CHAPTER 3

Kalman Filter-Based Estimation

A state estimation algorithm determines the values of a number of parameters of a system, such as its position and velocity, from measurements of the properties of that system. The Kalman filter forms the basis of most state estimation algorithms used in navigation systems. Its uses include maintaining an optimal satellite navigation solution, integration of GNSS user equipment with other navigation sensors, and alignment and calibration of an INS. State estimation is key to obtaining the best possible navigation solution from the various measurements available. A Kalman filter uses all the measurement information input to it over time, not just the most recent set of measurements.

This chapter provides an introduction to the Kalman filter and a review of how it may be adapted for practical use in navigation applications. Section 3.1 provides a qualitative description of the Kalman filter, with the algorithm and mathematical models introduced in Section 3.2. Section 3.3 discusses the practical application of the Kalman filter, while Section 3.4 reviews some more advanced estimation techniques, based on the Kalman filter, that are relevant to navigation problems. These include the extended Kalman filter (EKF), commonly used in navigation applications, the unscented Kalman filter (UKF), and the Kalman smoother, which can give improved performance in postprocessed applications. Finally, Section 3.5 provides a brief introduction to the particle filter. In addition, Appendix D on the CD describes least-squares estimation, summarizes the Schmidt-Kalman filter, and provides further information on the particle filter, while Appendix B on the CD provides background information on statistical measures, probability, and random processes.

Examples of the Kalman filter's applications in navigation are presented within Chapters 9 and 14 to 16, while the MATLAB software on the accompanying CD includes Kalman-filter based estimation algorithms for GNSS positioning and INS/GNSS integration. For a more formalized and detailed treatment of Kalman filters, there are many applied mathematics books devoted solely to this subject [1–6].

At this point, it is useful to introduce the distinction between systematic and random errors. A systematic error is repeatable and can thus be predicted from previous occurrences using a Kalman filter or another estimation algorithm. An example is a bias, or constant offset, in a measurement. A random error is nonrepeatable; it cannot be predicted. In practice, an error will often have both systematic and random components. An example is a bias that slowly varies in an unpredictable way. This can also be estimated using a Kalman filter.

3.1 Introduction

The Kalman filter is an estimation algorithm, rather than a filter. The basic technique was invented by R. E. Kalman in 1960 [7] and has been developed further by numerous authors since. It maintains real-time estimates of a number of parameters of a system, such as its position and velocity, that may continually change. The estimates are updated using a stream of measurements that are subject to noise. The measurements must be functions of the parameters estimated, but the set of measurements at a given time need not contain sufficient information to uniquely determine the values of the parameters at that time.

The Kalman filter uses knowledge of the deterministic and statistical properties of the system parameters and the measurements to obtain optimal estimates given the information available. It is a Bayesian estimation technique. It is supplied with an initial set of estimates and then operates recursively, updating its working estimates as a weighted average of their previous values and new values derived from the latest measurement data. By contrast, nonrecursive estimation algorithms derive their parameter estimates from the whole set of measurement data without prior estimates. For real-time applications, such as navigation, the recursive approach is more processor efficient, as only the new measurement data need be processed on each iteration. Old measurement data may be discarded.

To enable optimal weighting of the data, a Kalman filter maintains a set of uncertainties in its estimates and a measure of the correlations between the errors in the estimates of the different parameters. This is carried forward from iteration to iteration alongside the parameter estimates. It also accounts for the uncertainties in the measurements due to noise.

This section provides a qualitative description of the Kalman filter and the steps forming its algorithm. Some brief examples of Kalman filter applications conclude the section. A quantitative description and derivation follow in Section 3.2.

3.1.1 Elements of the Kalman Filter

Figure 3.1 shows the five core elements of the Kalman filter: the state vector and covariance, the system model, the measurement vector and covariance, the measurement model, and the algorithm.

The *state vector* is the set of parameters describing a system, known as *states*, which the Kalman filter estimates. Each state may be constant or time varying. For most navigation applications, the states include the components of position or position error. Velocity, attitude, and navigation sensor error states may also be estimated. Beware that some authors use the term state to describe the whole state vector rather than an individual component.

Associated with the state vector is an *error covariance matrix*. This represents the uncertainties in the Kalman filter's state estimates and the degree of correlation between the errors in those estimates. The correlation information within the error covariance matrix is important for three reasons. First, it enables the error distribution of the state estimates to be fully represented. Figure 3.2 illustrates this for north and east position estimates; when the correlation is neglected, the accuracy is overestimated in one direction and underestimated in another. Second, there is not

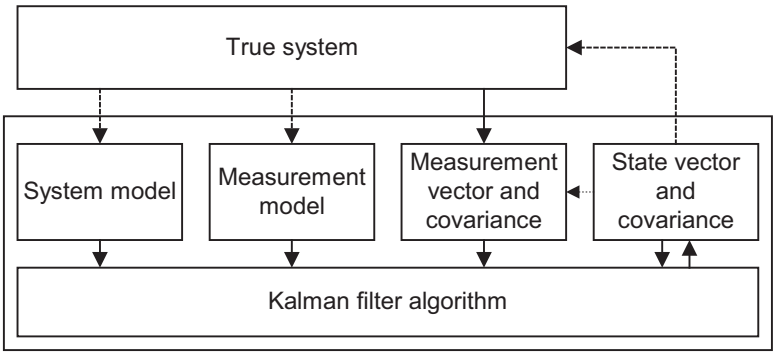

Figure 3.1 Elements of the Kalman filter. (*From:* [8]. © 2002 QinetiQ Ltd. Reprinted with permission.)

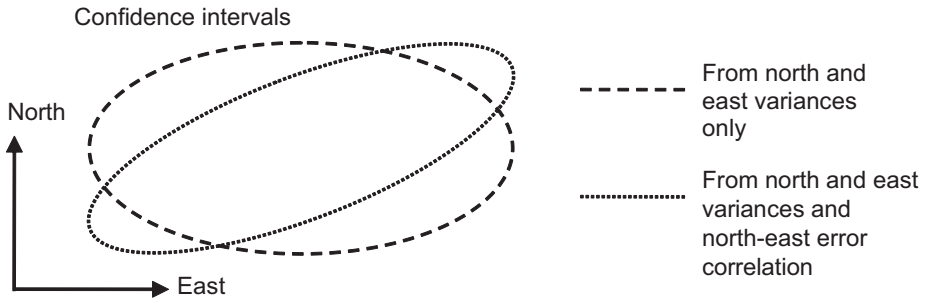

Figure 3.2 Example position error ellipses with and without error correlation.

always enough information from the measurements to estimate the Kalman filter states independently. The correlation information enables estimates of linear combinations of those states to be maintained while awaiting further measurement information. Finally, correlations between errors can build up over the intervals between measurements. Modeling this can enable one state to be determined from another (e.g., velocity from a series of positions). A Kalman filter is an iterative process, so the initial values of the state vector and covariance matrix must be set by the user or determined from another process.

The *system model*, also known as the process model or time-propagation model, describes how the Kalman filter states and error covariance matrix vary with time. For example, a position state will vary with time as the integral of a velocity state; the position uncertainty will increase with time as the integral of the velocity uncertainty; and the position and velocity estimation errors will become more correlated. The system model is deterministic for the states as it is based on known properties of the system.[*]

A state uncertainty should also be increased with time to account for unknown changes in the system that cause the state estimate to go out of date in the absence of new measurement information. These changes may be unmeasured dynamics or

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

random noise on an instrument output. For example, a velocity uncertainty must be increased over time if the acceleration is unknown. This variation in the true values of the states is known as *system noise* or process noise, and its assumed statistical properties are usually defined by the Kalman filter designer.

The *measurement vector* is a set of simultaneous measurements of properties of the system which are functions of the state vector. Examples include the set of range measurements from a radio navigation system and the difference in navigation solution between an INS under calibration and a reference navigation system. This is the information from which all of the state estimates are derived after initialization. Associated with the measurement vector is a *measurement noise covariance* matrix which describes the statistics of the noise on the measurements. For many applications, new measurement information is input to the Kalman filter at regular intervals. In other cases, the time interval between measurements can be irregular.

The *measurement model* describes how the measurement vector varies as a function of the true state vector (as opposed to the state vector estimate) in the absence of measurement noise. For example, the velocity measurement difference between an INS under calibration and a reference system is directly proportional to the INS velocity error. Like the system model, the measurement model is deterministic, based on known properties of the system.*

The *Kalman filter algorithm* uses the measurement vector, measurement model, and system model to maintain optimal estimates of the state vector.

3.1.2 Steps of the Kalman Filter

The Kalman filter algorithm consists of two phases, system propagation and measurement update, which together comprise up to 10 steps per iteration. These are shown in Figure 3.3. Steps 1–4 form the system-propagation phase and steps 5–10 the measurement-update phase. Each complete iteration of the Kalman filter corresponds to a particular point in time, known as an epoch.

The purpose of the system-propagation, or time-propagation, phase is to predict forward the state vector estimate and error covariance matrix from the time of validity of the last measurement set to the time of the current set of measurements using the known properties of the system. So, for example, a position estimate is predicted forward using the corresponding velocity estimate. This provides the Kalman filter's best estimate of the state vector at the current time in the absence of new measurement information. The first two steps calculate the deterministic and noise parts of the system model. The third step, *state propagation*, uses this to bring the state vector estimate up to date. The fourth step, *covariance propagation*, performs the corresponding update to the error covariance matrix, increasing the state uncertainty to account for the system noise.

In the measurement-update, or correction, phase, the state vector estimate and error covariance are updated to incorporate the new measurement information. Steps 5 and 6, respectively, calculate the deterministic and noise parts of the measurement model. The seventh step, *gain computation*, calculates the Kalman gain matrix. This

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

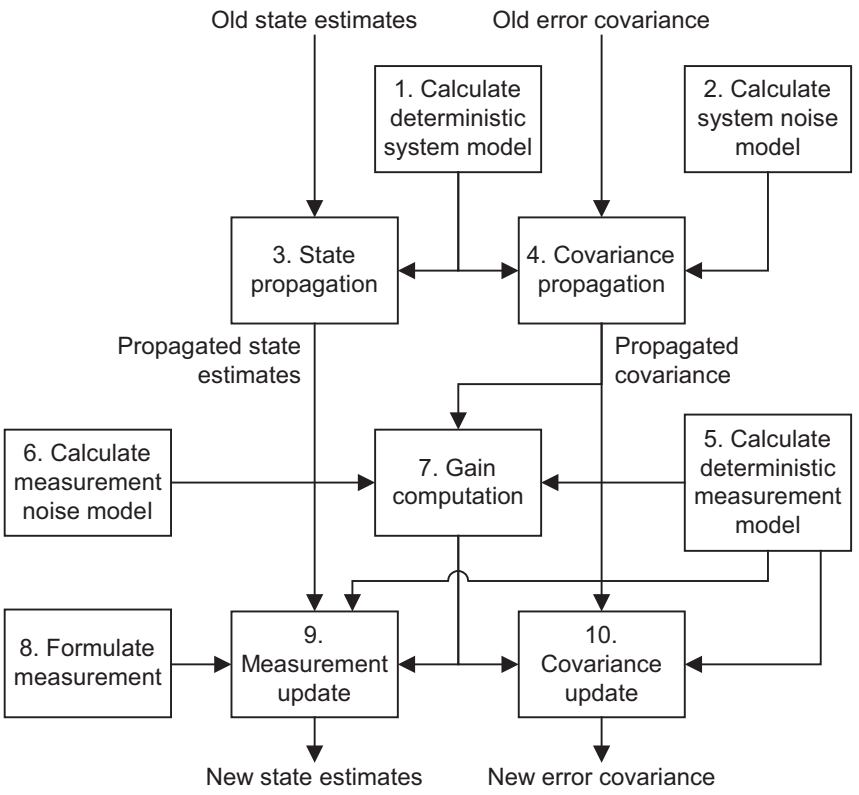

Figure 3.3 Kalman filter algorithm steps.

is used to optimally weight the correction to the state vector according to the uncertainty of the current state estimates and how noisy the measurements are. The eighth step formulates the measurement vector. The ninth step, the *measurement update*, updates the state estimates to incorporate the measurement data weighted with the Kalman gain. Finally, the *covariance update* updates the error covariance matrix to account for the new information that has been incorporated into the state vector estimate from the measurement data.

Figure 3.4 illustrates qualitatively how a Kalman filter can determine a position solution from successive incomplete measurements. At epoch 1, there is a 2-D position estimate with a large uncertainty. The measurement available at this epoch is a single line of position (LOP). This could be from a range measurement using a distant transmitter or from a bearing measurement. The measurement only provides positioning information along the direction perpendicular to the LOP. A unique position fix cannot be obtained from it. Implementing a Kalman filter measurement update results in the position estimate moving close to the measurement LOP. There is a large reduction in the position uncertainty perpendicular to the measurement LOP, but no reduction along the LOP.

At epoch 2, the Kalman filter system-propagation phase increases the position uncertainty to account for possible movement of the object. The measurement available at epoch 2 is also a single LOP, but in a different direction to that of the first measurement. This provides positioning information along a different direction to

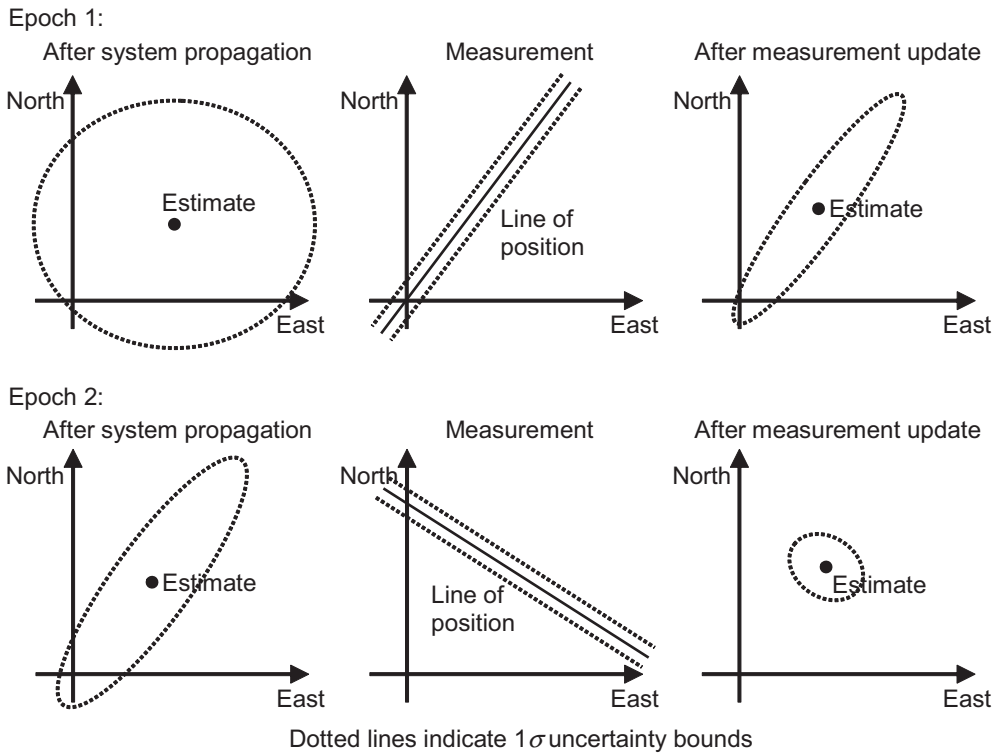

Figure 3.4 Kalman filter 2-D position determination from two successive incomplete measurements.

the first measurement. Consequently, implementing the Kalman filter measurement update results in a position estimate with a small uncertainty in both directions.

3.1.3 Kalman Filter Applications

Kalman filter-based estimation techniques have many applications in navigation. These include GNSS and terrestrial radio navigation, GNSS signal monitoring, INS/GNSS and multisensor integration, and fine alignment and calibration of an INS.

For stand-alone GNSS navigation, the states estimated are the user antenna position and velocity, and the receiver clock offset and drift. The measurements are the line-of-sight ranging measurements of each satellite signal made by the receiver. The GNSS navigation filter is described in Section 9.4.2. For terrestrial radio navigation, the height and vertical velocity are often omitted due to insufficient signal geometry, while the clock states may be omitted if the ranging measurements are two-way or differenced across transmitters (see Chapter 7). A single navigation filter may process both GNSS and terrestrial radio navigation measurements as discussed in Chapter 16.

GNSS signal monitoring uses the same measurements as GNSS navigation. However, the user antenna position and velocity are accurately known and a high precision receiver clock is used, so the time-correlated range errors may be estimated as Kalman filter states. With a network of monitor stations at different locations, the different contributing factors to the range errors may all be estimated as separate states.

For most INS/GNSS and multisensor integration architectures, the errors of the constituent navigation systems, including position and velocity errors, are estimated. In some architectures, the navigation solution itself is also estimated. The measurements processed vary with the type of integration implemented. Examples include position measurements, ranging measurements and sensor measurements. INS/GNSS integration techniques are described in Chapter 14, with multisensor integration described in Chapter 16.

For alignment and calibration of an INS, the states estimated are position, velocity, and attitude errors, together with inertial instrument errors, such as accelerometer and gyro biases. The measurements are the position, velocity, and/or attitude differences between the aligning-INS navigation solution and an external reference, such as another INS or GNSS.* More details are given in Section 5.6.3 and Chapters 14 and 15.

3.2 Algorithms and Models

This section presents and derives the Kalman filter algorithm, system model, and measurement model, including open- and closed-loop implementations and a discussion of Kalman filter behavior and state observability. Prior to this, error types are discussed and the main Kalman filter parameters defined. Although a Kalman filter may operate continuously in time, discrete-time implementations are most common as these are suited to digital computation. Thus, only the discrete-time version is presented here.*

3.2.1 Definitions

The time variation of all errors modeled within a discrete-time Kalman filter is assumed to fall into one of three categories: systematic errors, white noise sequences, and Gauss-Markov sequences. These are shown in Figure 3.5. *Systematic errors* are assumed to be constant—in other words, 100% time-correlated, though a Kalman filter's estimates of these quantities may vary as it obtains more information about them.

A *white noise sequence* is a discrete-time sequence of mutually uncorrelated random variables from a zero-mean distribution. Samples, w_i, have the property

$$\mathrm{E}(w_i w_j) = \begin{cases} \sigma_w^2 & i = j \\ 0 & i \neq j \end{cases}, \tag{3.1}$$

where E is the expectation operator and σ_w^2 is the variance. A white noise process is described in Section B.4.2 of Appendix B on the CD. Samples from a band-limited white noise process may be treated as a white noise sequence provided the sampling

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

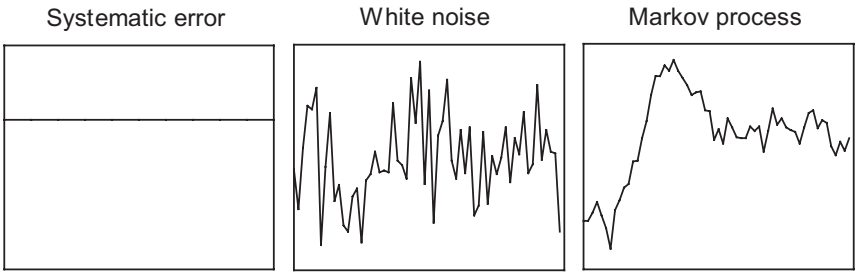

Figure 3.5 Example systematic error, white noise, and Markov process.

rate is much less than the double-sided noise bandwidth. The variance of a white noise sequence, obtained by integrating a white noise process over the time interval τ_w, is

$$\sigma_w^2 = \tau_w S_w, \tag{3.2}$$

where S_w is the power spectral density (PSD) of the white noise process. This is the variance per unit bandwidth. In general, the PSD is a function of frequency. However, for band-limited white noise, the PSD is constant within the white noise bandwidth, which must significantly exceed $1/\tau_w$ for (3.2) to apply. In a Kalman filter, white noise is normally assumed to have a Gaussian (or normal) distribution (see Section B.3.2 in Appendix B on the CD).

