Chapter 5

Square Root Solution Methods for the Orbit Determination Problem

5.1 INTRODUCTION

In the previous chapter, the solution to the least squares estimation problem including *a priori* information, $\overline{P}$ and $\overline{\mathbf{x}}$, is represented in the normal equation form as

$$(H^T W H + \overline{P}^{-1})\hat{\mathbf{x}} = H^T W y + \overline{P}^{-1}\overline{\mathbf{x}} \tag{5.1.1}$$

which can be expressed as

$$M\hat{\mathbf{x}} = N. \tag{5.1.2}$$

The solution for $\hat{\mathbf{x}}$ is obtained by computing the inverse of M. In practice, computational problems are encountered in forming and inverting the matrix $M = H^T W H + \overline{P}^{-1}$. An orthogonal transformation approach can be used to write Eq. (5.1.2) in the form

$$R\hat{\mathbf{x}} = b \tag{5.1.3}$$

where R is an $n \times n$ upper triangular matrix (R is not the observation error covariance matrix). Then $\hat{\mathbf{x}}$ can be obtained by backward substitution. This approach has the advantage that increased numerical accuracy is achieved over conventional matrix inversion methods for solving Eq. (5.1.2). Using the orthogonal transformation approach, accuracy can be achieved with a single-precision computation that is equal to the accuracy obtained by inverting M with the normal equation approach using double-precision computations.

The normal equation approach has several operational and conceptual advantages that have led to the widespread adoption of this technique for many operational orbit determination systems. Both the normal equation and orthogonal transformation approaches are described in the subsequent sections.

The solution of the linear system

$$M\hat{\mathbf{x}} = \mathbf{N} \tag{5.1.4}$$

is expressed as

$$\hat{\mathbf{x}} = M^{-1}\mathbf{N} \tag{5.1.5}$$

where the operation M^{-1} implies that the inverse of the $(n \times n)$ matrix M is computed and then postmultiplied by the column vector $\mathbf{N}$. A solution based on the Cholesky decomposition of M is more efficient and, in most cases, more accurate. The Cholesky decomposition is applicable only if M is symmetric and positive definite, a condition satisfied for the case considered here. The following discussion first outlines the Cholesky decomposition, and describes this approach to solving the normal equations. Then a discussion of solutions based on orthogonal transformations is presented in subsequent sections.

5.2 CHOLESKY DECOMPOSITION

Let M be a symmetric positive definite matrix, and let R be an upper triangular matrix computed such that

$$R^T R = M. \tag{5.2.1}$$

(Note that R is not the observation error covariance matrix.) Then Eq. (5.1.4) can be expressed as

$$R^T R\hat{\mathbf{x}} = \mathbf{N}. \tag{5.2.2}$$

If the definition

$$\mathbf{z} = R\hat{\mathbf{x}} \tag{5.2.3}$$

is used, Eq. (5.2.2) can be written as

$$R^T\mathbf{z} = \mathbf{N} \tag{5.2.4}$$

where R^T is lower triangular. The components of $\mathbf{z}$ can be determined using a forward recursion relation. Then, Eq. (5.2.3) can be solved using a backward recursion to obtain the elements of $\hat{\mathbf{x}}$.

The elements of the error covariance matrix, $P = (H^T W H + \overline{P}^{-1})^{-1} = M^{-1}$, can be obtained from the condition

$$P = M^{-1} = (R^T R)^{-1} = R^{-1}R^{-T} = SS^T \tag{5.2.5}$$

where S, the inverse of the upper triangular matrix, R, can be computed by an efficient backward recursion.

Equation (5.2.1) represents a set of $(n^2+n)/2$ equations for the $(n^2+n)/2$ unknown elements of the upper triangular matrix, R. The expression for the Cholesky decomposition of M is obtained by expanding Eq. (5.2.1) and solving term by term for the elements of R (e.g., R_{ij} is determined in terms of the elements of M).

Given the elements of the $n \times n$ positive definite matrix M and the $n \times 1$ column vector $\mathbf{N}$, the following Cholesky algorithm will yield a solution for the elements of R, $\mathbf{z}$, $\hat{\mathbf{x}}$ and the upper triangular matrix, S. Step 1 of the following algorithm determines the elements of R and the vector $\mathbf{z}$. Steps 2 and 3 perform a backward recursion to form $\hat{\mathbf{x}}$ and the elements of the matrix $S = R^{-1}$.

5.2.1 THE CHOLESKY ALGORITHM

The Cholesky algorithm for R is derived easily by equating the elements of $R^T R$ to the elements of M that are known. For example, from expanding $R^T R$ it is shown that $r_{11} = \sqrt{M_{11}}$.

1) For $i = 1, 2, \ldots, n$

$$r_{ii} = \left(M_{ii} - \sum_{k=1}^{i-1} r_{ki}^2 \right)^{1/2}$$

$$r_{ij} = \left(M_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj} \right) / r_{ii}; \quad j = i+1, \ldots, n. \tag{5.2.6}$$

The elements of $\mathbf{z}$ are obtained from an expansion of Eq. (5.2.4):

$$z_i = \left(N_i - \sum_{j=1}^{i-1} r_{ji} z_j \right) / r_{ii} \tag{5.2.7}$$

2) For $i = n, n-1, \ldots, 1$

$$\hat{x}_i = \left(z_i - \sum_{j=i+1}^{n} r_{ij} \hat{x}_j \right) / r_{ii}. \tag{5.2.8}$$

The elements of S are obtained from an expansion of $SR = I$.

3) For $i = 1, 2 \ldots, n$

$$s_{ii} = \frac{1}{r_{ii}}$$

$$s_{ij} = -s_{jj}\left[\sum_{k=i}^{j-1} r_{kj}s_{ik}\right]; \quad j = i+1, \ldots, n. \tag{5.2.9}$$

Examples of the application of this algorithm are given in Sections 5.6.1 and 5.6.5.

This Cholesky algorithm is a nonblock algorithm. That is, it does not use matrix multiplication. Because matrix multiplication is much faster in terms of floating point operations per second than matrix-vector operations on modern computers, a block Cholesky algorithm often will be faster than a nonblock version. In fact, the increase in speed may be a factor of three or more. See Golub and Van Loan (1996) for examples of a block Cholesky algorithm.

5.2.2 THE SQUARE ROOT FREE CHOLESKY ALGORITHM

Notice that calculation of the diagonal elements of R requires that n square roots be taken. Computationally the square root operation is expensive compared to multiplication, division, addition, or subtraction; hence, it is desirable to avoid square roots if possible. A square root free Cholesky algorithm may be developed by defining

$$M = UDU^T \tag{5.2.10}$$

where U is unit upper triangular (i.e., has ones on the diagonal), and D is a diagonal matrix. As in the previous section, Eq. (5.2.10) represents the $(n^2 + n)/2$ equation in the $(n^2 + n)/2$ unknown elements of U and D. The algorithm for the elements of U and D is obtained by expanding Eq. (5.2.10) and is given by

$$D_j U_{ij} = M_{ij} - \sum_{k=j+1}^{n} M'_{jk}U_{ik} \equiv M'_{ij} \quad \begin{matrix} j = n, \ldots, 1 \\ i = 1, \ldots, j-1 \end{matrix} \tag{5.2.11}$$

$$D_j = M_{jj} - \sum_{k=j+1}^{n} M'_{jk}U_{jk}. \tag{5.2.12}$$

The procedure is to set $j = n$ and cycle through the algorithm for $i = 1 \ldots n-1$, solving for D_n and the elements of U_{in} (i.e., the last column of U). Then set $j = n-1$ and cycle through for $i = 1 \ldots n-2$, solving for D_{n-1} and the $n-1$ column of U. Repeat this procedure for the remaining values of j.

Knowing U and D, it is possible to solve for $\hat{\mathbf{x}}$ from Eq. (5.1.2) and Eq. (5.2.10) as follows. Note that

$$U D U^T \hat{\mathbf{x}} = \mathbf{N} \tag{5.2.13}$$

and let

$$\mathbf{z} = DU^T \hat{\mathbf{x}}. \tag{5.2.14}$$

Then

$$U\mathbf{z} = \mathbf{N}. \tag{5.2.15}$$

The intermediate vector $\mathbf{z}$ can be determined from Eq. (5.2.15). The solution is a backward substitution (i.e., we solve for z_n, $z_{n-1} \ldots z_1$):

$$z_n = N_n \tag{5.2.16}$$

$$z_i = N_i - \sum_{j=i+1}^{n} U_{ij} z_j \qquad i = n-1, \ldots, 1. \tag{5.2.17}$$

Then the elements of $\hat{\mathbf{x}}$ can be determined from Eq. (5.2.14) by using a forward substitution, as

$$\hat{x}_i = \frac{z_i}{D_{ii}} - \sum_{j=1}^{i-1} U_{ji} \hat{x}_j \; i = 1, 2, \ldots n\,. \tag{5.2.18}$$

The associated estimation error covariance is obtained from

$$\begin{aligned} P &= (UDU^T)^{-1} \\ &= U^{-T} D^{-1} U^{-1} \end{aligned} \tag{5.2.19}$$

where

$$\begin{aligned} D_{ii}^{-1} &= 1/D_{ii} && i = 1, \ldots n \\ & && j = i+1, \ldots n \\ U_{ij}^{-1} &= -\sum_{k=i}^{j-1} U_{ik}^{-1} U_{kj} \\ U_{ij}^{-1} &= 0 && i > j \\ D_{ij}^{-1} &= 0 && i \neq j. \end{aligned} \tag{5.2.20}$$

Note that none of the algorithms required to compute $\hat{\mathbf{x}}$ involve the calculation of a square root.

5.3 LEAST SQUARES SOLUTION VIA ORTHOGONAL TRANSFORMATION

An alternate approach that avoids some of the numerical problems encountered in the normal equation approach is described in the following discussion. The method obtains the solution by applying successive orthogonal transformations to the information array, $(H, \mathbf{y})$. Enhanced numerical accuracy is obtained by this approach. Consider the quadratic performance index, $J(\mathbf{x})$, which minimizes the weighted sum of squares of the observation errors, $\boldsymbol{\epsilon} = \mathbf{y} - H\mathbf{x}$ (for the moment we will assume no *a priori* information; i.e., $\overline{P}^{-1} = 0, \overline{\mathbf{x}} = 0$):

$$J(\mathbf{x}) = \boldsymbol{\epsilon}^T W \boldsymbol{\epsilon} = \left\| W^{\frac{1}{2}}(H\mathbf{x} - \mathbf{y}) \right\|^2 = (H\mathbf{x} - \mathbf{y})^T W (H\mathbf{x} - \mathbf{y}). \tag{5.3.1}$$

If W is not diagonal, $W^{1/2}$ can be computed by the Cholesky decomposition. Or the prewhitening transformation described at the end of Section 5.7.1 can be applied so that $W = I$. For notational convenience we are using $-\boldsymbol{\epsilon}$ in Eq. (5.3.1).

The solution to the least squares estimation problem (as well as the minimum variance and the maximum likelihood estimation problem, under certain restrictions) is obtained by finding the value $\hat{\mathbf{x}}$ that minimizes the performance index $J(\mathbf{x})$. To achieve the minimum value of $J(\mathbf{x})$, we introduce the $m \times m$ orthogonal matrix, Q. An orthogonal matrix has the following properties:

1. $QQ^T = I.$ (5.3.2)

2. $Q^{-1} = Q^T$ hence $Q^T Q = I.$

3. If Q_1 and Q_2 are orthogonal matrices, then so is $Q_1 Q_2$.

4. For any vector $\mathbf{x}$,
$$\left\| Q\mathbf{x} \right\| = \left\| \mathbf{x} \right\| = (\mathbf{x}^T \mathbf{x})^{\frac{1}{2}}. \tag{5.3.3}$$
Multiplying by Q does not change the Euclidean norm of a vector.

5. If $\boldsymbol{\epsilon}$ is an m vector of random variables with $\boldsymbol{\epsilon} \sim (O, I)$ (i.e., $E(\boldsymbol{\epsilon}) = 0$ and $E(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T) = I$), then $\overline{\boldsymbol{\epsilon}} = Q\boldsymbol{\epsilon}$ has the same properties,
$$E(\overline{\boldsymbol{\epsilon}}) = QE(\boldsymbol{\epsilon}) = 0, \; E(\overline{\boldsymbol{\epsilon}}\,\overline{\boldsymbol{\epsilon}}^T) = QE(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T)Q^T = I. \tag{5.3.4}$$

It follows then that (5.3.1) can be expressed as

$$\begin{aligned} J(\mathbf{x}) &= \left\| QW^{\frac{1}{2}}(H\mathbf{x} - \mathbf{y}) \right\|^2 \\ &= \Big(H\mathbf{x} - \mathbf{y}\Big)^T W^{\frac{1}{2}} Q^T Q W^{\frac{1}{2}} \Big(H\mathbf{x} - \mathbf{y}\Big). \end{aligned} \tag{5.3.5}$$

Select Q such that

$$QW^{\frac{1}{2}}H = \begin{bmatrix} R \\ O \end{bmatrix} \text{ and define } QW^{\frac{1}{2}}\mathbf{y} \equiv \begin{bmatrix} \mathbf{b} \\ \mathbf{e} \end{bmatrix} \tag{5.3.6}$$

where

R is a $n \times n$ upper-triangular matrix of rank n

O is a $(m-n) \times n$ null matrix

$\mathbf{b}$ is a $n \times 1$ column vector

$\mathbf{e}$ is a $(m-n) \times 1$ column vector.

The results given by Eq. (5.3.6) assume that $m > n$ and H is of rank n. Using Eq. (5.3.6), Eq. (5.3.5) can be written as

$$J(\mathbf{x}) = \left\| \begin{bmatrix} R \\ O \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b} \\ \mathbf{e} \end{bmatrix} \right\|^2 . \tag{5.3.7}$$

Expanding leads to

$$J(\mathbf{x}) = \left\| R\mathbf{x} - \mathbf{b} \right\|^2 + \left\| \mathbf{e} \right\|^2 . \tag{5.3.8}$$

Only the first term in Eq. (5.3.8) is a function of $\mathbf{x}$, so the value of $\mathbf{x}$ that minimizes $J(\mathbf{x})$ is obtained by requiring that

$$R\hat{\mathbf{x}} = \mathbf{b} \tag{5.3.9}$$

and the minimum value of the performance index becomes (equating $J(\hat{x})$ in Eq. (5.3.1) and Eq. (5.3.8))

$$J(\hat{\mathbf{x}}) = \left\| \mathbf{e} \right\|^2 = \left\| W^{\frac{1}{2}}(H\hat{\mathbf{x}} - \mathbf{y}) \right\|^2 . \tag{5.3.10}$$

That is, $\|\mathbf{e}\|$ is the norm of the observation residual vector, which for a linear system will be equal to the weighted sum of the squares of observation residuals determined by using $\hat{\mathbf{x}}$ in Eq. (5.3.10).

5.4 GIVENS TRANSFORMATIONS

The procedure described in the previous section is direct and for implementation requires only that a convenient procedure for computing Q be obtained. One

such procedure can be developed based on the Givens plane rotation (Givens, 1958). Let $\mathbf{x}$ be a 2×1 vector having components $\mathbf{x}^T = [x_1 \ x_2]$ and let G be a 2×2 orthogonal matrix associated with the plane rotation through the angle θ. Then select G such that

$$G\mathbf{x} = \mathbf{x}' = \begin{pmatrix} x_1' \\ 0 \end{pmatrix}. \tag{5.4.1}$$

To this end, consider the transformation

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{5.4.2}$$

or

$$\begin{aligned} x_1' &= \cos\theta x_1 + \sin\theta x_2 \\ x_2' &= -\sin\theta x_1 + \cos\theta x_2. \end{aligned} \tag{5.4.3}$$

Equations (5.4.3) represent a system of two equations in three unknowns; that is, x_1', x_2', and θ. The Givens rotation is defined by selecting the rotation θ such that $x_2' = 0$. That is, let

$$x_1' = \cos\theta x_1 + \sin\theta x_2 \tag{5.4.4}$$

$$0 = -\sin\theta x_1 + \cos\theta x_2. \tag{5.4.5}$$

From Eq. (5.4.5), it follows that

$$\tan\theta = \frac{x_2}{x_1}, \sin\theta = \frac{x_2}{\sqrt{x_1^2 + x_2^2}}, \cos\theta = \frac{x_1}{\sqrt{x_1^2 + x_2^2}}. \tag{5.4.6}$$

The positive value associated with the square root operation is selected for the following discussion. Substituting the expression for $\sin\theta$ and $\cos\theta$ into Eq. (5.4.4) leads to

$$x_1' = \frac{x_1^2}{\sqrt{x_1^2 + x_2^2}} + \frac{x_2^2}{\sqrt{x_1^2 + x_2^2}} = \sqrt{x_1^2 + x_2^2}. \tag{5.4.7}$$

Consider the application of the transformation

$$G(\theta) = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{5.4.8}$$

to two general row vectors $\mathbf{h}_i$ and $\mathbf{h}_k$; for example,

$$G \begin{bmatrix} h_{ii} & h_{ii+1} & \dots & h_{in} \\ h_{ki} & h_{ki+1} & \dots & h_{kn} \end{bmatrix} = \begin{bmatrix} h_{ii}' & h_{ii+1}' & \dots & h_{in}' \\ 0 & h_{ki+1}' & \dots & h_{kn}' \end{bmatrix}. \tag{5.4.9}$$

That is, for any two general row vectors, $\mathbf{h}_i$ and $\mathbf{h}_k$, the transformation is applied to the first column so as to null h_{ki}. The transformation that accomplishes this is applied to each remaining column to obtain the transformed matrix. Hence,

$$\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} h_{ii} \\ h_{ki} \end{bmatrix} = \begin{bmatrix} h'_{ii} \\ 0 \end{bmatrix} \tag{5.4.10}$$

or

$$\begin{aligned} \sin\theta &= h_{ki}/\sqrt{h_{ii}^2 + h_{ki}^2} = h_{ki}/h'_{ii} \\ \cos\theta &= h_{ii}/\sqrt{h_{ii}^2 + h_{ki}^2} = h_{ii}/h'_{ii} \\ h'_{ii} &= \sqrt{h_{ii}^2 + h_{ki}^2}. \end{aligned} \tag{5.4.11}$$

Then for all other columns,

$$\begin{aligned} h'_{ij} &= h_{ij}\cos\theta + h_{kj}\sin\theta \\ & \qquad\qquad j = i+1, \ldots, n \\ h'_{kj} &= -h_{ij}\sin\theta + h_{kj}\cos\theta\,. \end{aligned} \tag{5.4.12}$$

By using this transformation repetitively as k goes from $i+1$ to m, the remaining elements of the i^{th} column can be nulled. Then by moving down the diagonal and applying the transformation to successive columns whose first element lies on the diagonal, a rank n matrix can be reduced to an upper triangular $n \times n$ matrix with a lower $(m-n) \times n$ null matrix. If the element to be nulled already has a value of zero, the transformation matrix will be the identity matrix and the corresponding transformation may be skipped.

