78, Tugnait79]. The Interacting MM algorithm is from [Blom84, Blom88]. The application of the IMM to air traffic control can be found in [Bar-Shalom98b] (Chapters 1 and 2). Section 11.7 is based on [Wang99]. Reference [Li93] discusses how one can predict the performance of an IMM algorithm without Monte Carlo simulations and illustrates this technique on examples, including a simplified ATC problem. The generalization of the IMM to second-order Markov systems was presented in [Blom85]. Smoothing with an IMM was discussed in [Helmick95]. A survey on the IMM and its numerous applications can be found in [Mazor98]. The book [Sworder99] presents a mathematical treatise on hybrid systems. A survey of MM methods can be found in [Li96].

In [Li95] a theoretical framework for evaluating the performance of hybrid estimation algorithms (with continuous and discrete uncertainties) is presented.

Adaptive sampling techniques for a maneuvering target are presented in [Blackman81]. This topic is discussed in the larger context of radar resource allocation in [Blair98, Kirubarajan98]. The use of target orientation measurements for tracking maneuvering targets is explored in [Kendrick81, Lefas84, Andrisani86, Sworder89].

## 11.10.2 Problems

**11-1 CRLB for a two-model parameter estimation problem.** Consider an estimation problem where the noise behaves according to one of two models. Using the binary random

variable  $\alpha$ , the observation can be written as

$$z = x + \alpha w_1 + (1 - \alpha)w_2$$

where  $x$  is the (unknown constant) parameter to be estimated,

$$P\{\alpha = 1\} = p_1 \quad P\{\alpha = 0\} = p_2 = 1 - p_1$$

with  $w_i \sim \mathcal{N}(0, \sigma_i^2)$  independent of each other and of  $\alpha$ . In other words, the measurement has with probability  $p_1$  accuracy  $\sigma_1$  and with probability  $p_2$  accuracy  $\sigma_2$ .

1. Write the likelihood function of  $x$ ,  $\Lambda(x) = p(z|x)$ .
2. Find  $\hat{x}^{\text{ML}}$ .
3. Evaluate the corresponding MSE.
4. Evaluate the CRLB for this estimation problem assuming  $p_1 = 0.5$ ,  $\sigma_1 = 1$ ,  $\sigma_2 = 100$ . (*Hint:* Use the fact that  $\sigma_1 \ll \sigma_2$  to approximate the integral that requires otherwise numerical evaluation.)
5. Compare the results of items 3 and 4 for the above numbers. Comment on the usefulness of the bound in this problem.

- 11-2 Two-model parameter estimation problem with a prior.** Given the random parameter  $x \sim \mathcal{N}(x_0, \sigma_0^2)$  to be estimated based on the same measurement as in problem 11-1.

1. Find the MMSE estimate of  $x$  given  $z$ .
2. Find the conditional variance of  $x$  given  $z$ .

Assume  $x$ ,  $\alpha$  and  $w_i$  are all independent.

- 11-3 EKF prediction equations for a system with unknown parameters.**

1. Derive the predicted augmented state from (11.9.2-14).
2. Prove (11.9.2-15) and derive the remaining terms of this covariance.

- 11-4 EKF simulation for state and parameter estimation.** Consider the system (11.9.2-1) with  $x(0) = 0$ ,  $a(k) = 0.5$ ,  $b(k) = 0$ , process noise zero mean with variance  $q_1 = 0.09$ . The observations are given by (11.9.2-2) with measurement noise zero mean with variance  $r = 0.09$ . The covariance matrix of the initial augmented state estimate is  $P(0|0) = \text{diag}(0.09, 0.04)$ .

1. Implement an EKF for this problem (to estimate  $x$ ,  $a$ , and  $b$ ) without an artificial process noise. Initialize with random initial estimates according to  $P(0|0)$ . List the normalized estimation error squared (NEES) for each component of the augmented state for 100 steps.
2. Perform 100 Monte Carlo runs for the above. List the averages of the normalized estimation error for bias check and the NEES for each component for 100 steps. Indicate the corresponding 95% confidence regions.
3. If the filter needs tuning, do it and present the results. Compare also the absolute RMS estimation errors with those from (2).

- 11-5 Fading memory average.** Prove (11.2.2-7).

**11-6 Sojourn time in a state for a Markov chain.** Prove

$$E[\tau_i] = \frac{1}{1 - p_{ii}}$$

where  $\tau_i$  is the expected sojourn time (in units of the sampling interval) of a Markov chain in state  $i$  (mode  $i$ ) and  $p_{ii}$  is the transition probability from state  $i$  to state  $i$ .