A *Gauss-Markov sequence* is a quantity that varies with time as a linear function of its previous values and a white noise sequence. When the properties of a Gauss-Markov sequence are known, it can be modeled in a Kalman filter. It typically varies slowly compared to the update interval. A first-order Gauss-Markov sequence may be represented as a linear function only of its previous value and noise. A Markov process is the continuous-time equivalent of a Markov sequence. A first-order Gauss-Markov process, x_{mi}, may be described by

$$\frac{\partial x_{mi}}{\partial t} = -\frac{x_{mi}}{\tau_{mi}} + w_i, \tag{3.3}$$

where t is time and τ_{mi} is the correlation time. It is often known as an exponentially correlated Markov process as it has an exponentially decaying auto-correlation function. Markov processes and sequences are described in more detail in Section B.4.3 of Appendix B on the CD. In a Kalman filter, they are normally assumed to have Gaussian distributions.

A principal assumption of Kalman filter theory is that the errors of the modeled system are systematic, white noise, or Gauss-Markov processes. They may also be linear combinations or integrals thereof. For example, a random walk process is integrated white noise, while a constant acceleration error leads to a velocity error that grows with time. Error sources modeled as states are assumed to be systematic, Markov processes, or their integrals. All noise sources are assumed to be white, noting that Markov processes have a white noise component. Real navigation system errors do not fall neatly into these categories, but, in many cases, can be approximated to them, provided the modeled errors adequately overbound their real counterparts. A good analogy is that you can fit a square peg into a round hole if you make the hole sufficiently large.

The set of parameters estimated by a Kalman filter, known as the *state vector*, is denoted by $\mathbf{x}$. The Kalman filter estimate of the state vector is denoted $\hat{\mathbf{x}}$, with the caret, ^, also used to indicate other quantities calculated using the state estimates.[*] Estimating absolute properties of the system, such as position, velocity, and attitude, as states is known as a *total-state* implementation. Estimation of the errors in a measurement made by the system, such as INS position, velocity, and attitude, as states is known as an *error-state* implementation. However, a state vector may comprise a mixture of total states and error states.

Note that it is not always sufficient for a Kalman filter only to estimate those states required to directly determine or correct the navigation solution. Significant systematic error sources and Markov processes that impact the states or measurements must be added to the state vector to prevent corruption of the navigation states. This is because a Kalman filter assumes that all error sources that are not modeled as states are white noise. The addition of these extra states is sometimes known as augmentation.

The *state vector residual*, $\delta\mathbf{x}$, is the difference between the true state vector and the Kalman filter estimates thereof. Thus,[†]

$$\delta\mathbf{x} = \mathbf{x} - \hat{\mathbf{x}}. \tag{3.4}$$

In an error-state implementation, the state vector residual represents the errors remaining in the system after the Kalman filter estimates have been used to correct it. The errors in the state estimates are obtained simply by reversing the sign of the state residuals.

The *error covariance matrix*, $\mathbf{P}$, defines the expectation of the square of the deviation of the state vector estimate from the true value of the state vector. Thus,[‡]

$$\mathbf{P} = \mathrm{E}\left((\hat{\mathbf{x}} - \mathbf{x})(\hat{\mathbf{x}} - \mathbf{x})^{\mathrm{T}}\right) = \mathrm{E}\left(\delta\mathbf{x}\delta\mathbf{x}^{\mathrm{T}}\right). \tag{3.5}$$

The $\mathbf{P}$ matrix is symmetric (see Section A.3 of Appendix A on the CD). The diagonal elements are the variances of each state estimate, while their square roots are the uncertainties. Thus,

$$P_{ii} = \sigma_i^2, \tag{3.6}$$

where σ_i is the uncertainty of the ith state estimate. The off-diagonal elements of $\mathbf{P}$, the covariances, describe the correlations between the errors in the different state estimates. They may be expressed as

$$P_{ij} = P_{ji} = \sigma_i \sigma_j \rho_{i,j} \tag{3.7}$$

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

[†]This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

[‡]End of QinetiQ copyright material.

where $\rho_{i,j}$ is the correlation coefficient, defined in Section B.2.1 of Appendix B on the CD. Note that $\rho_{i,j} = 1$ where $i = j$.

The errors in the estimates of different states can become significantly correlated with each other where there is insufficient information from the measurements to estimate those states independently. It is analogous to having a set of simultaneous equations where there are more unknowns than equations. This subject is known as observability and is discussed further in Section 3.2.5.

In an error-state implementation, all state estimates are usually given an initial value of zero. In a total-state implementation, the states may be initialized by the user, by a coarse initialization process, or with the estimates from the previous time the host equipment was used. The initialization values of the covariance matrix are generally determined by the Kalman filter designer and are normally selected cautiously. Thus, the state initialization values are a priori estimates, while the initial covariance matrix values indicate the confidence in those estimates.

In the continuous-time Kalman filter system and measurement models, the state vector and other parameters are shown as functions of time, t. In the discrete-time Kalman filter, the subscript k is used to denote the epoch or iteration to which the state the state vector and other parameters apply. Therefore, $\mathbf{x}_k \equiv \mathbf{x}(t_k)$.

It is necessary to distinguish between the state vector and error covariance after complete iterations of the Kalman filter and in the intermediate step between propagation and update. Thus, the time-propagated state estimates and covariance are denoted $\hat{\mathbf{x}}_k^-$ and $\mathbf{P}_k^-$ (some authors use $\hat{\mathbf{x}}_k(-)$ and $\mathbf{P}_k(-)$, $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$, or $\hat{\mathbf{x}}(k|k-1)$ and $\mathbf{P}(k|k-1)$). Their counterparts following the measurement update are denoted $\hat{\mathbf{x}}_k^+$ and $\mathbf{P}_k^+$ (some authors use $\hat{\mathbf{x}}_k(+)$ and $\mathbf{P}_k(+)$, $\hat{\mathbf{x}}_{k|k}$ and $\mathbf{P}_{k|k}$, or $\hat{\mathbf{x}}(k|k)$ and $\mathbf{P}(k|k)$).

The measurement vector, $\mathbf{z}$ (some authors use $\mathbf{y}$), is a set of measurements of the properties of the system described by the state vector. This could be a set of range measurements or the difference between two navigation systems' position and velocity solutions. It comprises a deterministic function, $\mathbf{h}(\mathbf{x})$, and noise, $\mathbf{w}_m$ (many authors use $\mathbf{v}$, while some use $\boldsymbol{\mu}$ or $\mathbf{w}$). Thus,[†]

$$\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{w}_m. \tag{3.8}$$

The *measurement innovation*, $\delta\mathbf{z}^-$ (some authors use $\boldsymbol{\mu}$ or $\mathbf{r}$), is the difference between the true measurement vector and that computed from the state vector estimate prior to the measurement update:[‡]

$$\delta\mathbf{z}^- = \mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}^-). \tag{3.9}$$

For example, it could be the difference between an actual set of range measurements and a set predicted using a Kalman filter's position estimate. The *measurement residual*, $\delta\mathbf{z}^+$, is the difference between the true measurement vector and that computed from the updated state vector:

[†]This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

[‡]End of QinetiQ copyright material.

$$\delta \mathbf{z}^+ = \mathbf{z} - \mathbf{h}\left(\hat{\mathbf{x}}^+\right). \tag{3.10}$$

Beware that some authors use the term residual to describe the innovation.

The measurement innovations and residuals are a mixture of state estimation errors and measurement errors that are uncorrelated with the state estimates, such as the noise on a set of range measurements. The standard Kalman filter assumes that these measurement errors form a zero-mean distribution, normally assumed to be Gaussian, that is uncorrelated in time, and models their standard deviations with the *measurement noise covariance matrix*, **R**. This defines the expectation of the square of the measurement noise. Thus,

$$\mathbf{R} = \mathrm{E}\left(\mathbf{w}_m \mathbf{w}_m^{\mathrm{T}}\right). \tag{3.11}$$

The diagonal terms of **R** are the variances of each measurement, and the off-diagonal terms represent the correlation between the different components of the measurement noise. The **R** matrix is also symmetric. For most navigation applications, the noise on each component of the measurement vector is independent so **R** is a diagonal matrix. The rest of the Kalman filter notation is defined as it is used.[*]

3.2.2 Kalman Filter Algorithm

With reference to Figure 3.3, the discrete-time Kalman filter algorithm comprises the following steps:[†]

1. Calculate the transition matrix, $\mathbf{\Phi}_{k-1}$.
2. Calculate the system noise covariance matrix, $\mathbf{Q}_{k-1}$.
3. Propagate the state vector estimate from $\hat{\mathbf{x}}_{k-1}^+$ and $\hat{\mathbf{x}}_k^-$.
4. Propagate the error covariance matrix from $\mathbf{P}_{k-1}^+$ to $\mathbf{P}_k^-$.
5. Calculate the measurement matrix, $\mathbf{H}_k$.
6. Calculate the measurement noise covariance matrix, $\mathbf{R}_k$.
7. Calculate the Kalman gain matrix, $\mathbf{K}_k$.
8. Formulate the measurement, $\mathbf{z}_k$.
9. Update the state vector estimate from $\hat{\mathbf{x}}_k^-$ to $\hat{\mathbf{x}}_k^+$.
10. Update the error covariance matrix from $\mathbf{P}_k^-$ to $\mathbf{P}_k^+$.[‡]

The Kalman filter steps do not have to be implemented strictly in this order, provided that the dependencies depicted in Figure 3.3 are respected. Although many Kalman filters simply alternate the system-propagation and measurement-update phases, other processing cycles are possible as discussed in Section 3.3.2.

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

[†]This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

[‡]End of QinetiQ copyright material.

The first four steps comprise the system-propagation phase of the Kalman filter, also known as the system-update, system-extrapolation, prediction, projection, time-update, or time-propagation phase. The system model is derived in Section 3.2.3.

Step 1 is the calculation of the *transition matrix*, $\mathbf{\Phi}_{k-1}$ (a few authors use $\mathbf{F}_{k-1}$). This defines how the state vector changes with time as a function of the dynamics of the system modeled by the Kalman filter. For example, a position state will vary as the integral of a velocity state. The rows correspond to the new values of each state and the columns to the old values.

The transition matrix is different for every Kalman filter application and is derived from a linear system model as shown in Section 3.2.3. It is nearly always a function of the time interval, τ_s, between Kalman filter iterations and is often a function of other parameters. When these parameters vary over time, the transition matrix must be recalculated on every Kalman filter iteration. Note that, in a standard Kalman filter, the transition matrix is never a function of any of the states; otherwise, the system model would not be linear.

Example A is a Kalman filter estimating position and velocity along a single axis in a nonrotating frame. The state vector and transition matrix are

$$\mathbf{x}_A = \begin{pmatrix} r^i_{ib,x} \\ v^i_{ib,x} \end{pmatrix}, \qquad \mathbf{\Phi}_A = \begin{pmatrix} 1 & \tau_s \\ 0 & 1 \end{pmatrix} \tag{3.12}$$

as position is the integral of velocity. Example B is a Kalman filter estimating 2-D position, again in a nonrotating frame. Its state vector and transition matrix are

$$\mathbf{x}_B = \begin{pmatrix} r^i_{ib,x} \\ r^i_{ib,y} \end{pmatrix}, \qquad \mathbf{\Phi}_B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{3.13}$$

as the transition matrix is simply the identity matrix where all states are independent. Examples 3.1 and 3.2 on the CD, both of which are editable using Microsoft Excel, comprise numerical implementations of a complete Kalman filter cycle based on Examples A and B, respectively.

Step 2 is the calculation of the *system noise covariance matrix*, $\mathbf{Q}_{k-1}$, also known as the process noise covariance matrix. It defines how the uncertainties of the state estimates increase with time due to unknown changes in the true values of those states, such as unmeasured dynamics and instrument noise. These changes are treated as noise sources in the Kalman filter's system model. The system noise is always a function of the time interval between iterations, τ_s. Depending on the application, it may be modeled as either time-varying or as constant (for a given time interval). The system noise covariance is a symmetric matrix and is often approximated to a diagonal matrix.

In Example A, system noise arises from changes in the velocity state over time. Example B does not include a velocity state, so system noise arises from changes in the two position states. System noise covariance matrices for these examples are presented in Section 3.2.3.

Step 3 comprises the propagation of the state vector estimate through time using

$$\hat{\mathbf{x}}_k^- = \boldsymbol{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1}^+. \tag{3.14}$$

Step 4 is the corresponding error covariance propagation. The standard form is

$$\mathbf{P}_k^- = \boldsymbol{\Phi}_{k-1}\mathbf{P}_{k-1}^+\boldsymbol{\Phi}_{k-1}^{\mathrm{T}} + \mathbf{Q}_{k-1}. \tag{3.15}$$

Note that the first $\boldsymbol{\Phi}$ matrix propagates the rows of the error covariance matrix, while the second, $\boldsymbol{\Phi}^{\mathrm{T}}$, propagates the columns. Following this step, each state uncertainty should be either larger or unchanged.

The remaining steps in the Kalman filter algorithm comprise the measurement-update or correction phase. The measurement model is derived in Section 3.2.4.

Step 5 is the calculation of the *measurement matrix*, $\mathbf{H}_k$ (some authors use $\mathbf{M}_k$, while $\mathbf{G}_k$ or $\mathbf{A}_k$ is sometimes used in GNSS navigation filters). This defines how the measurement vector varies with the state vector. Each row corresponds to a measurement and each column to a state. For example, the range measurements from a radio navigation system vary with the position of the receiver. In a standard Kalman filter, each measurement is assumed to be a linear function of the state vector. Thus,

$$\mathbf{h}(\mathbf{x}_k, t_k) = \mathbf{H}_k\mathbf{x}_k. \tag{3.16}$$

In most applications, the measurement matrix varies, so it must be calculated on each iteration of the Kalman filter. In navigation, $\mathbf{H}_k$ is commonly a function of the user kinematics and/or the geometry of transmitters, such as GNSS satellites.

In Examples A and B, the measurements are, respectively, single-axis position and 2-D position, plus noise. The measurement models and matrices are thus

$$z_A = r_{ib,x}^i + w_m, \qquad \mathbf{H}_A = \begin{pmatrix} 1 & 0 \end{pmatrix} \tag{3.17}$$

and

$$\mathbf{z}_B = \begin{pmatrix} r_{ib,x}^i + w_{m,x} \\ r_{ib,y}^i + w_{m,y} \end{pmatrix}, \qquad \mathbf{H}_B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{3.18}$$

Measurement updates using these models are shown in Examples 3.1 and 3.2 on the CD.

Step 6 is the calculation of the measurement noise covariance matrix, $\mathbf{R}_k$. Depending on the application, it may be assumed constant, modeled as a function of dynamics, and/or modeled as a function of signal-to-noise measurements.

Step 7 is the calculation of the *Kalman gain matrix*, $\mathbf{K}_k$. This is used to determine the weighting of the measurement information in updating the state estimates. Each row corresponds to a state and each column to a measurement. The Kalman gain depends on the error covariance matrices of both the true measurement vector, $\mathbf{z}_k$,

and that predicted from the state estimates, $\mathbf{H}_k\hat{\mathbf{x}}_k^-$, noting that the diagonal elements of the matrices are the squares of the uncertainties. From (3.8), (3.9), and (3.10), the error covariance of the true measurement vector is

$$\mathrm{E}\left(\left(\mathbf{z}_k - \mathbf{H}_k\mathbf{x}_k\right)\left(\mathbf{z}_k - \mathbf{H}_k\mathbf{x}_k\right)^{\mathrm{T}}\right) = \mathbf{R}_k, \tag{3.19}$$

and, from (3.5), the error covariance of the measurement vector predicted from the state vector is

$$\mathrm{E}\left(\left(\mathbf{H}_k\hat{\mathbf{x}}_k^- - \mathbf{H}_k\mathbf{x}_k\right)\left(\mathbf{H}_k\hat{\mathbf{x}}_k^- - \mathbf{H}_k\mathbf{x}_k\right)^{\mathrm{T}}\right) = \mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}}. \tag{3.20}$$

The Kalman gain matrix is

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}}\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k\right)^{-1}. \tag{3.21}$$

where $(\ )^{-1}$ denotes the inverse of a matrix. Matrix inversion is discussed in Section A.4 of Appendix A on the CD. Some authors use a fraction notation for matrix inversion; however, this can leave the order of matrix multiplication ambiguous. Note that, as the leading $\mathbf{H}_k$ matrix of (3.20) is omitted in the "numerator" of the variance ratio, the Kalman gain matrix transforms from measurement space to state space as well as weighting the measurement information. The correlation information in the off-diagonal elements of the $\mathbf{P}_k^-$ matrix couples the measurement vector to those states that are not directly related via the $\mathbf{H}_k$ matrix.

In Example A, the measurement is scalar, simplifying the Kalman gain calculation. If the covariance matrices are expressed as

$$\mathbf{P}_{A,k}^- = \begin{pmatrix} \sigma_r^2 & P_{rv} \\ P_{rv} & \sigma_v^2 \end{pmatrix}, \qquad R_{A,k} = \sigma_z^2, \tag{3.22}$$

substituting these and (3.17) into (3.21) gives a Kalman gain of

$$\mathbf{K}_{A,k} = \begin{pmatrix} \sigma_r^2 \\ P_{rv} \end{pmatrix}\frac{1}{\sigma_r^2 + \sigma_z^2}. \tag{3.23}$$

Note that the velocity may be estimated from the position measurements provided the prior position and velocity estimates have correlated errors.

Step 8 is the formulation of the measurement vector, $\mathbf{z}_k$. In some cases, such as radio navigation range measurements and Examples A and B, the measurement vector components are already present in the system modeled by the Kalman filter. In other cases, $\mathbf{z}_k$ must be calculated as a function of other system parameters. An example is the navigation solution difference between a system under calibration and a reference system.

For many applications, the measurement innovation, $\delta\mathbf{z}_k^-$, may be calculated directly by applying corrections derived from the state estimates to those parameters of which the measurements are a function. For example, the navigation solution of an INS under calibration may be corrected by the Kalman filter state estimates prior to being differenced with a reference navigation solution.

Step 9 is the update of the state vector with the measurement vector using

$$\begin{aligned}\hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-\right)\\ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\delta\mathbf{z}_k^-\end{aligned}. \tag{3.24}$$

The measurement innovation, $\delta\mathbf{z}_k^-$, is multiplied by the Kalman gain matrix to obtain a correction to the state vector estimate.

Step 10 is the corresponding update of the error covariance matrix with

$$\mathbf{P}_k^+ = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-. \tag{3.25}$$

As the updated state vector estimate is based on more information, the updated state uncertainties are smaller than before the update. Note that for an application where $\mathbf{\Phi}_{k-1}$ and Q_{k-1} are both zero, the Kalman filter is the same as a recursive least-squares estimator (see Section D.1 of Appendix D on the CD).

Figure 3.6 summarizes the data flow in a Kalman filter.

