

Chapter 5

STATE ESTIMATION IN DISCRETE-TIME LINEAR DYNAMIC SYSTEMS

5.1 INTRODUCTION

5.1.1 Outline

This chapter extends the estimation concepts presented previously to the case of dynamic (time-varying) quantities. The estimation of the *state vector* of a *stochastic linear dynamic system* is considered.

The state estimator for discrete-time linear dynamic systems driven by white noise — the (discrete-time) **Kalman filter** — is introduced in Section 5.2 and its properties are discussed. The continuous-time case is considered later, in Chapter 9. An example that illustrates the discrete time Kalman filter is given in Section 5.3.

The issue of consistency¹ of a dynamic estimator, which is crucial for *evaluation of estimator optimality* in every implementation, is discussed in Section 5.4.

The initialization of estimators and practical ways to make it consistent are presented in Section 5.5.

5.1.2 Discrete-Time Linear Estimation — Summary of Objectives

Extend the static estimation concepts to dynamic systems.

Derive the state estimator for discrete-time linear dynamic systems driven by white noise — the (discrete-time) Kalman filter.

Present the properties of the Kalman filter:

¹Or, “how to keep the filter honest”; the origin of the preoccupation with this issue is summarized in problem 5-6.

- Optimality in the linear Gaussian case and best linear state estimator in the (linear) non-Gaussian case
- Whiteness of the innovations
- The role of the Riccati equation for the state covariance and its relationship with the CRLB
- Stability and steady-state
- Connection with the observability and controllability of the system

Introduce the *likelihood function of a filter*, to be used later in adaptive filtering.

Discuss the issues of

- Consistency of a dynamic estimator and *estimator evaluation*
- Initialization of estimators

and practical ways to make it consistent.

5.2 LINEAR ESTIMATION IN DYNAMIC SYSTEMS — THE KALMAN FILTER

5.2.1 The Dynamic Estimation Problem

Consider a discrete-time linear dynamic system described by a vector difference equation with additive white Gaussian noise that models “unpredictable disturbances.” The dynamic (plant) equation is

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k) \quad k = 0, 1, \dots \quad (5.2.1-1)$$

where $x(k)$ is the n_x -dimensional state vector, $u(k)$ is an n_u -dimensional **known input** vector (e.g., control or sensor platform motion), and $v(k)$, $k = 0, 1, \dots$, is the sequence of zero-mean white Gaussian **process noise** (also n_x -vectors) with covariance

$$E[v(k)v(k)'] = Q(k) \quad (5.2.1-2)$$

The measurement equation is

$$z(k) = H(k)x(k) + w(k) \quad k = 1, \dots \quad (5.2.1-3)$$

with $w(k)$ the sequence of zero-mean white Gaussian **measurement noise** with covariance

$$E[w(k)w(k)'] = R(k) \quad (5.2.1-4)$$

The matrices F , G , H , Q , and R are assumed **known** and possibly time-varying. In other words, the system can be *time-varying* and the noises *non-stationary*. The initial state $x(0)$, in general unknown, is modeled as a *random variable*, Gaussian distributed with known mean and covariance. The two noise sequences and the initial state are assumed *mutually independent*.

The above constitutes the ***linear Gaussian (LG) assumption***.

In the dynamic equation (5.2.1-1), the process noise term $v(k)$ is sometimes taken as $\Gamma(k)v(k)$ with $v(k)$ an n_v -vector and $\Gamma(k)$ a known $n_x \times n_v$ matrix. Then the covariance matrix of the disturbance in the state equation, which is $Q(k)$ if $v(k)$ enters directly, is to be replaced by

$$E[[\Gamma(k)v(k)][\Gamma(k)v(k)]'] = \Gamma(k)Q(k)\Gamma(k)' \quad (5.2.1-5)$$

The linearity of (5.2.1-1) and (5.2.1-3) leads to the preservation of the Gaussian property of the state and measurements — this is a ***Gauss-Markov process***.

The following notation will be used: The conditional mean

$$\hat{x}(j|k) \triangleq E[x(j)|Z^k] \quad (5.2.1-6)$$

where

$$Z^k \triangleq \{z(i), i \leq k\} \quad (5.2.1-7)$$

denotes the sequence of observations available at time k , is the

- ***Estimate of the state*** if $j = k$ (also called filtered value)
- ***Smoothed value of the state*** if $j < k$
- ***Predicted value of the state*** if $j > k$

The ***estimation error*** is defined as

$$\tilde{x}(j|k) \triangleq x(j) - \hat{x}(j|k) \quad (5.2.1-8)$$

The ***conditional covariance matrix*** of $x(j)$ given the data Z^k or the ***covariance associated with the estimate*** (5.2.1-6) is

$$P(j|k) \triangleq E[[x(j) - \hat{x}(j|k)][x(j) - \hat{x}(j|k)]'|Z^k] = E[\tilde{x}(j|k)\tilde{x}(j|k)'|Z^k] \quad (5.2.1-9)$$

Remarks

The smoothed state has also been called recently the ***retrodicted state*** [Drummond93]. The term smoothing is, however, commonly used, even though retrodiction is the correct antonym of prediction. Sometimes the estimated state is called (incorrectly) the smoothed state.

Note that the ***covariance of the state*** is the same as the ***covariance of the estimation error*** — this is a consequence of the fact that the estimate is the conditional mean (5.2.1-6). (See also problem 5-3.)

It was shown earlier that the ***MMSE criterion*** for estimation leads to the ***conditional mean*** as the ***optimal estimate***.

As discussed in Section 3.2, if two vectors are jointly Gaussian, then the probability density of one conditioned on the other is also Gaussian. Thus the conditional mean (5.2.1-6) will be evaluated using this previous result.

The Estimation Algorithm

The estimation algorithm starts with the **initial estimate** $\hat{x}(0|0)$ of $x(0)$ and the associated **initial covariance** $P(0|0)$, assumed to be available. The second (conditioning) argument 0 stands for Z^0 , the **initial information**. Practical procedures to obtain the initial estimate and initial covariance will be discussed later.

One cycle of the dynamic estimation algorithm — the **Kalman filter (KF)** — will thus consist of mapping the estimate

$$\hat{x}(k|k) \triangleq E[x(k)|Z^k] \quad (5.2.1-10)$$

which is the conditional mean of the state at time k (the “current stage”) given the observations up to and including time k , and the associated covariance matrix

$$P(k|k) = E[(x(k) - \hat{x}(k|k))(x(k) - \hat{x}(k|k))'|Z^k] \quad (5.2.1-11)$$

into the corresponding variables at the next stage, namely, $\hat{x}(k+1|k+1)$ and $P(k+1|k+1)$.

This follows from the fact that a Gaussian random variable is *fully characterized* by its first two moments.

The values of past known inputs are subsumed in the conditioning, but (most of the time) will not be shown explicitly.

5.2.2 Dynamic Estimation as a Recursive Static Estimation

The recursion that yields the state estimate at $k+1$ and its covariance can be obtained from the static estimation equations (3.2.1-7) and (3.2.1-8)

$$\hat{x} \triangleq E[x|z] = \bar{x} + P_{xz}P_{zz}^{-1}(z - \bar{z}) \quad (5.2.2-1)$$

$$P_{xx|z} \triangleq E[(x - \hat{x})(x - \hat{x})'|z] = P_{xx} - P_{xz}P_{zz}^{-1}P_{zx} \quad (5.2.2-2)$$

by the following substitutions, indicated below by “ \rightarrow ”.

The *prior* (unconditional) expectations from the static case become *prior to the availability of the measurement at time $k+1$* in the dynamic case, that is, *given the data up to and including k* .

The *posterior* (conditional) expectations become *posterior to obtaining the measurement at time $k+1$* , that is, *given the data up to and including $k+1$* .

The variable to be estimated is the state at $k+1$

$$x \rightarrow x(k+1) \quad (5.2.2-3)$$

Its mean prior to $k+1$ — the **(one-step) predicted state** — is

$$\bar{x} \rightarrow \bar{x}(k+1) \triangleq \hat{x}(k+1|k) \triangleq E[x(k+1)|Z^k] \quad (5.2.2-4)$$

Based on the observation (measurement)

$$z \rightarrow z(k+1) \quad (5.2.2-5)$$

with prior mean — the ***predicted measurement***

$$\bar{z} \rightarrow \bar{z}(k+1) \triangleq \hat{z}(k+1|k) \triangleq E[z(k+1)|Z^k] \quad (5.2.2-6)$$

one computes the estimate posterior to $k+1$ — the ***updated state estimate*** (or, just the ***updated state***)

$$\hat{x} \rightarrow \hat{x}(k+1) \triangleq \hat{x}(k+1|k+1) \triangleq E[x(k+1)|Z^{k+1}] \quad (5.2.2-7)$$

Note

Two notations have been used in (5.2.2-4) and (5.2.2-6): with single argument and with two arguments. The single-argument notation — with “overbar” for one-step predicted values and “hat” for updated value — is patterned after the static case. The two-argument notation, which clearly indicates the time index of the conditioning data, is more general and is the preferred one because it applies to smoothing and general prediction as well. Nevertheless, one can find both in the literature.

The Covariances

The prior covariance matrix of the state variable $x(k+1)$ to be estimated — the ***state prediction covariance*** or ***predicted state covariance*** — is

$$P_{xx} \rightarrow \bar{P}(k+1) \triangleq P(k+1|k) \triangleq \text{cov}[x(k+1)|Z^k] = \text{cov}[\tilde{x}(k+1|k)|Z^k] \quad (5.2.2-8)$$

with the last equality following from (5.2.1-9).

The (prior) covariance of the observation $z(k+1)$ — the ***measurement prediction covariance*** — is

$$P_{zz} \rightarrow S(k+1) \triangleq \text{cov}[z(k+1)|Z^k] - \text{cov}[\tilde{z}(k+1|k)|Z^k] \quad (5.2.2-9)$$

The covariance between the variable to be estimated $x(k+1)$ and the observation $z(k+1)$ is

$$P_{xz} \rightarrow \text{cov}[x(k+1), z(k+1)|Z^k] = \text{cov}[\tilde{x}(k+1|k), \tilde{z}(k+1|k)|Z^k] \quad (5.2.2-10)$$

The posterior covariance of the state $x(k+1)$ — the ***updated state covariance*** — is

$$\begin{aligned} P_{xx|z} \rightarrow P(k+1) &\triangleq P(k+1|k+1) = \text{cov}[x(k+1)|Z^{k+1}] \\ &= \text{cov}[\tilde{x}(k+1|k+1)|Z^{k+1}] \end{aligned} \quad (5.2.2-11)$$

Note

Similarly to the single-argument and two-argument notations for the state, \bar{x} and \hat{x} , one has the notations \bar{P} and P for the one-step predicted and updated state covariances. As before, the two-argument notations are preferred and will be used in the sequel. However, the flowchart of the estimation algorithm will be given with both notations.

The Filter Gain

The weighting matrix from the estimation (“updating”) equation (5.2.2-1) becomes the *filter gain*

$$P_{xz} P_{zz}^{-1} \rightarrow W(k+1) \triangleq \text{cov}[x(k+1), z(k+1)|Z^k] S(k+1)^{-1} \quad (5.2.2-12)$$

Remark

The reasons for the designation of the above as filter gain are as follows:

1. The recursive estimation algorithm is a filter — it reduces the effect of the various noises on the quantity of interest (the state estimate).
2. The quantity (5.2.2-12) multiplies the observation $z(k+1)$, which is the input to the filter; that is, this quantity is a gain.

5.2.3 Derivation of the Dynamic Estimation Algorithm

The *predicted state* (5.2.2-4) follows by applying on the state equation (5.2.1-1) the operator of expectation conditioned on Z^k ,

$$E[x(k+1)|Z^k] = E[F(k)x(k) + G(k)u(k) + v(k)|Z^k] \quad (5.2.3-1)$$

Since the process noise $v(k)$ is *white and zero mean*, this results in

$$\hat{x}(k+1|k) = F(k)\hat{x}(k|k) + G(k)u(k) \quad (5.2.3-2)$$

Subtracting the above from (5.2.1-1) yields the *state prediction error*

$$\tilde{x}(k+1|k) \triangleq x(k+1) - \hat{x}(k+1|k) = F(k)\tilde{x}(k|k) + v(k) \quad (5.2.3-3)$$

Note the cancellation of the input $u(k)$ in (5.2.3-3) — it has no effect on the estimation error as long as it is *known*.

The *state prediction covariance* (5.2.2-8) is

$$E[\tilde{x}(k+1|k)\tilde{x}(k+1|k)'|Z^k] = F(k)E[\tilde{x}(k|k)\tilde{x}(k|k)'|Z^k]F(k)' + E[v(k)v(k)'] \quad (5.2.3-4)$$

which can be rewritten as

$$\boxed{P(k+1|k) = F(k)P(k|k)F(k)' + Q(k)} \quad (5.2.3-5)$$

The cross-terms in (5.2.3-4) are zero due to the fact that $v(k)$ is zero mean and white and, thus, orthogonal to $\tilde{x}(k|k)$.

The **predicted measurement** (5.2.2-6) follows similarly by taking the expected value of (5.2.1-3) conditioned on Z^k ,

$$E[z(k+1)|Z^k] = E[H(k+1)x(k+1) + w(k+1)|Z^k] \quad (5.2.3-6)$$

Since the measurement noise $w(k+1)$ is zero mean and white, this becomes

$$\boxed{\hat{z}(k+1|k) = H(k+1)\hat{x}(k+1|k)} \quad (5.2.3-7)$$

Subtracting the above from (5.2.1-3) yields the **measurement prediction error**

$$\tilde{z}(k+1|k) \triangleq z(k+1) - \hat{z}(k+1|k) = H(k+1)\tilde{x}(k+1|k) + w(k+1) \quad (5.2.3-8)$$

The **measurement prediction covariance** in (5.2.2-9) follows from (5.2.3-8), in a manner similar to (5.2.3-5), as

$$\boxed{S(k+1) = H(k+1)P(k+1|k)H(k+1)' + R(k+1)} \quad (5.2.3-9)$$

The covariance (5.2.2-10) between the state and measurement is, using (5.2.3-8),

$$\begin{aligned} E[\tilde{x}(k+1|k)\tilde{z}(k+1|k)'] &= E[\tilde{x}(k+1|k)[H(k+1)\tilde{x}(k+1|k) + w(k+1)]]' \\ &= P(k+1|k)H(k+1)' \end{aligned} \quad (5.2.3-10)$$

The **filter gain** (5.2.2-12) is, using (5.2.3-9) and (5.2.3-10),

$$\boxed{W(k+1) \triangleq P(k+1|k)H(k+1)'S(k+1)^{-1}} \quad (5.2.3-11)$$

Thus the **updated state estimate** (5.2.2-7) can be written according to (5.2.2-1) as

$$\boxed{\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + W(k+1)\nu(k+1)} \quad (5.2.3-12)$$

where

$$\boxed{\nu(k+1) \triangleq z(k+1) - \hat{z}(k+1|k) = \tilde{z}(k+1|k)} \quad (5.2.3-13)$$

is called the **innovation** or **measurement residual**. This is the same as (5.2.3-8), but the notation ν will be used in the sequel. Note that in view of this, S is also the **innovation covariance**.

Finally, the **updated covariance** (5.2.2-11) of the state at $k + 1$ is, according to (5.2.2-2),

$$\begin{aligned} P(k+1|k+1) &= P(k+1|k) - P(k+1|k) \\ &\quad \cdot H(k+1)' S(k+1)^{-1} H(k+1) P(k+1|k) \\ &= [I - W(k+1)H(k+1)]P(k+1|k) \end{aligned} \quad (5.2.3-14)$$

or, in symmetric form,

$$P(k+1|k+1) = P(k+1|k) - W(k+1)S(k+1)W(k+1)' \quad (5.2.3-15)$$

Equation (5.2.3-12) is called the **state update** equation, since it yields the updated state estimate, and (5.2.3-15) is the **covariance update** equation.

Note the similarity between the state update equation (5.2.3-12) and the recursive LS equation (3.4.2-15). The covariance update equation (5.2.3-15) is analogous to (3.4.2-11). The only difference is that in the LS case the prediction to $k + 1$ from k is the same as the updated value at k . This follows from the fact that in the LS formulation one deals with a constant parameter while in a dynamic system the state evolves in time.

Alternative Forms for the Covariance Update

Similarly to (3.4.2-6), there is a recursion for the inverse covariance

$$P(k+1|k+1)^{-1} = P(k+1|k)^{-1} + H(k+1)' R(k+1)^{-1} H(k+1) \quad (5.2.3-16)$$

Using the matrix inversion lemma, it can be easily shown that (5.2.3-16) is algebraically equivalent to (5.2.3-15). The filter using (5.2.3-16) instead of (5.2.3-15) is known as the **information matrix filter** (see Chapter 7).

As in (3.4.2-12), the filter gain (5.2.3-11) has the alternate expression

$$W(k+1) = P(k+1|k+1)H(k+1)'R(k+1)^{-1} \quad (5.2.3-17)$$

An alternative form for the covariance update equation (5.2.3-15), which holds for an *arbitrary gain* W (see Problem 5-5), called the **Joseph form covariance update**, is

$$\begin{aligned} P(k+1|k+1) &= [I - W(k+1)H(k+1)]P(k+1|k) \\ &\quad \cdot [I - W(k+1)H(k+1)]' + W(k+1)R(k+1)W(k+1)' \end{aligned} \quad (5.2.3-18)$$

While this is computationally more expensive than (5.2.3-15), it is less sensitive to round-off errors: It will not lead to negative eigenvalues, as (5.2.3-15) is prone to, due to the subtraction present in it. Numerical techniques that reduce the sensitivity to round-off errors are discussed in Chapter 7.

Equation (5.2.3-18) can be also used for evaluation of the **sensitivity of the filter to an incorrect gain** (see Section 5.6). This equation can also be used to obtain the optimal gain (see problem 5-9).

Intuitive Interpretation of the Gain

Note from (5.2.3-11) that the *optimal filter gain* is (taking a simplistic “scalar view” of it)

1. “Proportional” to the state prediction variance
2. “Inversely proportional” to the innovation variance

Thus, the gain is

- “Large” if the state prediction is “inaccurate” (has a large variance) and the measurement is “accurate” (has a relatively small variance)
- “Small” if the state prediction is “accurate” (has a small variance) and the measurement is “inaccurate” (has a relatively large variance)

A large gain indicates a “rapid” response to the measurement in updating the state, while a small gain yields a slower response to the measurement. In the frequency domain it can be shown that these properties correspond to a higher/lower **bandwidth of the filter**.

A filter whose *optimal gain* is higher yields less “noise reduction,” as one would expect from a filter with a higher bandwidth. This will be seen quantitatively in the next chapter.

Remark

Equations (5.2.3-9) and (5.2.3-15) yield **filter-calculated covariances**, which are exact if all the modeling assumptions used in the filter derivation hold. In practice this is not always the case and the validity of these filter-calculated estimation accuracies should be tested, as discussed in Section 5.4.

5.2.4 Overview of the Kalman Filter Algorithm

Under the *Gaussian assumption* for the initial state (or initial state error) and all the noises entering into the system, the Kalman filter is the **optimal MMSE state estimator**. If these random variables are *not Gaussian* and one has only their first two moments, then, in view of the discussion from Section 3.3, the Kalman filter algorithm is the **best linear state estimator**, that is, the **LMMSE state estimator**.

The flowchart of one cycle of the Kalman filter is presented in Fig. 5.2.4-1 with the two-argument notations and in Fig. 5.2.4-2 with the one-argument notations. Note that at every stage (cycle) k the entire past is summarized by the *sufficient statistic* $\hat{x}(k|k)$ and the associated covariance $P(k|k)$.

The left-side column represents the true system’s evolution from the state at time k to the state at time $k + 1$ with the input $u(k)$ and the process noise $v(k)$. The measurement follows from the new state and the noise $w(k + 1)$. The

known input (e.g., control, platform motion, or sensor pointing) enters (usually) the system with the knowledge of the latest state estimate and is used by the state estimator to obtain the predicted value for the state at the next time.