As an example, the transformation to null the fourth element in the third column is shown as follows:

$$\begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & C^{4,3} & S^{4,3} & & & \\ & & -S^{4,3} & C^{4,3} & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \\ & & & & & & & \ddots \\ & & & & & & & & 1 \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1n} & y_1 \\ 0 & h_{22} & h_{23} & \cdots & h_{2n} & y_2 \\ 0 & 0 & h_{33} & \cdots & h_{3n} & y_3 \\ 0 & 0 & h_{43} & \cdots & h_{4n} & y_4 \\ 0 & 0 & h_{53} & \cdots & h_{5n} & y_5 \\ 0 & 0 & h_{63} & \cdots & h_{6n} & y_6 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & h_{m3} & \cdots & h_{mn} & y_m \end{bmatrix}$$

$$= \begin{bmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1n} & y_1 \\ 0 & h_{22} & h_{23} & \cdots & h_{2n} & y_2 \\ 0 & 0 & h'_{33} & \cdots & h'_{3n} & y'_3 \\ 0 & 0 & 0 & \cdots & h'_{4n} & y'_4 \\ 0 & 0 & h_{53} & \cdots & h_{5n} & y_5 \\ 0 & 0 & h_{63} & \cdots & h_{6n} & y_6 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & h_{m3} & \cdots & h_{mn} & y_m \end{bmatrix}. \tag{5.4.13}$$

The prime superscript identifies the two rows that are affected by this transformation (rows three and four). Notice that the location of $-\sin\theta$ in the Givens transformation matrix corresponds to the location of the element to be nulled in the H matrix. For example, in the previous example both are element (4,3).

By using the transformation

$$Q^{5,3} = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & C^{5,3} & 0 & S^{5,3} & & & \\ & & 0 & 1 & 0 & & & \\ & & -S^{5,3} & 0 & C^{5,3} & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \\ & & & & & & & & \ddots \\ & & & & & & & & & 1 \end{bmatrix}, \tag{5.4.14}$$

the third and fifth rows will be transformed so that the term h_{53} will be zero.

5.4.1 *A Priori* Information and Initialization

The formulation given earlier does not specifically address the question of *a priori* information. Assume *a priori* information, $\overline{\mathbf{x}}$ and $\overline{P}$, are available. The procedure is initialized by writing the *a priori* information in the form of a data equation; that is, in the form of $\mathbf{y} = H\mathbf{x} + \boldsymbol{\epsilon}$. This is accomplished by writing

$$\overline{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta} \tag{5.4.15}$$

where $\mathbf{x}$ is the true value and $\boldsymbol{\eta}$ is the error in $\bar{\mathbf{x}}$. We assume that

$$E[\boldsymbol{\eta}] = 0,\ E[\boldsymbol{\eta}\boldsymbol{\eta}^T] = \overline{P}. \tag{5.4.16}$$

Compute $\overline{S}$, the upper triangular square root of $\overline{P}$,

$$\overline{P} = \overline{S}\,\overline{S}^T. \tag{5.4.17}$$

If $\overline{P}$ is not diagonal, the Cholesky decomposition may be used to accomplish this. Next compute $\overline{R}$, the square root of the *a priori* information matrix, $\overline{\Lambda}$,

$$\overline{\Lambda} = \overline{P}^{-1} = \overline{S}^{-T}\,\overline{S}^{-1} = \overline{R}^T\,\overline{R} \tag{5.4.18}$$

hence,

$$\overline{R} = \overline{S}^{-1}. \tag{5.4.19}$$

Multiplying Eq. (5.4.15) by $\overline{R}$ yields

$$\overline{R}\bar{\mathbf{x}} = \overline{R}\mathbf{x} + \overline{R}\boldsymbol{\eta}. \tag{5.4.20}$$

Define

$$\overline{\mathbf{b}} \equiv \overline{R}\bar{\mathbf{x}},\ \overline{\boldsymbol{\eta}} \equiv \overline{R}\boldsymbol{\eta}, \tag{5.4.21}$$

then

$$\overline{\mathbf{b}} = \overline{R}\mathbf{x} + \overline{\boldsymbol{\eta}} \tag{5.4.22}$$

where $\overline{\boldsymbol{\eta}} \sim (O, I)$. Note that Eq. (5.4.22) expresses the *a priori* information in the form of the data equation, $\mathbf{y} = H\mathbf{x} + \boldsymbol{\epsilon}$. Hence, the equations we wish to solve for $\hat{\mathbf{x}}$, using orthogonal transformations, are

$$\begin{aligned} \overline{\mathbf{b}} &= \overline{R}\mathbf{x} + \overline{\boldsymbol{\eta}} \\ \mathbf{y} &= H\mathbf{x} + \boldsymbol{\epsilon} \end{aligned} \tag{5.4.23}$$

where $\mathbf{x}$ is an n vector and $\mathbf{y}$ is an m vector.

The least squares solution for $\mathbf{x}$ in Eq. (5.4.23) is found by minimizing the performance index (we assume that $\boldsymbol{\epsilon}$ has been prewhitened so that $\boldsymbol{\epsilon} \sim (O, I)$; if not, replace $\boldsymbol{\epsilon}$ with $W^{1/2}\boldsymbol{\epsilon}$ in J)

$$\begin{aligned} J &= \|\overline{\boldsymbol{\eta}}\|^2 + \|\boldsymbol{\epsilon}\|^2 \\ &= \|\overline{R}\mathbf{x} - \overline{\mathbf{b}}\|^2 + \|H\mathbf{x} - \mathbf{y}\|^2 \\ &= \left\| \begin{bmatrix} \overline{R} \\ H \end{bmatrix} \mathbf{x} - \begin{bmatrix} \overline{\mathbf{b}} \\ \mathbf{y} \end{bmatrix} \right\|^2. \end{aligned} \tag{5.4.24}$$

After multiplying by an orthogonal transformation, Q, Eq. (5.4.24) may be written as

$$J = \left\{ \begin{bmatrix} \overline{R} \\ H \end{bmatrix} \mathbf{x} - \begin{bmatrix} \overline{\mathbf{b}} \\ \mathbf{y} \end{bmatrix} \right\}^T Q^T Q \left\{ \begin{bmatrix} \overline{R} \\ H \end{bmatrix} \mathbf{x} - \begin{bmatrix} \overline{\mathbf{b}} \\ \mathbf{y} \end{bmatrix} \right\}. \tag{5.4.25}$$

Choose Q so that

$$Q \begin{bmatrix} \overline{R} \\ H \end{bmatrix} = \begin{bmatrix} R \\ O \end{bmatrix}, \text{ and define } Q \begin{bmatrix} \overline{\mathbf{b}} \\ \mathbf{y} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{b} \\ \mathbf{e} \end{bmatrix} \tag{5.4.26}$$

where R is upper triangular. Eq. (5.4.24) can now be written as

$$J = \left\| \begin{bmatrix} R \\ O \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b} \\ \mathbf{e} \end{bmatrix} \right\|^2 \tag{5.4.27}$$

or

$$J = \left\| R\mathbf{x} - \mathbf{b} \right\|^2 + \left\| \mathbf{e} \right\|^2 \tag{5.4.28}$$

as noted before. The minimum value of J is found by choosing $\hat{\mathbf{x}}$ so that

$$R\hat{\mathbf{x}} - \mathbf{b} = 0. \tag{5.4.29}$$

The vector $\hat{\mathbf{x}}$ is obtained by the backward substitution described by Eq. (5.2.8), where z and r are replaced by b and R, respectively. Observe that $\hat{\mathbf{x}}$ usually would be determined after processing all observations. However, intermediate values of $\hat{\mathbf{x}}$ could be determined at any point in the process.

The minimum value of J is given by substituting $\hat{\mathbf{x}}$ into Eq. (5.4.24):

$$J = \|\mathbf{e}\|^2 = \sum_{i=1}^{m} e_i^2 = \left\| \overline{R}\hat{\mathbf{x}} - \overline{\mathbf{b}} \right\|^2 + \sum_{i=1}^{m} \left(H_i\hat{\mathbf{x}} - y_i \right)^2. \tag{5.4.30}$$

Note that the first term on the right-hand side of Eq. (5.4.30) corresponds to the norm of the error in the *a priori* value for $\mathbf{x}$ multiplied by the square root of the inverse of the *a priori* covariance matrix,

$$\|\overline{R}\hat{\mathbf{x}} - \overline{\mathbf{b}}\|^2 = \|\overline{R}(\hat{\mathbf{x}} - \overline{\mathbf{x}})\|^2 \tag{5.4.31}$$

which also can be expressed as

$$\left\| \overline{R}(\hat{\mathbf{x}} - \overline{\mathbf{x}}) \right\|^2 = (\hat{\mathbf{x}} - \overline{\mathbf{x}})^T \overline{P}^{-1} (\hat{\mathbf{x}} - \overline{\mathbf{x}}). \tag{5.4.32}$$

From Eqs. (5.4.30) and (5.4.31) it is seen that

$$\sum_{i=1}^{m} e_i^2 = \|\overline{R}(\hat{\mathbf{x}} - \overline{\mathbf{x}})\|^2 + \sum_{i=1}^{m} \left(H_i\hat{\mathbf{x}} - y_i \right)^2$$

or

$$\sum_{i=1}^{m} e_i^2 = \|\overline{R}\hat{\boldsymbol{\eta}}\|^2 + \sum_{i=1}^{m} \hat{\epsilon}_i^2 \tag{5.4.33}$$
$$= \hat{\boldsymbol{\eta}}^T \overline{R}^T \overline{R}\hat{\boldsymbol{\eta}} + \sum_{i=1}^{m} \hat{\epsilon}_i^2,$$

where

$$\hat{\boldsymbol{\eta}} = \hat{\mathbf{x}} - \overline{\mathbf{x}}, \quad \hat{\epsilon}_i = y_i - H_i\hat{\mathbf{x}}. \tag{5.4.34}$$

Consequently, $e_i \neq \hat{\epsilon}_i$; that is, the elements of the error vector $[\mathbf{e}]_{m\times 1}$ contain a contribution from errors in the *a priori* information as well as the observation residuals. The RMS of the observation residuals, $\hat{\epsilon}_i$, is given by

$$\text{RMS} = \sqrt{\frac{1}{m}\sum_{i=1}^{m} \hat{\epsilon}_i^2} \tag{5.4.35}$$

and from Eq. (5.4.33)

$$\sum_{i=1}^{m} \hat{\epsilon}_i^2 = \sum_{i=1}^{m} e_i^2 - \hat{\boldsymbol{\eta}}^T \overline{R}^T \overline{R}\hat{\boldsymbol{\eta}}. \tag{5.4.36}$$

If the procedure is initialized with n observations in place of *a priori* information, $\overline{\mathbf{x}}$ and $\overline{P}$, Eq. (5.4.33) becomes

$$\sum_{i=1}^{m-n} e_i^2 = \sum_{i=1}^{m} \hat{\epsilon}_i^2 \tag{5.4.37}$$

and again $e_i \neq \hat{\epsilon}_i$ because the first n observations serve the same function as *a priori* values of $\overline{\mathbf{x}}$ and $\overline{P}$. Here we have assumed that the weighting matrix is the identity matrix. If not, $(H_i\hat{\mathbf{x}} - y_i)^2$ in Eq. (5.4.30) and subsequent equations should be replaced by $W_i(H_i\hat{\mathbf{x}} - y_i)^2$.

An advantage of the orthogonal transformation approach, in addition to improved accuracy, is that the sum of squares of the residuals, e_i, based on the final value of $\hat{\mathbf{x}}$ is computed automatically as part of the solution procedure. To obtain this result with the conventional batch processor, one would have to go through the additional computation of evaluating Eq. (5.4.30) after solving the normal equations for $\hat{\mathbf{x}}$.

This procedure of solving the least squares problem using orthogonal transformations can be described as a square root information batch processor. If the state vector is independent of time, it could be thought of as a filter because the

best estimate of $\mathbf{x}$ could be generated after processing each observation. If the state vector is a function of time, the H matrix must be combined with the state transition matrix to map the state to an epoch time as described in Chapter 4. As formulated here, it could not be implemented as a filter without mapping the square root information matrix, R, and the state estimate, $\hat{\mathbf{x}}$, to the appropriate time for each observation. The square root information filter with time-dependent effects is described in Section 5.10.1.

In summary, given *a priori* information $\overline{R}$ and $\overline{\mathbf{b}}$ and observations

$$y_i = H_i\mathbf{x} + \epsilon_i, \quad i = 1, \ldots, m,$$

the matrix we wish to reduce to upper triangular form is

$$\begin{array}{cc} \overbrace{}^{n} & \overbrace{}^{1} \\ \left[\begin{array}{cc} \overline{R} & \overline{\mathbf{b}} \\ H_1 & y_1 \\ H_2 & y_2 \\ \vdots & \vdots \\ H_m & y_m \end{array}\right] & \begin{array}{l} \}\, n \\ \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} m \end{array} \end{array} \tag{5.4.38}$$

where $\overline{R}$ is upper triangular.

5.4.2 Givens Computational Algorithm

For purposes of the computational algorithm we will write Eq. (5.4.38) as

$$\begin{array}{cc} \overbrace{}^{n} \overbrace{}^{1} & \overbrace{}^{n+1} \\ \left[\begin{array}{cc} \overline{R} & \overline{\mathbf{b}} \\ H & \mathbf{y} \end{array}\right] \begin{array}{l} \}n \\ \}m \end{array} = & \left[\begin{array}{c} \widetilde{R} \\ \widetilde{H} \end{array}\right] \begin{array}{l} \}n \\ \}m. \end{array} \end{array} \tag{5.4.39}$$

Lowercase r and h in the following algorithm represent the elements of $\widetilde{R}$ and $\widetilde{H}$, respectively, in Eq. (5.4.39).

The algorithm using the Givens rotation for reducing the $(m+n) \times (n+1)$ matrix of Eq. (5.4.39) to upper triangular form can be expressed as follows:

Sum = 0.

1. Do $k = 1, ..., m$

2. Do $i = 1, \ldots, n$

   If $(h_{ki} = 0)$ Go to 2

$$
\begin{aligned}
r'_{ii} &= \sqrt{r_{ii}^2 + h_{ki}^2} \\
S_{ik} &= h_{ki}/r'_{ii} \\
C_{ik} &= r_{ii}/r'_{ii} \\
h_{ki} &= 0 \\
r_{ii} &= r'_{ii}
\end{aligned}
$$

3. Do $j = i + 1, \ldots, n + 1$

$$
\begin{aligned}
r'_{ij} &= C_{ik} r_{ij} + S_{ik} h_{kj} \\
h_{kj} &= -S_{ik} r_{ij} + C_{ik} h_{kj} \\
r_{ij} &= r'_{ij}
\end{aligned}
\tag{5.4.40}
$$

   Next j
   Next i
   $e_k = h_{kj}$
   Sum = Sum $+ e_k^2$
   Next k

After application of this algorithm, the $(n + m) \times (n + 1)$ matrix will appear as

$$
Q \left[\begin{array}{c:c} \overline{R} & \overline{\mathbf{b}} \\ \hdashline H & \mathbf{y} \end{array}\right] = \left[\begin{array}{c:c} \overbrace{R}^{n} & \overbrace{\mathbf{b}}^{1} \\ \hdashline O & \mathbf{e} \end{array}\right] \begin{matrix} \}n \\ \}m \end{matrix} \tag{5.4.41}
$$

which is the required form for solution of the least squares estimation problem as given by Eq. (5.3.6). Note that $r_{i,n+1} (i = 1, \ldots, n)$ and $h_{k,n+1} (k = 1, \ldots, m)$ given by the algorithm represent $\mathbf{b}$ and $\mathbf{e}$, respectively, in Eq. (5.4.41). Also, Sum $= \sum_{k=1}^{m} e_k^2$.

Once the array has been reduced to the form given by Eq. (5.4.41), subsequent observations can be included by considering the following array:

$$
\left[\begin{array}{cc} R & \mathbf{b} \\ H_{m+1} & y_{m+1} \\ 0 & e^2 \end{array}\right] = \tag{5.4.42}
$$

$$\begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1n} & b_1 \\ 0 & R_{22} & \cdots & R_{2n} & b_2 \\ 0 & 0 & \cdots & R_{3n} & b_3 \\ \vdots & & & & \\ 0 & 0 & \cdots & R_{nn} & b_n \\ H_{m+1,1} & H_{m+1,2} & \cdots & H_{m+1,n} & y_{m+1} \\ 0 & 0 & \cdots & 0 & e^2 \end{bmatrix}$$

where

$$e^2 = \sum_{k=1}^{m} e_k^2 = \text{Sum}.$$

Then by application of a Givens rotation to rows 1 and $n+1$, $H_{m+1,1}$ can be nulled. Successive applications moving down the main diagonal can be used to null the remaining $n-1$ elements of the $n+1^{\text{st}}$ row and reduce the array to upper triangular form:

$$\begin{bmatrix} R' & \mathbf{b}' \\ 0 & e_{m+1} \\ 0 & e^2 \end{bmatrix}.$$

Next e^2 is replaced by $e^2 + e_{m+1}^2$ and the procedure is repeated with the next observation, and so on. It is also obvious that a group of m' observations could be included by replacing the array (H_{m+1}, y_{m+1}) with an array in which H_{m+1} has dimension $(m' \times n)$ and $\mathbf{y}_{m+1}$ has dimension m'. The Givens rotation would be used as before to reduce the augmented array to upper triangular form. Note that Sum is set to zero only before processing the first observation or batch of observations. Also note that if there are different observation types, e.g., range and range rate, the values of e for each observation type should be stored in separate arrays.

The Givens algorithm operates on an individual row (i.e., observation) until it is in the proper form and then moves to the next row of the matrix. The algorithm can also be applied so that it operates column by column simply by interchanging the k and i loops. The same procedure just described for processing a new observation or batch of observations still applies. The following Givens algorithm operates on the successive columns.