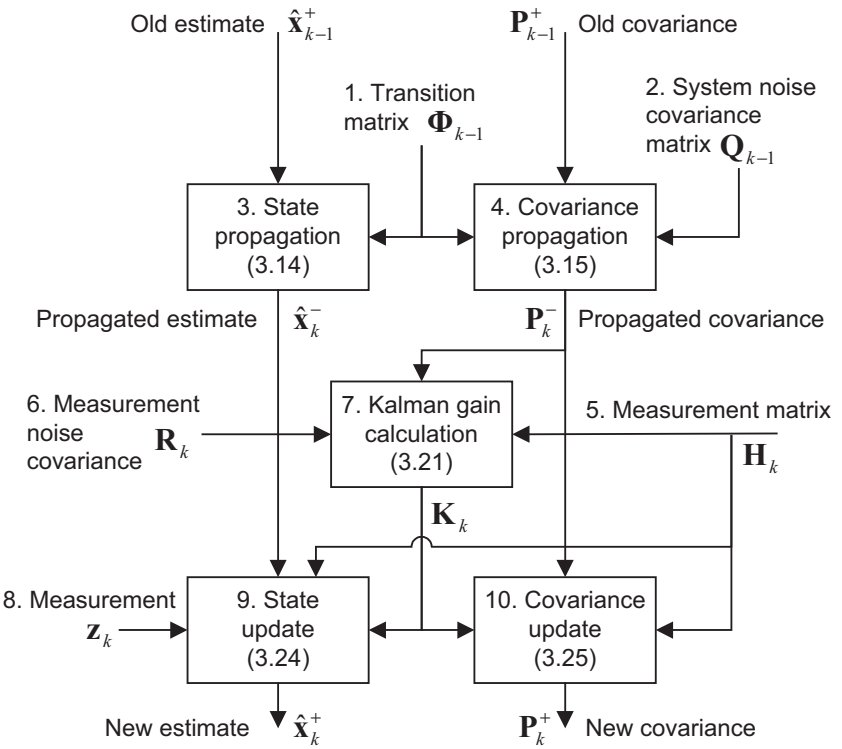

Figure 3.6 Kalman filter data flow.

The algorithm presented here is for an *open-loop implementation* of the Kalman filter, whereby all state estimates are retained in the Kalman filter algorithm. Section 3.2.6 describes the closed-loop implementation, whereby state estimates are fed back to correct the system.[*]

3.2.3 System Model

To propagate the state vector estimate, $\hat{\mathbf{x}}$, and error covariance, $\mathbf{P}$, forward in time, it is necessary to know how those states vary with time. This is the function of the system model. This section shows how the Kalman filter system propagation equations, (3.14) and (3.15), may be obtained from a model of the state dynamics, an application of linear systems theory.

An assumption of the Kalman filter is that the time derivative of each state is a linear function of the other states and of white noise sources. Thus, the true state vector, $\mathbf{x}(t)$, at time, t, of any Kalman filter is described by the following dynamic model:[†]

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{G}(t)\mathbf{w}_s(t), \tag{3.26}$$

where $\mathbf{w}_s(t)$ is the continuous system noise vector (many authors use $\mathbf{w}$ and some use $\boldsymbol{\omega}$ or $\mathbf{v}$), $\mathbf{F}(t)$ is the system matrix (some authors use $\mathbf{A}$), and $\mathbf{G}(t)$ is the continuous system noise distribution matrix. The system noise vector comprises a number of independent random noise sources, each assumed to have a zero-mean symmetric distributions, such as the Gaussian distribution. $\mathbf{F}(t)$ and $\mathbf{G}(t)$ are always known functions. To determine the system model, these functions must be derived from the known properties of the system.[‡]

In Example A, the two-state Kalman filter estimating position and velocity along a single axis, the acceleration is not estimated so it must be represented as system noise. Thus, the state vector and system noise for this example are

$$\mathbf{x}_A = \begin{pmatrix} r^i_{ib,x} \\ v^i_{ib,x} \end{pmatrix}, \qquad w_{s,A} = a^i_{ib,x}. \tag{3.27}$$

The state dynamics are simply

$$\dot{r}^i_{ib,x} = v^i_{ib,x}, \qquad \dot{v}^i_{ib,x} = a^i_{ib,x}. \tag{3.28}$$

Substituting (3.27) and (3.28) into (3.16) gives the system matrix and system noise distribution matrix:

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

[†]This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

[‡]End of QinetiQ copyright material.

$$\mathbf{F}_A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \qquad \mathbf{G}_A = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{3.29}$$

noting that, in this case, neither matrix is a function of time.

To obtain an estimate, the expectation operator, E, is applied. The expectation value of the true state vector, $\mathbf{x}(t)$, is the estimated state vector, $\hat{\mathbf{x}}(t)$. The expectation value of the system noise vector, $\mathbf{w}_s(t)$, is zero as the noise is assumed to be of zero mean. $\mathbf{F}(t)$ and $\mathbf{G}(t)$ are assumed to be known functions and thus commute with the expectation operator. Hence, taking the expectation of (3.26) gives*

$$\mathrm{E}(\dot{\hat{\mathbf{x}}}(t)) = \frac{\partial}{\partial t}\hat{\mathbf{x}}(t) = \mathbf{F}(t)\hat{\mathbf{x}}(t). \tag{3.30}$$

Solving (3.30) gives the state vector estimate at time t as a function of the state vector estimate at time $t - \tau_s$:

$$\hat{\mathbf{x}}(t) = \lim_{n\to\infty} \prod_{i=1}^{n} \exp\left(\mathbf{F}\left(t - \frac{i}{n}\tau_s \right) \frac{\tau_s}{n} \right) \hat{\mathbf{x}}(t - \tau_s), \tag{3.31}$$

noting (A.17) in Appendix A on the CD. When $\mathbf{F}$ may be treated as constant over the interval $t - \tau_s$ to t, the approximation

$$\hat{\mathbf{x}}(t) \approx \exp\bigl(\mathbf{F}(t)\tau_s\bigr)\hat{\mathbf{x}}(t - \tau_s) \tag{3.32}$$

may be made, noting that this is exact where $\mathbf{F}$ is actually constant [9].

In the discrete Kalman filter, the state vector estimate is modeled as a linear function of its previous value, coupled by the transition matrix, $\mathbf{\Phi}_{k-1}$, repeating (3.14):†

$$\hat{\mathbf{x}}_k^- = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1}^+.$$

The discrete and continuous forms of the Kalman filter are equivalent, with $\hat{\mathbf{x}}_k \equiv \hat{\mathbf{x}}(t_k)$ and $\hat{\mathbf{x}}_{k-1} = \hat{\mathbf{x}}(t_k - \tau_s)$. So, substituting (3.32) into (3.14),

$$\mathbf{\Phi}_{k-1} \approx \exp(\mathbf{F}_{k-1}\tau_s), \tag{3.33}$$

where, assuming data is available at times $t_{k-1} = t_k - \tau_s$ and t_k, but not at intervening intervals, the system matrix, $\mathbf{F}_{k-1}$, can be calculated either as $\frac{1}{2}(\mathbf{F}(t_k - \tau_s) + \mathbf{F}(t_k))$ or by taking the mean of the parameters of $\mathbf{F}$ at times $t_k - \tau_s$ and t_k and making a single calculation of $\mathbf{F}$. In general, (3.33) cannot be computed directly; the exponent of the

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

†This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

matrix is not the matrix of the exponents of its components.[‡] Numerical methods are available [10], but these are computationally intensive where the matrices are large. Therefore, the transition matrix is usually computed as a power-series expansion of the system matrix, $\mathbf{F}$, and propagation interval, τ_s:

$$\mathbf{\Phi}_{k-1} = \sum_{r=0}^{\infty} \frac{\mathbf{F}_{k-1}^r \tau_s^r}{r!} = \mathbf{I} + \mathbf{F}_{k-1}\tau_s + \tfrac{1}{2}\mathbf{F}_{k-1}^2\tau_s^2 + \tfrac{1}{6}\mathbf{F}_{k-1}^3\tau_s^3 + \cdots. \tag{3.34}$$

The Kalman filter designer must decide where to truncate the power-series expansion, depending on the likely magnitude of the states, the length of the propagation interval, and the available error margins. With a shorter propagation interval, a given accuracy may be attained with a shorter truncation. Different truncations may be applied to different terms and exact solutions may be available for some elements of the transition matrix. In some cases, such as Example A, $\mathbf{F}^2$ is zero, so the first-order solution, $\mathbf{I} + \mathbf{F}_{k-1}\tau_s$, is exact.

The true state vector can be obtained as a function of its previous value, $\mathbf{x}_{k-1}$, by integrating (3.26) between times $t_k - \tau_s$ and t_k under the approximation that $\mathbf{F}(t)$ and $\mathbf{G}(t)$ are constant over the integration interval and substituting (3.33):

$$\mathbf{x}_k = \mathbf{\Phi}_{k-1}\mathbf{x}_{k-1} + \mathbf{\Gamma}_{k-1}\mathbf{w}_{s,k-1}, \tag{3.35}$$

where $\mathbf{w}_{s,k-1}$ is the discrete system noise vector and $\mathbf{\Gamma}_{k-1}$ is the discrete system noise distribution matrix, such that

$$\mathbf{\Gamma}_{k-1}\mathbf{w}_{s,k-1} = \int_{t_k-\tau_s}^{t_k} \exp(\mathbf{F}_{k-1}(t_k - t'))\mathbf{G}_{k-1}\mathbf{w}_s(t')\,\mathrm{d}t', \tag{3.36}$$

Note that, as system noise is introduced throughout the propagation interval, it is subject to state propagation via $\mathbf{F}$ for the remainder of that propagation interval. The system noise distribution matrix, $\mathbf{G}_{k-1}$, is calculated in a similar manner to $\mathbf{F}_{k-1}$, either as $\frac{1}{2}\big(\mathbf{G}(t_k - \tau_s) + \mathbf{G}(t_k)\big)$ or by taking the mean of the parameters of $\mathbf{G}$ at times $t_k - \tau_s$ and t_k and making a single calculation of $\mathbf{G}$.

From (3.5), the error covariance matrix before and after the time propagation, and after the measurement update, is

$$\begin{aligned} \mathbf{P}_{k-1}^+ &= \mathrm{E}\big[(\hat{\mathbf{x}}_{k-1}^+ - \mathbf{x}_{k-1})(\hat{\mathbf{x}}_{k-1}^+ - \mathbf{x}_{k-1})^{\mathrm{T}}\big] \\ \mathbf{P}_k^- &= \mathrm{E}\big[(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)^{\mathrm{T}}\big] \\ \mathbf{P}_k^+ &= \mathrm{E}\big[(\hat{\mathbf{x}}_k^+ - \mathbf{x}_k)(\hat{\mathbf{x}}_k^+ - \mathbf{x}_k)^{\mathrm{T}}\big] \end{aligned}. \tag{3.37}$$

Subtracting (3.35) from (3.14),

$$\hat{\mathbf{x}}_k^- - \mathbf{x}_k = \mathbf{\Phi}_{k-1}\big(\hat{\mathbf{x}}_{k-1}^+ - \mathbf{x}_{k-1}\big) - \mathbf{\Gamma}_{k-1}\mathbf{w}_{s,k-1}. \tag{3.38}$$

[‡]End of QinetiQ copyright material.

The errors in the state estimates are uncorrelated with the system noise, so

$$\mathrm{E}\left[(\hat{\mathbf{x}}_k^{\pm} - \mathbf{x}_k)\mathbf{w}_s^{\mathrm{T}}(t)\right] = 0, \quad \mathrm{E}\left[\mathbf{w}_s(t)(\hat{\mathbf{x}}_k^{\pm} - \mathbf{x}_k)^{\mathrm{T}}\right] = 0. \tag{3.39}$$

Therefore, substituting (3.38) and (3.39) into (3.37) gives

$$\mathbf{P}_k^- = \boldsymbol{\Phi}_{k-1}\mathbf{P}_{k-1}^+\boldsymbol{\Phi}_{k-1}^{\mathrm{T}} + \mathrm{E}\left[\boldsymbol{\Gamma}_{k-1}\mathbf{w}_{s,k-1}\mathbf{w}_{s,k-1}^{\mathrm{T}}\boldsymbol{\Gamma}_{k-1}^{\mathrm{T}}\right]. \tag{3.40}$$

Defining the system noise covariance matrix as

$$\mathbf{Q}_{k-1} = \mathrm{E}\left[\boldsymbol{\Gamma}_{k-1}\mathbf{w}_{s,k-1}\mathbf{w}_{s,k-1}^{\mathrm{T}}\boldsymbol{\Gamma}_{k-1}^{\mathrm{T}}\right] \tag{3.41}$$

gives the covariance propagation equation, (3.15)

$$\mathbf{P}_k^- = \boldsymbol{\Phi}_{k-1}\mathbf{P}_{k-1}^+\boldsymbol{\Phi}_{k-1}^{\mathrm{T}} + \mathbf{Q}_{k-1}.$$

Note that some authors define Q differently. Substituting (3.36) into (3.41) gives the system noise covariance in terms of the continuous system noise:

$$\mathbf{Q}_{k-1} = \mathrm{E}\left[\int_{t_k-\tau_s}^{t_k}\int_{t_k-\tau_s}^{t_k} \exp(\mathbf{F}_{k-1}(t_k - t'))\mathbf{G}_{k-1}\mathbf{w}_s(t')\mathbf{w}_s^{\mathrm{T}}(t'')\mathbf{G}_{k-1}^{\mathrm{T}}\exp(\mathbf{F}_{k-1}^{\mathrm{T}}(t_k - t''))\,\mathrm{d}t'\,\mathrm{d}t''\right]. \tag{3.42}$$

If the system noise is assumed to be white, applying (B.102) from Section B.4.2 of Appendix B on the CD gives

$$\mathbf{Q}_{k-1} = \int_{t_k-\tau_s}^{t_k} \exp(\mathbf{F}_{k-1}(t_k - t'))\mathbf{G}_{k-1}\mathbf{S}_{s,k-1}\mathbf{G}_{k-1}^{\mathrm{T}}\exp(\mathbf{F}_{k-1}^{\mathrm{T}}(t_k - t'))\,\mathrm{d}t', \tag{3.43}$$

where $\mathbf{S}_{s,k-1}$ is a diagonal matrix comprising the single-sided PSDs of the components of the continuous system noise vector, $\mathbf{w}_s(t)$.

The system noise covariance is usually approximated. The simplest version is obtained by neglecting the time propagation of the system noise over an iteration of the discrete-time filter, giving

$$\mathbf{Q}_{k-1} \approx \mathbf{Q}'_{k-1} = \mathbf{G}_{k-1}\mathrm{E}\left(\int_{t_k-\tau_s}^{t_k}\int_{t_k-\tau_s}^{t_k} \mathbf{w}_s(t')\mathbf{w}_s^{\mathrm{T}}(t'')\,\mathrm{d}t'\,\mathrm{d}t''\right)\mathbf{G}_{k-1}^{\mathrm{T}} \tag{3.44}$$

in the general case or

$$\mathbf{Q}_{k-1} \approx \mathbf{Q}'_{k-1} = \mathbf{G}_{k-1}\mathbf{S}_{s,k-1}\mathbf{G}_{k-1}^{\mathrm{T}}\tau_s \tag{3.45}$$

where white noise is assumed. This is known as the impulse approximation and, like all approximations, should be validated against the exact version prior to use.

Alternatively, (3.15) and (3.42) to (3.43) may be approximated to the first order in $\mathbf{\Phi}_{k-1}\mathbf{Q}'_{k-1}\mathbf{\Phi}^{\mathrm{T}}{}_{k-1}$, giving

$$\mathbf{P}_k^- \approx \mathbf{\Phi}_{k-1}\left(\mathbf{P}_{k-1}^+ + \tfrac{1}{2}\mathbf{Q}'_{k-1}\right)\mathbf{\Phi}_{k-1}^{\mathrm{T}} + \tfrac{1}{2}\mathbf{Q}'_{k-1}. \tag{3.46}$$

Returning to Example A, if the acceleration is approximated as white Gaussian noise, the exact system noise covariance matrix is

$$\mathbf{Q}_A = \begin{pmatrix} \frac{1}{3}S_a\tau_s^3 & \frac{1}{2}S_a\tau_s^2 \\ \frac{1}{2}S_a\tau_s^2 & S_a\tau_s \end{pmatrix}, \tag{3.47}$$

where S_a is the PSD of the acceleration. This accounts for the propagation of the system noise onto the position state during the propagation interval. If the propagation interval is sufficiently small, the system noise covariance may be approximated to

$$\mathbf{Q}_A \approx \mathbf{Q}'_A = \begin{pmatrix} 0 & 0 \\ 0 & S_a\tau_s \end{pmatrix}. \tag{3.48}$$

In Example B, the two states have no dependency through the system model. Therefore, the exact system noise covariance is simply

$$\mathbf{Q}_B = \begin{pmatrix} S_{vx}\tau_s & 0 \\ 0 & S_{vy}\tau_s \end{pmatrix}, \tag{3.49}$$

where S_{vx} and S_{vy} are the PSDs of the velocity in the x- and y-axes, respectively. Calculations of $\mathbf{Q}_A$ and $\mathbf{Q}_B$ are shown in Examples 3.1 and 3.2, respectively, on the CD.

Time-correlated system noise is discussed in Section 3.4.3.

3.2.4 Measurement Model

To update the state vector estimate with a set of measurements, it is necessary to know how the measurements vary with the states. This is the function of the measurement model. This section presents the derivation of the Kalman filter measurement-update equations, (3.21), (3.24), and (3.25), from the measurement model.

In a standard Kalman filter, the measurement vector, $\mathbf{z}(t)$, is modeled as a linear function of the true state vector, $\mathbf{x}(t)$, and the white noise sources, $\mathbf{w}_m(t)$. Thus,

$$\mathbf{z}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{w}_m(t), \tag{3.50}$$

where $\mathbf{H}(t)$ is the measurement matrix and is determined from the known properties of the system. For example, if the state vector comprises the position error of a dead-reckoning system, such as an INS, and the measurement vector comprises the difference between the dead-reckoning system's position solution and that of a positioning system, such as GNSS, then the measurement matrix is simply the identity matrix.