The state estimation cycle consists of the following:

1. State and measurement prediction (also called *time update*)
2. State update (also called *measurement update*)

The state update requires the filter gain, obtained in the course of the covariance calculations. The covariance calculations are *independent* of the state and measurements (and control — assumed to be known) and can, therefore, be performed *offline*.

The Workhorse of Estimation — The Kalman Filter

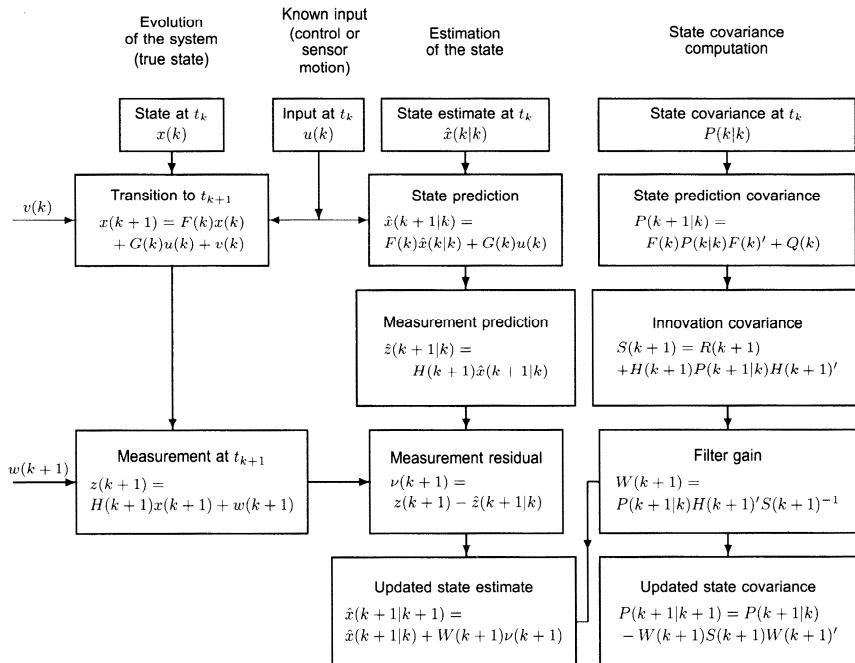


Figure 5.2.4-1: One cycle in the state estimation of a linear system.

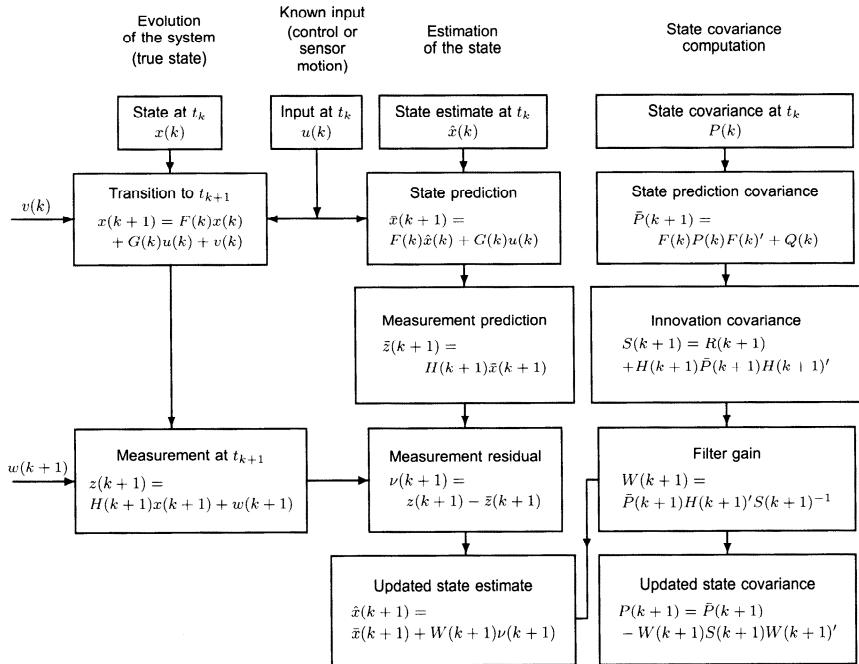


Figure 5.2.4-2: One cycle in the state estimation of a linear system (with single-argument notations).

Summary of the Statistical Assumptions of the Kalman Filter

The initial state has the known mean and covariance

$$E[x(0)|Z^0] = \hat{x}(0|0) \quad (5.2.4-1)$$

$$\text{cov}[x(0)|Z^0] = P(0|0) \quad (5.2.4-2)$$

where Z^0 denotes the initial (prior) information.

The process and measurement noise sequences are *zero mean and white* with *known covariance matrices*

$$E[v(k)] = 0 \quad (5.2.4-3)$$

$$E[v(k)v(j)'] = Q(k)\delta_{kj} \quad (5.2.4-4)$$

$$E[w(k)] = 0 \quad (5.2.4-5)$$

$$E[w(k)w(j)'] = R(k)\delta_{kj} \quad (5.2.4-6)$$

All the above are *mutually uncorrelated*

$$E[x(0)v(k)'] = 0 \quad \forall k \quad (5.2.4-7)$$

$$E[x(0)w(k)'] = 0 \quad \forall k \quad (5.2.4-8)$$

$$E[v(k)w(j)'] = 0 \quad \forall k, j \quad (5.2.4-9)$$

It can be easily shown that under the Gaussian assumption the whiteness and the uncorrelatedness of the noises imply the following:

$$E[v(k)|Z^k] = E[v(k)] = 0 \quad (5.2.4-10)$$

$$E[w(k)|Z^{k-1}] = E[w(k)] = 0 \quad (5.2.4-11)$$

Property (5.2.4-10) was used in (5.2.3-2), while (5.2.4-11) was used in (5.2.3-7).

Remark

The dynamic (plant) equation parameters — the matrices F , G — and the measurement equation parameters — the matrix H — are assumed known.

Computational Requirements

The *computational requirements* of the KF are approximately proportional to n^3 where $n = \max(n_x, n_z)$.

Some Extensions

The assumptions of

- White process noise
- White measurement noise
- Uncorrelatedness between the process and the measurement noise sequences

can be relaxed.

An autocorrelated (“colored”) noise has to be modeled as the output of a subsystem driven by white noise; that is, it has to be *prewhitened*, as discussed in Chapter 4. For an *autocorrelated process noise*, the state vector has to be augmented to incorporate this subsystem. An example of prewhitening of an autocorrelated process noise is presented in Section 8.2.

The situation where there is *correlation between the two noise sequences* is discussed in Section 8.3. The filter derivation for an *autocorrelated measurement noise*, which can be done without augmenting the state, is presented in Section 8.4.

Discrete-time *smoothing* is presented in Section 8.6.

Continuous-time state estimation as an extension of the discrete-time results is discussed in Chapter 9.

5.2.5 The Matrix Riccati Equation

As pointed out in Section 3.2, the covariance equations in the static MMSE estimation problem are independent of the measurements. Consequently, the covariance equations for the state estimation problem (in a linear dynamic system), derived in Subsection 5.2.3, can be iterated forward offline.

It can be easily shown that the following recursion can be written for the one-step prediction covariance

$$P(k+1|k) = F(k)\{P(k|k-1) - P(k|k-1)H(k)' \cdot [H(k)P(k|k-1)H(k)' + R(k)]^{-1}H(k)P(k|k-1)\}F(k)' + Q(k) \quad (5.2.5-1)$$

This is the *discrete-time (difference) matrix Riccati equation*, or just the *Riccati equation*. The above follows by substituting (5.2.3-9) and (5.2.3-11) into (5.2.3-15) and substituting the result into (5.2.3-5).

The solution of the above Riccati equation for a time-invariant system converges to a finite *steady-state covariance* if

1. The pair $\{F, H\}$ is *completely observable*.

If, in addition,

2. The pair $\{F, C\}$, where $Q \triangleq CC'$ (C is the *Cholesky factor* (see Subsection 7.4.2) — a square root of Q), is *completely controllable*,

then the steady-state covariance is a *unique positive definite matrix* — independent of the initial covariance.

The steady-state covariance matrix is the solution of the *algebraic matrix Riccati equation* (or just the *algebraic Riccati equation*)

$$P = F[P - PH'(HPH' + R)^{-1}HP]F' + Q \quad (5.2.5-2)$$

and this yields the *steady-state gain* for the Kalman filter.

The interpretation of the above conditions is as follows:

1. The observability condition on the state guarantees a “steady flow” of information about *each* state component — this prevents the uncertainty from becoming unbounded. This condition yields the existence of a (not necessarily unique) steady-state solution for the covariance matrix that is positive definite or positive semidefinite (i.e., with finite positive or nonnegative eigenvalues, respectively).

2. The controllability condition states that the process noise enters into each state component and prevents the covariance of the state from converging to zero. This condition causes the covariance to be positive definite (i.e., all the eigenvalues are positive).

Filter Stability

The convergence of the covariance to a *finite* steady state — that is, *the error becoming a stationary process in the MS sense* — is equivalent to *filter*

stability in the bounded input bounded output sense.

Stability of the filter does not require the dynamic system to be stable — only the observability condition (1) is required. As indicated above, observability alone does not guarantee uniqueness — the steady-state solution might depend on the initial covariance — but the existence (finiteness) of the solution is the key.

This is particularly important, since the state models used in tracking are *unstable* — they have an integration (from velocity to position). Stability means “bounded input bounded output,” and this condition is not satisfied by an integrator — its continuous-time transfer function has a pole at the origin and in discrete time it has a pole at 1.

Remarks

If the state covariance matrix is *positive semidefinite* rather than positive definite, that is, it has some zero eigenvalues that reflect the filter’s “belief” that it has “perfectly accurate” estimates of some state components, the gain will be zero for those state components — an *undesirable feature*.

In view of this, in many applications where there is no physical process noise, an *artificial process noise* or *pseudo-noise* is assumed (i.e., a matrix Q that will lead to condition (2) being satisfied).

The Riccati Equation and the CRLB

The lower bound on the minimum achievable covariance in state estimation is given by the (posterior) CRLB, which was presented for random parameters in (2.7.2-3). Since the state is a random variable, this is applicable to state estimation.

In the linear Gaussian case it can be shown that this is given by the solution of the Riccati equation. In the more general non-Gaussian case, the solution of the Riccati equation is the covariance matrix (actually matrix MSE) associated with the best linear state estimate; however, as discussed in Subsection 3.3.2, a nonlinear estimator can provide better estimates — with covariance smaller than the solution of the Riccati equation.

5.2.6 Properties of the Innovations and the Likelihood Function of the System Model

The Innovations — a Zero-Mean White Sequence

An important property of the *innovation sequence* is that it is an *orthogonal sequence*, that is,

$$\boxed{E[\nu(k)\nu(j)'] = S(k)\delta_{kj}} \quad (5.2.6-1)$$

where δ_{kj} is the Kronecker delta function.

This can be seen as follows. Without loss of generality, let $j \leq k - 1$. Use will be made of the smoothing property of the conditional expectations (see Subsection 1.4.12)

$$E[\nu(k)\nu(j)'] = E[E[\nu(k)\nu(j)'|Z^{k-1}]] \quad (5.2.6-2)$$

Note that $\nu(j)$ is a linear combination of the measurements up to j ; that is, given Z^{k-1} , it is *not a random variable anymore* and it can thus be taken outside the inner expectation. This yields

$$E[\nu(k)\nu(j)'] = E[E[\nu(k)|Z^{k-1}]\nu(j)'] \quad (5.2.6-3)$$

The inside expectation in (5.2.6-3) is, in view of (5.2.2-6),

$$E[z(k) - \hat{z}(k|k-1)|Z^{k-1}] = 0 \quad (5.2.6-4)$$

and therefore (5.2.6-1) follows for $k \neq j$.

The uncorrelatedness property (5.2.6-1) of the innovations implies that since they are Gaussian, the innovations are independent of each other and thus the innovation sequence is *strictly white*. Without the Gaussian assumption, the innovation sequence is wide sense white.

Thus the innovation sequence is *zero mean and white*.

Remark

Unlike the innovations, the state estimation errors are not white — they are *correlated in time* (see problem 5-11).

The Likelihood Function of the System Model

The joint pdf of the measurements up to k , denoted as

$$Z^k = \{z(j)\}_{j=1}^k \quad (5.2.6-5)$$

can be written as

$$p[Z^k] = p[z(k), Z^{k-1}] = p[z(k)|Z^{k-1}]p[Z^{k-1}] = \prod_{i=1}^k p[z(i)|Z^{i-1}] \quad (5.2.6-6)$$

where Z^0 is the prior information, shown explicitly only in the expression of (5.2.6-6).

If the above pdfs are Gaussian, then

$$\begin{aligned} p[z(i)|Z^{i-1}] &= \mathcal{N}[z(i); \hat{z}(i|i-1), S(i)] = \mathcal{N}[z(i) - \hat{z}(i|i-1); 0, S(i)] \\ &= \mathcal{N}[\nu(i); 0, S(i)] = p[\nu(i)] \end{aligned} \quad (5.2.6-7)$$

Using (5.2.6-7) in (5.2.6-6) yields

$$p[Z^k] = \prod_{i=1}^k p[\nu(i)] \quad (5.2.6-8)$$

that is, the joint pdf of the sequence of measurements Z^k is equal to the product of the marginal pdfs of the corresponding innovations. This shows the *informational equivalence of the measurements and the innovations*.

Since (5.2.6-8) is the joint pdf of Z^k conditioned on the system model (not indicated explicitly), it is the **likelihood function of the system model**. This will be used in Chapter 11 to evaluate the “goodness” of models in multiple model adaptive filtering.

5.2.7 The Innovations Representation

The counterpart of the Riccati equation that yields the recursion of the one-step prediction covariance $P(k+1|k)$ is the recursion of the one-step prediction of the state $\hat{x}(k+1|k)$, called the **innovations representation**.

This is obtained from (5.2.3-2) and (5.2.3-12), without the deterministic input, for simplicity, as

$$\begin{aligned} \hat{x}(k+1|k) &= F(k)\hat{x}(k|k-1) + F(k)W(k)\nu(k) \\ &= F(k)\hat{x}(k|k-1) + W_i(k)[z(k) - H(k)\hat{x}(k|k-1)] \end{aligned} \quad (5.2.7-1)$$

where

$$W_i(k) \triangleq F(k)W(k) \quad (5.2.7-2)$$

is the gain in the innovations representation (sometimes called ambiguously the filter gain).

Equation (5.2.7-1) can also be rewritten as the state equation

$$\hat{x}(k+1|k) = [F(k) - W_i(k)H(k)]\hat{x}(k|k-1) + W_i(k)z(k) \quad (5.2.7-3)$$

with the input being the (nonwhite) sequence $z(k)$ and the output being the innovation

$$\nu(k) = -H(k)\hat{x}(k|k-1) + z(k) \quad (5.2.7-4)$$

This motivates the name innovations representation for the system (5.2.7-3) and (5.2.7-4).

The Kalman Filter as a Whitening System

Note that while the input to this system is not white, its output is a white sequence. Thus the state estimation filter, written as (5.2.7-3) and (5.2.7-4), can be seen as a *whitening system for the measurement sequence*.

5.2.8 Some Orthogonality Properties

In the static estimation problem the LMMSE estimator was derived based on the principle of orthogonality, which states that the estimation error \tilde{x} has to be orthogonal to the observation(s) z , that is,

$$\tilde{x} \perp z \quad \Leftrightarrow \quad \langle \tilde{x}, z \rangle \triangleq E[\tilde{x}z] = 0 \quad (5.2.8-1)$$

The state estimator for linear dynamic systems — the Kalman filter — while derived under the Gaussian assumption, is (as pointed out earlier) the LMMSE estimator. Therefore, the orthogonality properties carry over.

Note that the estimate is a linear function of the measurements

$$\hat{x}(k|k) = \mathsf{L}_k(Z^k) \quad \forall k \quad (5.2.8-2)$$

or, in a more general manner,

$$\hat{x}(i|k) = \mathsf{L}_{i,k}(Z^k) \quad \forall i, k \quad (5.2.8-3)$$

where L denotes a *linear transformation* (because the estimates are linear functions of the measurements) and the measurement set Z^k includes the initial information Z^0 .

Thus, the estimation error

$$\tilde{x}(i|k) \triangleq x(i) - \hat{x}(i|k) \quad (5.2.8-4)$$

has the following *orthogonality properties*

$$\tilde{x}(i|k) \perp z(j) \quad \forall j \leq k \quad (5.2.8-5)$$

$$\tilde{x}(i|k) \perp \hat{x}(l|j) \quad \forall j \leq k, \quad \forall i, l \quad (5.2.8-6)$$

With the Gaussian assumption, all the orthogonality properties — which are equivalent to uncorrelatedness — also imply independence.

5.2.9 The Kalman Filter — Summary

The *MMSE state estimation model* for a dynamic system consists of the following:

- *Initial state* — unknown, assumed to be a *random variable* with a certain mean (initial estimate) and covariance (measure of the accuracy of the initial estimate).
- Evolution of the *system's state* — according to a possibly time-varying *linear difference equation* (*plant equation* or *dynamics*) driven by
 - A known input (the control);
 - An additive random disturbance — the *process noise* — a zero-mean white (uncorrelated) stochastic process with a known, possibly time-varying, covariance.
- *Measurements* — a *linear function* of the state with an additive random disturbance (*measurement noise*), which is a zero-mean white stochastic process with a known, possibly time-varying, covariance.

If, in addition, all the random variables of the problem, namely,

- The initial state,
- The process noises, and
- The measurement noises

are *Gaussian and mutually independent* (i.e., under the *LG assumption*), the MMSE estimate of the state of the system under consideration — the conditional mean of the state given the measurements — is given by the (discrete-time) Kalman filter.

The discrete-time Kalman filter computes recursively the MMSE estimate of the state of a dynamic system through the following stages:

- Starting from the *current updated state estimate* (estimate of the current state given the observations up to and including the current time), the *predicted value* of the *state* for the next sampling time is computed.
- Using the predicted state, the *predicted value* of the next *measurement* is calculated.
- When the new measurement is obtained, the difference between it and its predicted value — the *innovation* (residual) — is evaluated.
- The *updated state at the next time* is obtained as the sum of the predicted state and the correction term. The correction term is the product of the *filter gain* (obtained separately) and the innovation.

Covariance and filter gain calculation:

- Starting from the *current updated state covariance*, the *state prediction covariance* is computed.
- Using the state prediction covariance, the *measurement prediction covariance* (which is the same as the *innovation covariance*) is obtained.
- The *filter gain* is calculated from the state and measurement prediction covariances.
- The *updated covariance* associated with the next state is then computed.

The covariance prediction and update equations combined together result in the (*discrete time*) *matrix Riccati equation*.

For a *time-invariant system*, if it is *observable*, the state estimation covariance will be finite and the Riccati equation will converge to a steady-state solution. If, in addition, the process noise excites each state component, then the steady-state covariance is also positive definite and unique.

The gain of the filter reflects the relative accuracy of the predicted state versus the new measurement:

- If the new measurement is deemed “more accurate” than the predicted state, then the filter gain will be relatively high.
- If the predicted state is deemed “more accurate” than the new measurement, then the gain will be low.

If the state covariance matrix is positive semidefinite, this reflects the filter’s “belief” that it has “perfectly accurate” estimates of some state components — an undesirable feature.

The innovation sequence is *zero mean, white (uncorrelated)*, with covariance equal to the measurement prediction covariance.

The joint pdf of the sequence of measurements, which is the *likelihood function of the model*, is equal to the product of the marginal pdfs of the corresponding innovations, which are Gaussian under the LG assumption.

If the random variables in the state estimation problem for a linear system are *not Gaussian* and one only has their first two moments, then the Kalman filter is the *LMMSE estimator* and the covariances are really the corresponding MSE matrices. In this case, the likelihood function (needed in an adaptive multiple model estimation) is, typically, still assumed to be Gaussian, even though this is clearly an approximation.