Sum = 0.

1. Do $i = 1, \ldots, n$
2. Do $k = 1, \ldots, m$

If $(h_{ki} = 0)$ Go to 2

$$r'_{ii} = \sqrt{r_{ii}^2 + h_{ki}^2}$$
$$S_{ik} = h_{ki}/r'_{ii}$$
$$C_{ik} = r_{ii}/r'_{ii}$$
$$h_{ki} = 0$$
$$r_{ii} = r'_{ii}$$

3. Do $j = i+1, \ldots, n+1$

$$r'_{ij} = C_{ik} r_{ij} + S_{ik} h_{kj}$$
$$h_{kj} = -S_{ik} r_{ij} + C_{ik} h_{kj}$$
$$r_{ij} = r'_{ij}$$

Next j

Next k

Next i

Do $\ell = 1, \ldots, m$

Sum = Sum $+h_{\ell,n+1}^2$

Next ℓ

5.4.3 SQUARE ROOT FREE GIVENS TRANSFORMATION

The original Givens transformation requires the formation of square roots, which are more complex to compute than divisions or multiplications. The following procedure leads to a square root free algorithm.

Rather than seek the orthogonal transformation Q, which leads to $QW^{1/2}H = [R^T \vdots 0]^T$, we seek a factorization of R of the form

$$R = D^{\frac{1}{2}} U$$

where D is an $n \times n$ diagonal matrix and U is $n \times n$ unit upper triangular matrix,

i.e.

$$D^{1/2} = \begin{bmatrix} d_1^{1/2} & & & 0 \\ & d_2^{1/2} & & \\ & & \ddots & \\ 0 & & & d_n^{1/2} \end{bmatrix}, U = \begin{bmatrix} 1 & U_{12} & & \cdots & U_{1n} \\ 0 & 1 & U_{23} & \cdots & U_{2n} \\ \vdots & & & & \vdots \\ 0 & \cdots & \cdots & \cdots & 1 \end{bmatrix}. \tag{5.4.43}$$

Following Eq. (5.3.6) select the orthogonal matrix Q, such that

$$QW^{\frac{1}{2}}H = \begin{bmatrix} D^{\frac{1}{2}}U \\ 0 \end{bmatrix} = \begin{bmatrix} D^{\frac{1}{2}} \vdots 0 \\ \cdots \quad \cdots \\ 0 \vdots 0 \end{bmatrix} \begin{bmatrix} U \\ \cdots \\ 0 \end{bmatrix} \tag{5.4.44}$$

and

$$QW^{\frac{1}{2}}\mathbf{y} = \begin{bmatrix} \mathbf{b} \\ \mathbf{e} \end{bmatrix}. \tag{5.4.45}$$

Let

$$\mathbf{b} = D^{\frac{1}{2}}\widetilde{\mathbf{b}}. \tag{5.4.46}$$

It follows from Eq. (5.3.7) that the least squares performance index can be expressed as

$$J(\mathbf{x}) = \left\| \begin{bmatrix} D^{\frac{1}{2}} & U \\ 0 \end{bmatrix} \mathbf{x} - \begin{bmatrix} D^{\frac{1}{2}} & \widetilde{\mathbf{b}} \\ \mathbf{e} \end{bmatrix} \right\|^2 \tag{5.4.47}$$

which for a minimum requires

$$D^{\frac{1}{2}}U\hat{\mathbf{x}} = D^{\frac{1}{2}}\widetilde{\mathbf{b}}. \tag{5.4.48}$$

Since $D^{1/2}$ is common to both sides of Eq. (5.4.48), the solution is

$$U\,\hat{\mathbf{x}} = \widetilde{\mathbf{b}} \tag{5.4.49}$$

and $\hat{\mathbf{x}}$ is obtained by backward recursion. Because the diagonals of U are unitary, division by the n diagonal elements is eliminated.

Consider now the use of the square root free Givens transformation to obtain the orthogonal decomposition (Gentleman, 1973). The product $W^{1/2}H$ can be

expressed in component form as

$$W^{\frac{1}{2}}H = \begin{bmatrix} \sigma_1 h_{11} & \sigma_1 h_{12} & \cdots & \sigma_1 h_{1n} \\ \sigma_2 h_{21} & \sigma_2 h_{22} & \cdots & \sigma_2 h_{2n} \\ \vdots & & & \\ \sigma_m h_{m1} & \sigma_m h_{m2} & \cdots & \sigma_m h_{mn} \end{bmatrix} \tag{5.4.50}$$

where

$$\sigma_i = W_{ii}^{\frac{1}{2}}, W_{ij}^{\frac{1}{2}} = 0, i \neq j.$$

Now consider the application of the Givens rotation, where any two rows of Eq. (5.4.19) are expressed in the form

$$G \begin{bmatrix} \sqrt{d_i} & \sqrt{d_i}\, l_{i,i+1} & \cdots & \sqrt{d_i}\, l_{in} \\ \sqrt{\delta_k}\, h_{ki} & \sqrt{\delta_k}\, h_{k,i+1} & \cdots & \sqrt{\delta_k}\, h_{kn} \end{bmatrix}$$

$$= \begin{bmatrix} \sqrt{d_i'} & \sqrt{d_i'}\, l'_{i,i+1} & \cdots & \sqrt{d_i'}\, l'_{in} \\ 0 & \sqrt{\delta_k'}\, h'_{k,i+1} & \cdots & \sqrt{\delta_k'}\, h'_{kn} \end{bmatrix}. \tag{5.4.51}$$

From Eq. (5.4.50), it follows that

$$\sqrt{d_i} = \sigma_i h_{ii}$$

$$l_{ij} = \sigma_i h_{ij} / \sigma_i h_{ii}, j = i+1, \ldots, , n$$

$$\sqrt{\delta_k} = \sigma_k. \tag{5.4.52}$$

Then from Eq. (5.4.51), the Givens transformation, Eq. (5.4.2), applied to the first column leads to

$$\begin{aligned} \sqrt{d'} &= \cos\theta \sqrt{d_i} + \sin\theta \sqrt{\delta_k} h_{ki} \\ 0 &= -\sin\theta \sqrt{d_i} + \cos\theta \sqrt{\delta_k} h_{ki}. \end{aligned} \tag{5.4.53}$$

The second of Eq. (5.4.53) can be used to obtain

$$\tan\theta = \frac{\sin\theta}{\cos\theta} = \frac{\sqrt{\delta_k} h_{ki}}{\sqrt{d_i}}$$

and hence

$$\sin\theta = \sqrt{\delta_k} h_{ki} \Big/ \sqrt{d_i + \delta_k h_{ki}^2}$$

$$\cos\theta = \sqrt{d_i} \Big/ \sqrt{d_i + \delta_k h_{ki}^2}. \tag{5.4.54}$$

Substituting (5.4.54) into (5.4.53) leads to

$$\sqrt{d_i'} = \sqrt{d_i + \delta_k h_{ki}^2}. \tag{5.4.55}$$

Hence, Eq. (5.4.54) becomes

$$\sin\theta = \sqrt{\delta_k}\ h_{ki}/\sqrt{d_i'}; \cos\theta = \sqrt{d_i}/\sqrt{d_i'}. \tag{5.4.56}$$

Then for the general transformation, the ij^{th} element of the first row is

$$\begin{aligned}\sqrt{d_i'}\ell_{ij}' &= \cos\theta\sqrt{d_i}\ell_{ij} + \sin\theta\sqrt{\delta_k}\ h_{kj} \\ &= \frac{\sqrt{d_i}\sqrt{d_i}}{\sqrt{d_i'}}\ell_{ij} + \frac{\sqrt{\delta_k}h_{ki}\sqrt{\delta_k}}{\sqrt{d_i'}}h_{kj}.\end{aligned}$$

Now, dividing by $\sqrt{d_i'}$, yields

$$\ell_{ij}' = \frac{d_i}{d_i'}\ell_{ij} + \frac{\delta_k}{d_i'}h_{ki}h_{kj}. \tag{5.4.57}$$

Using the definitions

$$\overline{C_i} = d_i/d_i', \overline{S}_i = \delta_k h_{ki}/d_i'$$

Eq. (5.4.57) becomes

$$\ell_{ij}' = \overline{C}_i\ell_{ij} + \overline{S}_i h_{kj}. \tag{5.4.58}$$

Similarly, the kj^{th} element of the k^{th} row is

$$\sqrt{\delta_k'}h_{kj}' = -\sin\theta\sqrt{d_i}\ell_{ij} + \cos\theta\sqrt{\delta_k}h_{kj}.$$

Then, using Eq. (5.4.56) results in

$$\sqrt{\delta_k'}h_{kj}' = \frac{-\sqrt{\delta_k}h_{ki}}{\sqrt{d_i'}}\sqrt{d_i}\ l_{ij} + \frac{\sqrt{d_i}}{\sqrt{d_i'}}\sqrt{\delta_k}h_{kj}. \tag{5.4.59}$$

Dividing by $\sqrt{\delta_k'}$ leads to

$$h_{kj}' = \frac{\sqrt{d_i}}{\sqrt{d_i'}}\frac{\sqrt{\delta_k}}{\sqrt{\delta_k'}}h_{kj} - \frac{\sqrt{\delta_k}\sqrt{d_i}}{\sqrt{d_i'}\sqrt{\delta_k'}}h_{ki}l_{ij}. \tag{5.4.60}$$

Because $\sqrt{\delta_k'}$ is an arbitrary scaling factor for each element of the k^{th} row, it is convenient to let

$$\sqrt{\delta_k'} = \sqrt{\delta_k} \times \frac{\sqrt{d_i}}{\sqrt{d_i'}}. \tag{5.4.61}$$

It then follows that Eq. (5.4.60) can be expressed as

$$h_{kj}' = h_{kj} - h_{ki} l_{ij}. \tag{5.4.62}$$

The final values of l_{ij}' become the elements of U_{ij} and $\tilde{b}_i$, and h_{kj}' are the interim elements of the H matrix. The final value of h_{kj}' in each row is the observation error. This will become clear from the examples of Section 5.6.

5.4.4 Square Root Free Givens Computational Algorithm

If *a priori* information, $\overline{\mathrm{x}}$ and $\overline{P}$, are available the algorithm is initialized by computing $\overline{S}$ and $\overline{R}$ where

$$\overline{P} = \overline{S}\,\overline{S}^T \tag{5.4.63}$$

and

$$\overline{R} = \overline{S}^{-1}. \tag{5.4.64}$$

The *a priori* information we need is $\overline{D}$, $\overline{U}$, and $\widetilde{\mathrm{b}}$, where

$$\overline{R} = \overline{D}^{\frac{1}{2}}\overline{U}. \tag{5.4.65}$$

Hence,

$$\overline{d}_i = \overline{R}_{ii}^2 \quad i = 1 \ldots n \tag{5.4.66}$$

and

$$\overline{U} = \begin{bmatrix} 1 & \overline{U}_{12} & \cdots & \cdots & \overline{U}_{1n} \\ 0 & 1 & \overline{U}_{23} & \cdots & \overline{U}_{2n} \\ 0 & 0 & 1 & & \vdots \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 1 \end{bmatrix}, \tag{5.4.67}$$

where $\overline{d}_i$ are the square of the diagonal elements of $\overline{R}$; for example, $d_i = R_{ii}^2$, $i = 1, \ldots, n$, and $\overline{U}_{ij} = \overline{R}_{ij}/\overline{R}_{ii}$; $i = 1, \ldots n$; $j = i + 1, n$. Also, recall from Eq. (5.4.21) that

$$\overline{\mathrm{b}} = \overline{R}\,\overline{\mathrm{x}}$$

and the *a priori* value of $\tilde{\mathbf{b}} \equiv \overline{\tilde{\mathbf{b}}}$ is

$$\overline{\tilde{\mathbf{b}}} = \overline{D}^{-\frac{1}{2}}\,\overline{\mathbf{b}}. \tag{5.4.68}$$

Given observations y_i and $i = 1 \ldots m$, we now have the necessary information to execute the algorithm; that is, we wish to apply a series of orthogonal transformations, G, so that

$$G \begin{bmatrix} \overbrace{\overline{U}}^{n} & \overbrace{\overline{\tilde{\mathbf{b}}}}^{1} \\ H_1 & y_1 \\ \vdots & \\ H_m & y_m \end{bmatrix} \Bigg\} \begin{matrix} n \\ \\ m \end{matrix} = \begin{bmatrix} \overbrace{U}^{n} & \overbrace{\tilde{\mathbf{b}}}^{1} \\ & e_1 \\ 0 & \vdots \\ & e_m \end{bmatrix} \Bigg\} \begin{matrix} n \\ \\ m \end{matrix} . \tag{5.4.69}$$

The computational algorithm for operating on the data one row, or one observation, at a time is as follows:

Sum = 0

$U_{ii} = 1 \qquad i = 1, \ldots, n$

1. Do $k = 1, \ldots, m$

$$\delta_k = 1$$

2. Do $i = 1, \ldots, n$

If $(h_{ki} = 0)$ Go to 2

$$\begin{aligned} d_i' &= d_i + \delta_k h_{ki}^2 \\ \overline{C} &= d_i / d_i' \\ \overline{S} &= \delta_k h_{ki} / d_i' \\ y_k' &= y_k - \tilde{b}_i h_{ki} \\ \tilde{b}_i &= \tilde{b}_i \overline{C} + y_k \overline{S} \\ y_k &= y_k' \\ \delta_k &= \delta_k \overline{C} \\ d_i &= d_i' \end{aligned} \tag{5.4.70}$$

3. Do $j = i + 1, \ldots, n$

$$\begin{aligned} h_{kj}' &= h_{kj} - U_{ij} h_{ki} \\ U_{ij} &= U_{ij} \overline{C} + h_{kj} \overline{S} \\ h_{kj} &= h_{kj}' \end{aligned}$$

Next j
Next i
$e_k = \sqrt{\delta_k} y_k$
Sum = Sum + e_k^2

Next k

The diagonal elements of D are given by $d_i (i = 1, \ldots, n)$, the upper triangular elements of U are given by $U_{ij} (i = 1, \ldots, n,\ j = i + 1, \ldots, n + 1)$, the elements of $\tilde{\mathbf{b}}$ are given by $\tilde{b}_i$ $(i = 1, \ldots, n)$ and the elements of $\mathbf{e}$ are given by $e_k (k = 1, \ldots, m)$. The same procedure described for the Givens algorithm at the end of Section 5.4.2 can be used to handle multiple batches of observation data. Note that Sum and δ_k are set to zero and one, respectively, only for the first batch of observations.

The vector $\hat{\mathbf{x}}$ is obtained from U and $\tilde{\mathbf{b}}$ by performing a back substitution using Eq. (5.4.49). Note that D is not needed to compute $\hat{\mathbf{x}}$ but is needed to compute the estimation error covariance matrix, $P = U^{-1} D^{-1} U^{-T}$ (see Eq. (5.4.80)).

5.4.5 A Simplified Square Root Free Givens Transformation

The square root free Givens algorithm can be simplified further (Gentleman, 1973) by noting that we may write

$$\begin{aligned} y_k' &= y_k - \tilde{b}_i h_{ki} \\ \tilde{b}_i' &= \tilde{b}_i \overline{C} + y_k \overline{S} \end{aligned} \tag{5.4.71}$$

and

$$\begin{aligned} h_{kj}' &= h_{kj} - U_{ij} h_{ki} \\ U_{ij}' &= U_{ij} \overline{C} + h_{kj} \overline{S} \end{aligned} \tag{5.4.72}$$

as

$$\begin{aligned} y_k' &= y_k - \tilde{b}_i h_{ki} \\ \tilde{b}_i' &= \tilde{b}_i + y_k' \overline{S} \end{aligned} \tag{5.4.73}$$

and

$$\begin{aligned} h_{kj}' &= h_{kj} - U_{ij}\ h_{ki} \\ U_{ij}' &= U_{ij} + h_{kj}'\ \overline{S} \end{aligned} \tag{5.4.74}$$

by noting that

$$\overline{C} = 1 - \overline{S}\ h_{ki}. \tag{5.4.75}$$

Hence, the algorithm may be simplified by eliminating the need to compute $\overline{C}$ explicitly. The computational algorithm becomes

Sum = 0.

1. Do $k = 1, \ldots, m$

$$\delta_k = 1$$

2. Do $i = 1, \ldots, n$

If $(h_{ki} = 0)$ go to 2

$$\begin{aligned} d_i' &= d_i + \delta_k\ h_{ki}^2 \\ \overline{S} &= \delta_k\ h_{ki}/d_i' \\ y_k' &= y_k - \tilde{b}_i\ h_{ki} \\ \tilde{b}_i &= \tilde{b}_i + y_k'\ \overline{S} \\ y_k &= y_k' \\ \delta_k &= \delta_k\ d_i/d_i' \\ d_i &= d_i' \end{aligned} \tag{5.4.76}$$

3. Do $j = i + 1, \ldots, n$

$$\begin{aligned} h_{kj}' &= h_{kj} - U_{ij}\, h_{ki} \\ U_{ij} &= U_{ij} + h_{kj}'\ \overline{S} \\ h_{kj} &= h_{kj}' \end{aligned}$$

Next j

Next i

$$\begin{aligned} e_k &= \sqrt{\delta_k}\ y_k \\ \text{Sum} &= \text{Sum} + e_k^2 \end{aligned}$$

Next k

This version of the square root free Givens algorithm is preferred because it involves fewer operations than that of Eq. (5.4.70).

5.4.6 IMPLEMENTATION CONSIDERATIONS

The following observations on the square root free algorithm are given:

1. Note that no square roots are required.

2. The algorithm assumes that *a priori* information $\overline{D}$, $\overline{U}$, and $\overline{\tilde{\mathbf{b}}}$ computed from $\overline{P}$ and $\bar{\mathbf{x}}$ is available. These are computed using Eqs. (5.4.63) through (5.4.68):

$$\begin{aligned} \overline{P} &= \overline{S}\,\overline{S}^T \\ \overline{R} &= \overline{S}^{-1} \\ \bar{d}_i &= \overline{R}_{ii}^2, \quad i = 1 \ldots n \\ \overline{U} &= \overline{D}^{-1/2}\,\overline{R} \\ \overline{\tilde{\mathbf{b}}} &= \overline{D}^{-1/2}\,\overline{R}\,\bar{\mathbf{x}} = \overline{U}\bar{\mathbf{x}}. \end{aligned} \tag{5.4.77}$$

3. If no *a priori* information is given the algorithm may be initialized using

$$\begin{aligned} \overline{D} &= 10^{-16\ddagger} \\ \overline{U} &= I \\ \overline{\tilde{\mathbf{b}}} &= 0. \end{aligned} \tag{5.4.78}$$

4. When an unprimed variable appears on both sides of an equation, this is taken to represent a replacement.

5. Remember that an orthogonal transformation does not change the norm or Euclidian length of each column of the original matrix.