If the measurements are taken at discrete intervals, (3.50) becomes

$$\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{w}_{mk}. \tag{3.51}$$

Given this set of measurements, the new optimal estimate of the state vector is a linear combination of the measurement vector and the previous state vector estimate. Thus,

$$\hat{\mathbf{x}}_k^+ = \mathbf{K}_k\mathbf{z}_k + \mathbf{L}_k\hat{\mathbf{x}}_k^-, \tag{3.52}$$

where $\mathbf{K}_k$ and $\mathbf{L}_k$ are weighting matrices to be determined. Substituting in (3.51),

$$\hat{\mathbf{x}}_k^+ = \mathbf{K}_k\mathbf{H}_k\mathbf{x}_k + \mathbf{K}_k\mathbf{w}_{mk} + \mathbf{L}_k\hat{\mathbf{x}}_k^-. \tag{3.53}$$

A Kalman filter is an unbiased estimation algorithm, so the expectations of the errors in both the new and previous state vector estimates, $\hat{\mathbf{x}}_k^+ - \mathbf{x}_k$, and $\hat{\mathbf{x}}_k^- - \mathbf{x}_k$ are zero. The expectation of the measurement noise, $\mathbf{w}_{mk}$, is also zero. Thus, taking the expectation of (3.53) gives

$$\mathbf{L}_k = \mathbf{I} - \mathbf{K}_k\mathbf{H}_k. \tag{3.54}$$

Substituting this into (3.52) gives the state vector update equation [repeating (3.24)]:

$$\begin{aligned}\hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-\right)\\ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\delta\mathbf{z}_k^-\end{aligned}.$$

Substituting (3.51) into (3.24) and subtracting the true state vector,

$$\hat{\mathbf{x}}_k^+ - \mathbf{x}_k = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\left(\hat{\mathbf{x}}_k^- - \mathbf{x}_k\right) + \mathrm{K}_k\mathbf{w}_{mk}. \tag{3.55}$$

The error covariance matrix after the measurement update, $\mathbf{P}_k^+$, is then obtained by substituting this into (3.37), giving

$$\mathbf{P}_k^+ = \mathrm{E}\begin{bmatrix}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^{\mathrm{T}} + \mathbf{K}_k\mathbf{w}_{mk}(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)^{\mathrm{T}}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^{\mathrm{T}}\\ +(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)\mathbf{w}_{mk}^{\mathrm{T}}\mathbf{K}_k^{\mathrm{T}} + \mathbf{K}_k\mathbf{w}_{mk}\mathbf{w}_{mk}^{\mathrm{T}}\mathbf{K}_k^{\mathrm{T}}\end{bmatrix}. \tag{3.56}$$

The error in the state vector estimates is uncorrelated with the measurement noise so,[†]

$$\mathrm{E}\left[(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)\mathbf{w}_{mk}^{\mathrm{T}}\right] = 0, \qquad \mathrm{E}\left[\mathbf{w}_{mk}(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)^{\mathrm{T}}\right] = 0. \tag{3.57}$$

[†]This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

$\mathbf{K}_k$ and $\mathbf{H}_k$ commute with the expectation operator, so substituting (3.57) and (3.11) into (3.56) gives[‡]

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^{\mathrm{T}} + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}}, \tag{3.58}$$

noting that the measurement noise covariance matrix, $\mathbf{R}_k$, is defined by (3.11). This equation is known as the Joseph form of the covariance update.

There are two methods for determining the weighting function, $\mathbf{K}_k$, the minimum variance method [1, 2, 4], used here, and the maximum likelihood method [1, 3, 5]. Both give the same result. The criterion for optimally selecting $\mathbf{K}_k$ by the minimum variance method is the minimization of the error in the estimate, $\hat{\mathbf{x}}_k^+$. The variances of the state estimates are given by the diagonal elements of the error covariance matrix. It is therefore necessary to minimize the trace of $\mathbf{P}_k^+$ (see Section A.2 in Appendix A on the CD) with respect to $\mathbf{K}_k$:

$$\frac{\partial}{\partial \mathbf{K}_k}\left[\mathrm{Tr}\left(\mathbf{P}_k^+\right)\right] = 0. \tag{3.59}$$

Substituting in (3.58) and applying the matrix relation (A.42) from Appendix A on the CD gives

$$-2\left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}} + 2\mathbf{K}_k\mathbf{R}_k = 0. \tag{3.60}$$

Rearranging this gives (3.21)

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}}\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k\right)^{-1}.$$

As explained in [2], this result is independent of the units and/or scaling of the states.

By substituting (3.21) into (3.58), the error covariance update equation may be simplified to (3.25):

$$\mathbf{P}_k^+ = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-.$$

This may also be computed as

$$\mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k\left(\mathbf{H}_k\mathbf{P}_k^-\right), \tag{3.61}$$

which is more efficient where the measurement vector has fewer components than the state vector.

An alternative form of measurement update, known as sequential processing, is described in Section 3.2.7.

Returning to the simple example at the beginning of the subsection, a Kalman filter estimates INS position error using the INS–GNSS position solution difference as

[‡]End of QinetiQ copyright material.

the measurement, so the measurement matrix, $\mathbf{H}$, is the identity matrix. The problem may be simplified further if all components of the measurement have independent noise of standard deviation, σ_z, and the state estimates are uncorrelated and each have an uncertainty of σ_x. This is denoted Example C and may be expressed as

$$\mathbf{H}_{C,k} = \mathbf{I}_3, \qquad \mathbf{R}_{C,k} = \sigma_z^2 \mathbf{I}_3, \qquad \mathbf{P}_{C,k}^- = \sigma_x^2 \mathbf{I}_3. \tag{3.62}$$

Substituting this into (3.21), the Kalman gain matrix for this example is

$$\mathbf{K}_{C,k} = \frac{\sigma_x^2}{\sigma_x^2 + \sigma_z^2} \mathbf{I}_3. \tag{3.63}$$

From (3.24) and (3.25), the state estimates and error covariance are then updated using

$$\begin{aligned} \hat{\mathbf{x}}_{C,k}^+ &= \frac{\sigma_z^2 \hat{\mathbf{x}}_{C,k}^- + \sigma_x^2 \mathbf{z}_{C,k}}{\sigma_x^2 + \sigma_z^2} \\ \mathbf{P}_{C,k}^+ &= \frac{\sigma_z^2}{\sigma_x^2 + \sigma_z^2} \mathbf{P}_{C,k}^- = \frac{\sigma_x^2 \sigma_z^2}{\sigma_x^2 + \sigma_z^2} \mathbf{I}_3 \end{aligned}. \tag{3.64}$$

Suppose the measurement vector input to the Kalman filter, $\mathbf{z}$, is computed from another set of measurements, $\mathbf{y}$. For example, a position measurement might be converted from range and bearing to Cartesian coordinates. If the measurements, $\mathbf{y}$, have noise covariance, $\mathbf{C}_y$, the Kalman filter measurement noise covariance is determined using

$$\mathbf{R} = \left(\left. \frac{d\mathbf{z}}{d\mathbf{y}} \right|_{\mathbf{y}=\tilde{\mathbf{y}}} \right) \mathbf{C}_y \left(\left. \frac{d\mathbf{z}}{d\mathbf{y}} \right|_{\mathbf{y}=\tilde{\mathbf{y}}} \right)^{\mathrm{T}}, \tag{3.65}$$

where vector differentiation is described in Section A.5 of Appendix A on the CD. Note that $\mathbf{z}$ will typically not be a linear function of $\mathbf{y}$ as such transformations may be required where the original measurements, $\mathbf{y}$, are not a linear function of the state vector, $\mathbf{x}$. Nonlinear estimation is discussed in Sections 3.4.1, 3.4.2, and 3.5.

3.2.5 Kalman Filter Behavior and State Observability

Figure 3.7 shows how the uncertainty of a well-observed state estimate varies during the initial phase of Kalman filter operation, where the state estimates are converging with their true counterparts. Note that the state uncertainties are the root diagonals of the error covariance matrix, $\mathbf{P}$. Initially, when the state uncertainties are large, the Kalman gain will be large, weighting the state estimates towards the new measurement data. The Kalman filter estimates will change quickly as they converge with the true values of the states, so the state uncertainty will drop rapidly. However, assuming a constant measurement noise covariance, $\mathbf{R}$, this causes the Kalman gain to drop, weighting the state estimates more towards their previous values. This

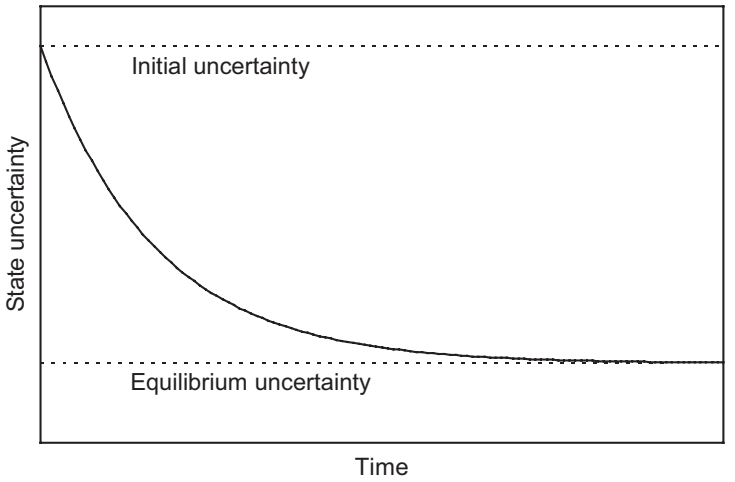

Figure 3.7 Kalman filter state uncertainty during convergence.

reduces the rate at which the states change, so the reduction in the state uncertainty slows. Eventually, the Kalman filter will approach equilibrium, whereby the decrease in state uncertainty with each measurement update is matched by the increase in uncertainty due to system noise. At equilibrium, the state estimates may still vary, but the level of confidence in those estimates, reflected by the state uncertainty, will be more or less fixed.

The rate at which a state estimate converges, if at all, depends on the *observability* of that state. There are two types of observability: deterministic, also known as geometric, and stochastic. *Deterministic observability* indicates whether there is sufficient measurement information, in the absence of noise, to independently determine all of the states, a condition known as full observability.

The Kalman filter's measurement model is analogous to a set of simultaneous equations where the states are the unknowns to be found, the measurements are the known quantities, and the measurement matrix, $\mathbf{H}$, provides the coefficients of the states. Therefore, on a single iteration, the Kalman filter cannot completely observe more states than there are components of the measurement vector, merely linear combinations of those states. However, if the measurement matrix changes over time or there is a time-dependent relationship between states through the transition matrix, $\mathbf{\Phi}$, then it is possible, over time, to observe more states than there are measurement components. The error covariance matrix, $\mathbf{P}$, records the correlations between the state estimates as well as their uncertainties. A good example in navigation is determination of velocity from the rate of change of position.

To determine whether the state vector $\mathbf{x}_1$ can be fully observed from a set of measurement vectors $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_k$, an observability matrix, $\mathbf{O}_{1:k}$ is defined by [2, 6, 11]

$$\begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \vdots \\ \mathbf{z}_k \end{pmatrix} = \mathbf{O}_{1:k}\mathbf{x}_1 + \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_k \end{pmatrix}, \tag{3.66}$$

where the noise vectors, $\mathbf{w}_i$, comprise both measurement and system noise. Thus, from (3.35) and (3.51),

$$\mathbf{O}_{1:k} = \begin{pmatrix} \mathbf{H}_1 \\ \mathbf{H}_2\mathbf{\Phi}_1 \\ \vdots \\ \mathbf{H}_k\mathbf{\Phi}_{k-1}\cdots\mathbf{\Phi}_2\mathbf{\Phi}_1 \end{pmatrix}. \tag{3.67}$$

where $\mathbf{x}_1$ is fully observable, an estimate may be obtained by applying the expectation operator to (3.66), assuming the noise distributions are zero mean. Thus,

$$\hat{\mathbf{x}}_1 = \left(\mathbf{O}_{1:k}^{\mathrm{T}}\mathbf{O}_{1:k}\right)^{-1}\mathbf{O}_{1:k}^{\mathrm{T}}\begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \vdots \\ \mathbf{z}_k \end{pmatrix}. \tag{3.68}$$

This only has a solution where the observability matrix has a pseudo-inverse, which requires $\mathbf{O}_{1:k}^{\mathrm{T}}\mathbf{O}_{1:k}$ to be nonsingular (see Section A.4 of Appendix A on the CD). This requires $\mathbf{O}_{1:k}$ to be of rank n, where n is the number of elements of the state vector, $\mathbf{x}$. The rank of a matrix is equal to the number of rows of the largest square submatrix (not necessarily continuous) with a nonzero determinant.

When $\mathbf{O}_{1:k}$ is of rank m, where $m < n$, there are m observable linear combinations of the states and $n - m$ unobservable combinations. The state vector is thus partially observable. A vector, $\mathbf{y}$, comprising m observable linear combinations of the states may be determined using [11]

$$\mathbf{y} = \mathbf{T}\mathbf{O}_m\mathbf{x}, \tag{3.69}$$

where $\mathbf{O}_m$ comprises m linearly independent rows of $\mathbf{O}_{1:k}$ and $\mathbf{T}$ is an arbitrary nonsingular $m\times m$ matrix.

An alternative method of determining full observability is to calculate the information matrix, $\mathbf{Y}$

$$\mathbf{Y} = \mathbf{H}_1^{\mathrm{T}}\mathbf{R}_1^{-1}\mathbf{H}_1 + \sum_{i=2}^{k}\left[\left(\prod_{j=1}^{i-1}\mathbf{\Phi}_{i-j}^{\mathrm{T}}\right)\mathbf{H}_i^{\mathrm{T}}\mathbf{R}_i^{-1}\mathbf{H}_i\left(\prod_{j=1}^{i-1}\mathbf{\Phi}_j\right)\right], \tag{3.70}$$

where $\prod_{j=1}^{n}\mathbf{\Phi}_j = \mathbf{\Phi}_n\cdots\mathbf{\Phi}_2\mathbf{\Phi}_1$. This is positive definite (see Section A.6 of Appendix A on the CD) where the state vector is fully observable [1, 6].

The observability of many parameters is dynamics dependent. For example, the attitude errors and accelerometer biases of an INS are not separately observable at constant attitude, but they are after a change in attitude as this changes the relationship between the states in the system model. Observation of many higher-order gyro and accelerometer errors requires much higher dynamics. However, if two states have the same effect on the measurements and vary with time and dynamics in the

same way, they will never be separately observable, so should be combined to avoid wasting processing resources.

Given that a state, or linear combination of states, is deterministically observable, the rate of convergence depends on the *stochastic observability*. This depends on the measurement sampling rate, the magnitude and correlation properties of the measurement noise, and the level of system noise. The higher the sampling rate (subject to correlation time constraints) and the lower the measurement and system noise, the greater the stochastic observability.

Conversely, system and measurement noise can mask the effects of those states that only have a small impact on the measurements, making those states effectively unobservable. For a state that is stochastically unobservable, the equilibrium state uncertainty will be similar to the initial uncertainty or may even be larger.

The combined observability of states and their combinations may be studied by analyzing the normalized error covariance matrix, $\mathbf{P}'_k$, after k Kalman filter (or covariance propagation and update) cycles. This is defined by

$$P'_{k,ij} = \frac{P_{k,ij}}{\sigma_{0,i}\sigma_{0,j}}, \tag{3.71}$$

where $\sigma_{0,i}$ is the initial uncertainty of the ith state. When a diagonal element of $\mathbf{P}'_k$ is close to zero, the corresponding state is strongly observable, whereas if it is close to unity (or larger), the state is weakly observable. As discussed above, linear combinations of weakly observable states may be strongly observable. These may be identified by calculating the eigenvalues and eigenvectors of $\mathbf{P}'_k$ (see Section A.6 of Appendix A on the CD). The eigenvectors corresponding to the smallest eigenvalues indicate the most strongly observed linear combinations of normalized states (i.e., $x_i/\sigma_{0,i}$).

When a Kalman filter is well designed, a reduction in the state uncertainty, as defined by the error covariance matrix, will be accompanied by a reduction in the corresponding state residual. Thus, the Kalman filter is convergent. However, poor design can result in state uncertainties much smaller than the corresponding state residuals or even residuals growing as the uncertainties drop, a phenomenon known as divergence. Section 3.3 discusses the causes of these problems and how they may be mitigated in a practical Kalman filter design.

3.2.6 Closed-Loop Kalman Filter

A linear system model is an assumption of the standard Kalman filter design. However, in many navigation applications, such as integration, alignment, and calibration of an INS, the true system model is not linear (i.e., the time differential of the state vector varies with terms to second order and higher in the state vector elements). One solution is to use a modified version of the Kalman filter algorithm, such as an extended Kalman filter (Section 3.4.1) or an unscented Kalman filter (Section 3.4.2). However, it is often possible to neglect the higher-order terms in the system model and still obtain a practically useful Kalman filter. The larger the values of the states that contribute to the neglected terms, the poorer a given linearity approximation will be.

A common technique for getting the best performance out of an error-state Kalman filter with a linearity approximation applied to the system model is the

closed-loop implementation. Here, the errors estimated by the Kalman filter are fed back every iteration, or at regular intervals, to correct the system itself, zeroing the Kalman filter states in the process. This feedback process keeps the Kalman filter states small, minimizing the effect of neglecting higher order products of states in the system model. Conversely, in the *open-loop implementation*, when there is no feedback, the states will generally get larger as time progresses.[†]

The best stage in the Kalman filter algorithm to feed back the state estimates is immediately after the measurement update. This produces zero state estimates at the start of the state propagation, (3.14), enabling this stage to be omitted completely. The error covariance matrix, **P**, is unaffected by the feedback process as the same amount is added to or subtracted from both the true and estimated states, so error covariance propagation, (3.15), is still required.

The closed-loop and open-loop implementations of the Kalman filter may be mixed such that some state estimates are fed back as corrections, whereas others are not. This configuration is useful for applications where feeding back states is desirable, but some states cannot be fed back as there is no way of applying them as corrections to the system. In designing such a Kalman filter, care must be taken in implementing the state propagation as for some of the fed-back states, $\mathbf{x}_k^-$ may be nonzero due to coupling with nonfed-back states through the system model.

When a full closed-loop Kalman filter is implemented (i.e., with feedback of every state estimate at every iteration), $\mathbf{H}_k\hat{\mathbf{x}}_k^-$ is zero, so the measurement, $\mathbf{z}_k$, and measurement innovation, $\delta\mathbf{z}_k^-$, are the same.

In navigation, closed-loop Kalman filters are common for the integration, alignment, and calibration of low-grade INS and may also be used for correcting GNSS receiver clocks.[‡]

3.2.7 Sequential Measurement Update

The sequential measurement-update implementation of the Kalman filter, also known as the scalar measurement update or sequential processing, replaces the vector measurement update, (3.21), (3.24), and (3.25), with an iterative process using only one component of the measurement vector at a time. The system propagation is unchanged from the standard Kalman filter implementation. For each measurement, denoted by the index *j*, the Kalman gain is calculated and the state vector estimate and error covariance matrix are updated before moving onto the next measurement. The notation $\hat{\mathbf{x}}_k^j$ and $\mathbf{P}_k^j$ is used to respectively denote the state vector estimate and error covariance that have been updated using all components of the measurement vector up to and including the *j*th. If the total number of measurements is *m*,

$$\begin{aligned} \hat{\mathbf{x}}_k^0 &\equiv \hat{\mathbf{x}}_k^-, & \mathbf{P}_k^0 &\equiv \mathbf{P}_k^- \\ \hat{\mathbf{x}}_k^m &\equiv \hat{\mathbf{x}}_k^+, & \mathbf{P}_k^m &\equiv \mathbf{P}_k^+ \end{aligned}. \tag{3.72}$$

[†]This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

[‡]End of QinetiQ copyright material.

When the components of the measurement vector are statistically independent, the measurement noise covariance matrix, $\mathbf{R}_k$, will be diagonal. In this case, the Kalman gain calculation for the jth measurement is

$$\mathbf{k}_k^j = \frac{\mathbf{P}_k^{j-1}\mathbf{H}_{k,j}^{\mathrm{T}}}{\mathbf{H}_{k,j}\mathbf{P}_k^{j-1}\mathbf{H}_{k,j}^{\mathrm{T}} + R_{k,j,j}}, \tag{3.73}$$

where $\mathbf{H}_{k,j}$ is the jth row of the measurement matrix, so $\mathbf{H}_{k,j}\mathbf{P}_k^j\mathbf{H}_{k,j}^{\mathrm{T}}$ is a scalar. Note that $\mathbf{k}_k^j$ is a column vector. The sequential measurement update equations are then

$$\begin{aligned}\hat{\mathbf{x}}_k^j &= \hat{\mathbf{x}}_k^{j-1} + \mathbf{k}_k^j\left(z_{k,j} - \mathbf{H}_{k,j}\hat{\mathbf{x}}_k^{j-1}\right)\\ &= \hat{\mathbf{x}}_k^{j-1} + \mathbf{k}_k^j\delta z_{k,j}^-\end{aligned}, \tag{3.74}$$

$$\mathbf{P}_k^j = \mathbf{P}_k^{j-1} - \mathbf{k}_k^j\left(\mathbf{H}_{k,j}\mathbf{P}_k^{j-1}\right), \tag{3.75}$$

noting that the jth component of the measurement innovation, $\delta\mathbf{z}_{k,j}^-$, must be calculated after the j–1th step of the measurement update has been performed. Because no matrix inversion is required to calculate the Kalman gain, the sequential form of the measurement update is always more computationally efficient (see Section 3.3.2) where the components of the measurement are independent.