Remark

One can summarize in a nutshell what the KF does: It estimates/predicts the state of a linear system with *zero-mean white noise* disturbances — it reduces the uncertainties due to such disturbances. While this seems very restrictive, through a number of extensions, this technique can be used in numerous problems where the disturbances do not satisfy the above conditions. The essence of the extensions is to map the problem at hand into one that meets (or nearly meets) the above constraints.

5.3 EXAMPLE OF A FILTER

5.3.1 The Model

Consider the system with state

$$\dot{x} = \begin{bmatrix} \xi \\ \dot{\xi} \end{bmatrix} \quad (5.3.1-1)$$

which evolves according to

$$x(k+1) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} v(k) \quad k = 0, 1, \dots, 99 \quad (5.3.1-2)$$

with initial condition

$$x(0) = \begin{bmatrix} 0 \\ 10 \end{bmatrix} \quad (5.3.1-3)$$

This represents a one-dimensional motion with position ξ and velocity $\dot{\xi}$ sampled at intervals T , which will be assumed as unity in the sequel.

Note that (5.3.1-2) is of the form

$$x(k+1) = Fx(k) + \Gamma v(k) \quad (5.3.1-4)$$

The process noise, a scalar, which models the acceleration, is a zero-mean white sequence with variance

$$E[v(k)^2] = q \quad (5.3.1-5)$$

The measurements consist of the state's position component corrupted by additive noise

$$z(k) = [1 \ 0] x(k) + w(k) \quad k = 1, \dots, 100 \quad (5.3.1-6)$$

where the measurement noise is a zero-mean white sequence with variance

$$E[w(k)^2] = r = 1 \quad (5.3.1-7)$$

The two noise sequences are mutually independent.

The filter was initialized according to the “two-point differencing” procedure discussed later in Section 5.5 in (5.5.3-3) to (5.5.3-5) with the first two measurements at $k = 1, 2$ and the filter started running at $k = 3$.

See Subsection 5.3.3 for a step-by-step demonstration of using DynaEst™ to set up a simulation scenario and design an estimator.

5.3.2 Results for a Kalman Filter

Using DynaEst, Figs. 5.3.2-1 through 5.3.2-3 present

1. The true and estimated trajectories of the system in the state space (position-velocity)
2. The variances of the predicted position, $P_{11}(k|k-1)$, and updated position, $P_{11}(k|k)$
3. The variances of the predicted velocity, $P_{22}(k|k-1)$, and updated velocity, $P_{22}(k|k)$
4. Filter gains

for three cases with different values of the process noise variance q .

In the first case, with $q = 0$, the controllability condition 2 of Subsection 5.2.5 does not hold and the state estimation covariance matrix is seen from Fig. 5.3.2-1b to converge to zero. In this case the filter gain also converges to zero as shown in Fig. 5.3.2-1c. The reason this is an undesirable feature is that the filter's "belief" of having a *perfect state estimate* (which "shuts it off") hinges on the *assumed perfect noiseless constant velocity motion*. In practice such an assumption usually does not hold, except for short periods of time.

The filter for $q = 0$ is equivalent to the LS estimation of the initial position and velocity (straight line fitting), as discussed in Subsection 3.5.1.

For the nonzero values of q , the filter is seen from Figs. 5.3.2-2 and 5.3.2-3 to reach steady state quite rapidly, in a few time steps.

In Fig. 5.3.2-2, which shows the results on a motion with $q = 1$, the steady-state filter gain is $[0.75 \ 0.50]'$. In Fig. 5.3.2-3, which corresponds to a strong process noise ($q = 9$) that models a "highly unpredictable" motion, the (optimal) steady-state gain is $[0.90 \ 0.94]'$ (i.e., higher). Note how the case with higher gain (due to the higher process noise level) leads to a larger state variance — less accurate state estimates.

5.3.3 A Step-by-Step Demonstration of DynaEstTM

The companion software *DynaEstTM*, written in MATLABTM, can be used to set up a simulation scenario, generate measurements, and design an estimator to obtain state estimates. Alternatively, one can import measurements from an external file — for example, obtained using a real sensor — and run the tracker on those real measurements. DynaEst can also be used to analyze the performance of the estimator in terms of RMS estimation errors and filter consistency. In the following, a step-by-step demonstration of designing the simulated scenario in Section 5.3.1 and the Kalman filter in 5.3.2 using DynaEst is presented.

Starting DynaEst Start MATLAB as you would normally and, from the MATLAB prompt, change to the directory where DynaEst files are stored. For example, if DynaEst files are stored in directory /matlab/dynaest,

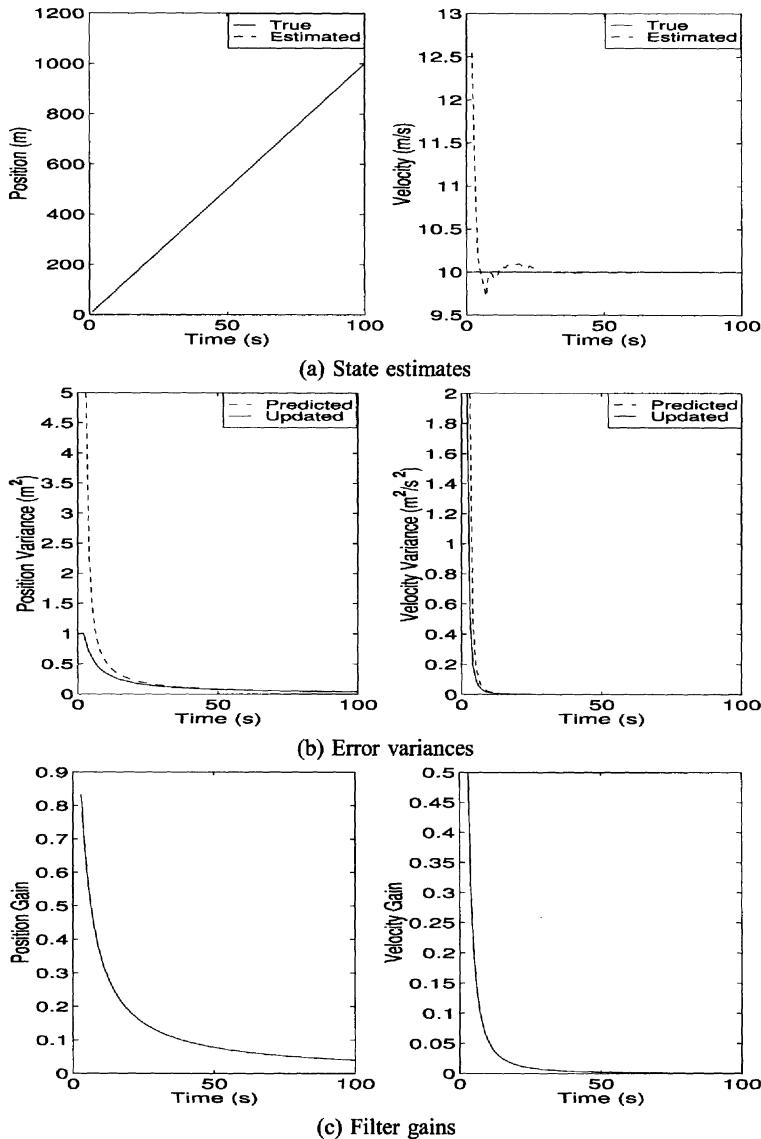


Figure 5.3.2-1: State estimates, error variances and filter gains for $q = 0$.

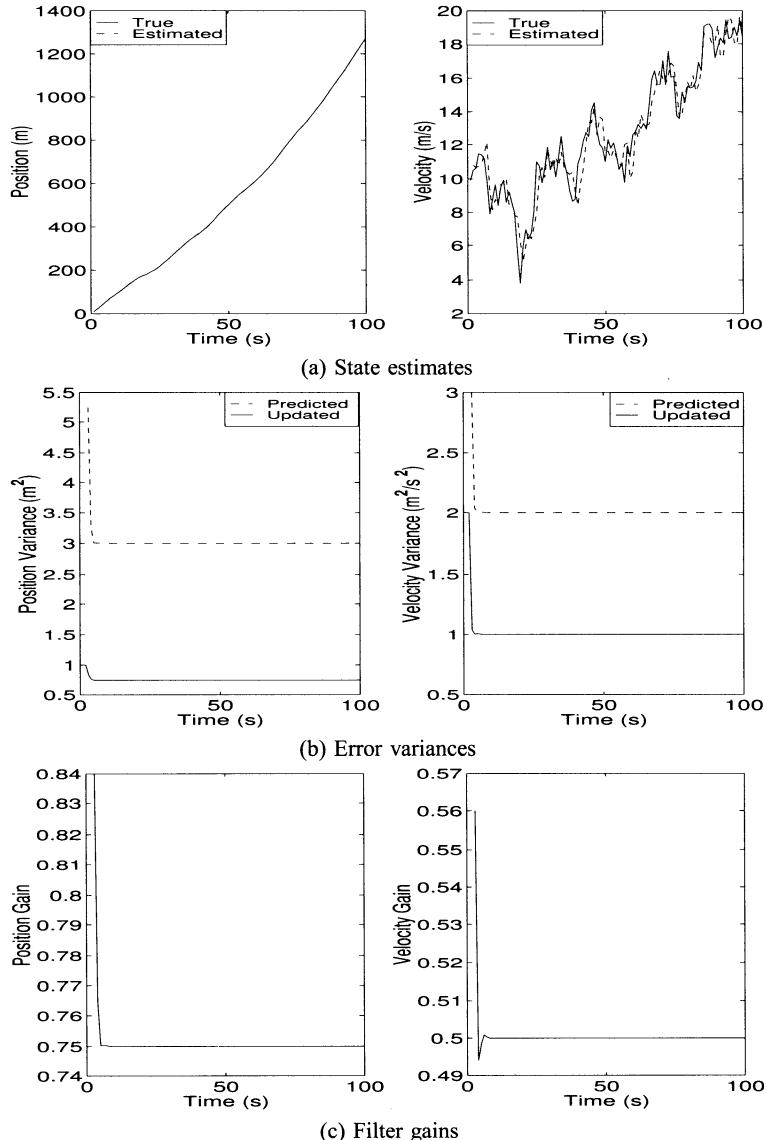


Figure 5.3.2-2: State estimates, error variances and filter gains for $q = 1$.

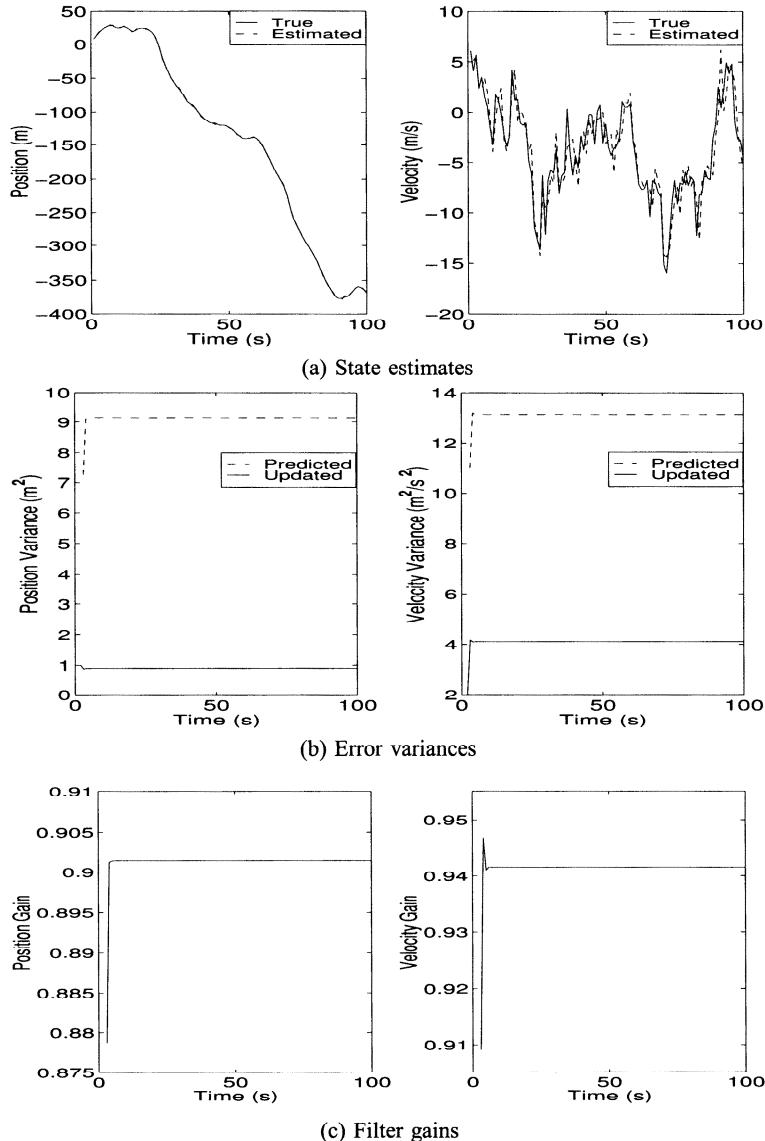


Figure 5.3.2-3: State estimates, error variances and filter gains for $q = 9$.



Figure 5.3.3-1: Creating a new project in DynaEst.

type `cd /matlab/dynaest`. And then DynaEst can be started by typing `dynaest` at the prompt.

Creating a Project In DynaEst, the simulation and estimation parameters are stored as a project in a file with extension `prj`. To create a new project, select the New option in the menu `File` as shown in Fig. 5.3.3-1. After specifying the parameters for simulation and estimation, the project can be saved in a file by issuing the `Save As` command in the `File` menu. The project file name is to be given following the command `Save As`. Alternatively, one can open an already saved project using the `Open` option under the same menu.

Data Source When the option to create a new project is selected, one can specify the data source for estimation. As mentioned earlier, one can generate **simulated measurements** or use **imported measurements** from a real sensor for estimation. To simulate the scenario in Subsection 5.3.1, select `Simulation` and press `Next` as shown in Fig. 5.3.3-2.

Ground Truth When the option to simulate the system is specified, one can also set the system dynamics (defined in the state space) interactively within DynaEst or from an external file. To simulate the scenario in Subsection 5.3.1, select `Define` interactively and press `Next` as shown in Fig. 5.3.3-3.

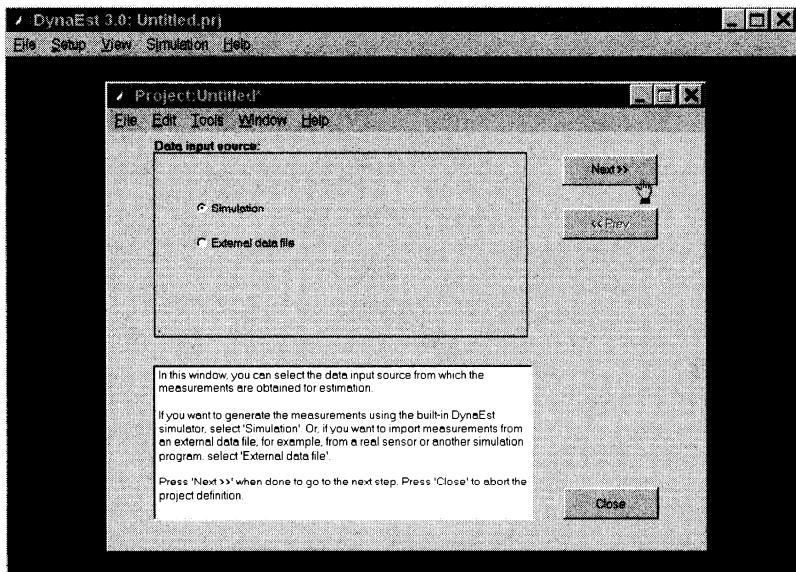


Figure 5.3.3-2: Selecting the data source for estimation.

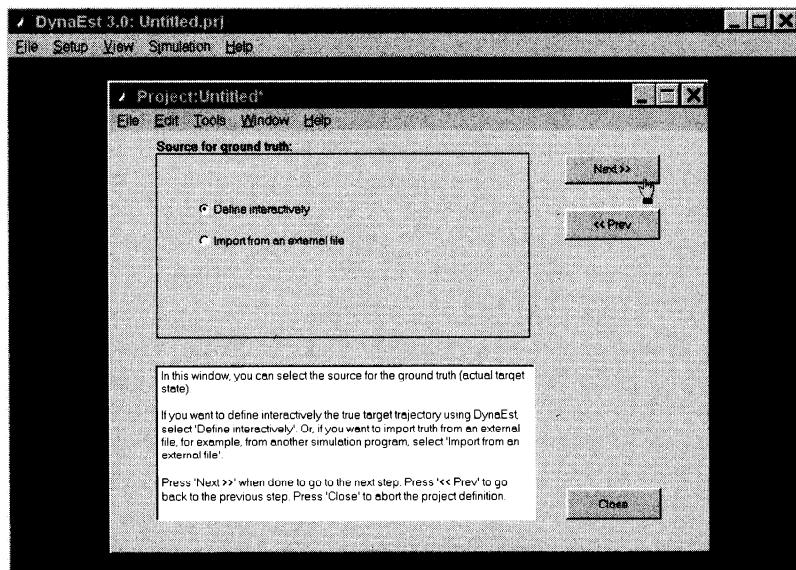


Figure 5.3.3-3: Specifying the ground truth.

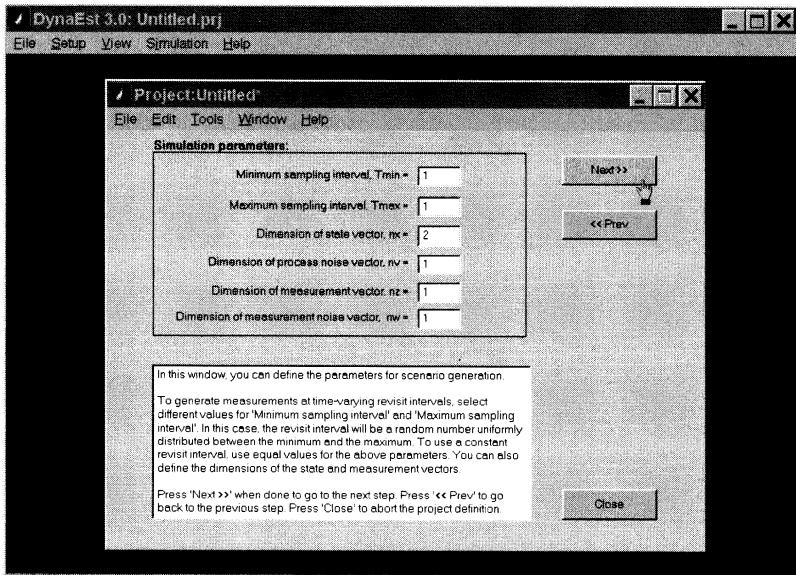


Figure 5.3.3-4: Specifying the system parameters.

System Parameters To simulate the scenario in Subsection 5.3.1, specify the parameters of the system defined in (5.3.1-1)–(5.3.1-7) and press Next as shown in Fig. 5.3.3-4. In this step, the dimensions of the state and noise vectors and system matrices are set up.

Simulation Parameters Since the measurements are obtained using simulation, one has to specify the number of Monte Carlo runs used for measurement generation and estimation. In addition, the total duration (number of sampling times or *revisits*) of simulation in each run has to be specified as well.

With DynaEst it is possible to generate true target trajectories consisting of a number of motion legs — for example, a constant velocity motion leg followed by a coordinated turn or linear acceleration/deceleration. Since the scenario in Subsection 5.3.1 consists of only a straight line segment, the number of legs is set to one. Specify the parameters and press Next as shown in Fig. 5.3.3-5.

Motion Equations Then the true system matrix $F(k)$ and the true noise gain matrix Γ in the true target state evolution, defined by (5.3.1-3), are specified. DynaEst has the capability to parse matrices given as functions of the sampling interval T . Define the matrices and press Next as shown in Figs. 5.3.3-6 and 5.3.3-7.