6. Each of the Givens algorithms (Eqs. (5.4.40), (5.4.70), and (5.4.76)) operates on the subject matrix row by row. They can be modified to operate column by column by interchanging the k and i loops.

The value of $\hat{\mathbf{x}}$ is computed from Eq. (5.4.49),

$$U\hat{\mathbf{x}} = \tilde{\mathbf{b}}$$

‡ $\overline{D}$ is an $n \times n$ diagonal matrix with 10^{-16} on the diagonal, using $\overline{D} = 0$ causes the algorithm to fail when computing $\tilde{b}$; hence, a small number should be used. This is equivalent to $\overline{P}$ being diagonal with values of 10^{16} and $\bar{\mathbf{x}} = 0$.

by using the back substitution algorithm obtained by a slight modification of Eq. (5.2.8):

$$\begin{aligned} &i = n,\ \ n-1, \ldots, 1 \\ &\hat{x}_i = \tilde{b}_i - \sum_{j=i+1}^{n} U_{ij}\hat{x}_j. \end{aligned} \tag{5.4.79}$$

The estimation error covariance is computed from

$$\begin{aligned} P &= R^{-1}\, R^{-T} \\ &= (D^{\frac{1}{2}}\, U)^{-1}(U^T D^{\frac{1}{2}})^{-1} \\ &= U^{-1} D^{-\frac{1}{2}}\, D^{-\frac{1}{2}} U^{-T} \\ &= U^{-1} D^{-1} U^{-T} \end{aligned} \tag{5.4.80}$$

where U^{-1} may be computed by using Eq. (5.2.20).

5.5 THE HOUSEHOLDER TRANSFORMATION

An alternate computational approach can be developed by using orthogonal reflections rather than the planar rotations used in the previous discussions. Such transformations, which are referred to as Householder transformations (Householder, 1958), have the advantage of nulling a complete column in a single operation. Consider the following matrix

$$T = I - 2\hat{\mathbf{u}}\hat{\mathbf{u}}^T$$

where $\hat{\mathbf{u}}^T\hat{\mathbf{u}} = 1$. The matrix T satisfies the following conditions:

1) T is symmetric.

2) T is idempotent: $T^2 = I$. (5.5.1)

3) T is orthogonal: $TT^T = I$.

Proof: The first condition follows from the definition. Then, since $\hat{\mathbf{u}}\hat{\mathbf{u}}^T$ is symmetric and $\hat{\mathbf{u}}^T\hat{\mathbf{u}} = 1$,

$$\begin{aligned} T^2 &= (I - 2\hat{\mathbf{u}}\hat{\mathbf{u}}^T)(I - 2\hat{\mathbf{u}}\hat{\mathbf{u}}^T) = I - 4\hat{\mathbf{u}}\hat{\mathbf{u}}^T + 4\hat{\mathbf{u}}(\hat{\mathbf{u}}^T\hat{\mathbf{u}})\hat{\mathbf{u}}^T \\ &= I - 4\hat{\mathbf{u}}\hat{\mathbf{u}}^T + 4\hat{\mathbf{u}}\hat{\mathbf{u}}^T = I. \end{aligned} \tag{5.5.2}$$

From Properties 1 and 2, it follows that

$$T^2 = TT^T = I. \tag{5.5.3}$$

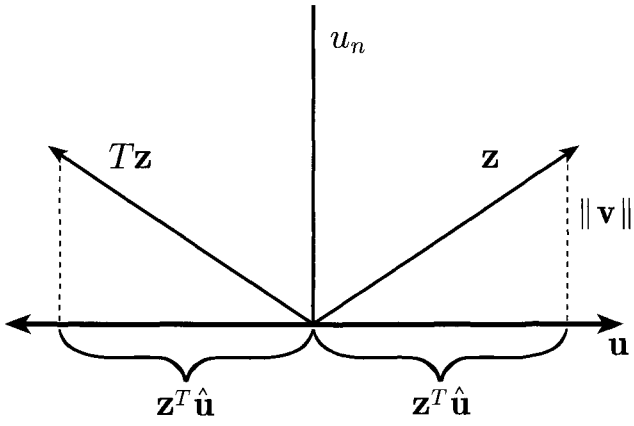

Figure 5.5.1: An elementary reflection.

Matrices of the form T are called elementary Hermitian matrices, elementary reflectors, or Householder transformations. The geometric notion of a reflection is shown in Fig. 5.5.1 and can be developed as follows. Let u_n be the plane perpendicular to $\mathbf{u}$. Let

$$\hat{\mathbf{u}} = \mathbf{u}/\|\mathbf{u}\|$$

and let $T\mathbf{z}$ be the reflection of $\mathbf{z}$ in the plane u_n—the mirror image of $\mathbf{z}$ where the plane u_n represents the mirror. That is,

$$\mathbf{z} = (\mathbf{z}^T\hat{\mathbf{u}})\hat{\mathbf{u}} + \mathbf{v} \tag{5.5.4}$$

and

$$T\mathbf{z} = -(\mathbf{z}^T\hat{\mathbf{u}})\hat{\mathbf{u}} + \mathbf{v}. \tag{5.5.5}$$

Eliminating $\mathbf{v}$ leads to

$$-T\mathbf{z} + \mathbf{z} = 2(\mathbf{z}^T\hat{\mathbf{u}})\hat{\mathbf{u}}.$$

Therefore,

$$\begin{aligned} T\mathbf{z} &= \mathbf{z} - 2(\mathbf{z}^T\hat{\mathbf{u}})\hat{\mathbf{u}} \\ &= \mathbf{z} - 2\hat{\mathbf{u}}(\hat{\mathbf{u}}^T\mathbf{z}) \\ &= \mathbf{z} - 2\hat{\mathbf{u}}\hat{\mathbf{u}}^T\mathbf{z} \end{aligned}$$

where we have used the fact that $\mathbf{z}^T\hat{\mathbf{u}} = \hat{\mathbf{u}}^T\mathbf{z}$ because this quantity is a scalar. Finally,

$$T\mathbf{z} = [I - 2\hat{\mathbf{u}}\hat{\mathbf{u}}^T]\mathbf{z}. \tag{5.5.6}$$

Hence,

$$\begin{aligned} T &= I - 2\frac{\mathbf{u}\mathbf{u}^T}{\mathbf{u}^T\mathbf{u}} \\ &= I - \beta\mathbf{u}\mathbf{u}^T \end{aligned} \tag{5.5.7}$$

where

$$\beta = 2/\mathbf{u}^T\mathbf{u} = 2\Big/\|\mathbf{u}\|^2. \tag{5.5.8}$$

The following additional properties are of use:

4) If u_j, the j^{th} component of $\mathbf{u}$, is zero, then $(T\mathbf{z})_j = z_j$. That is, if the j^{th} component of $\mathbf{u}$ is zero, then the transformation T leaves the j^{th} component of $\mathbf{z}$ unchanged.

Proof: This can be verified by writing down the product $\mathbf{u}\mathbf{u}^T$ and noting that the j^{th} row and j^{th} column vanish if u_j is zero.

5) If $\mathbf{u}$ is perpendicular to $\mathbf{z}$, then $T\mathbf{z} = \mathbf{z}$.

Proof: Since $T\mathbf{z} = [I - \beta\mathbf{u}\mathbf{u}^T]\mathbf{z} = \mathbf{z} - \beta\mathbf{u}(\mathbf{u}^T\mathbf{z})$, it follows that $\mathbf{u}^T\mathbf{z} = 0$ if $\mathbf{u}$ is perpendicular to $\mathbf{z}$ and hence that $T\mathbf{z} = \mathbf{z}$.

6) $T\mathbf{z} = \mathbf{z} - \gamma\mathbf{u}$, where $\gamma = 2\mathbf{z}^T\mathbf{u}/\mathbf{u}^T\mathbf{u}$. (5.5.9)

Proof:

$$\begin{aligned} T\mathbf{z} &= \left[I - \frac{2\mathbf{u}\mathbf{u}^T}{\mathbf{u}^T\mathbf{u}}\right]\mathbf{z} = \mathbf{z} - 2\mathbf{u}\mathbf{u}^T\mathbf{z}/\mathbf{u}^T\mathbf{u} \\ &= \mathbf{z} - \frac{2(\mathbf{z}^T\mathbf{u})\mathbf{u}}{\mathbf{u}^T\mathbf{u}} = \mathbf{z} - 2\left(\frac{\mathbf{z}^T\mathbf{u}}{\mathbf{u}^T\mathbf{u}}\right)\mathbf{u} \\ &= \mathbf{z} - \gamma\mathbf{u}. \end{aligned} \tag{5.5.10}$$

Here we use the fact that $\mathbf{u}^T\mathbf{z}$ is a scalar; hence, $\mathbf{u}^T\mathbf{z} = \mathbf{z}^T\mathbf{u}$. Note that the computation of T requires significantly more computation than the computation of $T\mathbf{z}$, if property 6 is used. Furthermore, using property 6, it is not necessary that T be stored; only the elements of $T\mathbf{z}$ need to be stored.

The feature of the Householder transformation, T, that is of crucial interest in this discussion is that T can be used to introduce zeros into any column vector, $\mathbf{z}$. One can always find an orthogonal matrix, T, such that $T\mathbf{z} = -\sigma\mathbf{e}_1$ where σ is a scalar and $\mathbf{e}_1$ is a unit vector in the direction of z_1, the first component of the

column vector $\mathbf{z}$. The constant σ can be determined up to a sign by the fact that T is orthogonal. That is,

$$\|T\mathbf{z}\|^2 = \| - \sigma \mathbf{e}_1\|^2 = \sigma^2 = \|\mathbf{z}\|^2 = \mathbf{z}^T\mathbf{z}.$$

Therefore,

$$\sigma = \pm \|\mathbf{z}\|. \tag{5.5.11}$$

To obtain the specific transformation, T, which accomplishes this reduction, the vector, $\mathbf{u}$, must be determined. If we define $\mathbf{u}$ as

$$\mathbf{u} = \mathbf{z} + \sigma\mathbf{e} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} + \begin{bmatrix} \sigma \\ 0 \\ \vdots \\ 0_n \end{bmatrix} \tag{5.5.12}$$

where

$$\mathbf{e} = [1 \ 0 \ 0 \cdots 0]^T \ \text{ is an } n \text{ vector}$$

$$\sigma = \text{sign } (z_1)(\mathbf{z}^T\mathbf{z})^{\frac{1}{2}} \tag{5.5.13}$$

then the orthogonal transformation $T = I - \beta\mathbf{u}\mathbf{u}^T$ is an elementary reflector and

$$T\mathbf{z} = -\sigma\mathbf{e}$$

where

$$\beta = \frac{2}{\mathbf{u}^T\mathbf{u}}. \tag{5.5.14}$$

Proof: Writing the expression for $T\mathbf{z}$ yields

$$\begin{aligned} T\mathbf{z} &= \mathbf{z} - \beta\mathbf{u}\,\mathbf{u}^T\mathbf{z} \\ &= \mathbf{z} - \beta(\mathbf{u}^T\mathbf{z})\mathbf{u} \\ &= \mathbf{z} - \beta(\mathbf{z}^T\mathbf{z} + \sigma\mathbf{e}^T\mathbf{z})\mathbf{u} \\ &= \mathbf{z} - \beta(\sigma^2 + \sigma\mathbf{e}^T\mathbf{z})\mathbf{u}. \end{aligned} \tag{5.5.15}$$

However,

$$\begin{aligned} \beta &= \frac{2}{\mathbf{u}^T\mathbf{u}} = 2/(\mathbf{z}^T\mathbf{z} + 2\sigma\mathbf{e}^T\mathbf{z} + \sigma^2) \\ &= 1/(\sigma^2 + \sigma\mathbf{e}^T\mathbf{z}) \end{aligned} \tag{5.5.16}$$

or

$$\beta = \frac{1}{\sigma u_1} \tag{5.5.17}$$

where u_1 is the first element of $\mathbf{u}$.

Substituting Eq. (5.5.16) into (5.5.15) yields

$$\begin{aligned} T\mathbf{z} &= \mathbf{z} - \mathbf{u} \\ &= -\sigma \mathbf{e}. \end{aligned} \tag{5.5.18}$$

From Eq. (5.5.14) and (5.5.17) note that

$$\mathbf{u}^T \mathbf{u} = 2\sigma u_1. \tag{5.5.19}$$

Also, from Eq. (5.5.9) we may write

$$\gamma = \beta \mathbf{z}^T \mathbf{u}$$

or

$$\gamma = \beta \mathbf{u}^T \mathbf{z}. \tag{5.5.20}$$

For each application of T to a given matrix, we zero out all elements below the main diagonal of the first column. In this operation all elements will be multiplied by β. Hence, we want to choose the sign of σ to maximize $\|u_1\|$. Because $u_1 = z_1 + \sigma$, we choose the sign of σ to be the same as that of z_1, the first element of $\mathbf{z}$.

In summary, the Householder equations to achieve the desired transformation are

$$\begin{aligned} &\sigma = \text{sign } (z_1)(\mathbf{z}^T \mathbf{z})^{1/2} \\ &z_i' = -\sigma \\ &u_1 = z_1 + \sigma \\ &u_i = z_i, \quad z_i' = 0, \quad i = 2, \ldots \text{number of rows} \\ &\beta = 1/(\sigma u_1). \end{aligned} \tag{5.5.21}$$

The remaining columns also must have T applied. From Eq. (5.5.20)

$$\gamma_j = \beta \, \mathbf{u}^T \mathbf{z}_j, \quad j = 2, \ldots \text{number of columns} \tag{5.5.22}$$

where γ_j is a scalar and $\mathbf{z}_j$ is the j^{th} column of the matrix being transformed. The transformed columns, $\mathbf{z}_j'$, are computed using Eq. (5.5.9):

$$\mathbf{z}_j' = \mathbf{z}_j - \gamma_j \mathbf{u}. \tag{5.5.23}$$

A more detailed algorithm is given in Section 5.5.2.

5.5.1 Application to the Solution of the Least Squares Problem

Our goal is to upper triangularize an $(n + m) \times (n + 1)$ matrix using the Householder transformation:

$$T_n T_{n-1} \dots T_1 \overbrace{\begin{bmatrix} \overline{R} & \overline{\mathbf{b}} \\ H & \mathbf{y} \end{bmatrix}}^{\begin{matrix} n & 1 \end{matrix}} \begin{matrix} \} n \\ \} m \end{matrix} = \overbrace{\begin{bmatrix} R & \mathbf{b} \\ 0 & \mathbf{e} \end{bmatrix}}^{\begin{matrix} n & 1 \end{matrix}} \begin{matrix} \} n \\ \} m. \end{matrix} \tag{5.5.24}$$

The procedure for accomplishing this is as follows. Let

$$A \equiv \begin{bmatrix} \overline{R} & \overline{\mathbf{b}} \\ H & \mathbf{y} \end{bmatrix}. \tag{5.5.25}$$

The first transformation $T_1 A$ will zero all elements of the first column except for the first element, which will have the opposite sign of $\overline{R}_{11}$ and magnitude equal to the Euclidian norm of the first column,

$$T_1 A = \begin{matrix} \\ 1\{ \\ m+n-1\{ \end{matrix} \overbrace{\quad\quad}^{1} \overbrace{\quad\quad\quad}^{n} \begin{matrix} \\ \}1 \\ \}m+n-1. \end{matrix} \begin{bmatrix} -\sigma_1 & \tilde{a}_{11} & \tilde{a}_{12} & \cdots & \tilde{a}_{1n} \\ \hline 0 & & & & \\ 0 & & \widetilde{A}_1 & & \\ \vdots & & & & \\ 0 & & & & \end{bmatrix} \tag{5.5.26}$$

The next transformation operates on $\widetilde{A}_1$ and does not change the first row or column.

$$T_2 \widetilde{A}_1 = \begin{matrix} \\ 1\{ \\ m+n-2\{ \end{matrix} \overbrace{\quad\quad}^{1} \overbrace{\quad\quad\quad}^{n-1} \begin{matrix} \\ \}1 \\ \}m+n-2 \end{matrix} \begin{bmatrix} -\sigma_2 & \tilde{a}_{21} & \tilde{a}_{22} & \cdots & \tilde{a}_{2,n-1} \\ \hline 0 & & & & \\ 0 & & \widetilde{A}_2 & & \\ \vdots & & & & \\ 0 & & & & \end{bmatrix} \tag{5.5.27}$$

This procedure is continued, zeroing elements below the diagonal until the upper $n \times n$ portion of A is upper triangular.

5.5.2 HOUSEHOLDER COMPUTATIONAL ALGORITHM

The computational algorithm for applying the Householder transformation is given by Bierman (1977). A few steps have been added here to aid in implementing the algorithm.