When the measurement noise covariance, $\mathbf{R}_k$, is not diagonal, indicating measurement noise that is correlated between measurements at a given epoch, a sequential measurement update may still be performed. However, it is first necessary to reformulate the measurement into statistically independent components using

$$\begin{aligned}\mathbf{z}_k' &= \mathbf{T}_k\mathbf{z}_k\\ \mathbf{R}_k' &= \mathbf{T}_k\mathbf{R}_k\mathbf{T}_k^{\mathrm{T}},\\ \mathbf{H}_k' &= \mathbf{T}_k\mathbf{H}_k\end{aligned} \tag{3.76}$$

where the transformation matrix, $\mathbf{T}_k$, is selected to diagonalize $\mathbf{R}_k$ using Cholesky factorization as described in Section A.6 of Appendix A on the CD. The measurement update is then performed with $\mathbf{z}_k'$, $\mathbf{R}_k'$, and $\mathbf{H}_k'$ substituted for $\mathbf{z}_k$, $\mathbf{R}_k$, and $\mathbf{H}_k$ in (3.73) to (3.75). Calculation of the transformation matrix requires inversion of an $m\times m$ matrix, as is required for the conventional Kalman gain calculation. Therefore, with correlated measurement components, the sequential measurement update can only provide greater computational efficiency if the same transformation matrix is used at every epoch, k, and it is relatively sparse.

A hybrid of the sequential and conventional measurement updates may also be performed whereby the measurement vector is divided into a number of subvectors, which are then used to update the state estimates and error covariance sequentially. This can be useful where there is noise correlation within groups of measurements, but not between those groups.

3.3 Implementation Issues

This section discusses the implementation issues that must be considered in designing a practical Kalman filter. These include tuning and stability, efficient algorithm design, numerical issues, and synchronization. An overall design process is also recommended. Detection of erroneous measurements and biased state estimates is discussed in Chapter 17.

3.3.1 Tuning and Stability

The tuning of a Kalman filter is the selection by the designer or user of values for three matrices. These are the system noise covariance matrix, $\mathbf{Q}_k$, the measurement noise covariance matrix, $\mathbf{R}_k$, and the initial values of the error covariance matrix, $\mathbf{P}_0^+$. It is important to select these parameters correctly. If the values selected are too small, the actual errors in the Kalman filter estimates will be much larger than the state uncertainties obtained from **P**. Conversely, if the values selected are too large, the reported uncertainties will be too large.[*] These can cause an external system that uses the Kalman filter estimates to apply the wrong weighting to them.

However, the critical parameter in Kalman filtering is the ratio of the error and measurement noise covariance matrices, $\mathbf{P}_{\bar{k}}^-$ and $\mathbf{R}_k$, as they determine the Kalman gain, $\mathbf{K}_k$. Figure 3.8 illustrates this. If **P/R** is too small, the Kalman gain will be too small and state estimates will converge with their true counterparts more slowly than necessary. The state estimates will also be slow to respond to changes in the system. Conversely, if **P/R** is too large, the Kalman gain will be too large. This will bias the filter in favor of more recent measurements, which may result in unstable or biased state estimates due to the measurement noise having too great an influence on them. Sometimes, the state estimates can experience positive feedback of the measurement noise through the system model, causing them to rapidly diverge from their truth counterparts.[*]

In an ideal Kalman filter application, tuning the noise models to give consistent estimation errors and uncertainties will also produce stable state estimates that track their true counterparts. However, in practice, it is often necessary to tune the filter to give 1σ state uncertainties substantially larger (two or three times is typical) than the corresponding error standard deviations in order to maintain stability. This is because the Kalman filter's model of the system is only an approximation of the real system.

There are a number of sources of approximation in a Kalman filter. Smaller error states are often neglected due to observability problems or processing-capacity limitations. The system and/or measurement models may have to be approximated to meet the linearity requirements of the Kalman filter equations. The stochastic properties of slowly time-varying states are often oversimplified. Nominally constant states may also vary slowly with time (e.g., due to temperature or pressure changes). Finally, the Kalman filter assumes that all noise sources are white, whereas, in practice, they

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

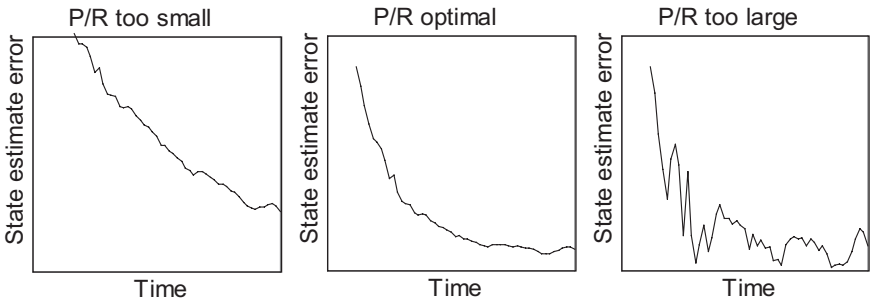

Figure 3.8 Kalman filter error propagation for varying **P**/**R** ratio.

will exhibit some time correlation due to band-limiting effects. Therefore, to overcome the limitations of the Kalman filter model, sufficient noise must be modeled to overbound the real system's behavior. By analogy, to fit the square peg of the real world problem into the round hole of the Kalman filter model, the hole must be widened to accommodate the edges of the peg.

A further issue is that allowing state uncertainties to become very small can precipitate numerical problems (see Section 3.3.3). Therefore, it is advisable to model system noise on all states and ensure that $\mathbf{Q}_k$ is positive definite (see Section A.6 of Appendix A on the CD). Alternatively, lower limits to the state uncertainties may be maintained.

For most applications, manufacturers' specifications and laboratory test data may be used to determine suitable initial values for the error covariance matrix. The same approach may be adopted for the system noise covariance matrices in cases in which the system model is a good representation of the truth and the system noise is close to white. In other cases, the system noise may be highly colored or dominated by the compensation of modeling approximations, in which case a more empirical approach will be needed, making use of test data gathered in typical operational environments. It may also be necessary to model the system noise as a function of vibration and/or user dynamics.

Similarly, when the measurement noise is close to being white, manufacturer's specifications or simple laboratory variance measurements may be used. However, it is often necessary to exaggerate **R** in order to account for time correlation in the measurement noise due to band-limiting or synchronization errors, while measurement noise can also vary with vibration and user dynamics. For radio navigation, the measurement noise is also affected by signal reception conditions.

In tuning a Kalman filter, it can be difficult to separate out the effects of measurement noise from those of the system noise and modeling limitations. Therefore, a good tuning philosophy is to fix $\mathbf{P}_0^+$, together with whichever of $\mathbf{Q}_k$ and $\mathbf{R}_k$ is easier to define analytically, then vary the remaining tuning matrix by trial and error to find the smallest value that gives stable state estimates. If this does not give satisfactory performance, the other tuning parameters can also be varied. Automatic real-time tuning techniques are discussed in Section 3.4.4.

Tuning a Kalman filter is essentially a tradeoff between convergence rate and stability. However, it is important to note that the convergence rate can also affect

Table 3.1 Multiplications and Additions Required by Kalman Filter Processes

Kalman Filter Process	*Equation*	*Multiplications Required*	*Additions Required*
System-propagation phase			
State propagation	(3.14)	n^2	$n(n-1)$
Covariance propagation	(3.15)	$2n^3$	$n^2(2n-1)$
System noise distribution matrix computation	(3.41)	$n(n+1)l$	$n^2(l-1)$
Measurement-update phase (vector implementation)			
Kalman gain calculation	(3.21)	$2mn^2 = 2m^2n$	$mn(m+n-2)+m^2$
Matrix inversion		$(3/2)m^3-(1/2)m$	$\sim m^3$
State vector update	(3.24)	$2mn$	$2nm$
Covariance update	(3.25)	mn^2+n^3	$n^2(n+m-1)$
	or (3.61)	$2mn^2$	$mn(2n-1)$
Measurement-update phase (sequential implementation, assuming diagonal **R**)			
Kalman gain calculation	(3.73)	$2mn^2+2mn$	$m(n^2-n+1)$
State vector update	(3.74)	$2mn$	$2nm$
Covariance update	(3.75)	$2mn^2$	$mn(2n-1)$

the long-term accuracy, as this is reached once the convergence rate matches the rate at which the true states change due to noise effects. For some Kalman filtering applications, integrity monitoring techniques (Chapter 17) can be used to detect and remedy state instability, in which case the tuning may be selected to optimize convergence.*

3.3.2 Algorithm Design

The processing load for implementation of a Kalman filter depends on the number of components of the state vector, n, measurement vector, m, and system noise vector, l, as shown in Table 3.1. When the number of states is large, the covariance propagation and update require the largest processing capacity. However, when the measurement vector is larger than the state vector, the Kalman gain calculation has the largest impact on processor load for the vector implementation of the measurement update. Therefore, implementing a sequential measurement update can significantly reduce the processor load when there are a large number of uncorrelated measurement components at each epoch.

In moving from a theoretical to a practical Kalman filter, a number of modifications can be made to improve the processing efficiency without significantly impacting on performance. For example, many elements of the transition, $\mathbf{\Phi}_k$, and measurement, $\mathbf{H}_k$, matrices are zero, so it is more efficient to use sparse matrix multiplication routines that only multiply the nonzero elements. However, there is a tradeoff between processing efficiency and algorithm complexity, with more complex algorithms taking longer to develop, code, and debug.*

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

Another option takes advantage of the error covariance matrix, $\mathbf{P}_k$, being symmetric about the diagonal. By computing only the diagonal elements and either the upper or lower triangle, the computational effort required to propagate and update the covariance matrix may be almost halved.

Sparse matrix multiplication cannot be used for the matrix inversion within the Kalman gain calculation, while its use in updating the covariance, (3.25), is limited to computing $\mathbf{K}_k\mathbf{H}_k$ and cases in which $\mathbf{H}_k$ has some columns that are all zeros. Consequently, the measurement-update phase of the Kalman filter will always require more computational capacity than the system-propagation phase. The interval between measurement updates may be limited by processing power. It may also be limited by the rate at which measurements are available or by the correlation time of the measurement noise. In any case, the measurement-update interval can sometimes be too large to calculate the transition matrix, $\mathbf{\Phi}_k$, over. This is because the system propagation interval, τ_s, must be sufficiently small for the system matrix, $\mathbf{F}$, to be treated as constant and the power-series expansion of $\mathbf{F}\tau_s$ in (3.34) to converge. However, the different phases of the Kalman filter do not have to be iterated at the same rate. The system propagation may be iterated at a faster rate than the measurement update, reducing the propagation interval, τ_s. Similarly, if a measurement update cannot be performed due to lack of valid data, the system propagation can still go ahead. The update rate for a given measurement stream should not be faster than the system-propagation rate.

The Kalman filter equations involving the covariance matrix, $\mathbf{P}$, impose a much higher computational load than those involving the state vector, $\mathbf{x}$. However, the accuracy requirement for the state vector is higher, particularly for the open-loop Kalman filter, requiring a shorter propagation interval to maximize the transition matrix accuracy. Therefore, it is sometimes more efficient to iterate the state vector propagation, (3.14), at a higher rate than the error covariance propagation, (3.15).

When the measurement update interval that processing capacity allows is much greater than the noise correlation time of the measurement stream, the noise on the measurements can be reduced by time averaging. In this case, the measurement innovation, $\delta\mathbf{z}^-$, is calculated at a faster rate and averaged measurement innovations are used to update the state estimates, $\hat{\mathbf{x}}$, and covariance, $\mathbf{P}$, at the rate allowed by the processing capacity. When the measurements, $\mathbf{z}$, rather than the measurement innovations, are averaged, the measurement matrix, $\mathbf{H}$, must be modified to account for the state propagation over the averaging interval [12]. Measurement averaging is also known as prefiltering.

Altogether, a Kalman filter algorithm may have four different iteration rates for the state propagation, (3.14), error covariance propagation, (3.15), measurement accumulation, and measurement update, (3.21), (3.24), and (3.25). Figure 3.9 presents an example illustration. Furthermore, different types of measurement input to the same Kalman filter, such as position and velocity or velocity and attitude, may be accumulated and updated at different rates.*

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

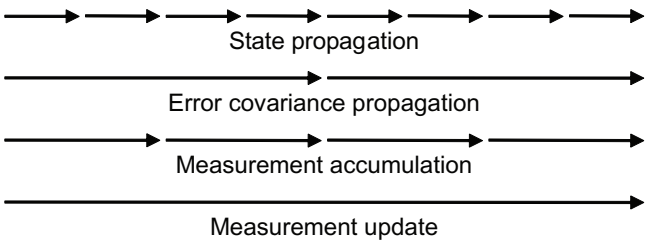

Figure 3.9 Example Kalman filter iteration rates.

3.3.3 Numerical Issues

When a Kalman filter is implemented on a computer, the precision is limited by the number of bits used to store and process each parameter. The fewer bits used, the larger the rounding errors on each computation will be. Thus, double-precision (64-bit) arithmetic is more robust than single precision (32 bit), which is more robust than 16-bit arithmetic, used in early implementations.

The effect of rounding errors on the state estimates can be accounted for by increasing the system noise covariance, **Q**, and, in many cases, is corrected by the Kalman filter's measurement update process. However, there are no corresponding corrections to the error covariance matrix, **P**. The longer the Kalman filter has been running and the higher the iteration rate, the greater the distortion of the matrix. This distortion manifests as breakage of the symmetry about the diagonal and can even produce negative diagonal elements, which represent imaginary uncertainty. Small errors in the **P** matrix are relatively harmless. However, large **P**-matrix errors distort the Kalman gain matrix, **K**. Gains that are too small produce unresponsive state estimates while gains that are too large can produce unstable, oscillatory state estimates. If an element of the Kalman gain matrix is the wrong sign, a state estimate is liable to diverge away from truth. Extreme covariance matrix distortion can also cause software crashes. Thus, the Kalman filter implementation must be designed to minimize computational errors in the error covariance matrix. In particular, **P** must remain positive definite (i.e., retain a positive determinant and positive eigenvalues).

There is a particular risk of numerical problems at the first measurement update following initialization in cases where the initial uncertainties are very large and the measurement noise covariance is small. This is because there can be a very large change in the error covariance matrix, with the covariance update comprising the multiplication of very large numbers with very small numbers. If problems occur, the initial state uncertainties should be set artificially small. As long as the values used are still larger than those expected after convergence, the state uncertainties will be corrected as the Kalman filter converges [4].

In general, rounding errors may be reduced by scaling the Kalman filter states so that all state uncertainties are of a similar order of magnitude in numerical terms, effectively reducing the dynamic range of the error covariance matrix. Rescaling of the measurement vector may also be needed to reduce the dynamic range of the $\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k$ matrix that is inverted to calculate the Kalman gain. Scaling is essential where fixed-point, as opposed to floating-point, arithmetic is used.

Another way of minimizing the effects of rounding errors is to modify the Kalman filter algorithm. The Joseph form of the covariance update replaces (3.25) with (3.58). It has greater symmetry than the standard form, but requires more than twice the processing capacity. A common approach is covariance factorization. These techniques effectively propagate $\sqrt{\mathbf{P}}$ rather than $\mathbf{P}$, reducing the dynamic range by a factor of 2 so that rounding errors have less impact. A number of factorization techniques are reviewed in [3, 5, 6], but the most commonly used is the Bierman-Thornton or UDU method [3, 13].

Ad hoc methods of stabilizing the error covariance matrix include forcibly maintaining symmetry by averaging the $\mathbf{P}$-matrix with its transpose after each system propagation and measurement update, and applying minimum values to the state uncertainties.

3.3.4 Time Synchronization

Different types of navigation system exhibit different data lags between the time at which sensor measurements are taken, known as the *time of validity*, and the time when a navigation solution based on those measurements is output. There may also be a communication delay between the navigation system and the Kalman filter processor. When Kalman filter measurements compare the outputs of two different navigation systems, it is important to ensure that those outputs correspond to the same time of validity. Otherwise, differences in the navigation system outputs due to the time lag between them will be falsely attributed by the Kalman filter to the states, corrupting the estimates of those states. The greater the level of dynamics encountered, the larger the impact of a given time-synchronization error will be. Poor time synchronization can be mitigated by using very low gains in the Kalman filter; however, it is better to synchronize the measurement data.

Data synchronization requires the outputs from the faster responding system, such as an INS, to be stored. Once an output is received from the slower system, such as a GNSS receiver, an output from the faster system with the same time of validity is retrieved from the store and used to form a synchronized measurement input to the Kalman filter. Figure 3.10 illustrates the architecture. It is usually better to interpolate the data in the store rather than use the nearest point in time. Data-lag compensation is more effective where all data is time-tagged, enabling precise synchronization. When time tags are unavailable, data lag compensation may operate using an assumed average time delay, provided this is known to within about 10 ms and the actual lag does not vary by more than about ±100 ms.[*] It is also possible to estimate the time lag of one data stream with respect to another as a Kalman filter state (see Section I.6 of Appendix I on the CD).

The system-propagation phase of the Kalman filter usually uses data from the faster responding navigation system. Consequently, the state estimates may be propagated to a time ahead of the measurement time of validity. The optimal solution is to postmultiply the measurement matrix, $\mathbf{H}$, by a transition matrix, $\mathbf{\Phi}$, that propagates from the state time of validity, t_s, to the measurement time of validity, t_m. Thus,

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinteQ copyright material.

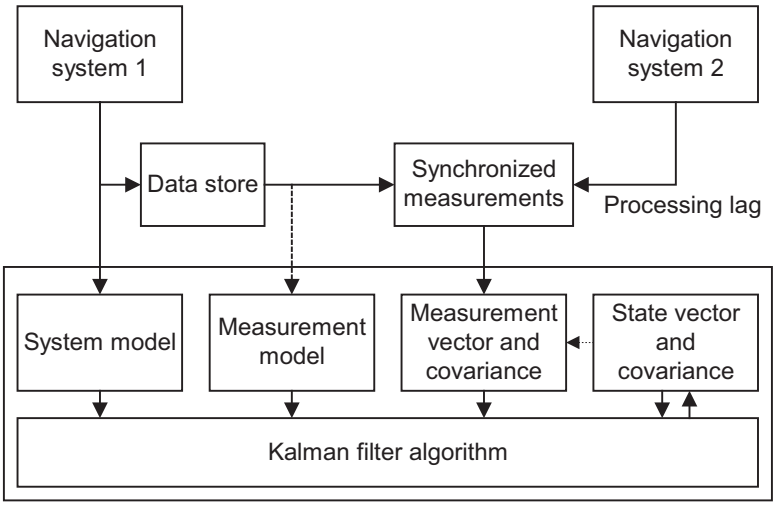

Figure 3.10 Data synchronization (open-loop Kalman filter).

$$\mathbf{H}(t_s) = \mathbf{H}(t_m)\boldsymbol{\Phi}(t_m, t_s). \tag{3.77}$$

Another option is simply to limit the extent of the system propagation to the measurement time of validity. However, the simplest solution is simply to increase the measurement noise covariance, **R**, to account for the effects of the timing offset, noting that these may be time correlated. The choice of solution is a tradeoff between performance impact, processing load, and complexity and will depend on the requirements for the application concerned.