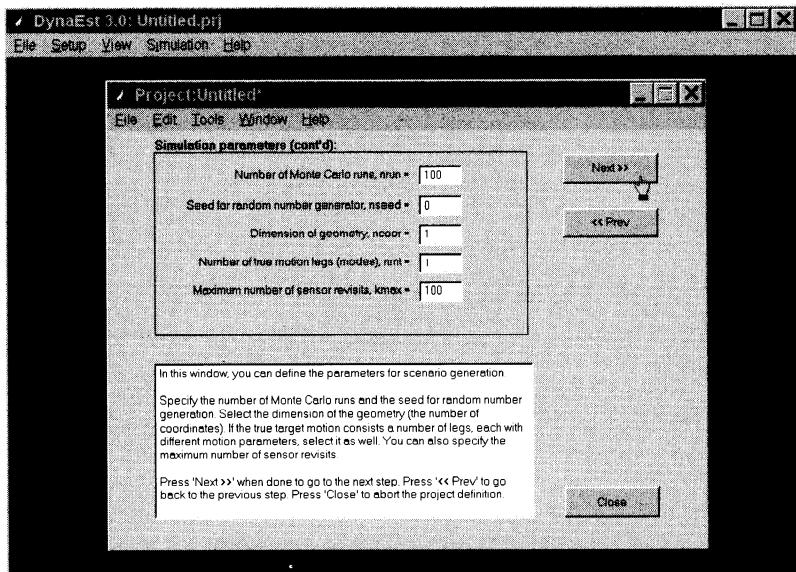


Figure 5.3.3-5: Specifying the simulation parameters.

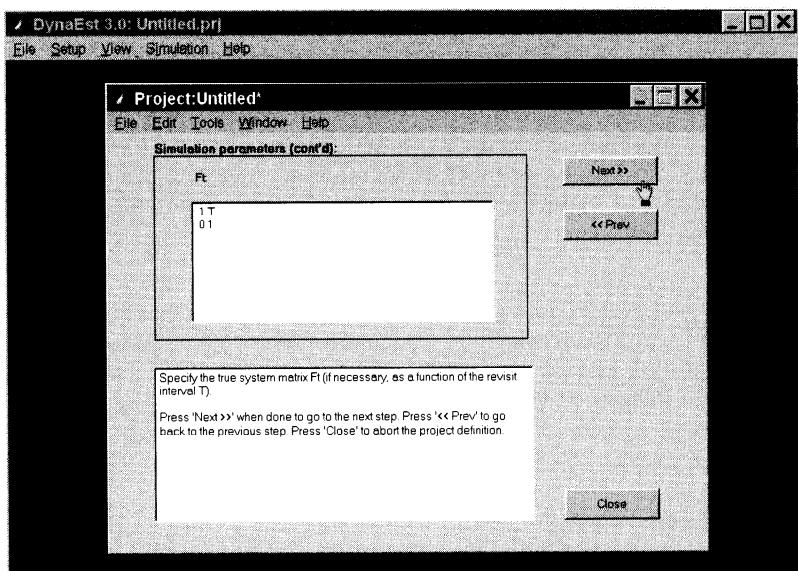


Figure 5.3.3-6: Specifying the system matrix.

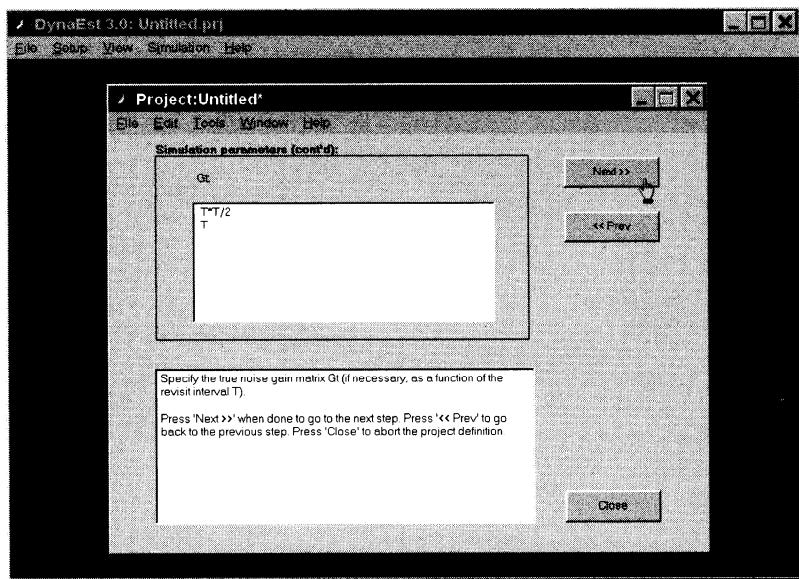


Figure 5.3.3-7: Specifying the noise gain matrix.

Measurement Equation The measurement equation (5.3.1-6) is specified as shown in Fig. 5.3.3-8.

Noise Parameters The process and measurement noise variances are specified as shown in Figs. 5.3.3-9 and 5.3.3-10, respectively. Note that $q = 1$ in Fig. 5.3.3-10.

Initial Conditions Finally, the initial state conditions are specified as part of the scenario definition as shown in Fig. 5.3.3-11.

Using the Defined Scenario With the above steps, the definition of the simulation scenario is complete and the scenario can be used for estimation, exporting measurements or for sensitivity analysis. For tracking, select the option as in Fig. 5.3.3-12 and proceed to the next step.

Selecting a Tracker DynaEst offers a variety of linear estimators, ranging from the simple α - β filter to the sophisticated IMM estimator. In this section, the design of a Kalman filter is demonstrated. Select the Kalman filter as the choice for estimation as shown in Fig. 5.3.3-13.

Selecting the Motion Model in the Estimator The crucial step in the estimator design is the selection of motion model and the noise parameters. In practice, one does not know the true values and has to make a good “engineer’s” choice for these design parameters. DynaEst offers a sequence of

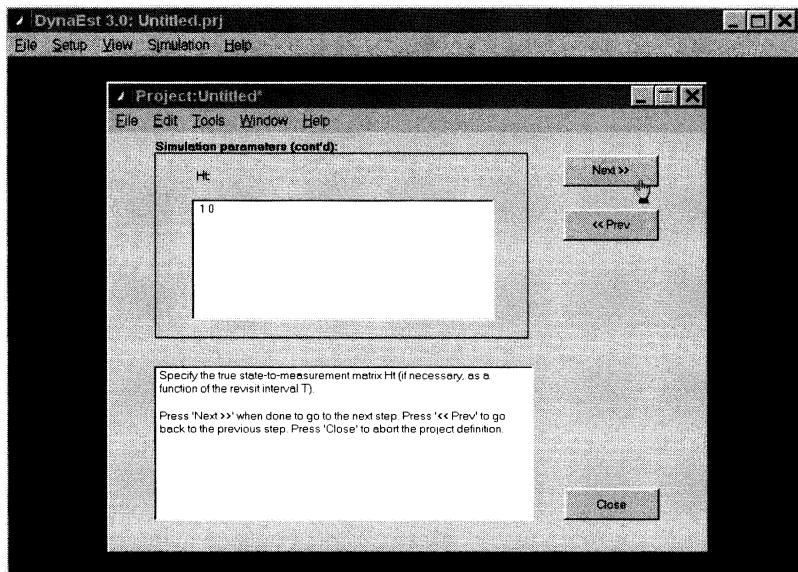


Figure 5.3.3-8: Specifying the state-to-measurement matrix.

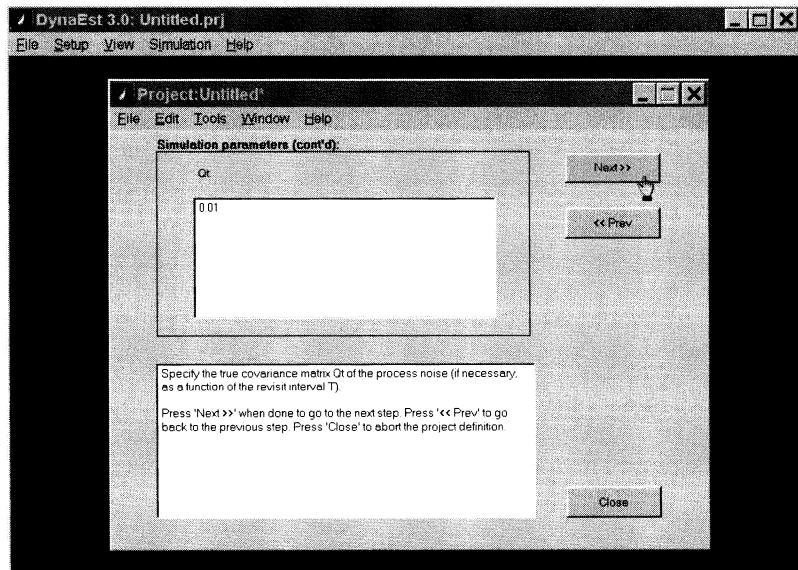


Figure 5.3.3-9: Specifying the process noise variance.

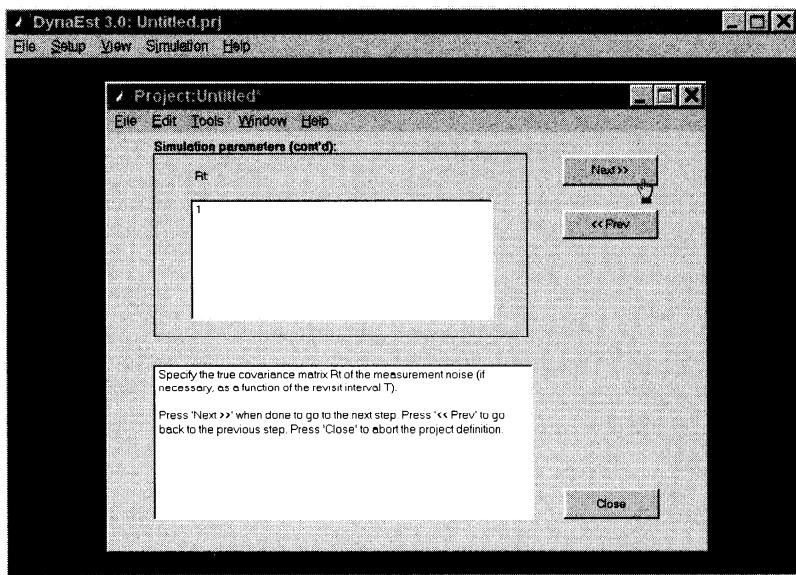


Figure 5.3.3-10: Specifying the measurement noise variance.

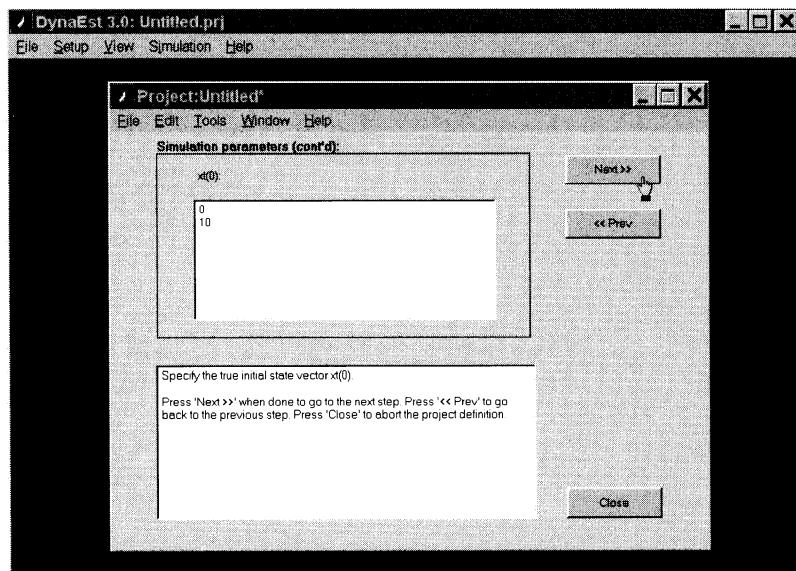


Figure 5.3.3-11: Specifying the initial conditions.

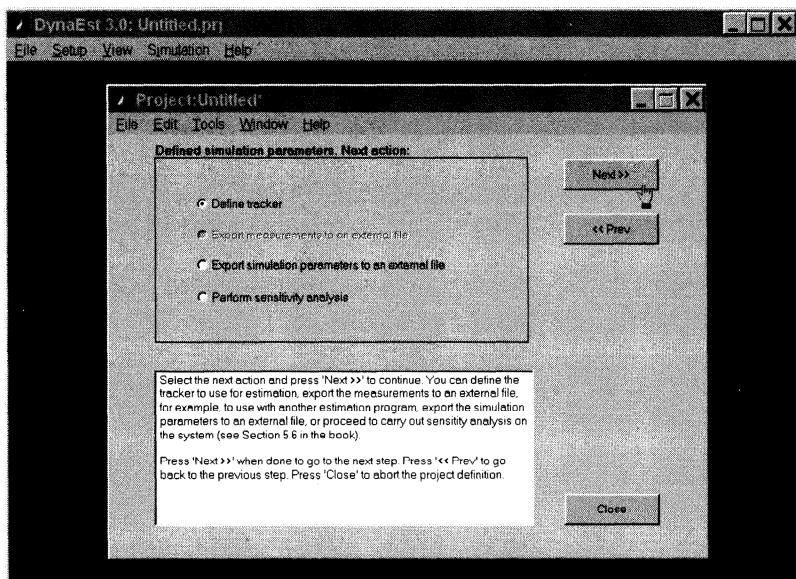


Figure 5.3.3-12: Using the defined scenario.

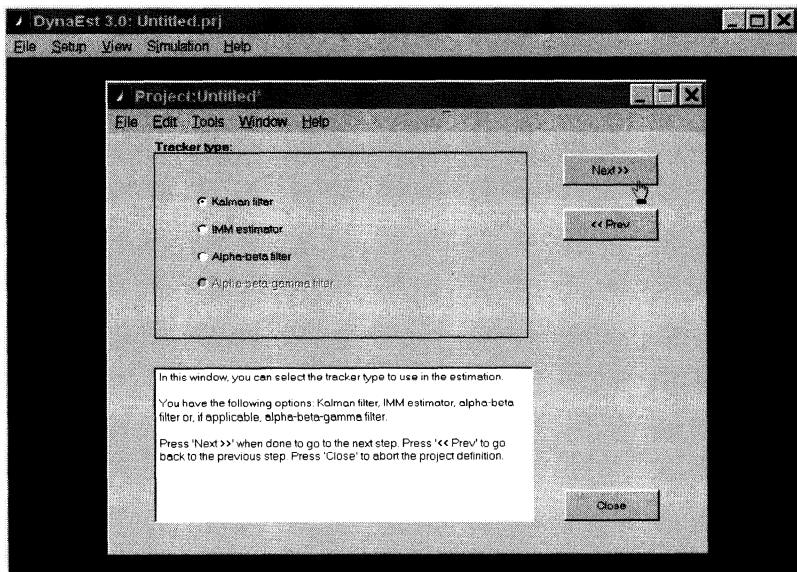


Figure 5.3.3-13: Selecting a tracker.

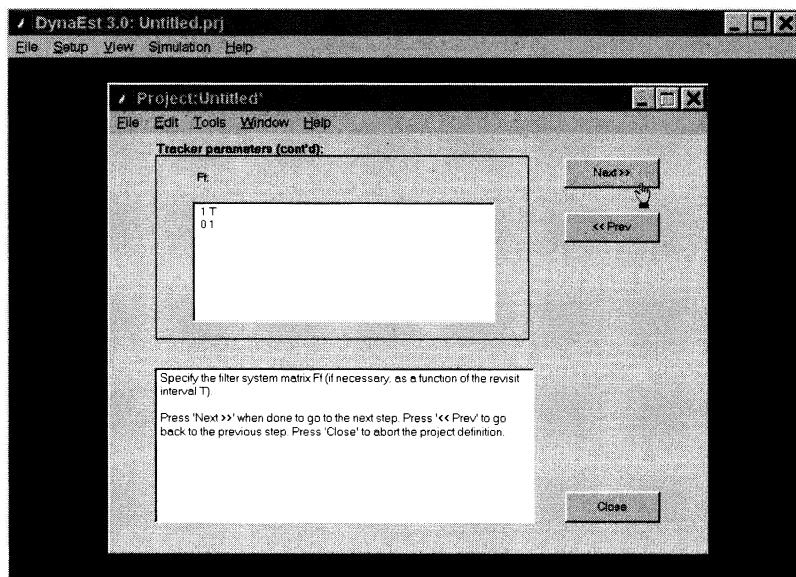


Figure 5.3.3-14: Designing the tracker.

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 estimator design process is shown in Fig. 5.3.3-14.

Viewing Estimation Results After the filter design parameters are specified, DynaEst executes the specified number of Monte Carlo runs and offers a choice to plot the estimation results as shown in Fig. 5.3.3-15.

Select the desired performance metric to view and obtain plots similar to those in Fig. 5.3.2-1–5.3.2-3.

The objective of DynaEst is to provide a user-friendly way to design and test linear estimators. However, it also provides the source code in MATLAB so that the user can modify the code to improve the built-in estimators or add additional ones. DynaEst contains far more functionalities than those demonstrated in this simple walk-through. The rest are left as an exercise to the reader.

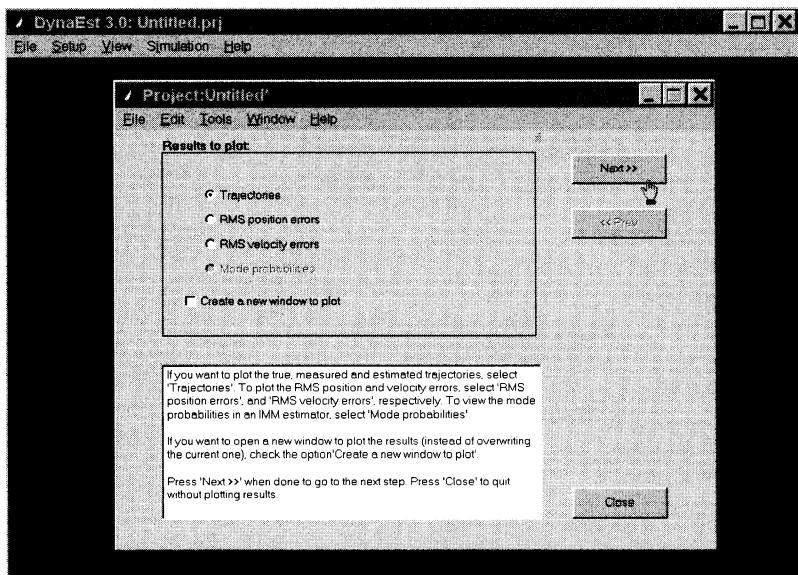


Figure 5.3.3-15: Viewing estimation results.

5.4 CONSISTENCY OF STATE ESTIMATORS

5.4.1 The Problem of Filter Consistency

In the problem of estimating a parameter that is constant, consistency of an estimator (i.e., a *static estimator*) was defined as *convergence of the estimate to the true value*. This implies that there is a steadily increasing amount of information (in the sense of Fisher) about the parameter that asymptotically reduces to zero the uncertainty about its true value.

When estimating the state of a dynamic system, in general, no convergence of its estimate occurs. What one has, in addition to the “current” estimate of the state, $\hat{x}(k|k)$, is the associated covariance matrix, $P(k|k)$.

The phenomenon of *divergence* has been observed: Sometimes the filter yields unacceptably large state estimation errors.² These can be due to one or more of the following:

- Modeling errors
- Numerical errors
- Programming errors

²In the 1960s, when one of the first Kalman filters performed poorly in an avionics application, it was called “the worst invention of the decade.” At that time the design of state estimators was still in the domain of “black magic.”

The question of what an **acceptable estimation error** is, will be discussed in the sequel.

Under the *Linear-Gaussian (LG)* assumption, the conditional pdf of the state $x(k)$ at time k is

$$p[x(k)|Z^k] = \mathcal{N}[x(k); \hat{x}(k|k), P(k|k)] \quad (5.4.1-1)$$

The modeling of the system consists of the dynamic equation, the measurement equation, and the statistical properties of the random variables entering into these equations. If all these are completely accurate, then (5.4.1-1) holds exactly. Since all models contain some approximations in practice, it is of interest to what extent one can verify (5.4.1-1).