**11-7 Hybrid system with i.i.d. modal state sequence.**

1. Find the Markov chain transition matrix that, regardless of the initial condition, yields (in one step) the fixed probability vector  $[p_1 \ p_2]'$ .
2. Given that the mean sojourn time (MST) in state 1 is  $\tau_1$  periods, find, for the above case,  $\tau_2$ , the MST in state 2.

**11-8 Direct mode observation in a hybrid system.** Consider the prior pdf of  $x$  as

$$p(x) = \sum_{i=1}^n \mu_i \delta(x - x_i)$$

and the observation

$$z = x + w \quad w \sim \mathcal{N}(0, 1)$$

Find the closed-form expressions of the posterior weights  $\hat{\mu}_i$  conditioned on  $z$ .

**11-9 Fault detection.** Consider a direct discrete-time WNA motion model in one coordinate with  $T = 1$ . Let the measurement be

$$z = \begin{bmatrix} h_{11} & 0 \\ 0 & h_{22} \end{bmatrix} x + w$$

with all noises zero-mean white with unity variance.

The system in the normal mode has  $h_{11} = 1$  and  $h_{22} = 1$ . The failure modes are:

- FM1.  $h_{11} = 0$
- FM2.  $h_{22} = 0$
- FM3.  $h_{11} = 0$  and  $h_{22} = 0$ .

The system starts in the normal mode, then goes into FM1 at  $k = 10$ , recovers at  $k = 20$ , goes into FM2 at  $k = 30$ , recovers at  $k = 40$ , goes into FM3 at  $k = 50$ , and recovers at  $k = 60$ .

Let the initial condition be

- Scenario A.  $x = [0 \ 0]'$
- Scenario B.  $x = [0 \ 10]'$

both with identity matrix initial covariance.

1. Design an IMM estimator for this system. Specify the design parameters and implement it with DynaEst™.
2. Based on 100 runs with the same failure sequence (but independent noises and random initial state estimates), find the average number of samples it takes to detect each failure mode and recovery for the two scenarios. Define what you mean by failure mode detection. Explain the difference in performance between the two scenarios.

- 11-10 Maneuvering index threshold vs. debiasing threshold.** Show that the maneuvering index threshold  $\lambda = 0.5$  from Section 11.8 is (almost) equivalent to the debiasing threshold (10.4.3-17).

### 11.10.3 Term Project — IMM Estimator for Air Traffic Control

Design IMM estimators for the following air traffic control problem:

The Ground Truth is a target moving with a constant speed of 250 m/s with initial state in Cartesian coordinates (with position in m)

$$\dot{x} = [\xi \quad \dot{\xi} \quad \eta \quad \dot{\eta}]' = [0 \quad 0 \quad 0 \quad 250]'$$

The sampling period is  $T = 10$  s. At  $k = 10$  ( $t = 100$  s) it starts a left turn of  $2^\circ/\text{s}$  for 30 s, then continues straight until  $k = 20$ , at which time it turns right with  $1^\circ/\text{s}$  for 45 s, then left with  $1^\circ/\text{s}$  for 90 s, then right with  $1^\circ/\text{s}$  for 45 s, then continues straight until  $k = 50$ . (The turns are not known to the filter!)

Measurements are made starting from  $k = 0$  on the position of this target in polar coordinates (range  $r$  and azimuth  $\theta$ ) by a radar located at  $[\xi_0, \eta_0] = [-10^4, 0]$ , with

$$r = \sqrt{(\xi - \xi_0)^2 + (\eta - \eta_0)^2} \quad \theta = \tan^{-1} \left( \frac{\eta - \eta_0}{\xi - \xi_0} \right)$$

with additive white Gaussian noise with covariance  $R = \text{diag}[2500 \text{ m}^2, (1^\circ)^2]$ .

Design the following two IMM estimators:

(A) IMM-L consisting of two (linear) WNA models (Section 6.3.2), with process noise (assumed zero-mean white Gaussian) with variances  $\sigma_{v_m}^2$ ,  $m = 1, 2$ . These, together with the Markov chain transition matrix between the models, are the IMM estimator design parameters.

(B) IMM-CT consisting of one WNA model and a coordinated turn model with the turn rate as an additional state component. The CT model has an additional design parameter, the turn rate process noise variance.

The IMM estimator is initialized from the measurements at  $k = 0$  and  $k = 1$  and starts running from  $k = 2$ . Each model has initial probability 0.5 and the same initial estimate.