For some high-integrity applications, there may a lag of tens of seconds between the measurement time of validity and its application. This is to enable multi-epoch fault detection tests (see Chapter 17) to be performed on the measurement stream prior to accepting the measurements. In this case, (3.77) should be used to link the measurements to the current states within the measurement model.

A similar problem is the case where the measurement vector comprises measurements with different times of validity. Again, the simplest option is to increase **R**, while an optimal approach is to postmultiply the relevant rows of the measurement matrix by transition matrices that propagate between the state and measurement times of validity. Another optimal approach is to divide the measurement vector up and perform a separate measurement update for each time of validity, interspersed with a Kalman filter system propagation.

When the closed-loop correction of the navigation system(s) under calibration by the Kalman filter is used, data-delay compensation introduces a delay in applying the corrections to the Kalman filter measurement stream. Further delays are introduced by the time it takes to process the Kalman filter measurement update and communicate the correction. Figure 3.11 illustrates this. As a result of these lags, one or more uncorrected measurement set may be processed by the Kalman filter, causing the closed-loop correction to be repeated. Overcorrection of a navigation system can cause instability with the navigation solution oscillating about the truth. The optimal solution to this problem is to apply corrections to the measurement innovations or

the data store in Figure 3.10. However, a simpler solution is to down-weight the Kalman gain, **K**, either directly or via the measurement noise covariance, **R**.

Sometimes measurements input to a Kalman filter may be the sum, average, or difference of data with different times of validity. In this case, the measurement and state vectors cannot be synchronized to a common time of validity. For summed and averaged measurements, **R** can be simply increased to compensate if the performance requirements allow for this. However, for time-differenced measurements, the Kalman filter must explicitly model the different times of validity, otherwise the measurement matrix would be zero. There are two ways of doing this. Consider a measurement model of the form

$$\mathbf{z}(t) = \mathbf{H}(t)\big(\mathbf{x}(t) - \mathbf{x}(t-\tau)\big) + \mathbf{w}_m(t). \tag{3.78}$$

One solution handles the time propagation within the system model by augmenting the state vector at time t with a replica valid at time $t-\tau$. These additional states are known as delayed states. The combined state vector, transition matrix, system noise covariance matrix, and measurement matrix, denoted by the superscript C thus become

$$\begin{aligned}
\mathbf{x}^{\mathrm{C}}(t) &= \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{x}(t-\tau) \end{pmatrix}, & \boldsymbol{\Phi}^{\mathrm{C}}(t,t-\tau) &= \begin{pmatrix} \boldsymbol{\Phi}(t,t-\tau) & \mathbf{0} \\ \mathbf{I} & \mathbf{0} \end{pmatrix} \\
\mathbf{Q}^{\mathrm{C}}(t,t-\tau) &= \begin{pmatrix} \mathbf{Q}(t,t-\tau) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, & \mathbf{H}^{\mathrm{C}}(t) &= \begin{pmatrix} \mathbf{H}(t) & -\mathbf{H}(t) \end{pmatrix}
\end{aligned}, \tag{3.79}$$

where $\boldsymbol{\Phi}(t,t-\tau)$ is the continuous-time transition matrix for the state vector between times $t-\tau$ and t, noting that $\boldsymbol{\Phi}(t_k,t_k-\tau_s) = \boldsymbol{\Phi}_{k-1}$. Similarly, $\mathbf{Q}(t,t-\tau)$ is the continuous-time system noise covariance matrix. This enables the standard Kalman filter measurement model, (3.50), to be used. In practice, only those components of $\mathbf{x}(t-\tau)$ to which the measurement matrix directly couples need be included in the state vector.

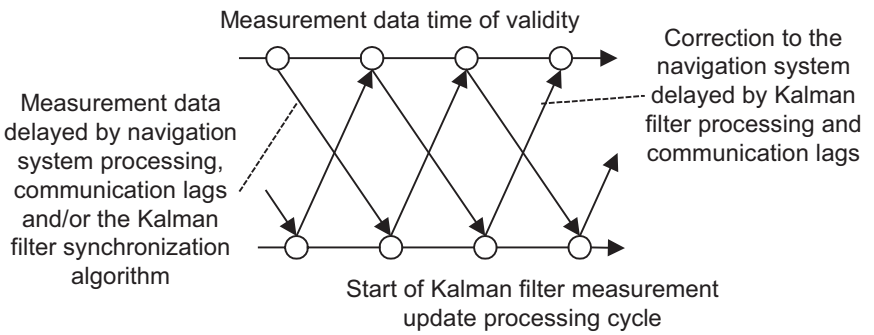

Figure 3.11 Processing lag in a closed-loop Kalman filter.

The other solution incorporates the time propagation of the state vector between epochs within the measurement model by replacing the measurement matrix with

$$\begin{aligned}\mathbf{H}'(t) &= \mathbf{H}(t)\left[\mathbf{I} - \boldsymbol{\Phi}(t-\tau,t)\right] \\ &= \mathbf{H}(t)\left[\mathbf{I} - \boldsymbol{\Phi}(t,t-\tau)^{-1}\right]\end{aligned} \tag{3.80}$$

and retaining the conventional single-epoch state vector. This imposes a lower processing load than the first method but neglects the effect of the system noise between times $t - \tau$ and t. Therefore, it should be validated against the first method before use. Both methods are extendable to measurement averaging and summing over multiple epochs.

3.3.5 Kalman Filter Design Process

A good design philosophy [14] for a Kalman filter is to first select as states all known errors or properties of the system which are modelable, observable, and contribute to the desired output of the overall system, generally a navigation solution. This is sometimes known as the *truth model*. System and measurement models should then be derived based on this state selection.

A software simulation should be developed, containing a version of the Kalman filter in which groups of states may be deselected and different phases of the algorithm run at different rates. With all states selected and all Kalman filter phases run at the fastest rate, the filter should be tuned and tested to check that it meets the requirements. Processor load need not be a major consideration at this stage.

Assuming the requirements are met, simulation runs should then be conducted with different groups of Kalman filter states de-selected and their effects modeled as system noise. Runs should also be conducted with phases of the Kalman filter run at a range of slower rates. Combinations of these configurations should also be investigated. Those changes that have the least effect on Kalman filter performance for a given reduction in processor load should then be implemented in turn until the computational load falls within the available processing capacity.

The reduced Kalman filter, sometimes known as the *design model*, should then be carefully retuned and assessed by simulation and trials to verify its performance.

3.4 Extensions to the Kalman Filter

The derivation of the Kalman filter algorithm is based on a number of assumptions about the properties of the states estimated and noise sources accounted for. However, these assumptions do not always apply to real navigation systems. This section looks at how the basic Kalman filter technique may be extended to handle a nonlinear measurement or system model, time-correlated noise, unknown system or measurement noise standard deviations, and non-Gaussian measurement distributions. In addition, Kalman smoothing techniques, which take advantage of the extra information available in postprocessed applications, are discussed.

3.4.1 Extended and Linearized Kalman Filter

In a standard Kalman filter, the measurement model is assumed to be linear (i.e., the measurement vector, $\mathbf{z}$, is a linear function of the state vector, $\mathbf{x}$). This is not always the case for real systems. In some applications, such as most INS alignment and calibration problems, a linear approximation of the measurement model is useful, though this can introduce small errors. However, for applications processing ranging measurements, such as a GNSS navigation filter, the measurement model is highly nonlinear.*

The system model is also assumed to be linear in the standard Kalman filter (i.e., $\dot{\mathbf{x}}$ is a linear function of $\mathbf{x}$). Closed-loop correction of the system using the state estimates (Section 3.2.6) can often be used to maintain a linear approximation in the system model. However, it is not always possible to perform the necessary feedback to the system. An example of this is total-state INS/GNSS integration (see Section 14.1.1), where the absolute position, velocity, and attitude are estimated rather than the errors therein.

A nonlinear version of the Kalman filter is the *extended Kalman filter*. In an EKF, the system matrix, $\mathbf{F}$, and measurement matrix, $\mathbf{H}$, can be replaced in the state propagation and update equations by nonlinear functions of the state vector, respectively, $\mathbf{f}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$. It is common in navigation applications to combine the measurement-update phase of the EKF with the system-propagation phase of the standard Kalman filter. The reverse combination may also be used, though it is rare in navigation.

The system dynamic model of the EKF is

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t),t\big) + \mathbf{G}(t)\mathbf{w}_s(t), \tag{3.81}$$

where the nonlinear function of the state vector, $\mathbf{f}$, replaces the product of the system matrix and state vector and the other terms are as defined in Section 3.2.3. The state vector propagation equation is thus

$$\hat{\mathbf{x}}_k^- = \hat{\mathbf{x}}_{k-1}^+ + \int_{t_k-\tau_s}^{t_k} \mathbf{f}\big(\hat{\mathbf{x}}(t'),t'\big)\,dt', \tag{3.82}$$

replacing (3.14). When $\mathbf{f}$ may be assumed constant over the propagation interval, this simplifies to

$$\hat{\mathbf{x}}_k^- = \hat{\mathbf{x}}_{k-1}^+ + \mathbf{f}\big(\hat{\mathbf{x}}_{k-1}^+,t_k\big)\tau_s, \tag{3.83}$$

In the EKF, it is assumed that the error in the state vector estimate is much smaller than the state vector, enabling a linear system model to be applied to the state vector residual:

$$\delta\dot{\mathbf{x}}(t) = \mathbf{F}(t)\delta\mathbf{x}(t) + \mathbf{G}(t)\mathbf{w}_s(t). \tag{3.84}$$

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

The conventional error covariance propagation equation, (3.15), may thus be used with the system matrix linearized about the state vector estimate using

$$\mathbf{F}_{k-1} = \left.\frac{\partial \mathbf{f}(\mathbf{x},t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1}^{+}}. \tag{3.85}$$

Provided the propagation interval, $\tau_s = t_k - t_{k-1}$, is sufficiently small for the approximation $\mathbf{f}(\mathbf{x},t_k) \approx \mathbf{f}(\mathbf{x},t_{k-1})$ to be valid, the transition matrix is calculated using (3.33):

$$\boldsymbol{\Phi}_{k-1} \approx \exp(\mathbf{F}_{k-1}\tau_s),$$

which is solved using a power-series expansion as in the conventional Kalman filter.

The measurement model of the EKF is

$$\mathbf{z}(t) = \mathbf{h}\big(\mathbf{x}(t),t\big) + \mathbf{w}_m(t), \tag{3.86}$$

where $\mathbf{h}$ is a nonlinear function of the state vector. The state vector is then updated with the true measurement vector using

$$\begin{aligned}\hat{\mathbf{x}}_k^{+} &= \hat{\mathbf{x}}_k^{-} + \mathbf{K}_k\left[\mathbf{z}_k - \mathbf{h}\big(\hat{\mathbf{x}}_k^{-},t_k\big)\right],\\ &= \hat{\mathbf{x}}_k^{-} + \mathbf{K}_k\delta\mathbf{z}_k^{-}\end{aligned} \tag{3.87}$$

replacing (3.24), where from (3.9) and (3.86), the measurement innovation is

$$\begin{aligned}\delta\mathbf{z}_k^{-} &= \mathbf{z}_k - \mathbf{h}\big(\hat{\mathbf{x}}_k^{-},t_k\big)\\ &= \mathbf{h}\big(\mathbf{x}_k,t_k\big) - \mathbf{h}\big(\hat{\mathbf{x}}_k^{-},t_k\big) + \mathbf{w}_{mk}\end{aligned}. \tag{3.88}$$

Once the state vector estimate has converged with its true counterpart, the measurement innovations will be small, so they can legitimately be modeled as a linear function of the state vector where the full measurements cannot. Thus,

$$\delta\mathbf{z}_k^{-} \approx \mathbf{H}_k\delta\mathbf{x}_k^{-} + \mathbf{w}_{mk}, \tag{3.89}$$

where

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x},t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k^{-}} = \left.\frac{\partial \mathbf{z}(\mathbf{x},t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k^{-}}. \tag{3.90}$$

A consequence of this linearization of $\mathbf{F}$ and $\mathbf{H}$ is that the error covariance matrix, $\mathbf{P}$, and Kalman gain, $\mathbf{K}$, are functions of the state estimates. This can occasionally cause stability problems and the EKF is more sensitive to the tuning of the $\mathbf{P}$-matrix initialization than a standard Kalman filter.*

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

For the EKF to be valid, the values of **F** and **H** obtained by, respectively, linearizing the system and measurement models about the state vector estimate must be very close to the values that would be obtained if they were linearized about the true state vector. Figures 3.12 and 3.13 show examples of, respectively, valid and invalid linearization of single-state system and measurement models.

One way to assess whether the system model linearization is valid is to test for the condition:

$$\left.\frac{\partial \mathbf{f}(\mathbf{x},t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1}^{+}+\Delta\mathbf{x}_{k-1}^{+i}} \approx \left.\frac{\partial \mathbf{f}(\mathbf{x},t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1}^{+}-\Delta\mathbf{x}_{k-1}^{+i}}, \qquad \Delta x_{k-1,j}^{+i} = \begin{cases} \sqrt{P_{k-1,i,i}^{+}} & j = i \\ 0 & j \neq i \end{cases} \tag{3.91}$$

for each state, i. This determines whether there is significant variation in the gradient of the system function, **f**, over the uncertainty bounds of the state vector estimate.

Similarly, the validity of the measurement model linearization may be assessed by testing for the condition

$$\left.\frac{\partial \mathbf{h}(\mathbf{x},t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k}^{-}+\Delta\mathbf{x}_{k}^{-i}} \approx \left.\frac{\partial \mathbf{h}(\mathbf{x},t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k}^{-}-\Delta\mathbf{x}_{k}^{-i}}, \qquad \Delta x_{k,j}^{-i} = \begin{cases} \sqrt{P_{k,i,i}^{-}} & j = i \\ 0 & j \neq i \end{cases}, \tag{3.92}$$

again, for each state, i. This determines whether the gradient of the measurement function, **h**, varies significantly over the uncertainty bounds of the state vector estimate. A more sophisticated approach is described in [15].

When the above conditions are not met, the error covariance computed by the EKF will tend to be overoptimistic, which can eventually lead to divergent state estimates. This is most likely to happen at initialization, when the error covariance

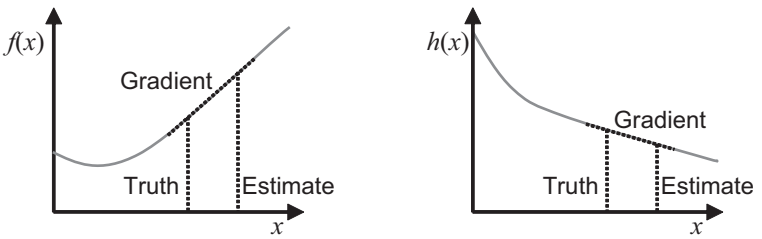

Figure 3.12 Example of valid system and measurement model linearization using an EKF (gradients are the same for the true and estimated values of x).

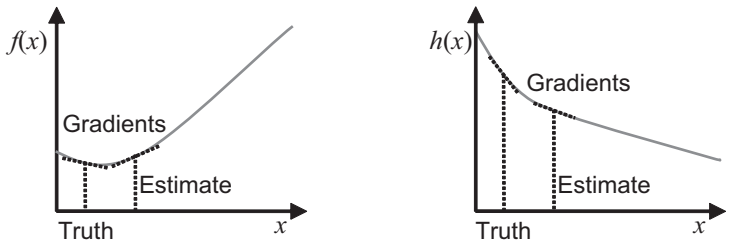

Figure 3.13 Example of invalid system and measurement model linearization using an EKF (gradients are different for the true and estimated values of x).

matrix, **P**, is normally at its largest. When the validity of the system model linearization is marginal, the system noise covariance, **Q**, may be increased to compensate. Similarly, the measurement noise covariance, **R**, may be increased where the validity of the measurement model linearization is marginal. However, if either linearization is clearly invalid, a higher-order approach must be used, such as the unscented Kalman filter (Section 3.4.2), the iterated EKF, or the second-order EKF [2, 6]. A new approach, designed to handle multiple classes of second-order problem, is the intelligent method for recursive estimation (IMRE) Kalman filter [16].

An alternative to the EKF that maintains an error covariance and Kalman gain that are independent of the state estimates is the linearized Kalman filter. This takes the same form as the EKF with the exception that the system and measurement models are linearized about a predetermined state vector, $\mathbf{x}^P$:

$$\mathbf{F}_{k-1} = \left.\frac{\partial \mathbf{f}(\mathbf{x}, t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{k-1}^P}, \quad \mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x}, t_k)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_k^P}. \tag{3.93}$$

For this to be valid, the above values of **F** and **H** must be very close to those that would be obtained if the system and measurement models were linearized about the true state vector. Therefore, it is not generally suited to cases where the EKF is invalid. A suitable application is guided weapons, where the approximate trajectory is known prior to launch and the Kalman filter is estimating the navigation solution.

3.4.2 Unscented Kalman Filter

The unscented Kalman filter, also known as the sigma-point Kalman filter, is a nonlinear adaptation of the Kalman filter that does not require the gradients of the system function, **f**, and measurement function, **h**, to be approximately constant over the uncertainty bounds of the state estimate [6, 17].

The UKF relies on the *unscented transformation* from an n-element state vector estimate, $\hat{\mathbf{x}}$, and its error covariance matrix, **P**, to a set of $2n$ parallel state vectors, known as sigma points. The transform is reversible as the mean and variance of the sigma points are the state vector estimate and error covariance matrix, respectively. There are a number of different types of unscented transformation [6]; the root-covariance type is used here. Like the standard Kalman filter and the EKF, the UKF assumes that all states and noise sources have distributions that can be described using only a mean and covariance (e.g., the Gaussian distribution).

For applications where only the system model is significantly nonlinear, the system-propagation phase of the UKF may be combined with the measurement-update phase of a conventional Kalman filter or EKF. Similarly, when only the measurement model is significantly nonlinear, a UKF measurement update may be combined with the system propagation of an EKF or conventional Kalman filter. In navigation, the UKF system propagation is useful for applications where there are large attitude uncertainties, while the UKF measurement update is useful where there is ranging between a transmitter and receiver a short distance apart.

The first step in the system-propagation phase of the UKF is to obtain the square root of the error covariance matrix, $\mathbf{S}_{k-1}^+$, by using Cholesky factorization (see Section A.6 of Appendix A on the CD) to solve

$$\mathbf{P}_{k-1}^{+} = \mathbf{S}_{k-1}^{+}\mathbf{S}_{k-1}^{+\mathrm{T}}. \tag{3.94}$$

Next, the sigma points are calculated using:

$$\mathbf{x}_{k-1}^{+(i)} = \begin{array}{ll} \hat{\mathbf{x}}_{k-1}^{+} + \sqrt{n}\mathbf{S}_{k-1,:,i}^{+} & i \leq n \\ \hat{\mathbf{x}}_{k-1}^{+} - \sqrt{n}\mathbf{S}_{k-1,:,(i-n)}^{+} & i > n \end{array}, \tag{3.95}$$

where the subscript $:,i$ denotes the ith column of the matrix.