Given:

$$\begin{matrix} \overbrace{}^{n} & \overbrace{}^{1} & \\ \left[\begin{matrix} \overline{R} & \overline{\mathbf{b}} \\ H & \mathbf{y} \end{matrix}\right] & \begin{matrix} \} n \\ \} m \end{matrix} & \equiv [A]_{(m+n)\times(n+1)} \end{matrix} \tag{5.5.28}$$

Do $k = 1,\ n$

$$\begin{aligned} \sigma &= \text{sign}(A_{kk}) \left(\sum_{i=k}^{m+n} [A_{ik}]^2 \right)^{1/2} \\ u_k &= A_{kk} + \sigma \\ A_{kk} &= -\sigma \\ u_i &= A_{ik} \quad i = k+1, \ldots, m+n \\ \beta &= \frac{1}{\sigma u_k} \end{aligned} \tag{5.5.29}$$

Do $j = k+1, \ldots, n+1$

$$\begin{aligned} \gamma &= \beta \sum_{i=k}^{m+n} u_i A_{ij} \\ A_{ij} &= A_{ij} - \gamma u_i \quad i = k, \ldots, m+n \end{aligned}$$

Next j

$$A_{ik} = 0 \quad i = k+1, \ldots, m+n$$

Next k

$$\text{Sum} = \sum_{i=1}^{m} [A_{n+i,n+1}]^2$$

Upon completion, the elements of A represent R, b, and e, respectively:

$$A = \begin{matrix} \overbrace{}^{n} & \overbrace{}^{1} \\ \left[\begin{array}{cc} R & \mathbf{b} \\ 0 & \mathbf{e} \end{array} \right] & \begin{matrix} \}n \\ \}m \end{matrix} \end{matrix} \tag{5.5.30}$$

and

$$\text{Sum} = \sum_{i=1}^{m} e_i^2.$$

The algorithm can be initialized with *a priori* $\overline{R}$ and $\overline{\mathbf{b}}$. If no *a priori* is available it can be initialized with $\overline{R} = 0$, $\overline{\mathbf{b}} = 0$. Alternatively we may define A as

$$A_{m\times(n+1)} = m\{ \begin{matrix} \overbrace{}^{n} & \overbrace{}^{1} \\ \left[\begin{array}{cc} H & \vdots \; \mathbf{y} \end{array} \right] \end{matrix} \tag{5.5.31}$$

and the operations with index i range to m instead of $m+n$. In this case the final result is

$$A = \begin{matrix} \overbrace{}^{n} & \overbrace{}^{1} \\ \left[\begin{array}{cc} R & \mathbf{b} \\ 0 & \mathbf{e} \end{array} \right] & \begin{matrix} \}n \\ \}m-n \end{matrix} \end{matrix} . \tag{5.5.32}$$

Recall from the discussion on Givens transformation in Section 5.4.1 that

$$\sum_{i=1}^{m} e_i^2 = \|\overline{R}(\hat{\mathbf{x}} - \overline{\mathbf{x}})\|^2 + \sum_{i=1}^{m} \hat{\epsilon}_i^2 \tag{5.5.33}$$

where

$$\hat{\epsilon}_i = y_i - H_i\hat{\mathbf{x}}. \tag{5.5.34}$$

If the initialization procedure of Eq. (5.5.31) is used, Eq. (5.5.33) becomes

$$\sum_{i=1}^{m-n} e_i^2 = \sum_{i=1}^{m} \hat{\epsilon}_i^2. \tag{5.5.35}$$

As with the Givens transformation, the value of $\hat{\mathbf{x}}$ is obtained by a backward substitution using Eq. (5.2.8); that is, $R\hat{\mathbf{x}} = \mathbf{b}$ and

$$\hat{x}_i = \left(b_i - \sum_{j=i+1}^{n} R_{ij}\hat{x}_j \right) \Big/ R_{ii} \quad i = n, \ldots, 1 \tag{5.5.36}$$

and the estimation error covariance is obtained from

$$\begin{aligned} P &= \Lambda^{-1} = (R^T R)^{-1} \\ &= R^{-1} R^{-T} \\ &= S S^T \end{aligned} \tag{5.5.37}$$

where S is obtained from Eq. (5.2.9):

$$\begin{aligned} &i = 1, \ldots, n \\ &S_{ii} = \frac{1}{R_{ii}} \\ &S_{ij} = -S_{jj} \left[\sum_{k=i}^{j-1} R_{kj} S_{ik} \right]; \quad j = i+1, \ldots, n . \end{aligned} \tag{5.5.38}$$

5.6 NUMERICAL EXAMPLES

First consider a case with no *a priori* information. Assume we are given the observation-state matrix, H, and the observation vector, $\mathbf{y}$, as follows:

$$\mathbf{y} = H\mathbf{x} + \boldsymbol{\epsilon}$$

where

$$\mathbf{y} = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix}, H = \begin{bmatrix} 1 & -2 \\ 2 & -1 \\ 1 & 1 \end{bmatrix}, \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{5.6.1}$$

Assume that the prewhitening transformation described in Section 5.7.1 has been applied so that $\boldsymbol{\epsilon} \sim (O, I)$; that is, $W = I$. Use the Cholesky decomposition algorithm and the Givens and Householder transformations to solve for $\hat{\mathbf{x}}$.

5.6.1 CHOLESKY DECOMPOSITION

The basic equations are

$$H^T H \mathbf{x} = H^T \mathbf{y}, \qquad M\mathbf{x} = \mathbf{N}$$

$$M = H^T H = \begin{bmatrix} 1 & 2 & 1 \\ -2 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & -2 \\ 2 & -1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 6 & -3 \\ -3 & 6 \end{bmatrix} \tag{5.6.2}$$

$$\mathbf{N} = H^T\mathbf{y} = \begin{bmatrix} 1 & 2 & 1 \\ -2 & -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}. \tag{5.6.3}$$

Find R such that

$$R^T R = M, \text{ where } R \text{ is upper triangular (UT).}$$

Using the Cholesky algorithm given by Eq. (5.2.6),

$$i = 1, \ldots, n, j = i+1, \ldots, n$$

$$\begin{aligned} r_{ii} &= \left(M_{ii} - \sum_{k=1}^{i-1} r_{ki}^2 \right)^{1/2} \\ r_{ij} &= \left(M_{ij} - \sum_{k=1}^{i-1} r_{ki}\, r_{kj} \right) / r_{ii}, \end{aligned} \tag{5.6.4}$$

the following results are obtained:

$$i = 1, 2, \text{and } j = 2 \text{ (for this example)}$$

$$r_{11} = \sqrt{6}, r_{12} = -3/\sqrt{6}, r_{22} = 3/\sqrt{2}$$

$$R = \begin{bmatrix} \sqrt{6} & -3/\sqrt{6} \\ 0 & 3/\sqrt{2} \end{bmatrix}$$

$$R^T\mathbf{z} = H^T\mathbf{y}$$

$$\begin{bmatrix} \sqrt{6} & 0 \\ -3/\sqrt{6} & 3/\sqrt{2} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{z} = \begin{bmatrix} 3/\sqrt{6} \\ 3/\sqrt{2} \end{bmatrix}$$

$$R\hat{\mathbf{x}} = \mathbf{z}$$

$$\begin{bmatrix} \sqrt{6} & -3/\sqrt{6} \\ 0 & 3/\sqrt{2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{6} \\ 3/\sqrt{2} \end{bmatrix}$$

$$\hat{\mathbf{x}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

If this were a more complex problem, we would use Eqs. (5.2.7) and (5.2.8) to compute $\mathbf{z}$ and $\hat{\mathbf{x}}$, respectively.

5.6.2 GIVENS TRANSFORMATION

For this example,

$$G[H \vdots \mathbf{y}] = \begin{bmatrix} R \vdots \mathbf{b} \\ \phi \vdots e \end{bmatrix}, \quad \begin{array}{l} \text{where } R \equiv 2 \times 2 \text{ UT}, \\ \phi \equiv 1 \times 2 \text{ null matrix} \\ \text{it follows then that } b \equiv 2 \times 1 \\ \text{and } e \equiv 1 \times 1. \end{array} \tag{5.6.5}$$

G is computed directly here using Eq. (5.4.8).

Operating on rows 1 and 2 ($i = 1, k = 2$) yields

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -2 \vdots & -1 \\ 2 & -1 \vdots & 1 \\ 1 & 1 \vdots & 2 \end{bmatrix}$$

$$h'_{ii} = \sqrt{h_{ii}^2 + h_{ki}^2}, \qquad h'_{11} = \sqrt{1 + 4} = \sqrt{5}$$

$$\sin\theta = S = h_{ki} / h'_{ii} = 2 / \sqrt{5}$$
$$\cos\theta = C = h_{ii} / h'_{ii} = 1 / \sqrt{5}$$

$$\begin{bmatrix} 1/\sqrt{5} & 2/\sqrt{5} & 0 \\ -2/\sqrt{5} & 1/\sqrt{5} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -2 & -1 \\ 2 & -1 & 1 \\ 1 & 1 & 2 \end{bmatrix} =$$
$$\begin{bmatrix} \sqrt{5} & -4/\sqrt{5} & 1/\sqrt{5} \\ 0 & 3/\sqrt{5} & 3/\sqrt{5} \\ 1 & 1 & 2 \end{bmatrix}.$$

Operating on rows 1 and 3 ($i = 1, k = 3$) yields

$$h'_{11} = \sqrt{6}$$

$$S = h_{31} / h'_{11} = 1 / \sqrt{6}, C = h_{11} / h'_{11} = \sqrt{5} / \sqrt{6}$$

$$\begin{bmatrix} \sqrt{5}/\sqrt{6} & 0 & 1/\sqrt{6} \\ 0 & 1 & 0 \\ -1/\sqrt{6} & 0 & \sqrt{5}/\sqrt{6} \end{bmatrix} \begin{bmatrix} \sqrt{5} & -4/\sqrt{5} & 1/\sqrt{5} \\ 0 & 3/\sqrt{5} & 3/\sqrt{5} \\ 1 & 1 & 2 \end{bmatrix}$$

$$= \begin{bmatrix} \sqrt{6} & -3/\sqrt{6} & 3/\sqrt{6} \\ 0 & 3/\sqrt{5} & 3/\sqrt{5} \\ 0 & 9/\sqrt{30} & 9/\sqrt{30} \end{bmatrix}.$$

The final operation is on rows 2 and 3 ($i = 2$, $k = 3$), which results in

$$h'_{22} = \sqrt{9/5 + 81/30} = \sqrt{135/30}$$
$$S = h_{32}/h'_{22} = 9/\sqrt{135}$$
$$C = h_{22}/h'_{22} = 3\sqrt{6}/\sqrt{135}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3\sqrt{6}/\sqrt{135} & 9/\sqrt{135} \\ 0 & -9/\sqrt{135} & 3\sqrt{6}/\sqrt{135} \end{bmatrix} \begin{bmatrix} \sqrt{6} & -3/\sqrt{6} & 3/\sqrt{6} \\ 0 & 3/\sqrt{5} & 3/\sqrt{5} \\ 0 & 9/\sqrt{30} & 9/\sqrt{30} \end{bmatrix}$$

$$= \left[\begin{array}{ccc:c} \sqrt{6} & -3/\sqrt{6} & & 3/\sqrt{6} \\ 0 & 3/\sqrt{2} & & 3/\sqrt{2} \\ 0 & 0 & & 0 \end{array}\right].$$

We now have the necessary information to solve for $\hat{\mathbf{x}}$. Using

$$R\hat{\mathbf{x}} = \mathbf{b}$$

yields

$$\begin{bmatrix} \sqrt{6} & -3/\sqrt{6} \\ 0 & 3/\sqrt{2} \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{6} \\ 3/\sqrt{2} \end{bmatrix}.$$

Solving for $\hat{\mathbf{x}}$ by a backward recursion yields

$$\hat{\mathbf{x}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

The associated covariance matrix is

$$P = (H^T H)^{-1} = (H^T G^T G H)^{-1} = R^{-1} R^{-T}$$

$$P = \begin{bmatrix} 2/9 & 1/9 \\ 1/9 & 2/9 \end{bmatrix}.$$

Note that in this example the observation residual, e, has a value of zero. This is because the three observations were perfect, from Eq. (5.5.34):

$$\hat{\epsilon} = \mathbf{y} - H\hat{\mathbf{x}} = 0.$$

5.6.3 HOUSEHOLDER TRANSFORMATION

This algorithm is given by Eq. (5.5.29). Because there is no *a priori* information, we use the initialization procedure of Eq. (5.5.31).

For this example,

$$[H \vdots \mathbf{y}] = A = \begin{bmatrix} 1 & -2 & -1 \\ 2 & -1 & 1 \\ 1 & 1 & 2 \end{bmatrix}.$$

Each transformation, T, results in

$$TA = \begin{bmatrix} -\sigma & \cdots \\ \vdots & \widetilde{A} \\ 0 & \end{bmatrix}.$$

Note that the algorithm given by Eq. (5.5.29) does not use the array, $\widetilde{A}$. It simply redefines the elements of A, thereby requiring less computer storage. Beginning with

$$k = 1$$

$$\begin{aligned} \sigma &= +\sqrt{1 + 2^2 + 1} = \sqrt{6} \\ A_{11} &= -\sigma = -\sqrt{6} \\ u_1 &= A_{11} + \sigma = 1 + \sqrt{6} = 3.4495 \\ u_2 &= A_{21} = 2 \\ u_3 &= A_{31} = 1 \\ \beta &= \frac{1}{\sigma u_1} = \frac{1}{\sqrt{6} + 6} = 0.11835 \end{aligned}$$

$$j = 2$$

$$\gamma = \beta \sum_{i=1}^{3} u_i A_{ij}$$
$$= 0.11835\,[3.4495\,(-2) \;+\; 2\,(-1) \;+\; 1]$$
$$= -0.9348$$

$$A_{12} = A_{12} - \gamma\, u_1$$
$$= -2 + 0.9348\,(3.4495)$$
$$= 1.2247$$
$$A_{22} = A_{22} - \gamma\, u_2 = -1 + 0.9348\,(2)$$
$$= 0.8696$$
$$A_{32} = 1 + 0.9348 = 1.9348$$

$$j = 3$$

$$\gamma = 0.11835\,[3.4495(-1) + 2 + 2]$$
$$= 0.06515$$

$$A_{13} = -1 - .06515\,(3.4495)$$
$$= -1.2247$$

$$A_{23} = 1 - 0.06515\,(2)$$
$$= 0.8697$$

$$A_{33} = A_{33} + \gamma\, u_3$$
$$= 2 - 0.06515$$
$$= 1.9348.$$

Hence,

$$\begin{bmatrix} \sigma \cdots \\ \vdots\ \tilde{A} \\ 0 \end{bmatrix} = \left[\begin{array}{c:cc} -2.4495 & 1.2247 & -1.2247 \\ \cdots\cdots & \vdots\ \cdots\cdots & \cdots\cdots \\ 0 & \vdots\ 0.8697 & 0.8697 \\ 0 & \vdots\ 1.9348 & 1.9348 \end{array}\right] \qquad (5.6.6)$$

$$k = 2$$

$$\begin{aligned}\sigma &= \sqrt{(0.8697)^2 + (1.9348)^2} \\ &= 2.1212 \\ \widetilde{A}_{11} &= -2.1212 \\ u_2 &= A_{22} + \sigma = 0.8697 + 2.1212 \\ &= 2.991 \\ u_3 &= A_{32} = 1.9348\end{aligned}$$

$$\beta = \frac{1}{\sigma\, u_2} = \frac{1}{(2.1212)(2.991)} = 0.1576$$

$$j = 3$$

$$\begin{aligned}\gamma &= \beta[u_2 u_3] \begin{bmatrix} A_{22} \\ A_{32} \end{bmatrix} \\ &= 0.1576\ [2.991\ \ 1.9348] \begin{bmatrix} 0.8697 \\ 1.9348 \end{bmatrix}\end{aligned}$$

$$\gamma = 1$$

$$\begin{aligned}\widetilde{A}_{ij} &= A_{ij} - \gamma\, u_i, \quad j = 3, \quad i = 2, 3 \\ \widetilde{A} &= \begin{bmatrix} 0.8697 \\ 1.9348 \end{bmatrix} - \begin{bmatrix} 2.991 \\ 1.9348 \end{bmatrix} = \begin{bmatrix} -2.1212 \\ 0 \end{bmatrix}\end{aligned} \tag{5.6.7}$$

so that the final result is

$$TA = \begin{bmatrix} -2.4495 & 1.2247 & -1.2247 \\ 0 & -2.1212 & -2.1212 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.6.8}$$

which agrees with the Givens result except for the sign, which has no effect on $\hat{\mathbf{x}}$ or P. Note that, as in the case for the Givens transformation, the observation residual is zero, implying that the observations were perfect.

5.6.4 A More Illustrative Example

For this example we will include *a priori* information and observation errors. This will illustrate how to initialize the least squares procedure with *a priori* information using the Cholesky decomposition and orthogonal transformations and

will also demonstrate the feature of computing the sum of squares of the observation residuals.

Assume that we are given observations

$$\mathbf{y} = H\mathbf{x} + \boldsymbol{\epsilon}$$

and that we have prewhitened the observation errors so that $\epsilon \sim (O, I)$. Values for $\mathbf{y}$ and H are

$$\mathbf{y} = \begin{bmatrix} -1.1 \\ 1.2 \\ 1.8 \end{bmatrix}, \quad H = \begin{bmatrix} 1 & -2 \\ 2 & -1 \\ 1 & 1 \end{bmatrix}, \tag{5.6.9}$$

with assumed *a priori* information

$$\overline{P} = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}, \overline{\mathbf{x}} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}. \tag{5.6.10}$$

Our objective is to determine the best estimate, $\hat{\mathbf{x}}$, of the constant vector $\mathbf{x}$ using the Cholesky decomposition, the Givens square root free, and Householder algorithms. We also wish to find the estimation error covariance matrix and the sum of squares of the observation residuals.