Practical Evaluation of Consistency

The **statistical characterization of the disturbances** is usually done with moments up to second order and the resulting filter will then (hopefully) give approximate first and second order moments of the state.

In view of this, (5.4.1-1) is replaced by the two moment conditions

$$E[x(k) - \hat{x}(k|k)] \triangleq E[\tilde{x}(k|k)] = 0 \quad (5.4.1-2)$$

$$E[[x(k) - \hat{x}(k|k)][x(k) - \hat{x}(k|k)]'] \triangleq E[\tilde{x}(k|k)\tilde{x}(k|k)'] = P(k|k) \quad (5.4.1-3)$$

that the filter should satisfy in spite of its inherent approximations.

Condition (5.4.1-2) is the **unbiasedness** requirement for the estimates (i.e., zero-mean estimation error), while (5.4.1-3) is the **covariance matching** requirement — that is, that the **actual MSE** (left-hand side) matches the **filter-calculated covariance** (right-hand side).

Note that if there is a bias, this will increase the MSE, which is the bias squared plus the variance in the scalar case. Thus the test to be discussed in the next subsection will deal with the MSE. This test will be based on the fact that, under the LG assumption, one has

$$E[\tilde{x}(k|k)'P(k|k)^{-1}\tilde{x}(k|k)] = n_x \quad (5.4.1-4)$$

that is, the average of the **squared norm of the estimation error** indicated above has to be equal to the dimension of the corresponding vector since it is chi-square distributed (see Subsection 1.4.17).

Consistency and Optimality

Since the filter gain is based on the filter-calculated error covariances, it follows that **consistency is necessary for filter optimality**: Wrong covariances yield wrong gain.

This is why **consistency evaluation** is vital for verifying a filter design — it amounts to **evaluation of estimator optimality**.

5.4.2 Definition and the Statistical Tests for Filter Consistency

A state estimator (filter) is called **consistent** if its state estimation errors satisfy (5.4.1-2) and (5.4.1-3). This is a ***finite-sample consistency*** property, that is, the estimation errors based on a finite number of samples (measurements) should be consistent with their theoretical statistical properties:

1. Have mean zero (i.e., the estimates are unbiased).
2. Have covariance matrix as calculated by the filter.

In contradistinction, the parameter estimator consistency is an asymptotic (infinite size sample) property.

The **consistency criteria of a filter** are as follows:

- (a) The state errors should be acceptable as zero mean and have magnitude commensurate with the state covariance as yielded by the filter.
- (b) The innovations should also have the same property.
- (c) The innovations should be acceptable as white.

The last two criteria are the only ones that can be tested in applications with *real data*. The first criterion, which is really the most important one, can be tested only in simulations.

Using the notation

$$\tilde{x}(k|k) = x(k) - \hat{x}(k|k) \quad (5.4.2-1)$$

define the **normalized (state) estimation error squared (NEES)** as

$$\boxed{\epsilon(k) = \tilde{x}(k|k)' P(k|k)^{-1} \tilde{x}(k|k)} \quad (5.4.2-2)$$

The test to be presented next is based on the above quadratic form, and it can verify simultaneously both properties 1 and 2.

Under hypothesis H_0 that the filter is consistent and the LG assumption, $\epsilon(k)$ is chi-square distributed with n_x degrees of freedom, where n_x is the dimension of x . Then

$$E[\epsilon(k)] = n_x \quad (5.4.2-3)$$

and the test is whether (5.4.2-3) can be accepted.

Monte Carlo Simulation Based Tests

The test will be based on the results of **Monte Carlo simulations (runs)** that provide N independent samples $\epsilon^i(k)$, $i = 1, \dots, N$, of the random variable $\epsilon(k)$. Let the sample average of $\epsilon(k)$ — the (N -run) **average NEES** — be

$$\bar{\epsilon}(k) = \frac{1}{N} \sum_{i=1}^N \epsilon^i(k) \quad (5.4.2-4)$$

Then $N\bar{\epsilon}(k)$ will have, under H_0 , a chi-square density with Nn_x degrees of freedom.

Hypothesis (5.4.2-3), that the *state estimation errors are consistent with the filter-calculated covariances* — criterion (a), also called the *chi-square test* — is accepted if

$$\bar{\epsilon}(k) \in [r_1, r_2] \quad (5.4.2-5)$$

where the **acceptance interval** is determined such that

$$P\{\bar{\epsilon}(k) \in [r_1, r_2] | H_0\} = 1 - \alpha \quad (5.4.2-6)$$

For example, with $\alpha = 0.05$, $n_x = 2$, and $N = 50$, one has from (1.5.4-6), for a two-sided interval, $r_1 = 1.5$ and $r_2 = 2.6$. The interval given in (5.4.2-5) is then the (two-sided) 95% **probability concentration region** for $\bar{\epsilon}(k)$.

If $N = 1$ (i.e., a single run), one can also use this test. The two-sided 95% interval in this case is $[0.05, 7.38]$. Note the much narrower range of the interval corresponding to $N = 50$ Monte Carlo runs — this illustrates the **variability reduction** in such repeated simulations.

Note that a bias in the state estimation error will increase (5.4.2-2), and, if significant, it will yield unacceptably large values for the statistic (5.4.2-4).

If (5.4.2-5) is not satisfied, then a separate bias test using the sample mean of (5.4.2-1) should be carried out to identify the source of the problem. This can be done by taking each component of the state error, divided by its standard deviation, which makes it (under ideal conditions) $\mathcal{N}(0, 1)$, and testing to see if its mean can be accepted as zero.

The test statistic for the **normalized mean estimation error (NMEE)** for component j of the state from runs $i = 1, \dots, N$ is

$$\mu_j(k) = \frac{1}{N} \sum_{i=1}^N \frac{\tilde{x}_j^i(k|k)}{\sqrt{P_{jj}(k|k)}} \quad (5.4.2-7)$$

Under ideal conditions, the above is distributed $\mathcal{N}(0, 1/N)$.

Denoting by ξ a zero-mean unity-variance normal random variable, let r_1 be such that

$$P\{\xi \in [-r_1, r_1]\} = 1 - \alpha \quad (5.4.2-8)$$

where, say, $\alpha = 0.05$. Then, since the standard deviation of $\bar{\mu}_j$ is $1/\sqrt{N}$, the $1 - \alpha$ probability region for it will be $[-r, r]$ where $r = r_1/\sqrt{N}$. For the above value of α one has the two-sided 95% region given by $r_1 = 1.96$.

Thus, using this acceptance region based on the normal density, the hypothesis that the true mean of the state estimation error is zero is accepted if

$$\bar{\mu}_j(k) \in [-r, r] \quad (5.4.2-9)$$

The commensurateness of the innovations with their filter-calculated covariances — criterion (b) — is tested in a similar manner.

Under the hypothesis that the filter is consistent, the **normalized innovation squared (NIS)**

$$\epsilon_\nu(k) = \nu(k)' S(k)^{-1} \nu(k) \quad (5.4.2-10)$$

has a chi-square distribution with n_z degrees of freedom, where n_z is the dimension of the measurement.

From N independent samples $\epsilon_\nu^i(k)$ one calculates the (N -run) **average NIS**

$$\bar{\epsilon}_\nu(k) = \frac{1}{N} \sum_{i=1}^N \epsilon_\nu^i(k) \quad (5.4.2-11)$$

which is then tested as in (5.4.2-5) but with acceptance region determined based on the fact that $N\bar{\epsilon}_\nu(k)$ is chi-square distributed with Nn_z degrees of freedom.

Similarly to the procedure for state errors, if (5.4.2-11) is too large, then a bias test (i.e., whether the mean of the innovations is nonzero) has to be carried out.

The **whiteness test** for the innovations can be done for a single component at a time as follows.

The following (N -run) **sample autocorrelation** statistic is used

$$\bar{\rho}_l(k, j) = \sum_{i=1}^N \nu_l^i(k) \nu_l^i(j) \left[\sum_{i=1}^N [\nu_l^i(k)]^2 \sum_{i=1}^N [\nu_l^i(j)]^2 \right]^{-1/2} \quad l = 1, \dots, n_z \quad (5.4.2-12)$$

For N large enough, a normal approximation of the density of (5.4.2-12) for $k \neq j$ is convenient (and reasonable in view of the central limit theorem). If the innovations are zero mean and white, then the mean of (5.4.2-12) is zero and its variance is $1/N$. (See problem 5-4.)

Alternatively, it can be done simultaneously for the entire innovation vector by using the following statistic

$$\bar{\rho}(k, j) = \frac{1}{\sqrt{n_z}} \sum_{i=1}^N \nu^i(k)' \left[\sum_{i=1}^N \nu^i(k) \nu^i(k)' \right]^{-1/2} \left[\sum_{i=1}^N \nu^i(j) \nu^i(j)' \right]^{-1/2} \nu^i(j) \quad (5.4.2-13)$$

Similarly to (5.4.2-12), for N large enough, (5.4.2-13) for $k \neq j$ is zero-mean with variance is $1/N$ if the innovations are zero-mean and white (see problem 5-4) and a normal approximation can be used.

Thus, using the same acceptance region based on the normal density as in (5.4.2-9), the hypothesis that the true correlation of the innovation sequence is zero — criterion (c) — is accepted if

$$\bar{\rho}(k, j) \in [-r, r] \quad (5.4.2-14)$$

Real-Time (Single-Run) Tests

All the above tests assume that N **independent runs** have been made. While they can be used on a single run ($N = 1$), they have a very high variability in this case, as illustrated above. The question is whether one can achieve a low variability of the test statistic based on a single run, as in a real-time implementation — that is, having **real-time consistency tests**.

Test (a) requires that such independent simulations be made; however, criteria (b) and (c) can be tested on a **single run** in time as follows.

These tests are based on replacing the **ensemble averages** by **time averages** based on the *ergodicity* of the innovation sequence.

The whiteness test statistic for innovations, which are j steps apart, from a single run can be written as the **time-average autocorrelation**

$$\bar{\rho}_l(j) = \sum_{k=1}^K \nu_l(k)\nu_l(k+j) \left[\sum_{k=1}^K \nu_l(k)^2 \sum_{k=1}^K \nu_l(k+j)^2 \right]^{-1/2} \quad l = 1, \dots, n_z \quad (5.4.2-15)$$

This statistic is, for large enough K , in view of the central limit theorem, normally distributed. Furthermore, similarly to (5.4.2-12), its variance can be shown to be $1/K$ (see problem 5-4). Also, similarly to (5.4.2-13), one can write a single statistic that incorporates the entire innovation vector.

Criterion (b) can be tested with the **time-average normalized innovation squared** statistic

$$\bar{\epsilon}_\nu = \frac{1}{K} \sum_{k=1}^K \nu(k)'S(k)^{-1}\nu(k) \quad (5.4.2-16)$$

If the innovations are white, zero mean, and with covariance $S(k)$, then $K\bar{\epsilon}_\nu$ has a chi-square distribution with Kn_z degrees of freedom.

The probability regions for (5.4.2-15) and (5.4.2-16) for acceptance of the “consistent filter” hypothesis are then set up as before.

5.4.3 Examples of Filter Consistency Testing

The example of Section 5.3 is continued to illustrate the use of the **consistency tests**. The following tests are carried out:

- Offline single-run (simulation) tests
- Offline multiple run (Monte Carlo simulation) tests
- Online single-run (real time) tests

Two cases will be considered:

1. A filter which is based on exactly the same model as the process — that is, **matched** to the system
2. A filter which is based on a different model than the system — that is, **mismatched** to the system

Single-Run Simulation Tests

Figure 5.4.3-1 shows, for a *single run*, the behavior of the state's **NEES** (5.4.2-2) for various values of the process noise variance q for filters that are perfectly matched. Out of 100 points, 3 to 6 are found outside the 95% probability region.

In this case a one-sided region was considered. The upper limit of this probability region is approximately 6 since, for a $n_x = 2$ degrees of freedom chi-square random variable, the 5% tail point is

$$\chi_2^2(0.95) = 5.99 \quad (5.4.3-1)$$

Note that in this case the two-sided 95% region is [0.05, 7.38]. Since the lower limit is practically zero, only the upper limit is of interest and it was taken for the 5% tail rather than for the 2.5% tail, which is 7.38. It should be noted that taking a 5% or a 2.5% (or a 1%) tail is rather arbitrary.

Monte Carlo Simulation Tests

Figures 5.4.3-2a to 5.4.3-2c illustrate, using DynaEstTM, the test statistics obtained from $N = 50$ Monte Carlo runs. Two-sided probability regions are used in the sequel.

Figure 5.4.3-2a shows the state's *N-run average NEES* (5.4.2-4). The two-sided 95% region for a 100 degrees of freedom chi-square random variable is

$$[\chi_{100}^2(0.025), \chi_{100}^2(0.975)] = [74.2, 129.6] \quad (5.4.3-2)$$

Dividing the above by $N = 50$, the 95% probability region (5.4.2-5) for the average normalized state estimation error squared becomes [1.5, 2.6]. Note that 6 out of the 100 points fall outside this 95% region, which is acceptable.

Figure 5.4.3-2b shows the (*N-run*) **average NIS** (5.4.2-11). Noting that the innovations are scalar, the 95% probability region will be based on the 50 degrees of freedom chi-square distribution, and is

$$[\chi_{50}^2(0.025), \chi_{50}^2(0.975)] = [32.3, 71.4] \quad (5.4.3-3)$$

Dividing by $N = 50$, the region becomes [0.65, 1.43]. As the plot shows, 4 out of the 100 points are outside the 95% region, again an acceptable situation.

Figure 5.4.3-2c shows the (*N-run*) **sample autocorrelation** of the innovations (5.4.2-12) one step apart ($k - j = 1$). The 95% region $[-1.96\sigma, 1.96\sigma]$ is, for $\sigma = \frac{1}{\sqrt{N}} = 0.141$, the interval [-0.277, 0.277]. In this case, 8 out of the 100 points fall outside the 95% region, which is also acceptable.

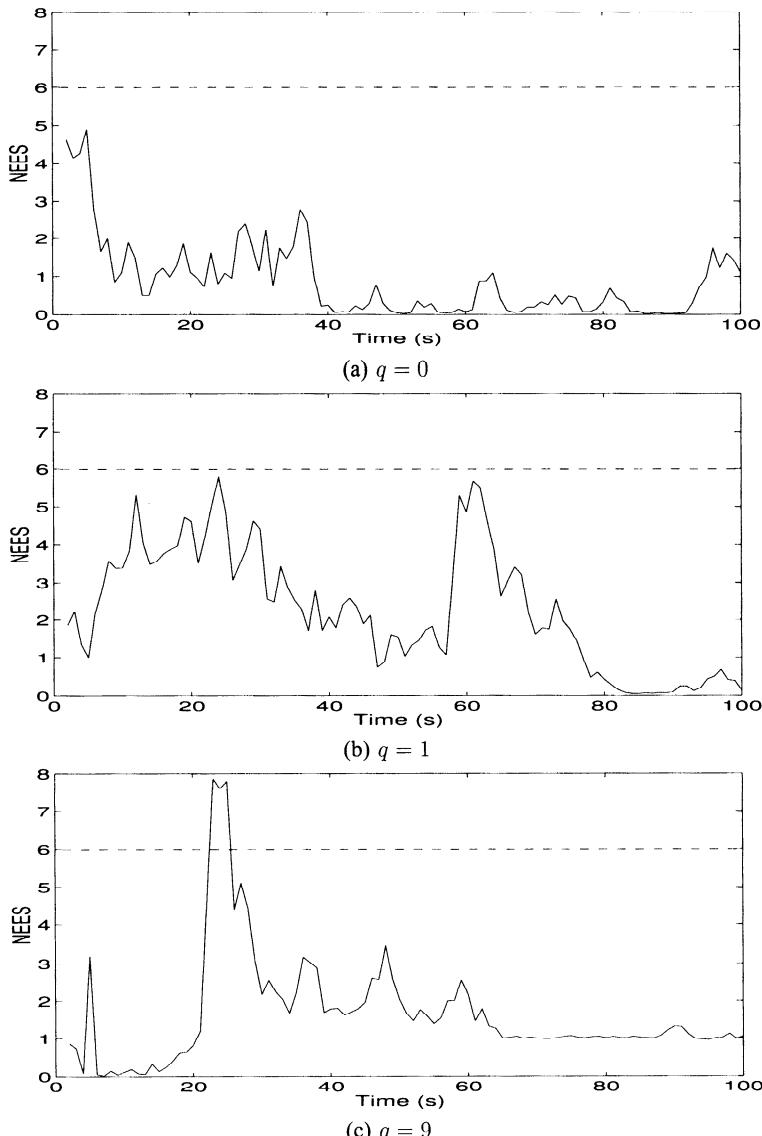


Figure 5.4.3-1: Normalized state estimation error squared from a single run with its 95% probability region.

Remark

These probability regions are also called **acceptance regions** because, if the test statistics fall in these regions, one can *accept* the hypothesis that the *filter is consistent*.

A Mismatched Filter

Next, a **mismatched filter** is examined. For this purpose it is assumed that the true (i.e., system) process noise has variance $q = 9$ while the model (i.e., filter) process noise has variance $q_F = 1$.

The normalized state estimation error squared, obtained with DynaEstTM, is shown in Fig. 5.4.3-3. The mismatch caused 41 points out of 100 to be outside the 95% probability region in a single run, clearly an unacceptable situation. The results of the Monte Carlo runs ($N = 50$) shown in Figs. 5.4.3-3b and 5.4.3-4 all show the serious mismatch: All points are outside the 95% region — actually they are all in the upper 2.5% tail region.

Real-Time Tests

Finally, the **real-time consistency tests** — that is, the *single-run tests that can be performed in real time* — are presented. For the correct filter the **time-average autocorrelation** of the innovations (5.4.2-15) obtained from 100 samples in time was

$$\bar{\rho}(1) = 0.152 \quad (5.4.3-4)$$

Under the assumption that the filter is correct, the error in the above estimate is normally distributed with mean zero and variance $1/100$

$$\bar{\rho} - \rho \sim \mathcal{N}(0, 0.1^2) \quad (5.4.3-5)$$

and its 95% probability region is $[-0.196, 0.196]$. The estimate (5.4.3-4) falls in this region, as expected, since the filter is matched to the system.

In the mismatched case the estimate was obtained as

$$\bar{\rho}(1) = 0.509 \quad (5.4.3-6)$$

which is clearly much too large and outside the region.

The other single-run test is for the **time-average normalized innovation squared** (5.4.2-16), also over 100 time steps.

For the matched filter, the result was

$$\bar{\epsilon}_\nu = 0.936 \quad (5.4.3-7)$$

(the ideal value is 1). The 95% confidence region is, based on the 100 degrees of freedom chi-square distribution, the interval $[0.74, 1.3]$.