Each sigma point is then propagated through the system model using

$$\mathbf{x}_{k}^{-(i)} = \mathbf{x}_{k-1}^{+(i)} + \mathbf{f}\left(\mathbf{x}_{k-1}^{+(i)}, t_k\right)\tau_s, \tag{3.96}$$

where $\mathbf{f}$ is assumed constant over the propagation interval; otherwise, (3.82) must be used. The propagated state estimate and its error covariance are then calculated using

$$\hat{\mathbf{x}}_{k}^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\mathbf{x}_{k}^{-(i)}, \tag{3.97}$$

$$\mathbf{P}_{k}^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\left(\mathbf{x}_{k}^{-(i)} - \hat{\mathbf{x}}_{k}^{-}\right)\left(\mathbf{x}_{k}^{-(i)} - \hat{\mathbf{x}}_{k}^{-}\right)^{\mathrm{T}} + \mathbf{Q}_{k-1}, \tag{3.98}$$

assuming that the system noise may be propagated linearly through the system model. Otherwise, the sigma point state vectors are augmented to incorporate system noise terms.

The measurement-update phase of the UKF begins by generating new sigma points using

$$\mathbf{x}_{k}^{-(i)} = \begin{array}{ll} \hat{\mathbf{x}}_{k}^{-} + \sqrt{n}\mathbf{S}_{k,:,i}^{-} & i \leq n \\ \hat{\mathbf{x}}_{k}^{-} - \sqrt{n}\mathbf{S}_{k,:,(i-n)}^{-} & i > n \end{array}, \qquad \mathbf{P}_{k}^{-} = \mathbf{S}_{k}^{-}\mathbf{S}_{k}^{-\mathrm{T}}. \tag{3.99}$$

This step may be omitted to save processing capacity, using the sigma points from the system propagation phase, instead. However, there is some degradation in performance.

The sigma point and mean measurement innovations are calculated using

$$\begin{aligned} \delta\mathbf{z}_{k}^{-(i)} &= \mathbf{z}_k - \mathbf{h}\left(\hat{\mathbf{x}}_{k}^{-(i)}, t_k\right) \\ \delta\mathbf{z}_{k}^{-} &= \frac{1}{2n}\sum_{i=1}^{2n}\delta\mathbf{z}_{k}^{-(i)} \end{aligned}. \tag{3.100}$$

The covariance of the measurement innovations is given by

$$\mathbf{C}_{\delta z,k}^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\left(\delta\mathbf{z}_{k}^{-(i)} - \delta\mathbf{z}_{k}^{-}\right)\left(\delta\mathbf{z}_{k}^{-(i)} - \delta\mathbf{z}_{k}^{-}\right)^{\mathrm{T}} + \mathbf{R}_k, \tag{3.101}$$

assuming that the measurement noise may be propagated linearly through the measurement model. Otherwise, the sigma point state vectors are augmented to incorporate measurement noise terms.

Finally, the Kalman gain, state vector update and error covariance update of the UKF are

$$\mathbf{K}_k = \left[\frac{1}{2n}\sum_{i=1}^{2n}\left(\mathbf{x}_k^{-(i)} - \hat{\mathbf{x}}_k^-\right)\left(\delta\mathbf{z}_k^{-(i)} - \delta\mathbf{z}_k^-\right)^{\mathrm{T}}\right]\left(\mathbf{C}_{\delta\mathbf{z},k}^-\right)^{-1}, \tag{3.102}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\delta\mathbf{z}_k^-, \tag{3.103}$$

$$\mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k\mathbf{C}_{\delta\mathbf{z},k}^-\mathbf{K}_k^{\mathrm{T}}. \tag{3.104}$$

The system-propagation phase of the UKF may be combined with the measurement-update phase of the standard Kalman filter or EKF, or vice versa.

3.4.3 Time-Correlated Noise

In Kalman filtering, it is assumed that all measurement errors, $\mathbf{w}_m$, are time uncorrelated; in other words, the measurement noise is white. In practice this is often not the case. For example, Kalman filters in navigation often input measurements output by another Kalman filter, a loop filter, or another estimation algorithm. There may also be time-correlated variation in the lever arm between navigation systems. A Kalman filter attributes the time-correlated parts of the measurement innovations to the states. Consequently, correlated measurement noise can potentially corrupt the state estimates.

There are three main ways to account for time-correlated measurement noise in a Kalman filter. The optimal solution is to estimate the time-correlated noise as additional Kalman filter states. However, this may not be practical due to observability or processing capacity limitations. The second, and simplest, option is to reduce the gain of the Kalman filter. The measurement update interval may be increased to match the measurement noise correlation time; the assumed measurement noise covariance, **R**, may be increased; or the Kalman gain, **K**, down-weighted. Measurement averaging may be used in conjunction with an increased update interval, provided the averaged measurement is treated as a single measurement for statistical purposes. These gain-reduction techniques will all increase the time it takes the Kalman filter to converge and the uncertainty of the estimates at convergence. The third method of handling time-correlated noise is to a use a Schmidt-Kalman filter with uncertain measurement noise parameters [18]. This effectively increases the error covariance matrix, **P**, to model the time-correlated noise and is described in Section D.2 of Appendix D on the CD.

Another assumption of Kalman filters is that the system noise, $\mathbf{w}_s$, is not time correlated. However, the system often exhibits significant systematic and other time-correlated errors that are not estimated as states due to observability or processing power limitations, but that affect the states that are estimated. These errors must be accounted for.[†]

[†]This and subsequent paragraphs are based on material written by the author for QinetiQ, so comprise QinetiQ copyright material.

When the correlation times are relatively short, these system errors may be modeled as white noise. However, the white noise must overbound the correlated noise, affecting the Kalman filter's convergence properties. For error sources correlated over more than a minute or so, a white noise approximation does not effectively model how the effects of these error sources propagate with time.[‡] The solution is to use a Schmidt-Kalman filter with uncertain system noise parameters [18]. Details are provided in Section D.2 of Appendix D on the CD.

Another Kalman filter formulation, designed for handling time-correlated GNSS measurement errors is described in [19].

3.4.4 Adaptive Kalman Filter

For most applications, the Kalman filter's system noise covariance matrix, **Q**, and measurement noise covariance matrix, **R**, are determined during the development phase by laboratory measurements of the system, simulation and trials. However, there are some cases where this cannot be done. For example, if an INS/GNSS integration algorithm or INS calibration algorithm is designed for use with a range of different inertial sensors, the system noise covariance will not be known in advance of operation. Similarly, if a transfer alignment algorithm (Section 15.1) is designed for use on different aircraft and weapon stores without prior knowledge of the flexure and vibration environment, the measurement noise covariance will not be known in advance.[*]

In other cases, the optimum Kalman filter tuning might vary over time as the context varies. For example, a GNSS navigation filter in a mobile device that may be stationary, on a walking pedestrian, or in a car, would require a different system noise model in each case. Similarly, the accuracy of GNSS ranging measurements varies with the signal-to-noise level, and multipath environment.

For both applications where the optimum tuning is unknown and applications where it varies, an adaptive Kalman filter may be used to estimate **R** and/or **Q** as it operates. There are two main approaches, innovation-based adaptive estimation (IAE) [20, 21] and multiple-model adaptive estimation (MMAE) [22].

The IAE method calculates the system noise covariance, **Q**, the measurement noise covariance, **R**, or both from the measurement innovation statistics. The first step is the calculation of the covariance of the last n measurement innovations, **C**:

$$\tilde{\mathbf{C}}_{\delta\mathbf{z},k}^{-} = \frac{1}{n}\sum_{j=k-n}^{k} \delta\mathbf{z}_j^{-}\delta\mathbf{z}_j^{-\mathrm{T}}. \tag{3.105}$$

This can be used to compute **Q** and/or **R**:

$$\begin{aligned} \tilde{\mathbf{Q}}_k &= \mathbf{K}_k\tilde{\mathbf{C}}_{\delta\mathbf{z},k}^{-}\mathbf{K}_k^{\mathrm{T}} \\ \tilde{\mathbf{R}}_{\mathrm{k}} &= \tilde{\mathbf{C}}_{\delta\mathbf{z},k}^{-} - \mathbf{H}_k\mathbf{P}_k^{-}\mathbf{H}_k^{\mathrm{T}} \end{aligned}. \tag{3.106}$$

[‡]End of QinetiQ copyright material.

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

Initial values of **Q** and **R** must be provided for use while the first set of measurement innovation statistics is compiled. These should be selected cautiously. Minimum and maximum values should also be imposed to stabilize the filter in the event of faults.

The MMAE method uses a bank of parallel Kalman filters with different values of the system and/or measurement noise covariance matrices, **Q** and **R**. Different initial values of the error covariance matrix, **P**, may also be used. Each of the Kalman filter hypotheses, denoted by the index i, is allocated a probability as follows [3, 4]:

$$p_{k,i} = \frac{\Lambda_{k,i}}{\sum_{j=1}^{l} \Lambda_{k,j}}, \qquad (3.107)$$

$$\Lambda_{k,i} = \frac{p_{k-1,i}}{\sqrt{(2\pi)^m \left|\mathbf{H}_{k,i}\mathbf{P}_{k,i}^{-}\mathbf{H}_{k,i}^{\mathrm{T}} + \mathbf{R}_{k,i}\right|}} \exp\left[-\tfrac{1}{2}\delta\mathbf{z}_{k,i}^{-\mathrm{T}}\left(\mathbf{H}_{k,i}\mathbf{P}_{k,i}^{-}\mathbf{H}_{k,i}^{\mathrm{T}} + \mathbf{R}_{k,i}\right)^{-1}\delta\mathbf{z}_{k,i}^{-} \right]$$

where m is the number of components of the measurement vector, l is the number of filter hypotheses, and Λ is a likelihood. Note that the matrix inversion is already performed as part of the Kalman gain calculation. The filter hypothesis with the smallest normalized measurement innovations is most consistent with the measurement stream, so is allocated the largest probability.

Over time, the probability of the best filter hypothesis will approach unity while the others approach zero. To make best use of the available processing capacity, weak hypotheses should be deleted and the strongest hypothesis periodically subdivided to refine the filter tuning and allow it to respond to changes in the system.

The overall state vector estimate and error covariance are obtained as follows:

$$\hat{\mathbf{x}}_k^{+} = \sum_{i=1}^{l} p_{k,i}\hat{\mathbf{x}}_{k,i}^{+}, \qquad (3.108)$$

$$\mathbf{P}_k^{+} = \sum_{i=1}^{l} p_{k,i}\left[\mathbf{P}_{k,i}^{+} + \left(\hat{\mathbf{x}}_{k,i}^{+} - \hat{\mathbf{x}}_k^{+}\right)\left(\hat{\mathbf{x}}_{k,i}^{+} - \hat{\mathbf{x}}_k^{+}\right)^{\mathrm{T}}\right], \qquad (3.109)$$

noting that the error covariance matrix must account for the spread in the state vector estimates of the filter hypotheses as well as the error covariance of each hypothesis.

Comparing the IAE and MMAE adaptive Kalman filter techniques, the latter is more computationally intensive, as a bank of Kalman filters must be processed instead of just one. However, in an IAE Kalman filter, the system noise covariance, measurement noise covariance, error covariance, and Kalman gain matrices may all be functions of the state estimates, whereas they are independent in the MMAE filter bank (assuming conventional Kalman filters rather than EKFs). Consequently, the MMAE is less prone to filter instability.

3.4.5 Multiple-Hypothesis Filtering

An assumption of the standard Kalman filter is that the measurements have unimodal distributions (e.g., Gaussian), enabling the measurement vector to be modeled as a

mean, $\mathbf{z}$, and covariance, $\mathbf{R}$. However, this is not the case for every navigation system. Ranging systems can produce bimodal position measurements where there are insufficient signals for a unique fix, while some feature-matching techniques (Chapter 13) can produce a fix in the form of a highly irregular position distribution. To process these measurements in a Kalman filter-based estimation algorithm, they must first be expressed as a sum of Gaussian distributions, known as hypotheses, each with a mean, $\mathbf{z}_i$, a covariance, $\mathbf{R}_i$, and also a probability, p_i. A probability score, p_0, should also be allocated to the null hypothesis, representing the probability that none of the other hypotheses are correct. The probability scores sum to unity:

$$\sum_{i=0}^{n_k} p_{k,i} = 1, \tag{3.110}$$

where n_k is the number of hypotheses and k denotes the Kalman filter iteration as usual.

There are three main methods of handling multiple-hypothesis measurements using Kalman filter techniques: best fix, weighted fix, and multiple-hypothesis filtering. The best-fix method is a standard Kalman filter that accepts the measurement hypothesis with the highest probability score and rejects the others. It should incorporate a prefiltering algorithm that rejects all of the measurement hypotheses where none is dominant. This method has the advantage of simplicity and can be effective where one hypothesis is clearly dominant on most iterations.

Weighted-fix techniques input all of the measurement hypotheses, weighted according to their probabilities, but maintain a single set of state estimates. An example is the probabilistic data association filter (PDAF) [23, 24], which is predominantly applied to target tracking problems. The system-propagation phase of the PDAF is the same as for a standard Kalman filter. In the measurement-update phase, the Kalman gain calculation is performed for each of the measurement hypotheses:

$$\mathbf{K}_{k,i} = \mathbf{P}_k^- \mathbf{H}_k^{\mathrm{T}} \left[\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_{k,i} \right]^{-1}. \tag{3.111}$$

The state vector and error covariance matrix are then updated using

$$\begin{aligned} \hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \sum_{i=1}^{n_k} p_{k,i} \mathbf{K}_{k,i} \left(\mathbf{z}_{k,i} - \mathbf{H}_k \hat{\mathbf{x}}_k^- \right) \\ &= \hat{\mathbf{x}}_k^- + \sum_{i=1}^{n_k} p_{k,i} \mathbf{K}_{k,i} \delta \mathbf{z}_{k,i}^- \\ &= \sum_{i=1}^{n_k} p_{k,i} \hat{\mathbf{x}}_{k,i}^+ \end{aligned}, \tag{3.112}$$

$$\mathbf{P}_k^+ = \left[\mathrm{I} - \left(\sum_{i=1}^{n_k} p_{k,i} \mathbf{K}_{k,i} \right) \mathbf{H}_k \right] \mathbf{P}_k^- + \sum_{i=1}^{n_k} p_{k,i} \left(\hat{\mathbf{x}}_{k,i}^+ - \hat{\mathbf{x}}_k^+ \right) \left(\hat{\mathbf{x}}_{k,i}^+ - \hat{\mathbf{x}}_k^+ \right)^{\mathrm{T}}, \tag{3.113}$$

where

$$\begin{aligned}\hat{\mathbf{x}}_{k,i}^{+} &= \hat{\mathbf{x}}_{k}^{-} + \mathbf{K}_{k,i}\left(\mathbf{z}_{k,i} - \mathbf{H}_{k}\hat{\mathbf{x}}_{k}^{-}\right) \\ &= \hat{\mathbf{x}}_{k}^{-} + \mathbf{K}_{k,i}\delta\mathbf{z}_{k,i}^{-}\end{aligned}. \tag{3.114}$$

Note that, where the measurement hypotheses are widely spread compared to the prior state uncertainties, the state uncertainty (root diagonals of **P**) can be larger following the measurement update; this cannot happen in a standard Kalman filter.

Compared to the best-fix technique, the PDAF has the advantage that it incorporates all true measurement hypotheses, but the disadvantage that it also incorporates all of the false hypotheses. It is most suited to applications where false hypotheses are not correlated over successive measurement sets or the truth is a combination of overlapping Gaussian measurement hypotheses.

Where false measurement hypotheses are time correlated, a multiple-hypothesis Kalman filter (MHKF) enables multiple state vector hypotheses to be maintained

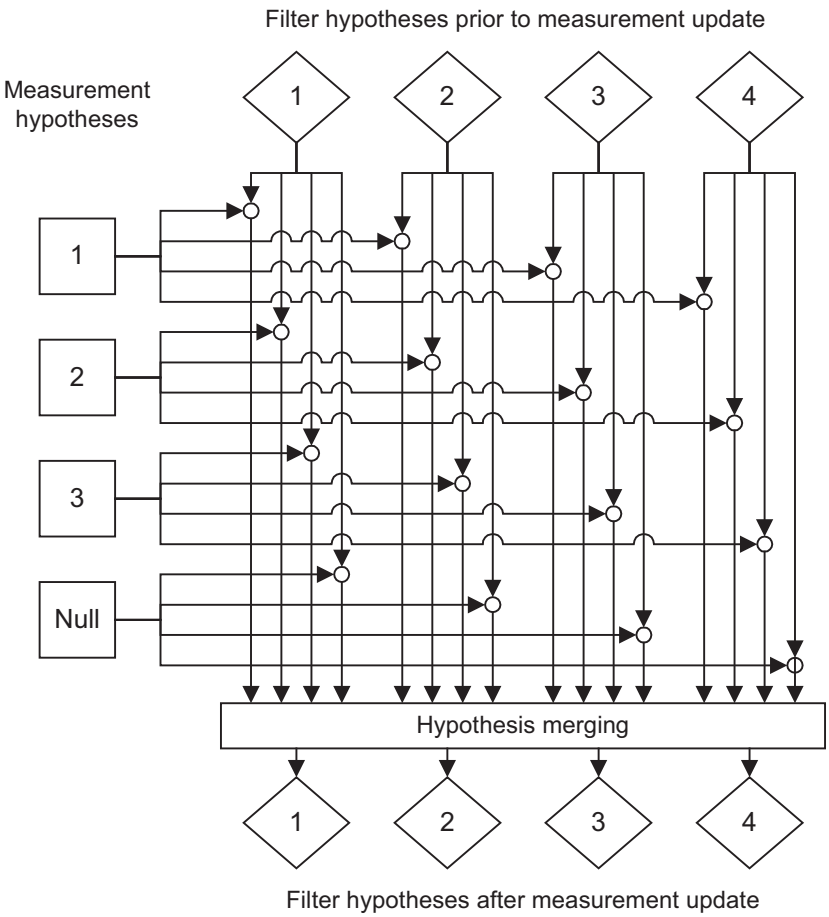

Figure 3.14 Multiple-hypothesis Kalman filter measurement update ($l = 4$, $n_k = 3$).

in parallel using a bank of Kalman filters. The technique was originally developed for target tracking [25], so is often known as multiple-hypothesis tracking (MHT). As the true hypothesis is identified over a series of filter cycles, the false measurement hypotheses are gradually eliminated from the filter bank. Like the MMAE filter, the MHKF maintains a set of l state vector and error covariance matrix hypotheses that are propagated independently through the system model using the conventional Kalman filter equations. Each of these hypotheses has an associated probability score.

For the measurement update phase, the filter bank is split into $(n_k + 1)l$ hypotheses, combining each state vector hypothesis with each measurement hypothesis and the null measurement hypothesis. Figure 3.14 shows the principle. A conventional Kalman filter update is then performed for each hypothesis and a probability score allocated that multiplies the probabilities of the state and measurement hypotheses. The new hypotheses must also be scored for consistency between the state vector and measurement hypotheses; a probability weighting similar to that used for the MMAE [see (3.107)] is suitable. Following this, the probability scores must be renormalized, noting that the scores for the null measurement hypotheses should remain unchanged.