5.6.5 Cholesky Decomposition

If the observations are given unit weight, for example, $W = I$, the normal equations are

$$(H^T H + \overline{P}^{-1})\mathbf{x} = H^T \mathbf{y} + \overline{P}^{-1}\overline{\mathbf{x}} \tag{5.6.11}$$

or

$$M\mathbf{x} = \mathbf{N} \tag{5.6.12}$$

where

$$M = H^T H + \overline{P}^{-1} \tag{5.6.13}$$

$$\mathbf{N} = H^T \mathbf{y} + \overline{P}^{-1}\overline{\mathbf{x}}. \tag{5.6.14}$$

Substituting H, $\overline{P}$, $\mathbf{y}$, and $\overline{\mathbf{x}}$ into Eqs. (5.6.13) and (5.6.14) yields

$$M = \begin{bmatrix} 6.01 & -3 \\ -3 & 6.01 \end{bmatrix}, \quad \mathbf{N} = \begin{bmatrix} 3.12 \\ 2.82 \end{bmatrix}. \tag{5.6.15}$$

Using the Cholesky algorithm given by Eq. (5.2.6) to compute R, where $M = R^T R$, yields

$$R = \begin{bmatrix} 2.452 & -1.224 \\ 0 & 2.124 \end{bmatrix}. \tag{5.6.16}$$

From Eq. (5.2.4)

$$R^T \mathbf{z} = \mathbf{N},$$

from Eq. (5.2.7)

$$\mathbf{z} = \begin{bmatrix} 1.273 \\ 2.0609 \end{bmatrix}, \tag{5.6.17}$$

and from Eq. (5.2.8)

$$\hat{\mathbf{x}} = \begin{bmatrix} 1.003 \\ .970 \end{bmatrix}. \tag{5.6.18}$$

The covariance of the estimation error is given by Eq. (5.2.5),

$$P = E(\hat{\mathbf{x}} - \mathbf{x})(\hat{\mathbf{x}} - \mathbf{x})^T = R^{-1}R^{-T} = S\,S^T. \tag{5.6.19}$$

S is given by Eq. (5.2.9)

$$S = \begin{bmatrix} .408 & .235 \\ 0 & .471 \end{bmatrix}.$$

Hence,

$$P = \begin{bmatrix} .222 & .111 \\ .111 & .222 \end{bmatrix}. \tag{5.6.20}$$

The sum of squares of the observation errors, including errors in the *a priori* information, is given by

$$\begin{aligned} e^2 &= (\hat{\mathbf{x}} - \overline{\mathbf{x}})^T \overline{P}^{-1} (\hat{\mathbf{x}} - \overline{\mathbf{x}}) + \sum_{i=1}^{3} (y_i - H_i \hat{\mathbf{x}})^2 \\ &= .0205 + .0266 + .0269 + .0299 \\ e^2 &= .1039. \end{aligned} \tag{5.6.21}$$

5.6.6 Square Root Free Givens Transformation

The initialization procedure for the orthogonal transformation algorithms is described in Section 5.4.1. This involves computing the square root of the *a priori*

information matrix, $\overline{R}$, and $\overline{\mathbf{b}} = \overline{R}\,\overline{\mathbf{x}}$. Recall that $\overline{P} = \overline{S}\,\overline{S}^T$ and $\overline{R} = \overline{S}^{-1}$. For this example

$$\overline{R} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad \overline{\mathbf{b}} = \begin{bmatrix} 0.2 \\ 0.2 \end{bmatrix}. \tag{5.6.22}$$

We will use the square root free Givens rotation given by Eq. (5.4.70). For this we need $\overline{D}$ and $\overline{U}$, where $\overline{R} = \overline{D}^{\frac{1}{2}}\,\overline{U}$. Hence,

$$\overline{D} = \begin{bmatrix} .01 & 0 \\ 0 & .01 \end{bmatrix}, \quad \overline{U} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{5.6.23}$$

The *a priori* value of $\widetilde{\mathbf{b}}$ is given by

$$\begin{aligned} \overline{\widetilde{\mathbf{b}}} &= \overline{D}^{-\frac{1}{2}}\overline{\mathbf{b}} = \overline{D}^{-\frac{1}{2}}\overline{R}\,\overline{\mathbf{x}} \\ &= \overline{U}\,\overline{\mathbf{x}}, \end{aligned} \tag{5.6.24}$$

so

$$\overline{\widetilde{\mathbf{b}}} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}.$$

The matrix we wish to compress is

$$\begin{bmatrix} \overline{U} & \overline{\widetilde{\mathbf{b}}} \\ H & \mathbf{y} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 2 \\ 1 & -2 & -1.1 \\ 2 & -1 & 1.2 \\ 1 & 1 & 1.8 \end{bmatrix}. \tag{5.6.25}$$

Recall that Givens algorithm as defined by Eq. (5.4.70) operates on the matrix one row at a time. After processing the first observation we have

$$D = \begin{bmatrix} 1.0100 & 0 \\ 0 & 0.0496 \end{bmatrix}, \quad U = \begin{bmatrix} 1.00 & -1.9802 \\ 0 & 1.00 \end{bmatrix}$$

$$\widetilde{\mathbf{b}} = \begin{bmatrix} -1.0693 \\ 1.6407 \end{bmatrix}, \quad e = 0.0402.$$

Processing the second observation results in

$$D = \begin{bmatrix} 5.0100 & 0 \\ 0 & 1.8164 \end{bmatrix}, \quad U = \begin{bmatrix} 1.0 & -0.7984 \\ 0 & 1.0 \end{bmatrix}$$

$$\tilde{\mathbf{b}} = \begin{bmatrix} 0.2635 \\ 1.1418 \end{bmatrix}, \quad e = -0.1127.$$

After processing the third observation,

$$D = \begin{bmatrix} 6.010 & 0 \\ 0 & 4.5125 \end{bmatrix}, U = \begin{bmatrix} 1.0 & -0.4992 \\ 0 & 1.0 \end{bmatrix}$$
$$\tilde{\mathbf{b}} = \begin{bmatrix} 0.5191 \\ 0.9701 \end{bmatrix}, \quad e = -0.2994. \tag{5.6.26}$$

To compare with the Householder results we must multiply by $D^{1/2}$:

$$\begin{bmatrix} R & \mathbf{b} \\ 0 & \mathbf{e} \end{bmatrix} = \begin{bmatrix} D^{\frac{1}{2}}U & D^{\frac{1}{2}}\tilde{\mathbf{b}} \\ 0 & \mathbf{e} \end{bmatrix} \begin{matrix} \} 2 \\ \} 3 \end{matrix} \quad \overbrace{}^{2} \quad \overbrace{}^{1}$$
$$= \begin{bmatrix} 2.4515 & -1.2237 & 1.2727 \\ 0 & 2.1243 & 2.0607 \\ 0 & 0 & 0.0402 \\ 0 & 0 & -0.1127 \\ 0 & 0 & -0.2994 \end{bmatrix}. \tag{5.6.27}$$

In actual practice, to conserve computer storage, we would read and process the observations one at a time and store the observations residuals as

$$e^2 = \sum_{i=1}^{m} e_i^2.$$

$\hat{\mathbf{x}}$ is computed using Eq. (5.4.49),

$$U\hat{\mathbf{x}} = \tilde{\mathbf{b}}$$

or

$$\begin{bmatrix} 1.00 & -0.4992 \\ 0 & 1.00 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = \begin{bmatrix} 0.5191 \\ 0.9701 \end{bmatrix}$$

and

$$\begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = \begin{bmatrix} 1.0034 \\ 0.9701 \end{bmatrix}. \tag{5.6.28}$$

The sum of squares of observation residuals is given by

$$\begin{aligned} e^2 &= \sum_{i=1}^{3} e_i^2 = (0.0402)^2 + (-0.1127)^2 + (-0.2994)^2 \\ &= 0.1039. \end{aligned}$$

The covariance of the estimation error is given by Eq. (5.4.80), $P = U^{-1} D^{-1} U^{-T}$. Equation (5.2.20) may be used to compute U^{-1},

$$U^{-1} = \begin{bmatrix} 1.0 & 0.4992 \\ 0 & 1.0 \end{bmatrix}$$

and

$$\begin{aligned} P &= \begin{bmatrix} 1.0 & 0.4992 \\ 0 & 1.0 \end{bmatrix} \begin{bmatrix} .1664 & 0 \\ 0 & .2216 \end{bmatrix} \begin{bmatrix} 1.0 & 0 \\ 0.4992 & 1.0 \end{bmatrix} \\ &= \begin{bmatrix} .2216 & .1106 \\ .1106 & .2216 \end{bmatrix}. \end{aligned}$$

Notice that the results for $\hat{\mathbf{x}}$, P, and e^2 agree with the Cholesky results.

5.6.7 The Householder Transformation

The matrix we wish to transform is given by Eq. (5.6.25). In terms of $\overline{R}$ and $\overline{\mathbf{b}}$ the matrix is given by

$$\begin{bmatrix} \overline{R} & \overline{\mathbf{b}} \\ H & \mathbf{y} \end{bmatrix} = \begin{bmatrix} 0.1 & 0 & 0.2 \\ 0 & 0.1 & 0.2 \\ 1 & -2 & -1.1 \\ 2 & -1 & 1.2 \\ 1 & 1 & 1.8 \end{bmatrix}. \tag{5.6.29}$$

The Householder transformation algorithm given by Eq. (5.5.29) nulls the elements of each column below the main diagonal. The first transformation yields

$$\begin{bmatrix} -2.4515 & 1.2237 & -1.2727 \\ 0 & 0.1 & 0.2 \\ 0 & -1.5204 & -1.6772 \\ 0 & -0.0408 & 0.0457 \\ 0 & 1.4796 & 1.2228 \end{bmatrix}.$$

The second transformation results in

$$\begin{bmatrix} -2.4515 & 1.2237 & -1.2727 \\ 0 & -2.1243 & -2.0607 \\ 0 & 0 & -0.1319 \\ 0 & 0 & 0.0871 \\ 0 & 0 & -0.2810 \end{bmatrix}. \tag{5.6.30}$$

Several points should be noted:

1. The Householder values of R and b are identical to the Givens results. Hence, the solution for $\hat{\mathbf{x}}$ and P will be identical.

2. Although the individual values of e_i differ, both algorithms yield identical values of $e^2 = 0.1039$. This also agrees with the Cholesky result.

3. The Euclidean norm of each column is preserved by an orthogonal transformation.

4. The square root free Givens algorithm as derived here operates on the matrix row by row, whereas the Householder algorithm transforms column by column. The Givens algorithm can be modified to operate column by column (see Section 5.4.2).

5. The orthogonal transformations do not require the formation of $H^T H$. Hence, they will generally be more accurate than Cholesky decomposition.

5.7 SQUARE ROOT FILTER ALGORITHMS

Although the sequential estimation algorithms have had wide use in autonomous navigation and control applications, there has been a reluctance to adopt the sequential estimation algorithm for real-time orbit determination mission support.

The primary reason for this reluctance is due to the phenomenon of filter divergence, during which the estimate of the state can depart in an unbounded manner from the true value of the state. There are two fundamental reasons for filter divergence. The first of these is due to inaccuracies in the mathematical model used to describe the dynamic process or in the model used to relate the observations to the state.

A second factor that can cause filter divergence is associated with the errors that occur in the measurement update of the state error covariance matrix. In particular, this matrix can become nonpositive definite, a situation that is a theoretical impossibility due to the effects of using finite word length arithmetic to compute the update of the state error covariance matrix at the point where an observation is incorporated. Since this type of divergence is related to errors introduced during the computational procedure, it should be possible to reformulate the computational process to minimize the effects of such errors.

To accomplish this objective, several filter modifications, referred to as square root covariance filters, have been proposed in which the state error covariance matrix is replaced by its square root. The state error covariance matrix is obtained by multiplying the square root matrix by its transpose and will always be symmetric and positive semidefinite. Note that the algorithms presented in the remainder of this chapter are designed to accommodate scalar observations.

The motivation for considering the square root measurement update algorithms stems from the loss of significant digits that occurs in computing the measurement update of the state error covariance matrix at the observation epoch (Kaminski *et al.*, 1971) . When the eigenvalues have a wide spread, the error introduced in the computational process can destroy the symmetry and positive definite character of the covariance matrix and filter divergence may occur. The square root measurement update philosophy, which has been proposed to alleviate this condition, can be expressed as follows.

Define W, the state error covariance matrix square root such that

$$P = WW^T. \tag{5.7.1}$$

Note that P, if computed using Eq. (5.7.1), can never be nonpositive definite even in the presence of round-off or truncation errors. Furthermore, since P is symmetric and positive definite, there will exist an orthogonal matrix M such that

$$P^* = M^T PM, \tag{5.7.2}$$

where P^* is a diagonal matrix whose elements are the eigenvalues of P and M is the corresponding matrix of eigenvectors (Graybill, 1961). Define W^* as the matrix whose diagonal elements are equal to the square root of the diagonal elements of P^*:

$$W^*_{ii} = \sqrt{P^*_{ii}}\, i = 1, \ldots, n \tag{5.7.3}$$

where $P_{ii}^* > 0$. Then note that

$$W^* W^{*T} = P^* = M^T P M = M^T W W^T M .$$

Hence, $W^* = M^T W$ and, since M is an orthogonal matrix, it follows that

$$W = M W^*. \tag{5.7.4}$$

If the matrix is symmetrical and positive definite, there are other methods of computing the square root matrix. For example, see the Cholesky decomposition discussed in Section 5.2.

The numerical conditioning of W is generally much better than that of P. The *conditioning number* $C(P)$ of P can be defined as (Lawson and Hanson, 1974)

$$C(P) = \gamma_{\max} / \gamma_{\min}, \tag{5.7.5}$$

where γ_{max} is the maximum eigenvalue of P and γ_{min} is the minimum eigenvalue. In base 10 arithmetic with p significant digits, numerical difficulties with matrix inversion and the precision of $\hat{\mathbf{x}}$ may be encountered as $C(P) \rightarrow 10^p$. However, for W,

$$C(W) = \sqrt{C(P)}.$$

Hence, numerical difficulties should not be encountered until

$$C(W) = 10^p$$

or

$$C(P) = 10^{2p}.$$

5.7.1 The Square Root Measurement Update Algorithms

Using these ideas, the covariance measurement update equation, Eq. (4.7.10), can be expressed in square root form as follows:

$$P = \overline{P} - \overline{P} H^T \left[H \overline{P} H^T + R \right]^{-1} H \overline{P}. \tag{5.7.6}$$

Now, let $P = W W^T$ and make this substitution in Eq. (5.7.6) to obtain

$$W W^T = \overline{W}\,\overline{W}^T - \overline{W}\,\overline{W}^T H^T \left[H \overline{W}\,\overline{W}^T H^T + R \right]^{-1} H \overline{W}\,\overline{W}^T. \tag{5.7.7}$$

Using the following definitions

$$\widetilde{F} = \overline{W}^T H^T, \alpha = (\widetilde{F}^T \widetilde{F} + R)^{-1}, \tag{5.7.8}$$

Eq. (5.7.7) can be expressed as

$$WW^T = \overline{W}[I - \widetilde{F}\alpha\widetilde{F}^T]\overline{W}^T. \tag{5.7.9}$$

If a matrix $\widetilde{A}$ can be found such that

$$\widetilde{A}\widetilde{A}^T = I - \widetilde{F}\alpha\widetilde{F}^T, \tag{5.7.10}$$

then Eq. (5.7.9) can be expressed as

$$WW^T = \overline{W}\widetilde{A}\widetilde{A}^T\overline{W}^T. \tag{5.7.11}$$

Hence,

$$W = \overline{W}\widetilde{A}. \tag{5.7.12}$$

The square root measurement update algorithm can be expressed as follows:

$$\begin{aligned}
\widetilde{F} &= \overline{W}^T H^T \\
\alpha &= (R + \widetilde{F}^T\widetilde{F})^{-1} \\
K &= \overline{W}\widetilde{F}\alpha \\
W &= \overline{W}\widetilde{A} \\
\hat{\mathbf{x}} &= \overline{\mathbf{x}} + K(\mathbf{y} - H\overline{\mathbf{x}}),
\end{aligned}$$

where $\widetilde{A} = [I - \widetilde{F}\alpha\widetilde{F}^T]^{1/2}$. The primary differences in the various algorithms for computing the measurement update in square root form lie in the manner in which the matrix $\widetilde{A}$ is computed. The method first used in practice is that given by Potter (Battin, 1999).

The Potter Square Root Update

If attention is restricted to processing a single scalar observation, α will be a scalar. For this application, Potter considered the problem of finding the matrix $\widetilde{A}$ such that

$$\widetilde{A}\widetilde{A}^T = [I - \alpha\widetilde{F}\widetilde{F}^T] = [I - \gamma\alpha\widetilde{F}\widetilde{F}^T][I - \gamma\alpha\widetilde{F}\widetilde{F}^T]^T, \tag{5.7.13}$$

where γ is an unspecified scalar parameter whose value is to be selected to satisfy Eq. (5.7.13). Expanding the right-hand side of Eq. (5.7.13) yields

$$I - \alpha\widetilde{F}\widetilde{F}^T = I - 2\alpha\gamma\widetilde{F}\widetilde{F}^T + \alpha^2\gamma^2(\widetilde{F}\widetilde{F}^T)(\widetilde{F}\widetilde{F}^T).$$

Canceling the identity and factoring $\widetilde{F}\widetilde{F}^T$ after noting that $\widetilde{F}^T\widetilde{F}$ is a scalar leads to

$$(1 - 2\gamma + \alpha\gamma^2\widetilde{F}^T\widetilde{F})\alpha\widetilde{F}\widetilde{F}^T = 0. \tag{5.7.14}$$

The solution, $\widetilde{F}\widetilde{F}^T = 0$, is trivial. Hence, the solution of Eq. (5.7.14) leads to the following condition:

$$1 - 2\gamma + \gamma^2 \alpha \widetilde{F}^T \widetilde{F} = 0. \tag{5.7.15}$$

From Eq. (5.7.15), it follows that γ must satisfy the following relation

$$\gamma = \frac{2 \pm \sqrt{4 - 4\alpha \widetilde{F}^T \widetilde{F}}}{2\alpha \widetilde{F}^T \widetilde{F}}.$$

After some algebra, this can be simplified to

$$\gamma = \frac{1}{1 \pm \sqrt{R\alpha}}, \tag{5.7.16}$$

where the $+$ sign is chosen to ensure that a singular value, $\gamma = \infty$, does not occur.

Using Eq. (5.7.16) the computational algorithm, based on the sequential estimation algorithm discussed in Chapter 4, can be expressed as follows:

Given $\overline{W}$, H, R, $\overline{\mathbf{x}}$ and $\mathbf{y}$

Compute:

$$\begin{aligned}
1) \quad \widetilde{F} &= \overline{W}^T H^T \\
2) \quad \alpha &= (\widetilde{F}^T \widetilde{F} + R)^{-1} \\
3) \quad \gamma &= 1/(1 + \sqrt{R\alpha}) \\
4) \quad K &= \alpha \overline{W} \widetilde{F} \\
5) \quad \hat{\mathbf{x}} &= \overline{\mathbf{x}} + K(\mathbf{y} - H\overline{\mathbf{x}}) \\
6) \quad W &= \overline{W} - \gamma K \widetilde{F}^T.
\end{aligned} \tag{5.7.17}$$

Note that even if $\overline{W}$ is triangular, the computation involved in Eq. (5.7.17) will result in a nontriangular form for W. Also the computation of Eq. (5.7.17) involves two divisions and a square root that are not involved in the conventional Kalman algorithm. Consequently, Eq. (5.7.17) will be slower than the conventional algorithm; however, it will be more accurate. If $\overline{P}$ is given instead of $\overline{W}$, a Cholesky decomposition may be used to compute $\overline{W}$ in order to initialize the algorithm.

The time update for the state error covariance square root at time t_k for the Potter algorithm follows directly from the time update of the state error covariance. In the general case where there is process noise

$$\begin{aligned}\overline{P}_k &= \Phi(t_k, t_{k-1}) W_{k-1} W_{k-1}^T \Phi^T(t_k, t_{k-1}) \\ &\quad + \Gamma(t_k, t_{k-1}) V_k V_k^T \Gamma^T(t_k, t_{k-1}) \\ &= \overline{W}_k \overline{W}_k^T\end{aligned} \tag{5.7.18}$$

where

$$Q_k = V_k V_k^T.$$

A Givens algorithm to compute an upper (or lower) triangular matrix, $\overline{W}_k$, is given in Section 5.8, Eqs. (5.8.6) through (5.8.13). If there is no process noise, set $V_k = 0$ in this algorithm. Methods for maintaining the measurement update in triangular form are discussed next.