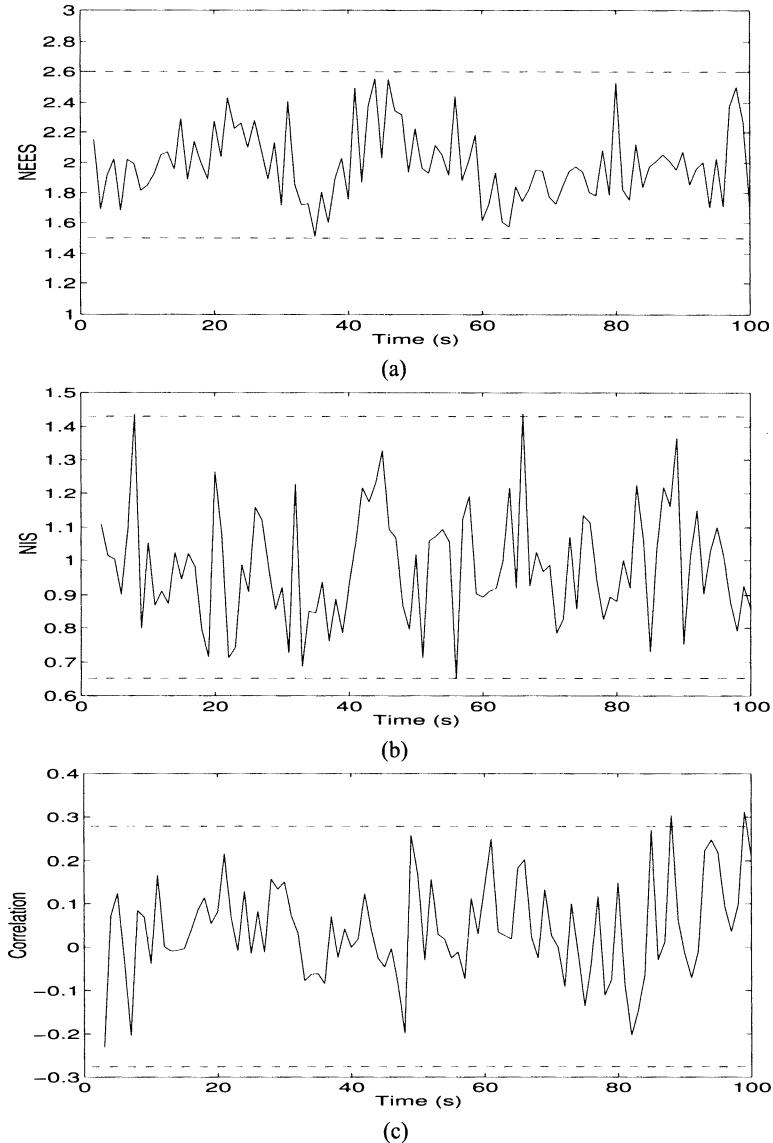


Figure 5.4.3-2: (a) Normalized state estimation error squared, (b) normalized innovation squared, (c) innovation autocorrelation, from 50 Monte Carlo runs with their 95% probability regions for $q = 1$.

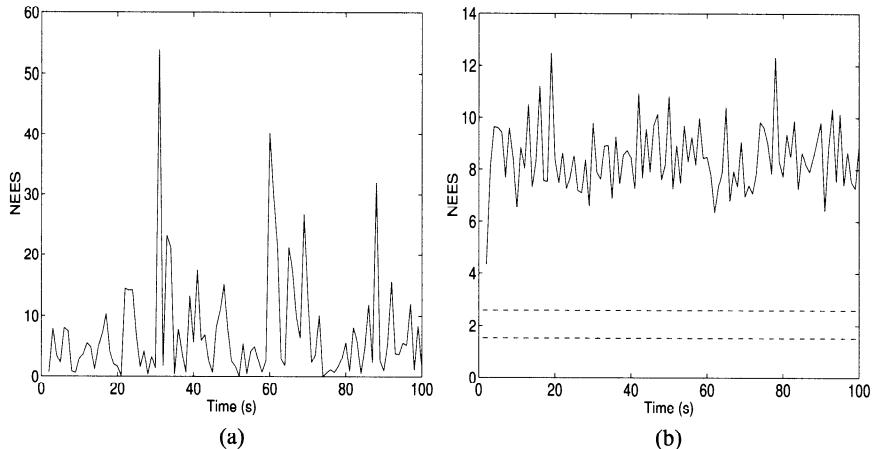


Figure 5.4.3-3: Mismatched filter: (a) normalized state estimation error squared from a single run, (b) 50-run Monte Carlo average with its 95% probability region.

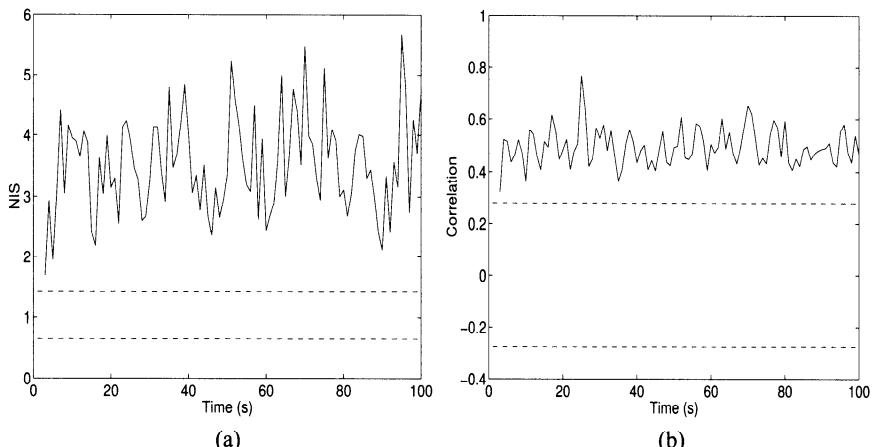


Figure 5.4.3-4: Mismatched filter: (a) normalized innovation squared, (b) innovation autocorrelation, from 50 Monte Carlo runs with their 95% probability regions.

For the mismatched filter, the result was

$$\bar{\epsilon}_v = 2.66 \quad (5.4.3-8)$$

which is clearly unacceptable.

The above illustrates how one can detect filter mismatch in a single run using suitable statistics and their probability regions.

Filter Tuning

The process noise is used in practice to model disturbances — for example, unknown inputs like target maneuvers in tracking. The procedure to match the process noise variance to suitably model such disturbances is called **filter tuning**. Since such inputs are not real noise in the probabilistic sense, it is said that the filter uses **pseudo-noise** or **artificial noise**.

The procedure for tuning is to make the filter consistent, that is, the three criteria (a)–(c) from Subsection 5.4.2 should be satisfied. While this is easier said than done, in practice one has to strive to make the filter as close to being consistent as feasible while at the same time achieving small RMS estimation errors. Note that the RMS errors are **unnormalized errors**, while the test statistics are normalized — that is, divided (in a matrix sense) by the filter-calculated variances.

This is discussed in more detail in Subsection 11.6.7 and Section 11.7, where the design of several filters for a realistic situation is illustrated.

5.4.4 Absolute Errors

In all problems one is also (actually even more) interested in the absolute performance of a filter, which is evaluated by the **RMS errors**. In the following, the expressions for these errors in **position**, **velocity**, **speed**,³ and **course**⁴ are given.

For a scalar ξ , the **RMS error from N Monte Carlo runs** with observed error $\tilde{\xi}_i$ in run i is

$$\text{RMS}(\tilde{\xi}) = \sqrt{\frac{1}{N} \sum_{i=1}^N \tilde{\xi}_i^2} \quad (5.4.4-1)$$

For a position vector $x = [\xi \ \eta]$, the RMS position error is

$$\text{RMS}(\tilde{x}) = \sqrt{\frac{1}{N} \sum_{i=1}^N (\tilde{\xi}_i^2 + \tilde{\eta}_i^2)} \quad (5.4.4-2)$$

³Speed, often confused with velocity, is the *magnitude* of the velocity, which is a vector.

⁴Course is the direction of the velocity vector, typically in the horizontal plane. The term **heading** is often used incorrectly in its place. Heading is the pointing direction of the aircraft or vessel — the direction you think you are going — while course is the direction you are actually going. This is relevant in motion in fluid media, like air or water, where winds aloft or currents can be significant.

Similarly, for the velocity $\dot{x} = [\dot{\xi} \ \dot{\eta}]$, the RMS velocity error is

$$\text{RMS}(\dot{x}) = \sqrt{\frac{1}{N} \sum_{i=1}^N (\tilde{\xi}_i^2 + \tilde{\eta}_i^2)} \quad (5.4.4-3)$$

For the speed

$$u_i = \sqrt{\dot{\xi}_i^2 + \dot{\eta}_i^2} \quad (5.4.4-4)$$

estimated as

$$\hat{u}_i = \sqrt{\tilde{\xi}_i^2 + \tilde{\eta}_i^2} \quad (5.4.4-5)$$

the RMS error is given, according to (5.4.4-1), by, with $\tilde{u}_i = u_i - \hat{u}_i$,

$$\text{RMS}(\tilde{u}) = \sqrt{\frac{1}{N} \sum_{i=1}^N \tilde{u}_i^2} \quad (5.4.4-6)$$

Similarly, for the course (measured from the η axis (North) clockwise)

$$\theta_i = \tan^{-1} \begin{bmatrix} \dot{\xi}_i \\ \dot{\eta}_i \end{bmatrix} \quad (5.4.4-7)$$

estimated as

$$\hat{\theta}_i = \tan^{-1} \begin{bmatrix} \hat{\xi}_i \\ \hat{\eta}_i \end{bmatrix} \quad (5.4.4-8)$$

the RMS error is given, according to (5.4.4-1), by, with $\tilde{\theta}_i = \theta_i - \hat{\theta}_i$,

$$\text{RMS}(\tilde{\theta}) = \sqrt{\frac{1}{N} \sum_{i=1}^N \tilde{\theta}_i^2} \quad (5.4.4-9)$$

5.4.5 Filter Consistency — Summary

Consistency testing is crucial for *estimator optimality evaluation*.

A state estimator is *consistent* if the first- and second-order moments of its estimation errors are as the theory predicts:

- Their means are zero — the estimates are unbiased.
- Their covariance matrices are as calculated by the filter.

The statistic that tests the mean and the covariance is the *normalized estimation error squared (NEES)*. This is done for

- the state and
- the innovations (measurement prediction errors).

Under the Gaussian assumption, these statistics are chi-square distributed and should be with a high probability in certain intervals — the corresponding probability regions. These tests are also called “chi-square” tests.

Bias in the estimates, if significant, or error magnitudes too large compared to the filter-calculated standard deviations will be detected by this statistic.

This test can be used as follows:

- Offline (in simulations) for state estimation errors — the truth is available for comparison.
- Offline or online (in real time) for the innovation.

The test for the *innovation’s whiteness* is the *sample autocorrelation* whose magnitude has to be below a certain threshold. This can be carried out offline or online.

These tests become very powerful; that is, they can detect *inconsistent (mismatched)* filters when used in Monte Carlo runs.

Monte Carlo runs evaluate *ensemble averages* of the test statistics. The averages obtained from *independent runs* — with *all the random variables independent from run to run* — decrease the variability and therefore increase the power of the tests.

Acceptance regions — *upper and lower bounds* — within which the test statistics should be for consistent filters, have been established.

If the normalized error statistic exceeds the upper bound, then

- there is *significant bias* in the estimates, or
- the *errors are too large* compared to the filter-calculated covariance, or
- the *covariance is too small*.

In this case the filter is *optimistic*.

If the normalized error statistic is below the lower bound, then the *covariance is too large*. In this case the filter can be said to be *pessimistic*.

For *real-time filter performance monitoring*, one can implement tests based on *time averaging* for innovation magnitude and whiteness.

5.5 INITIALIZATION OF STATE ESTIMATORS

5.5.1 Initialization and Consistency

A state estimation filter is called consistent if its estimation errors are “commensurate” or “compatible” with the filter-calculated covariances. At *initialization*, it is just as important that the covariance associated with the initial estimate reflects realistically its accuracy.

According to the Bayesian model, the *true initial state is a random variable*, assumed to be normally distributed with a known mean — the initial estimate

— and a given covariance matrix

$$x(0) \sim \mathcal{N}[\hat{x}(0|0), P(0|0)] \quad (5.5.1-1)$$

The norm (“chi-square”) test for the initial estimation error is

$$\tilde{x}(0|0)'P(0|0)^{-1}\tilde{x}(0|0) \leq c_1 \quad (5.5.1-2)$$

where c_1 is the upper limit of the, say, 95% confidence region from the chi-square distribution with the corresponding number of degrees of freedom.

Sometimes one has to choose the initial covariance — the “choice” has to be such that (5.5.1-2) is satisfied. A large error in the initial estimate, if the latter is deemed highly accurate, will persist a long time because it leads to a low filter gain and thus the new information from the measurements receives weighting that is too low. (See problem 5-6.)

For the one-dimensional case, (5.5.1-2) can be stated as follows: The initial error should be not more than, say, two times the associated standard deviation — this is called a “ 2σ ” error. If the initial variance is such that the initial error is “ 1σ ” and the subsequent measurements are accurate, then the initial error will decrease rapidly.

5.5.2 Initialization in Simulations

As discussed before, the Bayesian model for initialization is as follows:

- The initial state is a random variable.
- The (prior) pdf of the initial state is known and assumed Gaussian.

Thus, in simulations one should generate the initial state with a random number generator according to (5.5.1-1). The initial estimate is, again according to the Bayesian model, a *known quantity* together with the initial covariance matrix.

This approach, while rigorous according to the assumptions of the filter, is not so appealing because the true initial state, which defines the *scenario*, will be different in each simulation.

A more appealing approach is the following:

- Choose the initial true state.
- Generate the initial estimate according to

$$\hat{x}(0|0) \sim \mathcal{N}[x(0), P(0|0)] \quad (5.5.2-1)$$

The above relies on the fact that, switching the roles of the initial state $x(0)$ and the initial estimate $\hat{x}(0|0)$, one has the algebraic identity

$$\begin{aligned} p[x(0)|\hat{x}(0|0)] &= \mathcal{N}[x(0); \hat{x}(0|0), P(0|0)] \\ &= \mathcal{N}[\hat{x}(0|0); x(0), P(0|0)] \\ &= p[\hat{x}(0|0)|x(0)] \end{aligned} \quad (5.5.2-2)$$

in view of the special form of the normal distribution. In other words, the *scenario* $x(0)$ is *fixed* and the *initial condition* $\hat{x}(0|0)$ of the filter is *random*.

5.5.3 A Practical Implementation in Tracking

The practical implementation of the initialization (5.5.2-1) can be done as follows. Consider two state components, say, position ξ and velocity $\dot{\xi}$ in a given coordinate. If only position measurements

$$z(k) = \xi(k) + w(k) \quad (5.5.3-1)$$

are available, then for the true values $\xi(k)$, $k = -1, 0$, one generates the corresponding measurement noises, say

$$w(k) \sim \mathcal{N}[0, R] \quad (5.5.3-2)$$

Then, denoting by T the sampling interval, one has

$$\hat{\xi}(0|0) = z(0) \quad (5.5.3-3)$$

$$\hat{\dot{\xi}}(0|0) = \frac{z(0) - z(-1)}{T} \quad (5.5.3-4)$$

and the corresponding 2×2 block of the initial covariance matrix is then⁵

$$P(0|0) = \begin{bmatrix} R & R/T \\ R/T & 2R/T^2 \end{bmatrix} \quad (5.5.3-5)$$

This method, called *two-point differencing*, guarantees consistency of the initialization of the filter, which starts updating the state at $k = 1$.

If several (Monte Carlo) runs are made, then the same initialization procedure has to be followed with new (*independent*) noises generated in every run according to (5.5.3-2). “Reuse” of the same initial conditions in Monte Carlo runs will lead to biased estimates (see problem 5-6).

Remark

The above amounts to a *first-order polynomial fitting* of the first two measurements. If (and only if) there are *significant* higher derivatives, then one should use more than two points and a *higher-order polynomial fitting* via LS is to be carried out.

One-Point Initialization

In some problems the variance of the velocity obtained from two-point differencing might be much larger than some known bound on the speed. In such a case one can initialize the position from a single observation and use zero as initial estimate for the velocity with an associated standard deviation equal to, say, half of the known maximum speed.

Note, however, that this will cause a bias in Monte Carlo runs for a fixed scenario, since the initial velocity error is fixed rather than random and zero-mean. Appropriate care should be exercised when interpreting the results.

⁵Neglecting the process noise (see problem 5-13).

Automatic Start-up Without Initial Estimate

A general and exact technique for start-up of a filter without initial estimates via the *information filter* form of the KF is discussed in Subsection 7.2.3.

5.5.4 Filter Initialization — Summary

The error in the initial state estimate has to be *consistent* with the initial state covariance. When an initial covariance is “chosen,” it should be such that the error is at most 2 times the corresponding standard deviation.

According to the Bayesian assumptions in the Kalman filter,

- *the true initial state is a random variable and*
- *the initial estimate is a fixed known quantity.*

In simulations one can reverse this point of view by fixing the initial state (the scenario) and generating the initial estimate with a random number generator as follows:

The initial estimate is generated with mean equal to the true initial state and with covariance equal to the initial state covariance.

In practice, the initialization can be done from two consecutive position measurements by

1. using the latest measurement as initial position estimate,
2. differencing them to obtain the velocity estimate,
3. calculating the corresponding covariance matrix.

This amounts to a first-order polynomial fit. Higher-order polynomial fits can also be used for initializing the estimation of states that have higher derivatives.

This initialization method should also be followed in Monte Carlo simulations where, with *noises independent from run to run*, one will then obtain initial errors also independent from run to run.

In some circumstances, one-point initialization is appropriate.

There is a general procedure to initialize filters, to be discussed later.

5.6 SENSITIVITY

The derivation of the optimal filter gain in (5.2.3-11) assumes that the mathematical description of the system given by (5.2.1-1)–(5.2.1-4) is exact. That is, the system matrices and the noise statistics used in the Kalman filter model match those of the system. It also assumes that the chosen filter state fully describes the system state.

In practice, exact information about the system is not available, and therefore the model assumed by the filter designer is different from the truth. In some cases, in order to reduce computational load, the designer might implement a reduced-order filter with fewer states than in the “full model” or use suboptimal gains which are more easily computed than the optimal one.

In view of the modeling errors and implementation approximations, **sensitivity analysis**, where the sensitivity of the filter to any such mismatch is determined, is in order.

5.6.1 Model Mismatch

Let the model assumed by the filter be described by

$$x_f(k+1) = F_f(k)x_f(k) + G_f(k)u_f(k) + v_f(k) \quad (5.6.1-1)$$

and let the measurement process be described by

$$z_f(k) = H_f(k)x_f(k) + w_f(k) \quad (5.6.1-2)$$

where $x_f(k)$ is the filter state and the mutually independent model noise components $v_f(k)$ and $w_f(k)$ are

$$v_f(k) \sim \mathcal{N}(0, Q_f(k)) \quad (5.6.1-3)$$

and

$$w_f(k) \sim \mathcal{N}(0, R_f(k)) \quad (5.6.1-4)$$

Clearly, these are different from the actual system equations defined by the system matrices $F(k)$, $G(k)$ and $H(k)$ and by the error statistics $Q(k)$ and $R(k)$. As a result, the “optimal” gain $W(k+1)$ evaluated by the filter, which in this case is given by

$$\begin{aligned} W(k+1) &= P_f(k+1|k)H_f(k+1)' \\ &\cdot [H_f(k+1)P_f(k+1|k)H_f(k+1)' + R_f(k+1)]^{-1} \end{aligned} \quad (5.6.1-5)$$

which yields the associated **filter-calculated covariance matrices**

$$P_f(k+1|k) = F_f(k)P_f(k|k)F_f(k)' + Q_f(k) \quad (5.6.1-6)$$

and, using the Joseph form of the covariance update (5.2.3-18), which is valid for arbitrary filter gain,

$$\begin{aligned} P_f(k+1|k+1) &= [I - W(k+1)H_f(k+1)]P_f(k+1|k) \\ &\cdot [I - W(k+1)H_f(k+1)]' + W(k+1)R_f(k+1)W(k+1)' \end{aligned} \quad (5.6.1-7)$$

is not really optimal for the system given by (5.2.1-1)–(5.2.1-4), but *only* for the filter model assumed in (5.6.1-1)–(5.6.1-4). With these values, the state estimate $\hat{x}(k+1|k+1)$ is given by

$$\hat{x}(k+1|k) = F_f(k)\hat{x}(k|k) + G_f(k)u_f(k) \quad (5.6.1-8)$$

$$\begin{aligned} \hat{x}(k+1|k+1) &= \hat{x}(k+1|k) + W(k+1) \\ &\cdot [z(k+1) - H_f(k+1)\hat{x}(k+1|k)] \end{aligned} \quad (5.6.1-9)$$

To analyze the evolution of the error between the actual state $x(k)$ and the filter estimated state $\hat{x}(k|k)$, define the following:

$$\tilde{x}(k+1|k) = x(k+1) - \hat{x}(k+1|k) \quad (5.6.1-10)$$

$$\tilde{x}(k+1|k+1) = x(k+1) - \hat{x}(k+1|k+1) \quad (5.6.1-11)$$

which are the actual errors in the predicted and updated state, respectively, with associated **MSE matrices**⁶ $P(k+1|k)$ and $P(k+1|k+1)$. Also define the following model mismatches:

$$\tilde{F}(k) = F(k) - F_f(k) \quad (5.6.1-12)$$

$$\tilde{H}(k) = H(k) - H_f(k) \quad (5.6.1-13)$$

Then the state prediction error can be written as⁷

$$\begin{aligned} \tilde{x}(k+1|k) &= x(k+1) - \hat{x}(k+1|k) = F(k)x(k) + v(k) - F_f\hat{x}(k|k) \\ &= F_f(k)\tilde{x}(k|k) + \tilde{F}(k)x(k) + v(k) \end{aligned} \quad (5.6.1-14)$$

Combining this with (5.2.1-1), one can define an augmented state equation

$$\begin{bmatrix} x(k+1) \\ \tilde{x}(k+1|k) \end{bmatrix} = \begin{bmatrix} F(k) & 0 \\ \tilde{F}(k) & F_f(k) \end{bmatrix} \begin{bmatrix} x(k) \\ \tilde{x}(k|k) \end{bmatrix} + \begin{bmatrix} I_{n_x \times n_x} \\ I_{n_x \times n_x} \end{bmatrix} v(k) \quad (5.6.1-15)$$

where $I_{n_x \times n_x}$ is an $n_x \times n_x$ identity matrix.