1. Calculate the true state according to the above specifications for  $k = 0, \dots, 50$  (the truth evolves without noise).
2. Generate the noisy measurements along the trajectory.
3. Implement the IMM estimators for this problem. Indicate the rationale for the choice of the estimator design parameters.
4. Calculate the following average results for  $N = 100$  runs for the two designs (try to have the estimated position RMS errors not to exceed the single measurement position RMS error during the maneuver — a few percent excess is OK — while having as much as possible

“noise reduction” in the straight portions of the trajectory):

NORXE	$\triangleq$	$\text{AVE}\{\tilde{x}_1(k k)/\sqrt{P_{11}(k k)}\}$
FPOS	$\triangleq$	$\sqrt{\text{AVE}\{P_{11}(k k) + P_{33}(k k)\}}$
FVEL	$\triangleq$	$\sqrt{\text{AVE}\{P_{22}(k k) + P_{44}(k k)\}}$
RMSPOS	$\triangleq$	RMS position error ( <i>Note:</i> both coordinates combined)
RMSVEL	$\triangleq$	RMS velocity error ( <i>Note:</i> both coordinates combined)
RMSSPD	$\triangleq$	RMS speed error ( <i>Note:</i> speed $\triangleq$ magnitude of velocity vector)
RMSCRS	$\triangleq$	RMS course error ( <i>Note:</i> course $\triangleq$ direction of velocity vector)
NEES	$\triangleq$	$\text{AVE}\{\bar{x}(k k)'P(k k)^{-1}\bar{x}(k k)\}$
MOD2PR	$\triangleq$	$\text{AVE}\{\text{model 2 probability}\}$

where

$$\text{RMS}(y) \triangleq \sqrt{\frac{1}{N} \sum_{j=1}^N (y^j)^2} \quad \text{AVE}(y) \triangleq \frac{1}{N} \sum_{j=1}^N y^j$$

and  $y^j$  is the outcome of  $y$  in run  $j$ . Provide the expressions you used for calculating RMSPOS, RMSVEL, RMSSPD and RMSCRS. Indicate which is your final (“best”) design.

5. Indicate the distributions of NORXE and NEES and their 95% probability regions. Justify the final choice of the design parameters. Summarize the effect of the Markov chain transition matrix on the final results based on the designs illustrated in item 4.
6. Plot for both designs RMSPOS and FPOS; RMSVEL and FVEL; NORXE with its probability region; RMSSPD; RMSCRS; NEES with its probability region; and MODPR. Comment on the estimator bias and consistency.

7. Repeat item 4 for the best single-model filter (a standard KF) of your choice (a single design). Indicate how the design parameter was chosen and comment on its performance as compared to the best IMM.

#### DELIVERABLES:

A concise report is due.

Each student will have to make a 10- to 15-min. presentation in class. It is suggested to prepare 8–10 viewgraphs for this presentation, describing the designs and performance plots. The most important plots are:

- A single run with the true and estimated trajectories (best IMM). Plot the true trajectory as a solid line, marking the position (+) at the sampling time; the measurement ( $\square$ ) connected with a line to the corresponding true position; and the updated position ( $\circ$ ), also connected with a line to the corresponding true position at each sampling time. Provide magnified plots of the turn portions.

and the following comparison plots:

- RMSPOS for the two IMMs (RMSPOSIMM-L, RMSPOSIMM-CT) and the KF (RMSPOSKF), together, for comparison; also, indicate the (baseline) raw unfiltered position error (RMSPOSRAW).
- RMSSPD for the two IMMs and the KF (together, for comparison).
- RMSCRS for the two IMMs and the KF (together, for comparison).

**490** 11 ADAPTIVE ESTIMATION AND MANEUVERING TARGETS

- NEES for the two IMM's and the KF (together, for comparison).
- MOD2PR for the two IMM's (together, for comparison).

Develop an incremental animated display for the tracker with the following specifications:

- Plot the true position (black dot) every 0.1 s at a rate  $n$  times faster than real time, with the display speedup  $n \in [1, 10]$ . This high density sequence of points should appear as a continuous line.
- Each position measurement should appear as a box “□” connected with a red line to the true position at the corresponding time.
- The display of the measurement should be delayed from its true time by a constant  $\Delta$ , which is independent of the display speedup. Use  $\Delta = 0.1$  s to start with, then adjust to make the display more appealing if necessary.
- Each updated position should appear as bullet “●”, the starting point of a black arrow of length and direction given by the estimated velocity vector multiplied by the sampling interval  $T$ . The tip of this arrow will be then the predicted measurement for the next time under a linear motion model.
- The starting point of the arrow should be connected with a blue line to the true position at the corresponding time.
- The display of the measurement should be delayed from its true time by a constant  $2\Delta$ , which is independent of the display speedup. Use  $\Delta = 0.1$  s to start with, then adjust to make the display more appealing if necessary.
- You can use your creativity to provide sound effects (e.g., at each measurement time) to keep the audience entertained.

The display should grow incrementally, rather than being restarted at each sampling time.