Triangular Square Root Measurement Update

The algorithm for performing the measurement update on the state error covariance matrix square root, $\overline{W}$, can be expressed as

$$W = \overline{W}\widetilde{A}. \tag{5.7.19}$$

If $\overline{W}$ is lower triangular and if $\widetilde{A}$ is lower triangular, then W will be lower triangular, and the computational process expressed in Eq. (5.7.19) will be conducted in a more efficient manner. Choose $\widetilde{A}$ as the solution to

$$\widetilde{A}\widetilde{A}^T = [I - \alpha \widetilde{F}\widetilde{F}^T], \tag{5.7.20}$$

and require that $\widetilde{A}$ be lower triangular. $\widetilde{A}$ will contain $(n^2 + n)/2$ unknowns. Since the right-hand side is symmetric, Eq. (5.7.20) contains $(n^2 + n)/2$ unique equations for the $(n^2 + n)/2$ unknowns. By analytically expanding the results we can determine the expressions for $\tilde{a}_{ij}$, the elements of $\widetilde{A}$. The procedure used is as follows.

If $\overline{W}$ is lower triangular, the equation

$$\widetilde{F} = \overline{W}^T H^T$$

can be expressed as

$$\begin{bmatrix} \widetilde{F}_1 \\ \vdots \\ \widetilde{F}_n \end{bmatrix} = \begin{bmatrix} \overline{W}_{11} & \cdots\cdots & \overline{W}_{n1} \\ & \overline{W}_{22} \cdots\cdots & \overline{W}_{n2} \\ & & \ddots \quad \vdots \\ & O & \overline{W}_{nn} \end{bmatrix} \begin{bmatrix} H_1 \\ \vdots \\ H_n \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{j=1}^{n} \overline{W}_{j1} H_j \\ \vdots \\ \overline{W}_{nn} H_n \end{bmatrix} \tag{5.7.21}$$

or, for the general i^{th} element,

$$\widetilde{F}_i = \sum_{j=i}^{n} \overline{W}_{ji} H_j, \quad i = 1, \ldots, n. \tag{5.7.22}$$

From Eq. (5.7.20), the matrix B can be defined as

$$B = \widetilde{A}\widetilde{A}^T = I - \alpha \widetilde{F}\widetilde{F}^T. \tag{5.7.23}$$

The general ij element of the product matrix in Eq. (5.7.23) can be expressed as

$$B_{ij} = \sum_{k=1}^{j} a_{ik} a_{jk} = \delta_{ij} - \alpha \widetilde{F}_i \widetilde{F}_j, i = 1, \ldots, n; j = 1, \ldots, i. \tag{5.7.24}$$

From Eq. (5.7.24), it follows that

$$\begin{aligned} B_{11} &= a_{11}^2 = 1 - (\widetilde{F}_1^2 \alpha) \\ B_{21} &= a_{21} a_{11} = -(\widetilde{F}_2 \widetilde{F}_1 \alpha) \\ B_{22} &= a_{21}^2 + a_{22}^2 = 1 - (\widetilde{F}_2^2 \alpha) \\ B_{31} &= a_{31} a_{11} = -(\widetilde{F}_3 \widetilde{F}_1 \alpha) \\ B_{32} &= a_{31} a_{21} + a_{32} a_{22} = -(\widetilde{F}_3 \widetilde{F}_2 \alpha) \\ B_{33} &= a_{31}^2 + a_{32}^2 + a_{33}^2 = 1 - (\widetilde{F}_3^2 \alpha) \\ B_{41} &= a_{41} a_{11} = -(F_4 F_1 \alpha). \\ &\vdots \end{aligned} \tag{5.7.25}$$

If the definition

$$\beta_i = \frac{1}{\alpha} - \sum_{j=1}^{i-1} \widetilde{F}_j^2 = R + \sum_{j=i}^{n} \widetilde{F}_j^2, \tag{5.7.26}$$

where

$$\frac{1}{\alpha} = R + \sum_{j=1}^{n} \widetilde{F}_j^2$$

is used, then

$$\beta_{i-1} = \beta_i + \widetilde{F}_{i-1}^2 \; i = n+1, \ldots, 2 \tag{5.7.27}$$

where β_i satisfies the conditions

$$\beta_{n+1} = R, \beta_1 = \tfrac{1}{\alpha}. \tag{5.7.28}$$

If Eq. (5.7.25) are solved recursively, starting with a_{11}, the solution can be expressed as

$$a_{ii} = \sqrt{\beta_{i+1}/\beta_i} \tag{5.7.29}$$

$$a_{ij} = -\widetilde{F}_i \widetilde{F}_j / \sqrt{\beta_{j+1}\beta_j} \qquad \begin{matrix} i = 1, \ldots, n \\ j = 1, \ldots, i-1. \end{matrix}$$

Once the elements of $\widetilde{A}$ have been computed, the elements of Eq. (5.7.19) can be expressed as follows:

$$W_{ij} = \sum_{k=j}^{i} \overline{W}_{ik} a_{kj} \qquad \begin{matrix} i = 1, \ldots, n \\ j = 1, \ldots, i. \end{matrix} \tag{5.7.30}$$

The algorithm for the state estimate with a lower triangular square root covariance update is:

Given: $\overline{W}$, H, R, $\overline{\mathbf{x}}$, and y, where $\overline{W}$ is lower triangular
Compute:

$$
\begin{aligned}
&1.\ \widetilde{F}_i = \sum_{j=1}^{n} \overline{W}_{ji} H_j && i = 1, \ldots, n \\
&2.\ \beta_{n+1} = R \\
&\quad \beta_j = \beta_{j+1} + \widetilde{F}_j^2 && j = n, \ldots, 1 \\
&\quad \beta_1 = 1/\tilde{\alpha} \\
&3.\ d_i = 1/\sqrt{\beta_{i+1}\beta_i} && i = 1, \ldots, n \\
&\quad a_{ii} = \beta_{i+1} d_i \\
&\quad a_{ij} = -\widetilde{F}_i \widetilde{F}_j d_j && j = 1, \ldots, i-1 \\
&4.\ W_{ij} = \sum_{k=j}^{i} \overline{W}_{ik} a_{kj} && \begin{matrix} i = 1, \ldots, n \\ j = 1, \ldots, i \end{matrix} \\
&5.\ K_i = \alpha \Big(\sum_{j=1}^{i} \overline{W}_{ij} \widetilde{F}_j \Big) && i = 1, \ldots, n \\
&6.\ \hat{\mathbf{x}}_i = \overline{\mathbf{x}} + K(y - H\overline{\mathbf{x}}).
\end{aligned}
\tag{5.7.31}
$$

If $\mathbf{y}$ is a vector of observations, this procedure must be repeated for each element of $\mathbf{y}$.

The previous algorithm yields a lower triangular square root factorization of the measurement update error covariance matrix. An upper triangularization also can be used. The algorithm for this was developed by Carlson (1973). The version presented here is due to Bierman (1977) and is a slightly modified version of Carlson's algorithm to enhance accuracy.

Given: $\overline{W}$, H, R, $\overline{\mathbf{x}}$, and y, where $\overline{W}$ is upper triangular
Compute:

$$
\begin{aligned}
&1.\ \widetilde{F} = \overline{W}^T H^T \\
&\quad \beta_o = R, \quad K_2^T = \Big[\overline{W}_{11} \widetilde{F}_1, \overbrace{0, 0, \ldots 0}^{n-1} \Big] \\
&\quad \text{for } j = 1, \ldots n \text{ cycle through steps 2 through 7} \\
&\quad (\text{for } j = 1 \text{ do not evaluate steps 6 and 7})
\end{aligned}
\tag{5.7.32}
$$

$$
\begin{aligned}
&2.\ \beta_j = \beta_{j-1} + \widetilde{F}_j^2 \\
&3.\ d_j = (\beta_{j-1}/\beta_j)^{1/2} \\
&4.\ \gamma_j = \widetilde{F}_j/(\beta_j d_j) \\
&5.\ W_{jj} = \overline{W}_{jj} d_j \\
&6.\ W_{ij} = \overline{W}_{ij} d_j - \gamma_j K_j(i) \qquad i = 1, \ldots j-1 \\
&7.\ K_{j+1}(i) = K_j(i) + \widetilde{F}_j \overline{W}_{ij} \qquad i = 1, \ldots j \\
&8.\ K = K_{n+1}/\beta_n \\
&9.\ \hat{\mathbf{x}} = \overline{\mathbf{x}} + K(y - H\overline{\mathbf{x}}).
\end{aligned}
\tag{5.7.33}
$$

Although at first glance it does not seem to matter whether one uses an upper or lower triangular form for W, Carlson (1973) points out that the time update for the upper triangular form is computationally more efficient. This can be seen by partitioning the state vector such that

$$
\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix},
$$

where $\mathbf{x}_1$ contains the dynamical parameters such as position and velocity and $\mathbf{x}_2$ contains constants such as gravity coefficients or time-varying quantities such as gas leaks that affect the elements of $\mathbf{x}_1$ but are not affected by $\mathbf{x}_1$. Hence, the state transition matrix for $\mathbf{x}$ can be partitioned as

$$
\Phi = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ O & \Phi_{22} \end{bmatrix},
$$

where Φ_{11} is a dense matrix, Φ_{12} may contain some zeros, and Φ_{22} is typically diagonal. When performing the time update, if W is upper triangular, only the upper left partition of the product $\overline{W} = \Phi W$ becomes nontriangular and requires retriangularization.

Recall that these algorithms are designed to process the observations (acutally observation deviations) as scalars. Hence, if $\mathbf{y}$ is a vector the elements of $\mathbf{y}$ must be processed one at a time. We assume that the observations are uncorrelated so that R is a diagonal matrix.

If R is not a diagonal matrix, that is, the observation errors are correlated, we can perform a whitening and decorrelation transformation given by Bierman (1977), and described next.

Given a set of observations

$$\mathbf{y} - H\mathbf{x} = \boldsymbol{\epsilon}\,, \tag{5.7.34}$$

where

$$E[\boldsymbol{\epsilon}] = 0$$
$$E[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T] = R$$

and R is not diagonal but is positive definite. Compute the triangular square root of R

$$R = VV^T\,. \tag{5.7.35}$$

The Cholesky algorithm may be used to ensure that V is triangular. Next multiply Eq. (5.7.34) by V^{-1}

$$V^{-1}\mathbf{y} = V^{-1}H\mathbf{x} + V^{-1}\boldsymbol{\epsilon} \tag{5.7.36}$$

let

$$\tilde{\mathbf{y}} = V^{-1}\mathbf{y},\ \widetilde{H} = V^{-1}H,\ \tilde{\boldsymbol{\epsilon}} = V^{-1}\boldsymbol{\epsilon}$$

then

$$\tilde{\mathbf{y}} = \widetilde{H}x + \tilde{\boldsymbol{\epsilon}} \tag{5.7.37}$$

It is easily shown that $\widetilde{\boldsymbol{\epsilon}}$ has zero mean and unit variance,

$$E[\tilde{\boldsymbol{\epsilon}}] = V^{-1}E[\boldsymbol{\epsilon}] = 0$$
$$E[\tilde{\boldsymbol{\epsilon}}\,\tilde{\boldsymbol{\epsilon}}^T] = V^{-1}E[\boldsymbol{\epsilon}\,\boldsymbol{\epsilon}^T]V^{-T} = V^{-1}RV^{-T} = I \tag{5.7.38}$$

Hence, we would process the observations, $\tilde{\mathbf{y}}$, instead of $\mathbf{y}$.

5.7.2 Square Root Free Measurement Update Algorithms

The square root operations in the Potter and Triangular algorithms shown earlier lead to increased computational load. Square root free equivalents of these algorithms can be used to obtain enhanced computational performance. A comprehensive discussion of these square root free algorithms is given by Thornton (1976) and Bierman (1977).

A square root free algorithm, known as the *U–D covariance factorization*, developed by Bierman and Thornton is described next. Let

$$P = UDU^T \tag{5.7.39}$$

where U is unit upper triangular and D is diagonal. Substituting Eq. (5.7.39) into the expression for the covariance matrix update

$$P = [I - KH]\overline{P} \tag{5.7.40}$$

and restricting attention to the scalar observation case leads to the following expression:

$$UDU^T = \overline{U}\,\overline{D}\,\overline{U}^T - \overline{U}\,\overline{D}\,\overline{U}^T H^T \alpha H \overline{U}\,\overline{D}\,\overline{U}^T \tag{5.7.41}$$

where

$$\alpha = (H\overline{P}H^T + R)^{-1} = \left(H\overline{U}\,\overline{D}\,\overline{U}^T H^T + R\right)^{-1}. \tag{5.7.42}$$

Now, by factoring, Eq. (5.7.41) can be expressed as follows:

$$UDU^T = \overline{U}\left[\overline{D} - \overline{D}\,\overline{U}^T H^T \alpha H \overline{U}\,\overline{D}\right]\overline{U}^T. \tag{5.7.43}$$

Let

$$F = \overline{U}^T H^T, V = \overline{D}F, \tag{5.7.44}$$

then Eq. (5.7.43) can be expressed as

$$P = UDU^T = \overline{U}\left[\overline{D} - V\alpha V^T\right]\overline{U}^T, \tag{5.7.45}$$

where

$$\overline{U} = \begin{bmatrix} 1 & \overline{U}_{12} & \cdots & \overline{U}_{1n} \\ & 1 & & \vdots \\ & & \ddots & \overline{U}_{n-1n} \\ & & & 1 \end{bmatrix}; \overline{D} = \begin{bmatrix} \overline{d}_1 & & 0 \\ & \overline{d}_2 & \\ 0 & & \ddots \\ & & & \overline{d}_n \end{bmatrix}. \tag{5.7.46}$$

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_n \end{bmatrix} = \overline{U}^T H^T;\ V = \overline{D}F = \begin{bmatrix} \overline{d}_1 F_1 \\ \overline{d}_2 F_2 \\ \vdots \\ \overline{d}_n F_n \end{bmatrix}. \tag{5.7.47}$$

Now let $\widetilde{U}$ and $\widetilde{D}$ be the factors of $[\overline{D} - V\alpha V^T]$. Then Eq. (5.7.45) can be written as

$$UDU^T = \overline{U}\widetilde{U}\widetilde{D}\widetilde{U}^T\overline{U}^T. \tag{5.7.48}$$

Since $\overline{U}$ and $\widetilde{U}$ are upper triangular, their product will be upper triangular, and can be expressed as

$$\begin{aligned} U &= \overline{U}\widetilde{U} \\ D &= \widetilde{D}\ . \end{aligned} \tag{5.7.49}$$

Hence, the problem of factoring the covariance measurement update for P has been reduced to factoring the symmetric matrix $[\overline{D} - V\alpha V^T]$ into $\widetilde{U}$ and $\widetilde{D}$. This can be done using the square root free Cholesky decomposition or using the more computationally efficient algorithm presented in Thornton (1976) or Bierman (1977). This algorithm together with an example also appears in Maybeck (1979).

The computational algorithm for the measurement update of P can be summarized then as follows:

Given: $\overline{U}$, $\overline{D}$, H, R, $\overline{\mathbf{x}}$, and y

Compute:

1. $F = \overline{U}^T H^T$
2. $V_i = \overline{d}_i F_i, \qquad i = 1, \ldots, n$
3. $\alpha_1 = (R + V_1 F_1)$
   $d_1 = \frac{\overline{d}_1 R}{\alpha_1}$
   $b_1 = V_1$
4. $\left.\begin{aligned} \alpha_j &= \alpha_{j-1} + F_j V_j \\ d_j &= \overline{d}_j \alpha_{j-1} \,/\, \alpha_j \\ b_j &= V_j \\ p_j &= -F_j \,/\, \alpha_{j-1} \end{aligned}\right\} \quad j = 2, \ldots, n$
5. $\left.\begin{aligned} U_{ij} &= \overline{U}_{ij} + b_i p_j \\ b_i &= b_i + \overline{U}_{ij} V_j \end{aligned}\right\} \quad \begin{aligned} j &= 2, \ldots, n \\ i &= 1, \ldots, j-1 \end{aligned}$
6. $K = \mathbf{b} \,/\, \alpha_n$
7. $\hat{\mathbf{x}} = \overline{\mathbf{x}} + K(y - H\overline{\mathbf{x}})$.

Note that $\mathbf{b}$ is an n vector, $[b_1, b_2 \ldots b_n]^T$. If the estimation error covariance matrix is needed, it may be calculated by using Eq. (5.7.39).

Thus far, with the exception of the Potter Algorithm, we have not addressed the time update of the estimation error covariance matrix. This will be addressed in Sections 5.8 and 5.9.

5.8 TIME UPDATE OF THE ESTIMATION ERROR COVARIANCE MATRIX

Section 5.7 deals primarily with the measurement update of the estimation error covariance matrix, P. Various discrete algorithms have been developed for the time update. The most obvious approach for the time update would be to map WW^T or UDU^T using the conventional measurement update equation given in Chapter 4, for W_k and V_k,

$$\begin{aligned}\overline{P}_{k+1} &= \Phi(t_{k+1},t_k)P_k\Phi^T(t_{k+1},t_k) + \Gamma(t_{k+1},t_k)Q_k\Gamma^T(t_{k+1},t_k)\\ &= \Phi(t_{k+1},t_k)W_kW_k^T\Phi^T(t_{k+1},t_k)\\ &\quad +\Gamma(t_{k+1},t_k)V_kV_k^T\Gamma^T(t_{k+1},t_k)\end{aligned} \tag{5.8.1}$$

where

$$Q_k = V_kV_k^T\,.$$

$\overline{P}_{k+1}$ may then be triangularized using a Cholesky transformation. After this, the measurement update is applied to process the next vector of observations.