Since one is interested in the actual error statistics in the presence of modeling errors, namely, $P(k+1|k)$ and $P(k+1|k+1)$, define the following **mean-square value (MSV) matrices**, also called **correlation matrices**⁸:

$$E \left\{ \begin{bmatrix} x(k) \\ \tilde{x}(k|k) \end{bmatrix} \begin{bmatrix} x(k) \\ \tilde{x}(k|k) \end{bmatrix}' \right\} \triangleq \begin{bmatrix} X(k) & C(k|k)' \\ C(k|k) & P(k|k) \end{bmatrix} \quad (5.6.1-16)$$

and

$$E \left\{ \begin{bmatrix} x(k+1) \\ \tilde{x}(k+1|k) \end{bmatrix} \begin{bmatrix} x(k+1) \\ \tilde{x}(k+1|k) \end{bmatrix}' \right\} \triangleq \begin{bmatrix} X(k+1) & C(k+1|k)' \\ C(k+1|k) & P(k+1|k) \end{bmatrix} \quad (5.6.1-17)$$

In the above, X is the MSV matrix of the state vector, P is an estimation MSE, and C is a cross-term.

⁶In can be shown that, usually, the actual errors $\tilde{x}(k+1|k)$ and $\tilde{x}(k+1|k+1)$ are not zero-mean in the presence of modeling errors, resulting in biased estimates. Thus, the term MSE matrices of these quantities is used in place of covariance matrices.

⁷Without loss of generality, in the sequel it is assumed that $u(k) = u_f(k) = 0$. The derivation can be easily generalized for the more general case, resulting in longer equations.

⁸Using the terminology from stochastic processes, this is the autocorrelation (or correlation) of the corresponding random sequence at a single point in time, or its MS value, thus the term MSV matrix.

Then

$$\begin{bmatrix} X(k+1) & C(k+1|k)' \\ C(k+1|k) & P(k+1|k) \end{bmatrix} = \begin{bmatrix} F(k) & 0 \\ \tilde{F}(k) & F_f(k) \end{bmatrix} \begin{bmatrix} X(k) & C(k|k)' \\ C(k|k) & P(k|k) \end{bmatrix} \cdot \begin{bmatrix} F(k) & 0 \\ \tilde{F}(k) & F_f(k) \end{bmatrix}' + \begin{bmatrix} Q(k) & Q(k) \\ Q(k) & Q(k) \end{bmatrix} \quad (5.6.1-18)$$

which yields

$$\begin{aligned} P(k+1|k) &= F_f(k)P(k|k)F_f(k)' + Q(k) + \tilde{F}(k)X(k)\tilde{F}(k)' \\ &\quad + F_f(k)C(k|k)\tilde{F}(k)' + \tilde{F}(k)C(k|k)'F_f(k)' \end{aligned} \quad (5.6.1-19)$$

Similarly, the state update error defined in (5.6.1-11) is given by

$$\begin{aligned} \tilde{x}(k+1|k+1) &= x(k+1) - \hat{x}(k+1|k) - W(k+1)[z(k+1) \\ &\quad - H_f(k+1)\hat{x}(k+1|k)] \end{aligned} \quad (5.6.1-20)$$

which can be rewritten as

$$\begin{bmatrix} x(k+1) \\ \tilde{x}(k+1|k+1) \end{bmatrix} = K(k+1) \begin{bmatrix} x(k+1) \\ \tilde{x}(k+1|k) \end{bmatrix} + \begin{bmatrix} 0 \\ -W(k+1) \end{bmatrix} w(k+1) \quad (5.6.1-21)$$

where

$$K(k+1) \triangleq \begin{bmatrix} I & 0 \\ -W(k+1)\tilde{H}(k+1) & I - W(k+1)H_f(k+1) \end{bmatrix} \quad (5.6.1-22)$$

with associated MSV matrix

$$\begin{bmatrix} X(k+1) & C(k+1|k+1)' \\ C(k+1|k+1) & P(k+1|k+1) \end{bmatrix} = K(k+1) \cdot \begin{bmatrix} X(k+1) & C(k+1|k)' \\ C(k+1|k) & P(k+1|k) \end{bmatrix} K(k+1)' + \begin{bmatrix} 0 & 0 \\ 0 & W(k+1)R(k+1)W(k+1)' \end{bmatrix} \quad (5.6.1-23)$$

Then the actual updated MSE recursion equation is given by

$$\begin{aligned} P(k+1|k+1) &= [I - W(k+1)H_f(k+1)]P(k+1|k) \\ &\quad [I - W(k+1)H_f(k+1)]' + W(k+1)R(k+1)W(k+1)' \\ &\quad + W(k+1)\tilde{H}(k+1)X(k+1)\tilde{H}(k+1)'W(k+1)' \\ &\quad - [I - W(k+1)H_f(k+1)]C(k+1|k)\tilde{H}(k+1)'W(k+1)' \\ &\quad - W(k+1)\tilde{H}(k+1)C(k+1|k)'[I - W(k+1)H_f(k+1)] \end{aligned} \quad (5.6.1-24)$$

together with

$$\begin{aligned} C(k+1|k+1) &= [I - W(k+1)H_f(k+1)] C(k+1|k) \\ &\quad - W(k+1)\tilde{H}(k+1)X(k+1) \end{aligned} \quad (5.6.1-25)$$

The actual MSE matrices $P(k+1|k)$ and $P(k+1|k+1)$ given by (5.6.1-19) and (5.6.1-24) can be compared with the filter calculated covariance matrices $P_f(k+1|k)$ and $P_f(k+1|k+1)$ in (5.6.1-6) and (5.6.1-7), respectively. It can be seen that the actual MSE matrices are “larger” than the filter calculated covariance matrices — the increased uncertainty accounts for the modeling errors. Also, in the absence of modeling errors, the filter calculated covariances are the same as the actual MSE.

Note that the calculation of actual MSE matrices requires the knowledge of true system matrices $F(k)$, $H(k)$, $Q(k)$ and $R(k)$, which are not always available. When true system matrices are not available, their nominal values, which represent an “average” scenario for the problem under consideration, can be used.

The auxiliary MSV matrices $X(k)$, $C(k+1|k)$ and $C(k|k)$ are evaluated only to update the actual MSE matrices $P(k+1|k)$ and $P(k|k)$. Their initialization is carried out as follows:

$$X(0) = E[x(0)x(0)'] = \hat{x}(0|0)\hat{x}(0|0)' + P(0|0) \quad (5.6.1-26)$$

and

$$C(0|0) = P(0|0) = P_0 \quad (5.6.1-27)$$

where P_0 is the actual MSE of the initial state estimate. For the filter covariance calculation the initial state covariance is $P_f(0|0)$, which can be different from $P(0|0)$. This is relevant, e.g., for the evaluation of the effect of an incorrect initial covariance (see problem 5-6).

Figure 5.6.1-1 shows the flowchart of a cycle in a model-mismatched Kalman filter together with the evaluation of the actual MSE matrices.

Remark

Another point of interest is the asymptotic behavior of the actual errors. For example, by combining (5.6.1-14) and (5.6.1-21), the evolution of the updated state estimation error can be written as

$$\begin{aligned} \tilde{x}(k+1|k+1) &= [I - W(k+1)H_f(k+1)] F_f(k)\tilde{x}(k|k) \\ &\quad - W(k+1)\tilde{H}(k+1)x(k+1) \\ &\quad + [I - W(k+1)H_f(k+1)] [\tilde{F}(k)x(k) + v(k)] \end{aligned} \quad (5.6.1-28)$$

It can be noted that the above error system depends on the true state $x(k)$ in a multiplicative manner, which indicates that the error can become unbounded for unbounded true state. Thus, modeling errors can result in unbounded estimation errors.

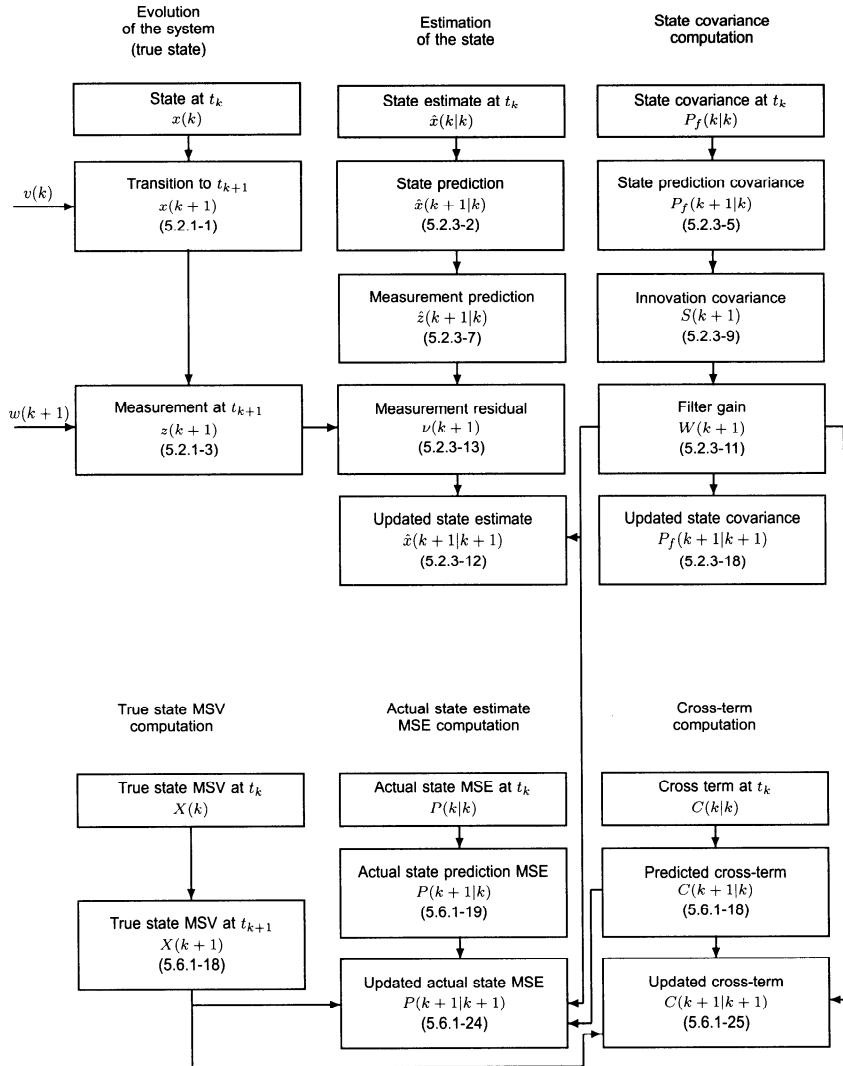


Figure 5.6.1-1: One cycle in a model-mismatched Kalman filter.

5.6.2 Reduced-Order Filters

A special case of modeling inaccuracy is when some components of the actual state vector are ignored and a Kalman filter to estimate the retained state components is implemented. This reduced-order filter may be due to incomplete knowledge about the additional states or because of the need for reduced computational load. For example, the “nearly constant velocity” model (see Section 6.4), which assumes that the target under track evolves at almost a constant velocity, is commonly used in tracking applications. This effectively ignores the (unknown) higher-order components, for example, the acceleration and jerk.

Let the size of the true state vector $x(k)$ in (5.2.1-1) be n and that of the reduced order state used in the filter be n_r . Without loss of generality, assume that filter state $x_r(k)$ consists of the first n_r elements of $x(k)$. That is,

$$x_r(k) = \begin{bmatrix} I_{n_r \times n_r} & \mathbf{0}_{n_r \times n-n_r} \end{bmatrix} x(k) = \mathcal{T}x(k) \quad (5.6.2-1)$$

Then

$$\begin{aligned} x_r(k+1) &= \mathcal{T}x(k+1) = \mathcal{T}F(k)\mathcal{T}'x_r(k) + \mathcal{T}v(k) \\ &= F_r(k)x_r(k) + v_r(k) \end{aligned} \quad (5.6.2-2)$$

and, since, typically, the neglected state components do not appear in the measurements,

$$z(k) = H_r(k)x_r(k) + w(k) = H_r(k)\mathcal{T}x(k) + w(k) \quad (5.6.2-3)$$

Note that the full-dimensional state estimate (with the last $n - n_r$ components zero) is

$$\hat{x}_f = \mathcal{T}'\hat{x}_r \quad (5.6.2-4)$$

Its state transition matrix is, using (5.6.2-1) and (5.6.2-3) in (5.6.2-2), given by

$$F_f(k) = \mathcal{T}'F_r(k)\mathcal{T} \quad (5.6.2-5)$$

and its measurement matrix is, from (5.6.2-3),

$$H_f(k) = H_r(k)\mathcal{T} \quad (5.6.2-6)$$

The prediction and the update filter errors, which were defined in (5.6.1-10) and (5.6.1-11) for general modeling errors, respectively, are given by

$$\begin{aligned} \tilde{x}(k+1|k) &= x(k+1) - \hat{x}_f(k+1|k) \\ &= x(k+1) - \mathcal{T}'\hat{x}_r(k+1|k) \end{aligned} \quad (5.6.2-7)$$

$$\begin{aligned} \tilde{x}(k+1|k+1) &= x(k+1) - \hat{x}_f(k+1|k+1) \\ &= x(k+1) - \mathcal{T}'\hat{x}_r(k+1|k+1) \end{aligned} \quad (5.6.2-8)$$

Also, define the model mismatches, similar to (5.6.1-12) and (5.6.1-13), as

$$\tilde{F}(k) = F(k) - F_f(k) = F(k) - T' F_r(k) T \quad (5.6.2-9)$$

$$\tilde{H}(k) = H(k) - H_f(k) = H(k) - H_r(k) T \quad (5.6.2-10)$$

With these values, the actual prediction and update MSE matrices can be obtained by substituting (5.6.2-5) and (5.6.2-6) into (5.6.1-18) – (5.6.1-25).

In deciding whether to use a reduced-order filter, one needs to be satisfied that the difference between the resulting actual error statistics and the full-state filter's error statistics is indeed acceptable.

Decoupling

Reduced-order filtering can be used to decouple the state estimates and implement the estimator using two or more reduced-order filters. This is especially useful when $F(k)$ has a block-diagonal structure, or is approximately block-diagonal. For example, assume that the state vector $x(k)$ (of size $n_1 + n_2$) can be written in terms of two decoupled state subvectors $x_1(k)$ and $x_2(k)$ (of sizes n_1 and n_2 , respectively) as

$$\begin{aligned} x(k+1) &= \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} \\ &= \begin{bmatrix} F_1(k) & 0 \\ 0 & F_2(k) \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} v_1(k) \\ v_2(k) \end{bmatrix} \end{aligned} \quad (5.6.2-11)$$

and

$$\begin{aligned} z(k+1) &= \begin{bmatrix} z_1(k+1) \\ z_2(k+1) \end{bmatrix} \\ &= \begin{bmatrix} H_1(k) & 0 \\ 0 & H_2(k) \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} w_1(k) \\ w_2(k) \end{bmatrix} \end{aligned} \quad (5.6.2-12)$$

The estimator for the above decoupled system can be implemented with two filters, one estimating $x_1(k)$ and the other $x_2(k)$. The motivation for decoupling comes from the “divide-and-conquer” philosophy — the decoupled system’s computations complexity is of the order $n_1^3 + n_2^3$ as opposed to $(n_1 + n_2)^3$.

An application of decoupled filtering is found in air traffic surveillance, where, usually, an aircraft’s motion in the horizontal place is independent of its vertical motion. In that case, the aircraft motion can be tracked using one filter for estimating the horizontal state (for example, positions and velocities in the North and East directions) and another for the vertical state (the altitude and the vertical velocity; see, e.g., [Yeddanapudi97, Wang99]).

5.6.3 Suboptimal Gains

Another common approximation is to use suboptimal filter gains, often precomputed or fixed ones, in order to reduce the computation load in evaluating the optimal gain $W(k)$ for each update. While the optimal gain can be precomputed offline as shown in Fig. 5.2.4-1, storing the entire gain history may not be practicable due to high storage requirements.

Two possible approximations are either to use a fixed gain, usually the steady-state gain W_∞ , or to use a piecewise linear approximation of the optimal gain. In the former case, the covariance update in (5.6.1-7), which is valid for any arbitrary gain, becomes

$$P_f(k+1|k+1) = [I - W_\infty H_f(k+1)] P_f(k|k) [I - W_\infty H_f(k+1)]' + W_\infty R_f(k+1) W_\infty' \quad (5.6.3-1)$$

The steady-state gain W_∞ can be evaluated by solving first for the steady-state covariance P_∞ in (5.2.5-2) and then using (5.2.3-17) to find W_∞ . Since the transient characteristics may not be acceptable when a fixed steady-state gain is used, one can use a piecewise linear gain as shown in Figure 5.6.3-1. Another motivation for this is that the steady-state covariance and gain do not exist for systems with time-varying $F(k)$, $R(k)$ or $H(k)$.

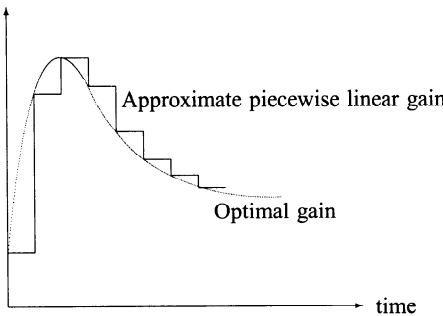


Figure 5.6.3-1: Piecewise linear approximation for filter gain.

The selection strategy for using a particular approximate state model or filter gain can be formulated as an optimization problem where the “cost” of using such approximation is minimized. For example, a suitable piecewise linear gain sequence can be chosen by minimizing the average quadratic error from the optimal gain [Gelb74].

5.6.4 Examples of Modeling Errors and Filter Approximations

Consider the evolution of the state $x(k)$, consisting of position and velocity, of a target, where the position is measured at a constant revisit interval T . The

target's state evolution is given by

$$x(k+1) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} v(k), \quad v(k) \sim \mathcal{N}(0, 1) \quad (5.6.4-1)$$

with initial state $x(0) \sim \mathcal{N}([0 \ -10]', I)$ and the measured position is

$$z(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k) + w(k), \quad w(k) \sim \mathcal{N}(0, 1) \quad (5.6.4-2)$$

In the following examples, the effects of modeling or programming errors and implementation approximations are analyzed using DynaEst™.