It is clearly impractical for the number of state vector hypotheses to increase on each iteration of the Kalman filter, so the measurement update process must conclude with a reduction in the number of hypotheses to l. This is done by merging hypotheses. The exact approach varies between implementations, but, generally, similar hypotheses are merged with each other and the weakest hypotheses, in terms of their probability scores, are merged into their nearest neighbor. Hypotheses with probability scores below a certain minimum may simply be deleted. A pair of hypotheses, denoted by indices α and β, are merged into a new hypothesis, denoted by γ, using

$$p_{k,\gamma} = p_{k,\alpha} + p_{k,\beta}, \tag{3.115}$$

$$\hat{\mathbf{x}}^+_{k,\gamma} = \frac{p_{k,\alpha}\hat{\mathbf{x}}^+_{k,\alpha} + p_{k,\beta}\hat{\mathbf{x}}^+_{k,\beta}}{p_{k,\gamma}}, \tag{3.116}$$

$$\mathbf{P}^+_{k,\gamma} = \sum_{i=\alpha,\beta} p_{k,i}\left[\mathbf{P}^+_{k,i} + \left(\hat{\mathbf{x}}^+_{k,i} - \hat{\mathbf{x}}^+_{k,\gamma}\right)\left(\hat{\mathbf{x}}^+_{k,i} - \hat{\mathbf{x}}^+_{k,\gamma}\right)^{\mathrm{T}}\right]. \tag{3.117}$$

The overall state vector estimate and error covariance can either be the weighted average of all the hypotheses, obtained using (3.108) and (3.109), or the highest-probability hypothesis, depending on the needs of the application. When closed-loop correction (Section 3.2.6) is used, it is not possible to feed back corrections from the individual filter hypotheses as this would be contradictory; the closed-loop feedback must come from the filter bank as a whole. The corrections fed back to the system must also be subtracted from all of the state vector hypotheses to maintain constant

differences between the hypotheses. Thus, the state estimates are not zeroed at feedback, so state vector propagation using (3.14) must take place in the same manner as for the open-loop Kalman filter.

The iterative Gaussian mixture approximation of the posterior (IGMAP) method [26], which can operate with either a single or multiple hypothesis state vector, combines the fitting of a set of Gaussian distributions to the measurement probability distribution and the measurement-update phase of the estimation algorithm into a single iterative process. By moving the approximation as a sum of Gaussian distributions from the beginning to the end of the measurement-update cycle, the residuals of the approximation process are reduced, producing more accurate state estimates. The system-propagation phase of IGMAP is the same as for a conventional Kalman filter or MHKF. However, IGMAP does require more processing capacity than a PDAF or MHKF.

The need to apply a Gaussian approximation to the measurement noise and system noise distributions can be removed altogether by using a Monte Carlo estimation algorithm, such as a particle filter (Section 3.5). However, this imposes a much higher processing load than Kalman filter-based estimation.

3.4.6 Kalman Smoothing

The Kalman filter is designed for real-time applications. It estimates the properties of a system at a given time using measurements of the system up to that time. However, for applications such as surveillance and testing, where the properties of a system are required after the event, a Kalman filter effectively throws away half the measurement data as it does not use measurements taken after the time of interest.[*]

The Kalman smoother is the extension of the Kalman filter that uses measurement information from after the time at which state estimates are required as well as before that time. This leads to more accurate state estimates for nonreal-time applications. There are two main methods, the forward-backward filter [2, 27], and the Rauch, Tung, and Striebel (RTS) method [4, 28].

The forward-backward filter comprises two Kalman filters, a forward filter and a backward filter. The forward filter is a standard Kalman filter. The backward filter is a Kalman filter algorithm working backward in time from the end of the data segment to the beginning. The two filters are treated as independent, so the backward filter must not be initialized with the final solution of the forward filter. The smoothed estimates are obtained simply by combining the estimates of the two filters, weighted according to the ratio of their error covariance matrices:[*]

$$\begin{aligned}\hat{\mathbf{x}}_k^+ &= \left(\mathbf{P}_{f,k}^{+\,-1} + \mathbf{P}_{b,k}^{+\,-1}\right)^{-1}\left(\mathbf{P}_{f,k}^{+\,-1}\hat{\mathbf{x}}_{f,k}^+ + \mathbf{P}_{b,k}^{+\,-1}\hat{\mathbf{x}}_{b,k}^+\right)\\ \mathbf{P}_k^+ &= \left(\mathbf{P}_{f,k}^{+\,-1} + \mathbf{P}_{b,k}^{+\,-1}\right)^{-1}\end{aligned}, \tag{3.118}$$

[*]This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

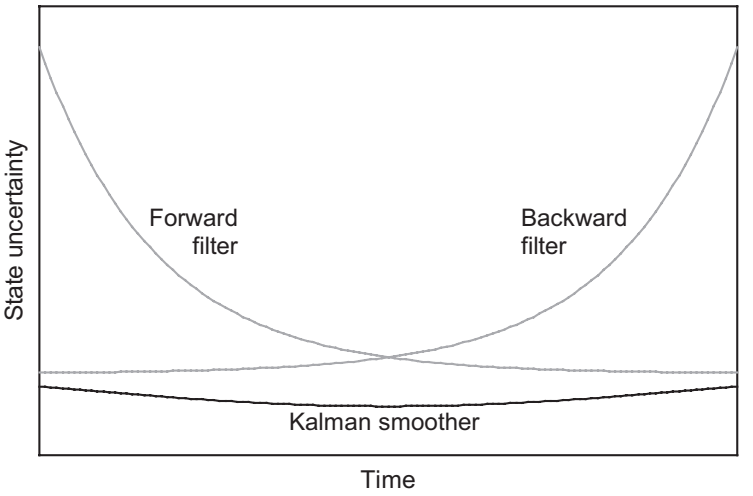

Figure 3.15 Forward-backward Kalman smoother state uncertainty.

where the subscripts *f* and *b* refer to the forward and backward filters, respectively.* The index, *k*, refers to the same point in time for both filters, so the backward filter must count backward. Figure 3.15 shows how the state uncertainty varies with time for the forward, backward and combined filters. It is only necessary to store the state vectors and error covariance matrices and perform the matrix inversion at the points of interest. Note that it is not necessary to run the forward filter beyond the last point of interest and the backward filter beyond the first point of interest.

In the RTS method, a conventional Kalman filter runs forward in time, but storing the state vector, $\mathbf{x}$, and the error covariance matrix, $\mathbf{P}$, after each system propagation and measurement update. The transition matrix, $\mathbf{\Phi}$, is also stored. Once the end of the data set is reached, smoothing begins, starting at the end and working back to the beginning. The smoothing gain on each iteration, $\mathbf{A}_k$, is given by

$$\mathbf{A}_k = \mathbf{P}_k^+ \mathbf{\Phi}_k^{\mathrm{T}} \left(\mathbf{P}_{k+1}^- \right)^{-1}. \tag{3.119}$$

The smoothed state vector, $\hat{\mathbf{x}}_k^s$, and error covariance, $\mathbf{P}_k^s$, are then given by

$$\begin{aligned} \hat{\mathbf{x}}_k^s &= \hat{\mathbf{x}}_k^+ + \mathbf{A}_k \left(\hat{\mathbf{x}}_{k+1}^s - \hat{\mathbf{x}}_{k+1}^- \right) \\ \mathbf{P}_k^s &= \mathbf{P}_k^+ + \mathbf{A}_k \left(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^- \right) \mathbf{A}_k^{\mathrm{T}} \end{aligned}. \tag{3.120}$$

When the smoothed solution is required at all points, the RTS method is more efficient, whereas the forward-backward method is more efficient where a smoothed solution is only required at a single point.

Kalman smoothing can also be used to provide a quasi-real-time solution by making use of information from a limited period after the time of interest. A continuous solution is then output at a fixed lag. This can be useful for tracking applications, such as logistics, security, and road-user charging, that require bridging of GNSS outages.

*This paragraph, up to this point, is based on material written by the author for QinetiQ, so comprises QinetiQ copyright material.

3.5 The Particle Filter

This section provides an introduction to the particle filter [29, 30], a nonlinear non-Gaussian Bayesian estimation technique, using the terminology and notation of Kalman filter-based estimation. Further information is available in standard texts [6, 31–33], noting that much of the particle filtering literature assumes a background in advanced statistics.

In state estimation, the measurements, noise sources, and the state estimates themselves are all probability distributions; the exact values are unknown. In Kalman filter-based estimation, all distributions are modeled using the mean and covariance. This is sufficient for modeling Gaussian distributions but is not readily applicable to all types of navigation system. Pattern-matching systems produce inherently non-Gaussian measurement distributions, which are often multimodal. Ranging and angular positioning using environmental features can produce multimodal measurements where the landmark identity is ambiguous. An INS can also have a non-Gaussian error distribution where the attitude uncertainty is too large for the small-angle approximation to be applied.

A particle filter is a type of sequential Monte Carlo estimation algorithm (some authors equate the two). As such, the state estimates are represented as a set of discrete state vectors, known as particles, which are spread throughout their joint probability distribution. Figure 3.16 illustrates this for a bivariate Gaussian distribution. This requires at least an order of magnitude more processing power than the mean and covariance representation used in Kalman filter-based estimation. However, it has the key advantage that any shape of probability distribution may be represented, as illustrated by Figure 3.17. There is no need to approximate the distribution to a multivariate Gaussian (or sum thereof in the case of a MHKF).

The more particles used, the more accurately the probability distribution of the state estimates is represented. Similarly, the more complex the distribution, the greater the number of particles required to represent it to a given accuracy. Most particle filters deploy at least a thousand particles and some use a million or more. The mean, $\hat{\mathbf{x}}_k^+$, and covariance, $\mathbf{P}_k^+$, of the state estimate at epoch k are given by

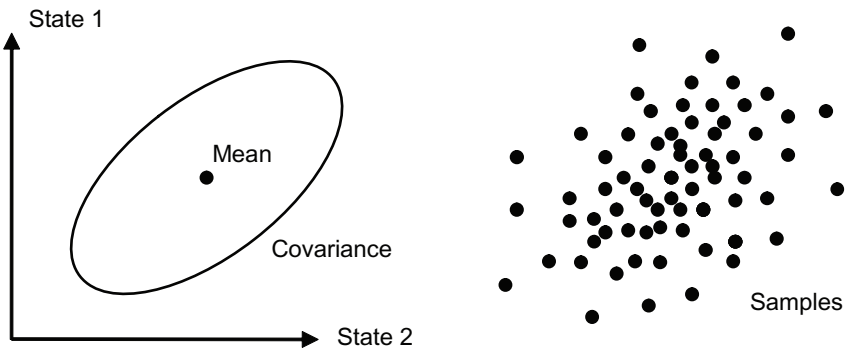

Figure 3.16 Representation of two states with a bivariate Gaussian distribution using a mean and covariance (left) and a set of particles (right).

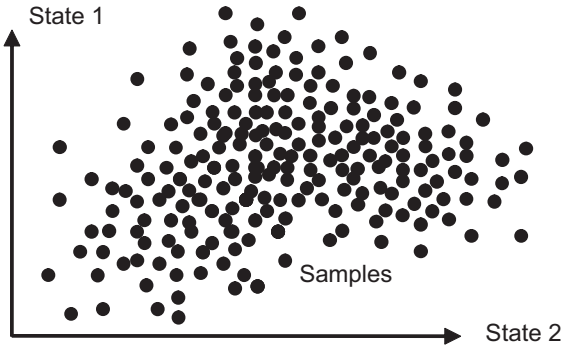

Figure 3.17 Representation of two states with a non-Gaussian distribution using a set of particles.

$$\hat{\mathbf{x}}_k^{\pm} = \sum_{i=1}^{N} p_{\mathrm{X},k}^{\pm(i)} \hat{\mathbf{x}}_k^{(i)}, \qquad \mathbf{P}_k^{\pm} = \sum_{i=1}^{N} p_{\mathrm{X},k}^{\pm(i)} \left(\hat{\mathbf{x}}_k^{(i)} - \hat{\mathbf{x}}_k^{\pm}\right)\left(\hat{\mathbf{x}}_k^{(i)} - \hat{\mathbf{x}}_k^{\pm}\right)^{\mathrm{T}}, \tag{3.121}$$

where $\hat{\mathbf{x}}_k^{(i)}$ and $p_{\mathrm{X},k}^{\pm(i)}$ are, respectively, the state vector estimate and probability of the ith particle, N is the number of particles; the superscripts, – and +, denote before and after the measurement update, respectively; and

$$\sum_{i=1}^{N} p_{\mathrm{X},k}^{\pm(i)} = 1. \tag{3.122}$$

A particle filter has three phases, shown in Figure 3.18. The system-propagation and measurement-update phases are equivalent to their Kalman filter counterparts, while the resampling phase has no Kalman filter equivalent.

Each particle is propagated through the system model separately. The first step, performed independently for each particle, is to sample the discrete system noise vector, $\mathbf{w}_{s,k-1}^{(i)}$, from a distribution, common to all particles. This distribution may be constant or vary with time and/or other known parameters. However, it does not vary with the estimated states. The particle's state vector estimate is then propagated using

$$\hat{\mathbf{x}}_k^{(i)} = \boldsymbol{\varphi}_{k-1}\left(\hat{\mathbf{x}}_{k-1}^{(i)}, \mathbf{w}_{s,k-1}^{(i)}\right), \tag{3.123}$$

where $\boldsymbol{\varphi}_{k-1}$ is a transition function, common to all particles. It need not be a linear function of either the states or the system noise and may or may not vary with time and other known parameters. Alternatively, a similar approach to the EKF and UKF system models may be adopted. By analogy with (3.82),

$$\begin{aligned} \hat{\mathbf{x}}_k^{(i)} &= \hat{\mathbf{x}}_{k-1}^{(i)} + \int_{t_k-\tau_s}^{t_k} \mathbf{f}\left(\hat{\mathbf{x}}_{k-1}^{(i)}, \mathbf{w}_{s,k-1}^{(i)}, t'\right) dt' \\ &\approx \hat{\mathbf{x}}_{k-1}^{(i)} + \mathbf{f}\left(\hat{\mathbf{x}}_{k-1}^{(i)}, \mathbf{w}_{s,k-1}^{(i)}, t_k\right)\tau_s \end{aligned}. \tag{3.124}$$

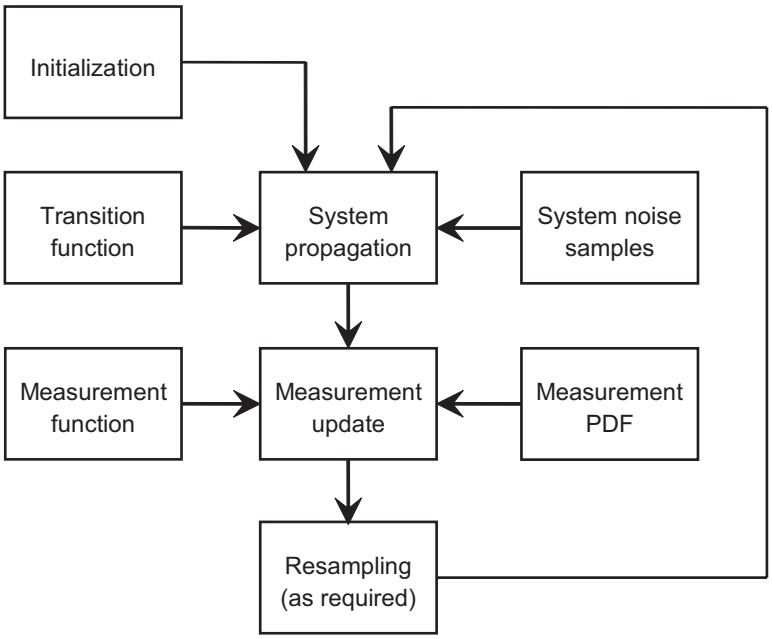

Figure 3.18 Phases of the particle filter.

If the system model is linear, this simplifies to

$$\hat{\mathbf{x}}_k^{(i)} = \hat{\mathbf{x}}_{k-1}^{(i)} + \boldsymbol{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1}^{(i)} + \boldsymbol{\Gamma}_{k-1}\mathbf{w}_{s,k-1}^{(i)}. \tag{3.125}$$

The propagation of multiple state vectors through a model of the system, each with a different set of randomly sampled noise sources is a type of Monte Carlo simulation (see Section J.5 of Appendix J on the CD). This is why the particle filter is classified as a Monte Carlo estimation algorithm.

The system propagation phase of a particle filter changes the state estimates of each particle but leaves the probabilities unchanged, thus $p_{\mathrm{X},k}^{-(i)} \equiv p_{\mathrm{X},k-1}^{+(i)}$. By contrast, the measurement update phase changes the probabilities but not the state estimates. The first step in the measurement update is to obtain a prediction of the measurement vector from each particle's state vector estimate. Thus,

$$\hat{\mathbf{z}}_k^{(i)} = \mathbf{h}\left(\hat{\mathbf{x}}_k^{(i)}\right), \tag{3.126}$$

where $\mathbf{h}$ is the deterministic measurement function, as defined by (3.8) and used in the EKF and UKF; it need not be linear.

Next, the predicted measurements are compared with the probability distribution function obtained from the actual measurement process, $\tilde{f}_{\mathrm{Z},k}$, to obtain the relative likelihood of each particle. This is multiplied by the prior probability to obtain the absolute likelihood. Thus,

$$\Lambda_{\mathbf{X},k}^{(i)} = p_{\mathbf{X},k}^{-(i)}\tilde{f}_{\mathrm{Z},k}\left(\hat{\mathbf{z}}_k^{(i)}\right). \tag{3.127}$$

If the measurement distribution is m-variate Gaussian, its PDF is

$$\tilde{f}_{\mathbf{z},k}\left(\hat{\mathbf{z}}_k^{(i)}\right) = \frac{1}{(2\pi)^{m/2}\left|\mathbf{R}_k\right|^{1/2}} \exp\left[-\tfrac{1}{2}\left(\hat{\mathbf{z}}_k^{(i)} - \tilde{\mathbf{z}}_k\right)^{\mathrm{T}} \mathbf{R}_k^{-1}\left(\hat{\mathbf{z}}_k^{(i)} - \tilde{\mathbf{z}}_k\right)\right], \tag{3.128}$$

where $\tilde{\mathbf{z}}_k$ is the measured mean and $\mathbf{R}_k$ is the measurement noise covariance.

The updated probabilities of each particle are then obtained by renormalizing the likelihoods, giving

$$p_{\mathrm{X},k}^{+(i)} = \frac{\Lambda_{\mathrm{X},k}^{(i)}}{\sum_{j=1}^{n} \Lambda_{\mathrm{X},k}^{(j)}}. \tag{3.129}$$

The final phase of the particle filter is resampling. Without resampling, the probabilities of many of the particles would shrink over successive epochs until they were too small to justify the processing capacity required to maintain them. At the same time, the number of particles available to represent the core of the state estimate distribution would shrink, reducing the accuracy. A particle filter is most efficient when the particles have similar probabilities. Therefore, in the resampling phase, low-probability particles are deleted and high-probability particles are duplicated. The independent application of system noise to each particle ensures that duplicate particles become different at the next system propagation phase of the particle filter. Section D.3.1 of Appendix D on the CD describes resampling in more detail. The most commonly used resampling algorithms allocate equal probability (i.e., $1/N$) to the resampled particles.

Resampling makes the particle filter more receptive to new measurement information, but it also adds noise to the state estimation process, degrading the accuracy. Therefore, it is not desirable to perform it on every filter cycle. Resampling can be triggered after a fixed number of cycles or based on the effective sample size, N_{eff}, given by

$$N_{eff} = \left[\sum_{i=1}^{N}\left(p_{\mathrm{X},k}^{+(i)}\right)^2\right]^{-1}, \tag{3.130}$$

dropping below a certain threshold, such as $N/2$ or $2N/3$.

To initialize a particle filter, it is necessary to generate a set of particles by sampling randomly from the initial distribution of the states. For a uniform or Gaussian distribution, this is straightforward (see Sections J.4.1 and J.4.3 of Appendix J on the CD). For more complex distributions, importance sampling must be used as described in Section D.3.2 of Appendix D on the CD.

Hybrid filters that combine elements of the particle filter and the Kalman filter may also be implemented. These are discussed in Section D.3.3 of Appendix D on the CD.

Problems and exercises for this chapter are on the accompanying CD.