Approximate Model

The target motion is modeled as exactly a constant velocity ($q_f = 0$), and a Kalman filter is implemented accordingly. In this case, the filter state equation is

$$x_f(k+1) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x_f(k) \quad (5.6.4-3)$$

Figure 5.6.4-1a shows the true and estimated states of the target under track. The actual and the filter calculated estimation variances are shown in Fig. 5.6.4-1b. It can be seen that while the filter calculated variances incorrectly converge to zero, the actual covariances continue to diverge. This is a manifestation of the mismatch between the true target state given by (5.6.4-1) and the filter model given by (5.6.4-3).

Reduced-Order Model

The target state is estimated using a reduced-order filter. In this case, the target's velocity state component is ignored and only the position is estimated. That is, the filter model for the target state is

$$x_f(k+1) = x_f(k) + v_f(k), \quad v(k) \sim \mathcal{N}(0, q_f) \quad (5.6.4-4)$$

For this model, the state reduction matrix T , defined in (5.6.2-1), is given by $T = [1 \ 0]$.

Figures 5.6.4-2 and 5.6.4-3 show the state estimates and error variances (obtained using DynaEst™) for $q_f = 1$ and $q_f = 4$, respectively. From these one can notice the following: (1) In both cases the actual variances do not match the filter calculated ones; (2) Increasing q_f improves the estimation results. The increased q_f serves as "artificial process noise" compensating for the reduced order model which ignores the velocity component.

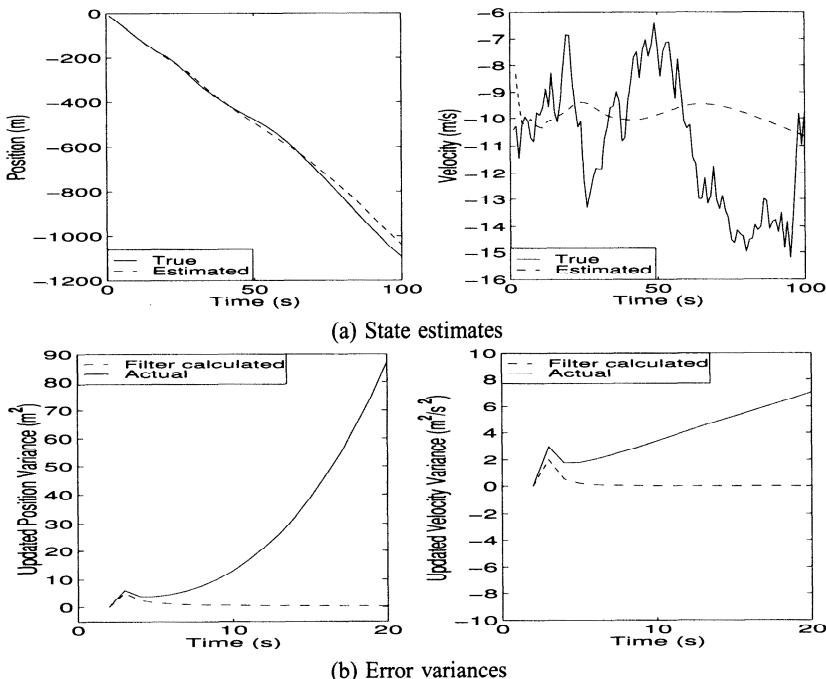


Figure 5.6.4-1: State estimates and error variances for the mismatched model with $q_f = 0$.

Fixed Filter Gain

A fixed gain of $W_f = [0.8 \ 0.5]'$ is used in place of the optimal one. The optimal and the fixed gains are shown in Fig. 5.6.4-4a. The corresponding variances are shown in Fig. 5.6.4-4b. It can be seen that the variances with incorrect gain are slightly higher than the optimal values. In other problems the difference can be larger.

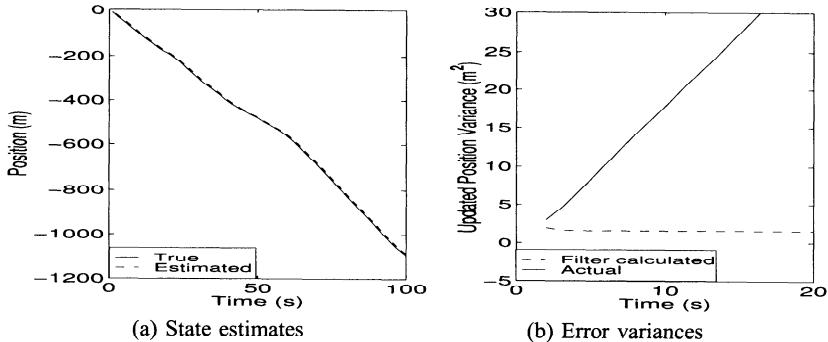


Figure 5.6.4-2: State estimates and error variances for the reduced-order filter with $q_f = 1$.

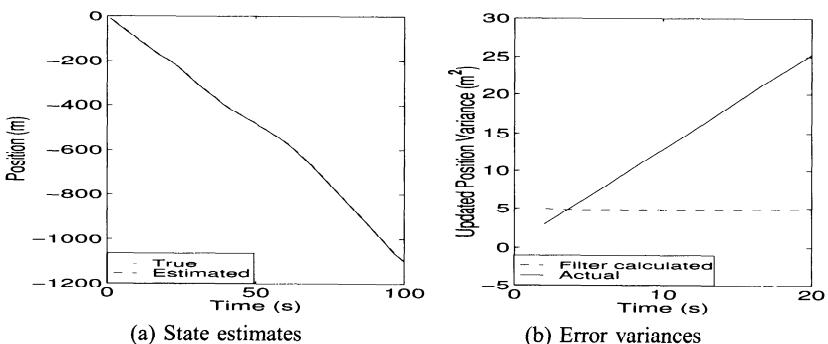


Figure 5.6.4-3: State estimates and error variances for the reduced-order filter with $q_f = 4$.

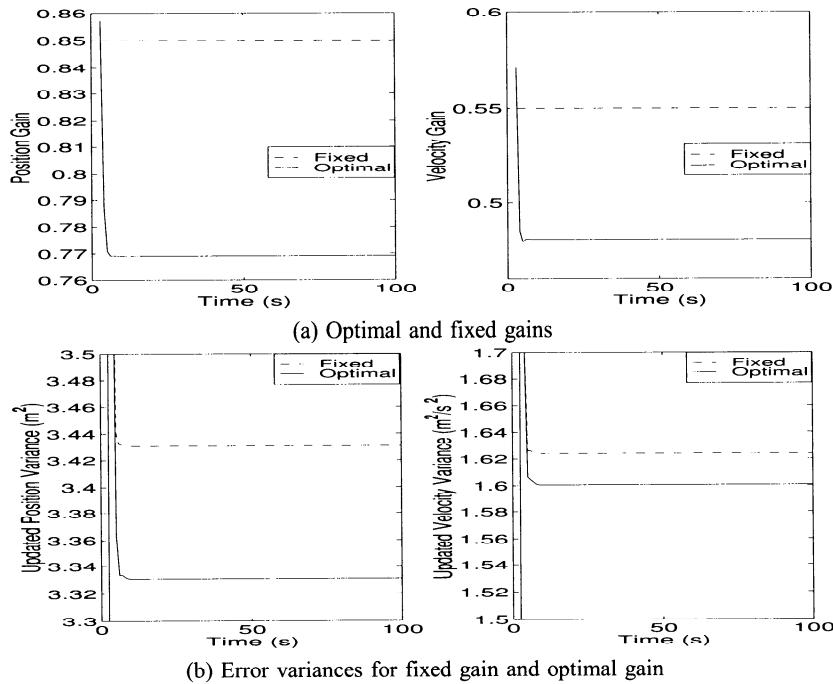


Figure 5.6.4-4: Error variances with suboptimal gains.

5.7 NOTES AND PROBLEMS

5.7.1 Bibliographical Notes

The Kalman filter originated with the work of Kalman and Bucy [Kalman60, Kalman61]. The idea of recursive estimation appeared earlier in the work of Swerling [Swerling59]. The topic of the Kalman filtering is covered in many texts, for instance, [Meditch69, Sage71, Gelb74, Anderson79, Kailath81, Maybeck79, Maybeck82, Balakrishnan84, Lewis86]. The proof of the sequential estimation for dynamic systems as a direct extension of the static case presented in Section 5.2 is simpler than the many different proofs in the literature. Issues of observability and controllability and their implications on the Kalman filter are discussed in detail in [Anderson79, Kailath81].

The concept of consistency of a state estimator (Section 5.4) has been mentioned in the literature under the name of “covariance matching.” The discussion of Section 5.4 is based on [Bar-Shalom83]. The lack of consistency of a Kalman filter has been called “divergence” (that is how severe it was in some cases) and has been the subject of extensive investigations [Sorenson85]. Sensitivity analysis of mismatched filters and the use of reduced-order models are discussed in detail in [Gelb74, Lewis86].

The technique of initialization of filters presented in Section 5.5 has been known for many years, but overlooked in many instances.

The models considered here had all known “parameters” — the matrices F , G , H , Q , and R . Techniques for the “identification” of these system parameters can be found in [Ljung87].

Treatment of a constant bias in recursive filtering using state augmentation is discussed in [Friedland69].

5.7.2 Problems

5-1 Filter implementation and consistency checking. Consider the scalar system

$$x(k+1) = fx(k) + u(k) + v(k)$$

where $u(k) = 1$ and $v(k) \sim \mathcal{N}(0, q)$ and white, with measurement

$$z(k) = hx(k) + w(k)$$

where $w(k) \sim \mathcal{N}(0, r)$ and white. The initial condition for the system is $x(0)$.

- Find the expression of the steady-state variance

$$P_\infty = \lim_{k \rightarrow \infty} P(k|k)$$

of the estimated state $\hat{x}(k|k)$.

- With the parameters of the problem $f = h = 1$, $q = 0.01$, $r = 1$, simulate a trajectory using a random number generator starting from $x(0) = 0$ for $k = 1, \dots, 50$.
- Let the initial estimate of the state be $\hat{x}(0|0) = z(0)/h$. Determine the corresponding $P(0|0)$ as a function of the parameters of the problem (h, r) .

4. For the values given in 2, estimate the state up to $k = 50$, starting as in 3. List the following:

$$k, x(k), v(k), w(k), z(k), \hat{x}(k|k-1), P(k|k-1), \hat{z}(k|k-1), S(k), \\ \nu(k), \nu(k)/\sqrt{S(k)}, W(k), \hat{x}(k|k), P(k|k), \tilde{x}(k|k)/\sqrt{P(k|k)}$$

5. Compare the values of $P(k|k)$ from 4 to the result of 1.
6. List $P(k|k)$ for $k = 0, 1, \dots, 50$ for the following values of the initial variance: $P(0|0) = 0, 1, 10$.
7. One desires to run the filter as in (4) with the various values of $P(0|0)$ as in (6). What should be changed in the filter simulation?

5-2 Random number generator testing. Describe the tests for a “correct” $\mathcal{N}(0, 1)$ random number generator from which we have n numbers. Indicate the distributions and probability regions for the

1. Sample mean.
2. Sample variance.
3. Sample correlation.

5-3 Covariance of the state versus covariance of the estimation error. Prove that, with $\tilde{x} \triangleq x - E[x|z]$, one has $\text{cov}[x|z] = \text{cov}[\tilde{x}|z]$.

5-4 Asymptotic distribution of the sample correlation. Show that the sample correlation (5.4.2-12) tends to $\mathcal{N}(0, 1/N)$. *Hint:* Use the law of large numbers (where convenient) and the central limit theorem.

5-5 MSE of the state for an update with an arbitrary gain.

1. Prove that the Joseph form covariance update (5.2.3-18) holds for *arbitrary* gain at time $k + 1$. *Hint:* Write the propagation equation of the error from $\tilde{x}(k+1|k)$ to $\tilde{x}(k+1|k+1)$.
2. State the condition for its stability.
3. Check this stability condition on the results of problem 5-1.

5-6 Initialization of a filter (how NOT to do it — a real story). A tracking filter was initialized in a set of Monte Carlo runs with the target’s initial range of 80 kft, initial range estimate of 100 kft and initial range variance of 10^6 ft².

1. Characterize the assumed quality of the initial estimate.
2. How will the average estimation error over the Monte Carlo runs behave?
3. You are a “young Turk” engineer who wants to prove mastery of estimation. Suggest a simple fix to the above initialization procedure that involves changing only the initial variance.

5-7 Orthogonality of the innovations to the state prediction. Show that

$$\nu(k) \perp \hat{x}(j|k-1) \quad \forall j > k-1$$

5-8 Orthogonality of estimation error to previous estimates. Show that

$$\tilde{x}(i|k) \perp \hat{x}(i|j) \quad \forall j \leq k$$

5-9 Alternative derivation of the Kalman filter gain. Show that the minimization of the trace of (5.2.3-18) with respect to the filter gain yields (5.2.3-11). *Hint:* Use the formulas from problem 1-10.

5-10 State estimation errors' autocorrelation. Prove that the state estimation errors are not white:

$$E[\tilde{x}(k+1|k+1)\tilde{x}(k|k)'] = [I - W(k+1)H(k+1)]F(k)P(k|k)$$

5-11 Kalman filter with nonzero noise means. Derive the Kalman filter equations for the formulation from Subsection 5.2.1 with the following modifications:

$$E[v(k)] = \bar{v}(k) \quad E[w(k)] = \bar{w}(k)$$

where the nonzero noise means are known. Their known covariances are Q and R . All the remaining assumptions are the same. Provide the derivations only for the equations that will be different than those in Subsection 5.2.4. Indicate which equations are not modified and why.

5-12 Bias in the measurements. Consider the problem from Section 5.3 with the modification that the measurement noise has an *unknown mean* \bar{w} (the sensor bias). Append this to the state as an extra component assuming it to be constant in time.

1. Indicate the new state space representation with the augmented state and *zero-mean noises* (specify the matrices F , Γ , and H).
2. What happens if we run a Kalman filter on this representation? Can one estimate this sensor bias? Justify mathematically your answer.

5-13 Effect of the process noise on the initialization. Modify (5.5.3-5) to account for the process noise.

5-14 Gating. Given a standard state estimation problem with the following specifics: the state prediction covariance matrix $P(k+1|k) = \text{diag}(5, 16)$, the measurement matrix $H = I_{2 \times 2}$, the measurement noise standard deviations as 2 and 3, respectively, and uncorrelated from each other.

1. Find the 95% point (threshold) for the NIS.
2. Given the predicted measurement $\hat{z}(k+1|k) = [10, 15]'$ indicate if the following measurements are accepted as consistent based on the threshold from above: (a) $z(k+1) = [7, 20]'$; (b) $z(k+1) = [16, 5]'$; (c) $z(k+1) = [19, 25]'$. This procedure is called "gating" or "validation" and is used extensively to select the measurement(s) to be used in the state update when their origin is uncertain.
3. Repeat the above two items for the 99% threshold.

264 5 STATE ESTIMATION IN DISCRETE-TIME LINEAR DYNAMIC SYSTEMS

5-15 Three-point initialization. Consider three position measurements $z(i)$, $i = 1, 2, 3$, taken at fixed intervals T . Assuming a constant acceleration motion, find

1. The LS estimates of the position, velocity and acceleration at $i = 3$
2. The corresponding covariance matrix.

5-16 Fusion of estimates. Consider the two state estimates (vectors)

$$z_i \stackrel{\Delta}{=} \hat{x}_i = x + \tilde{x}_i \quad i = 1, 2$$

with the estimation errors \tilde{x}_i zero-mean, with covariances

$$E[\tilde{x}_i \tilde{x}'_j] = P_{ij} \quad i, j = 1, 2$$

find the MLE $\hat{x}(z_1, z_2)$ and its covariance assuming the above errors are jointly Gaussian with the above moments.

5-17 Sensitivity to process noise variance. Consider a first-order dynamic system with $F = 1$, $H = 1$, process noise variance q and measurement noise variance r .

1. Find the optimal filter gain in steady state, W_∞ , as a function of q and r .
2. Show that it can be written as a function of $\mu = q/r$ only.
3. Assume that a fixed-gain nonoptimal filter is used to estimate x , based on the (incorrect) assumption that the process noise variance is q_f . Give the expression of this gain, W_{∞_f} . Indicate the equations to be used to evaluate the resulting estimation MSE for x . Solve for the steady-state value $P_\infty(q, r, q_f)$ of this MSE.
4. Indicate the condition on W_{∞_f} for the existence of the steady-state value $P_\infty(q, r, q_f)$.
5. Plot $P_\infty(q, r, q_f)$ for $q = 1$, $r = 1$ for $q_f \in [10^{-2}, 10^2]$.

5-18 Simultaneous estimation of sensor bias and target location. A sensor is believed to be at the origin of the real axis. However, its true location is biased and known only to the following extent: It is on the real axis in the neighborhood of the origin at $b(k)$, which is a Gaussian zero-mean random sequence with autocorrelation

$$E[b(k)b(j)] = \frac{\sigma_v^2}{1 - \alpha^2} \alpha^{|k-j|}$$

This sensor measures the range (distance) to a fixed *unknown* location point on the real axis at x_1 in the presence of additive Gaussian zero-mean white noise with s.d. σ_r .

- (i) Formulate the estimation problem for the above using the appropriate state space representation.
- (ii) Is the system completely observable? What condition has to be met?
- (iii) Find the initial-state covariance matrix based on the above.
- (iv) Find the steady-state estimation covariance matrix.

5.7.3 Computer Applications

The computational parts of these computer application exercises can be solved using the companion software DynaEstTM.

- 5-1 Evaluation of a Kalman filter with different sampling intervals.** Consider the single-input, single-output linear system with transfer function

$$H(s) = \frac{Z_1(s)}{\dot{V}(s)} = \frac{1}{s^2}$$

driven by a white noise $\tilde{v}(t)$ with power spectral density \tilde{q} .

- Find its continuous-time state-space representation with states (position and velocity)

$$x_1 = z_1 \quad x_2 = \dot{z}_1$$

- Assume the output (position) is measured every T seconds

$$z(k) = z_1(k) = x_1(k) + w(k)$$

in the presence of a zero mean white noise with variance R_1 . Find the discrete-time state representation and evaluate the process noise covariance matrix.

- For values $x_1(0) = 0\text{m}$, $x_2(0) = 10\text{m/s}$, $T = 2\text{s}$, $\tilde{q} = 1\text{m}^2/\text{s}^3$, $R_1 = 9\text{m}^2$, initialize a (discrete-time) KF for this system with two-point differencing and run one sample trajectory for 100 s. Plot the following:

- (i) True position, the measurements and the estimated position
 - (ii) True velocity and the estimated velocity
 - (iii) The estimated position standard deviation
 - (iv) The estimated velocity standard deviation.
- Redo the filter for $T = 5\text{s}$.

- 5-2 Evaluation of a Kalman filter with different measurements.** For the same system as in computer application exercise 5-1, assume the output measured every T seconds is

$$z(k) = \begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} = x(k) + w(k)$$

in the presence of a zero mean white noise with covariance

$$R = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}$$

Redo item 3 from computer application 5-1 for $r_{11} = 9\text{m}^2$, $r_{22} = 9\text{m}^2/\text{s}^4$ with correlation coefficient ρ between the position and velocity measurements

- $\rho = 0$
- $\rho = 0.5$
- $\rho = -0.5$

Order these three configurations according to the resulting estimation quality and explain why they turned out as they did.

266 5 STATE ESTIMATION IN DISCRETE-TIME LINEAR DYNAMIC SYSTEMS

5-3 Observability of a linear dynamic system. Consider the following linear dynamic systems with

(a)

$$F_a = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, Q_a = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, H_a = [1 \quad -1], R_a = 1 \text{ and } P_a(0|0) = \begin{bmatrix} 1 & 0 \\ 0 & 100 \end{bmatrix}$$

(b)

$$F_b = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & \alpha \end{bmatrix}, Q_b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \alpha = 0.5, H_b = [1 \quad -1 \quad 0], R_b = 1 \text{ and}$$

$$P_b(0|0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1. Using DynaEst, list $P_a(k|k)$ and $P_b(k|k)$ for $k = 0, 1, 2, 5, 10, 20, 50, 100$.
2. Explain the results.
3. What range of values of α is “good?” Justify.