Although this approach is computationally efficient, it is not as accurate as an orthogonal transformation approach that does not require the formation of the covariance matrix with the attendant numerical problems. A more accurate approach would be to find a propagation equation for the square root of $\overline{P}$. This can be accomplished by noting that the right-hand side of Eq. (5.8.1) may be written as

$$\overline{P}_{k+1} = B_{k+1}B_{k+1}^T \tag{5.8.2}$$

where

$$B_{k+1} = \left[\Phi(t_{k+1},t_k)W_k \,\vdots\, \Gamma(t_{k+1},t_k)V_k\right]. \tag{5.8.3}$$

Note, however, that B_{k+1} would not be a square matrix but would be of dimension $n\times(n+m)$, where n is the dimension of the state vector and m is the dimension of the process noise vector. Various methods for converting Eq. (5.8.3) into an $n\times n$ triangular matrix are described by Bierman (1977), Thornton (1976), Maybeck (1979), Dyer and McReynolds (1969), and Kaminski (1971). The key to the triangularization of Eq. (5.8.3) is to note that an $(n+m)\times(n+m)$ orthogonal transformation, T, can be applied to Eq. (5.8.2) so that (recall that $TT^T = I$)

$$B_{k+1}TT^TB_{k+1}^T = \tilde{W}_{k+1}\tilde{W}_{k+1}^T \tag{5.8.4}$$

where

$$\tilde{W}_{k+1} = B_{k+1}T = \left[\phi \vdots \overline{W}_{k+1}\right] \tag{5.8.5}$$

and where $\overline{W}_{k+1}$ is now an $n \times n$ upper triangular matrix and ϕ is an $n \times m$ null matrix. Various methods that may be used to accomplish the triangularization are the Gram-Schmidt orthogonalization (Bierman, 1977), the Householder transformation, or the Givens transformation (Thornton, 1976).

The following Givens square root factorization algorithm is taken from Thornton (1976), and yields an $n \times n$ upper triangularization for $\overline{W}$ and an $n \times m$ null matrix, ϕ.

Let B be a full rank $n \times (n+m)$ matrix with column vectors B_i, $i = 1 \ldots, n+m$. The following algorithm yields an $n \times n$ upper triangular factor $\overline{W}$ such that $\overline{W}\,\overline{W}^T = B\,B^T$. For $j = n, \ldots, 1$, cycle through Eqs. (5.8.6) through (5.8.12).

$$k = m + j\,. \tag{5.8.6}$$

For $i = k-1, \ldots 1$ evaluate recursively Eqs. (5.8.7) through (5.8.12).

$$B'_k(j) = \sqrt{(B_i(j))^2 + (B_k(j))^2} \tag{5.8.7}$$

$$C = B_k(j)/B'_k(j) \tag{5.8.8}$$

$$S = B_i(j)/B'_k(j) \tag{5.8.9}$$

$$V = B_i \tag{5.8.10}$$

$$B_i = CB_i - SB_k \tag{5.8.11}$$

$$B_k = SV + CB_k\,. \tag{5.8.12}$$

When completed this algorithm yields

$$B = \overbrace{\left[\phi \vdots \overline{W}\right]}^{m \quad n}\Big\}n \tag{5.8.13}$$

where $\overline{W}$ is upper triangular.

As pointed out by Thornton (1976), a lower triangular factor $\overline{W}$ may be obtained from Eqs (5.8.6) through (5.8.12) if the indicies i and j are reordered so that $j = 1, \ldots, n$ and $i = j+1, \ldots, k$. At the conclusion of this algorithm B has the form

$$B = \overbrace{\left[\overline{W} \vdots \phi\right]}^{n \quad m}\Big\}n$$

where $\overline{W}$ is lower triangular.

The previous algorithms yield $\overline{W}$ as either upper or lower triangular. For the $U-D$ factorization we need $\overline{U}$ and $\overline{D}$ in order to perform the time update for P,

$$\overline{P} = \overline{U}\,\overline{D}\,\overline{U}^T$$

The following algorithm employs a generalized Gram-Schmidt orthogonalization to preserve numerical accuracy and was developed by Thornton and Bierman (1975), and it also appears in Maybeck (1979). The measurement update for P at the k^{th} stage is given by

$$P_k = U_k\, D_k\, U_k^T. \tag{5.8.14}$$

Let

$$Y_{k+1} = [\Phi(t_{k+1}, t_k)U_k \vdots \Gamma(t_{k+1}, t_k)] \tag{5.8.15}$$

$$\widetilde{D}_{k+1} = \begin{bmatrix} D_k & 0 \\ 0 & Q_k \end{bmatrix}. \tag{5.8.16}$$

Then it can be seen that $Y_{k+1}\,\widetilde{D}_{k+1}\,Y_{k+1}^T$ satisfies Eq. (5.8.1). We may now apply the following algorithm to obtain $\overline{U}$ and $\overline{D}$, the time updated values of U and D. Let

$$Y_{k+1}^T = [\,\mathbf{a}_1 \vdots \mathbf{a}_2 \vdots \cdots \vdots \mathbf{a}_n\,] \tag{5.8.17}$$

Where each vector $\mathbf{a}_i$ is of dimension $n+m$. Again n is the dimension of the state vector and m is the dimension of the process noise vector.

For $\ell = n,\ n-1, \ldots, 1$:

$$\begin{aligned} \mathbf{C}_\ell &= \widetilde{D}\mathbf{a}_\ell \quad (\text{i.e., } C_{\ell j} = \widetilde{D}_{jj}a_{\ell j},\ \ j = 1, 2, \ldots, n+m) \\ \overline{D}_{\ell\ell} &= \mathbf{a}_\ell^T \mathbf{C}_\ell \\ \mathbf{d}_\ell &= \mathbf{C}_\ell / D_{\ell\ell} \\ \overline{U}_{j\ell} &= \mathbf{a}_j^T \mathbf{d}_\ell \\ \mathbf{a}_j &\leftarrow \mathbf{a}_j - \overline{U}_{j\ell}\mathbf{a}_\ell \end{aligned} \tag{5.8.18}$$

where $\leftarrow$ denotes replacement; that is, write over the old variable to reduce storage requirements. For the final iteration, $\ell = 1$ and only $\mathbf{C}_1$ and D_{11} are computed.

5.9 CONTINUOUS STATE ERROR COVARIANCE PROPAGATION

In this section a method that allows the integration of the continuous state error covariance differential equations in square root form is described. The derivation follows the approach used in Tapley and Choe (1976) and Tapley and Peters (1980), but the results are based on the $P \equiv UDU^T$ decomposition. This algorithm can be combined with a triangular measurement update algorithm to obtain a complete square root estimation algorithm for which square root operations are avoided. In addition, the effects of state process noise are included without approximation.

The differential equation for propagation of the state error covariance matrix can be expressed as

$$\dot{\overline{P}}(t) = A(t)\overline{P}(t) + \overline{P}(t)\,A^T(t) + Q(t)\,, \tag{5.9.1}$$

where $\overline{P}(t)$ is the *a priori* state error covariance matrix, $A(t)$ is the $n \times n$ linearized dynamics matrix, and $Q(t)$ is the process noise covariance matrix. Each of the matrices in Eq. (5.9.1) is time dependent in the general case. However, for simplicity, the time dependence will not be noted specifically in the following discussion.

If the following definitions are used,

$$\overline{P} \equiv \overline{U}\,\overline{D}\,\overline{U}^T, \overline{Q} \equiv Q/2\,, \tag{5.9.2}$$

and if the first part of Eq. (5.9.2) is differentiated with respect to time and substituted into Eq. (5.9.1), the results can be rearranged to form

$$\begin{aligned}&(\dot{\overline{U}}\,\overline{D} + \overline{U}\,\dot{\overline{D}}\,/\,2 - \overline{Q}\,\overline{U}^{-T} - A\overline{U}\,\overline{D})\overline{U}^T \\ &+\overline{U}(\overline{D}\,\dot{\overline{U}}^T + \dot{\overline{D}}\,\overline{U}^T/2 - \overline{U}^{-1}\overline{Q}^T - \overline{D}\,\overline{U}^T\,A^T) = 0\,.\end{aligned} \tag{5.9.3}$$

Noting that the first term of Eq. (5.9.3) is the transpose of the second term, and making the following definition

$$C(t) \equiv (\dot{\overline{U}}\,\overline{D} + \overline{U}\,\dot{\overline{D}}\,/\,2 - \overline{Q}\,\overline{U}^{-T} - A\overline{U}\,\overline{D})\overline{U}^T \tag{5.9.4}$$

one obtains

$$C(t) + C^T(t) = 0\,. \tag{5.9.5}$$

Relation (5.9.5) requires that $C(t)$ be either the null matrix or, more generally, skew symmetric.

Equation (5.9.4) can be simplified by selectively carrying out the multiplication of the $-Q\overline{U}^{-T}$ term by $\overline{U}^T$ to yield, after terms are arranged

$$(\dot{\overline{U}}\,\overline{D} + \overline{U}\,\dot{\overline{D}}\,/\,2 - A\overline{U}\,\overline{D})\overline{U}^T = \overline{Q} + C(t) \equiv \tilde{C}(t)\ . \tag{5.9.6}$$

Equation (5.9.6) defines the differential equations for $\overline{U}$ and $\overline{D}$ to the degree of uncertainty in $C(t)$. Since the unknown matrix $C(t)$ is skew symmetric, there exist $n(n-1)/2$ unknown scalar quantities in Eq. (5.9.6). The problem considered here is one of specifying the elements of $C(t)$ so that $\overline{U}$ is maintained in triangular form during the integration of Eq. (5.9.6). (The derivation pursued here assumes that $\overline{U}$ is lower triangular and $\overline{D}$ is diagonal, although an algorithm for an upper triangular $\overline{U}$ can be obtained as easily.) The following definitions are made to facilitate the solution to the problem posed:

$$T \equiv A\overline{U}\,\overline{D} \quad M \equiv \dot{\overline{U}}\,\overline{D} + \overline{U}\,\dot{\overline{D}}\,/\,2 - T. \tag{5.9.7}$$

With these definitions, Eq. (5.9.6) is expressed as

$$M\overline{U}^T = \tilde{C}(t) = \overline{Q} + C(t)\ . \tag{5.9.8}$$

Since $\overline{U}$ and $\dot{\overline{U}}$ in Eq. (5.9.6) are lower triangular, and since from Eq. (5.9.5) $C(t)$ is skew symmetric, several observations can be made regarding Eq. (5.9.8). There are $n(n-1)/2$ unknown elements in $\tilde{C}$. The products $\dot{\overline{U}}\,\overline{D}$ and $\overline{U}\,\dot{\overline{D}}$ are lower triangular, creating $n(n+1)\,/\,2$ unknowns. Therefore, the $n \times n$ system of equations in Eq. (5.9.8) has $[n(n-1)\,/\,2 + n(n+1)\,/\,2] = n \times n$ unknowns that can be determined uniquely.

An expansion of Eq. (5.9.8) into matrix elements indicates the method of solution

$$\begin{bmatrix} M_{11} & -T_{12} & \cdots & -T_{1n} \\ M_{21} & M_{22} & \cdots & -T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n1} & M_{n2} & \cdots & M_{nn} \end{bmatrix} \begin{bmatrix} 1 & \overline{U}_{21} & \cdots & \overline{U}_{n1} \\ \vdots & 1 & \cdots & \overline{U}_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \vdots & \cdots & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \overline{q}_{11} & -C_{21} & \cdots & -C_{n1} \\ C_{21} & \overline{q}_{22} & \cdots & -C_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ C_{n1} & C_{n2} & \cdots & \overline{q}_{mn} \end{bmatrix}\ . \tag{5.9.9}$$

In Eq. (5.9.9), $\overline{Q}$ is assumed to be a diagonal matrix with elements $\overline{q}_{ii} = q_{ii}/2$

$(i = 1, \ldots, n)$. (This assumption can be generalized to allow other nonzero terms in the $\overline{Q}$ matrix with only a slight increase in algebraic complexity.) Each row of the upper triangular portion of the $\widetilde{C}$ matrix in Eq. (5.9.9) is determined as the product of the corresponding row of the M matrix with the appropriate column of the $\overline{U}^T$ matrix. After an upper triangular row of $\widetilde{C}$ is determined, the condition from Eq. (5.9.5) that $\widetilde{C}_{ij} = -\widetilde{C}_{ji}$ $(i = 1, \ldots, n; \quad j = 1, \ldots, i-1)$ is invoked to evaluate the corresponding lower triangular column of $\widetilde{C}$. Then, a column of the lower triangular elements of M can be evaluated. Once the elements of the M matrix are determined, the new row of the upper triangular $\widetilde{C}$ elements can be computed along with a column of the $\dot{\overline{U}}$ and $\dot{\overline{D}}$ elements. This process is repeated until all $\dot{\overline{U}}$ and $\dot{\overline{D}}$ values are determined. The implementation of this approach proceeds as follows. From Eqs. (5.9.6) and (5.9.7) one can write

$$M + T = \dot{\overline{U}}\,\overline{D} + \overline{U}\,\dot{\overline{D}}\,/\,2\,. \tag{5.9.10}$$

The expansion of Eq. (5.9.10) in summation notation gives

$$M_{ij} + T_{ij} = \sum_{k=1}^{n} \dot{\overline{U}}_{ik}\overline{d}_{kj} + \sum_{k=1}^{n} \frac{\overline{U}_{ik}\dot{\overline{d}}_{kj}}{2}$$

$$(i = 1, \ldots, n; \quad j = 1, \ldots, i)\,. \tag{5.9.11}$$

But, since $\overline{D}$ is diagonal, Eq. (5.9.11) becomes

$$M_{ij} + T_{ij} = \dot{\overline{U}}_{ij}\overline{d}_{jj} + \overline{U}_{ij}\dot{\overline{d}}_{jj}\,/\,2$$

$$(i = 1, \ldots, n; j = 1, \ldots, i) \tag{5.9.12}$$

for $i = j$, $\overline{U}_{ij} \equiv 1$, and $\dot{\overline{U}}_{ij} \equiv 0$. Therefore, Eq. (5.9.12) becomes

$$\dot{\overline{d}}_{ii} = 2\,(M_{ii} + T_{ii}) \quad (i = 1, \ldots, n)\,. \tag{5.9.13}$$

For $i > j$, Eq. (5.9.12) is rearranged to obtain the differential equation

$$\dot{\overline{U}}_{ij} = (M_{ij} + T_{ij} - \overline{U}_{ij}\dot{\overline{d}}_{jj}\,/\,2)/\overline{d}_{jj}$$

$$(i = 2, \ldots, n; \quad j = 1, \ldots, \quad i-1)\,. \tag{5.9.14}$$

Equations (5.9.13) and (5.9.14) are the forms of the differential equations to be employed in the derivative routine of a numerical integrator. The elements of T_{ij} and M_{ij} are computed as defined in Eq. (5.9.7). The pertinent equations can be combined to obtain the following algorithm.

5.9.1 TRIANGULAR SQUARE ROOT ALGORITHM

Measurement Update

If the Extended Sequential Filter algorithm (for which the reference trajectory is updated at each observation point) is adopted, the measurement update algorithm for the UDU^T factorization has the following form. Using the observation $Y_{k+1} = G(\mathbf{X}_{k+1}, t_{k+1})$, calculate

$$H_{k+1} = [\partial G\,(\overline{\mathbf{X}}_{k+1},\, t_{k+1}) \,/\, \partial\, \overline{\mathbf{X}}_{k+1}]\,. \tag{5.9.15}$$

For $i = 1, \ldots, n$

$$\overline{F}_i = H_i + \sum_{k=i+1}^{n} H_k \overline{U}_{ki} \tag{5.9.16}$$

$$V_i = \overline{d}_i \overline{F}_i\,. \tag{5.9.17}$$

Set $\beta_{n+1} = R_{k+1}$ (where R_{k+1} is the variance of the measurement noise at the $k+1^{\text{st}}$ observation epoch) and calculate

$$\beta_i = \beta_{i+1} + V_i \overline{F}_i \quad (i = n, \ldots, 1)\,. \tag{5.9.18}$$

Calculate diagonal covariance elements

$$d_i = \overline{d}_i \beta_{i+1} \,/\, \beta_i \quad (i = 1, \ldots, n) \tag{5.9.19}$$

$$\alpha = \beta_1\,. \tag{5.9.20}$$

For $i = 2, \ldots, n$ and $j = 1, \ldots, i-1$, calculate

$$P_j = \overline{F}_j \,/\, \beta_{j+1} \tag{5.9.21}$$

$$B_{ij} = V_i + \sum_{k=j+1}^{i-1} \overline{U}_{ik} V_k \tag{5.9.22}$$

$$U_{ij} = \overline{U}_{ij} - B_{ij} P_j \quad (i = 2, \ldots n;\quad j = 1, \ldots, i-1)\,. \tag{5.9.23}$$

Compute the observation residual

$$y_{k+1} = Y_{k+1} - G\,(\overline{\mathbf{X}}_{k+1},\, t_{k+1})\,. \tag{5.9.24}$$

Calculate gain, and update the state using

$$\overline{K}_i = V_i + \sum_{j=1}^{i-1} \overline{U}_{ij}\, V_j \quad (i = 1, \ldots, n) \tag{5.9.25}$$

$$\hat{\mathbf{X}}_i = \overline{\mathbf{X}}_i + \overline{K}_i\, y_{k+1} \,/\, \alpha \quad (i = 1, \ldots, n)\,. \tag{5.9.26}$$

Propagation

Given the elements of the square root state error covariance in lower triangular $\overline{U}\,\overline{D}\,\overline{U}^T$ form, $\overline{Q} \equiv Q/2$ and $A(t)$, the differential equations $\dot{\overline{U}}_{ij}$ and $\dot{\overline{d}}_{ii}$ can be computed as follows:

$$T_{ij} = \sum_{k=j}^{n} A_{ik}\overline{U}_{kj}\overline{d}_{jj} \quad (i = 1, \ldots, n; \quad j = 1, \ldots, n) \tag{5.9.27}$$

$$\overline{C}_{ij} = \sum_{k=1}^{i} M_{ik}\overline{U}_{jk} - \sum_{k=i+1}^{j} T_{ik}\overline{U}_{jk} \quad (i = 1, \ldots, n; \quad j = i+1, \ldots, n) \tag{5.9.28}$$

$$M_{ii} = \overline{q}_{ii} - \sum_{k=1}^{i-1} M_{ik}\overline{U}_{ik} \quad (i = 1, \ldots, n) \tag{5.9.29}$$

$$M_{ij} = -\overline{C}_{ji} - \sum_{k=1}^{j-1} M_{ik}\overline{U}_{jk} \quad (i = 1, \ldots, n; \quad j = 1, \ldots, i-1) \tag{5.9.30}$$

$$\dot{\overline{d}}_{ii} = 2(M_{ii} + T_{ii}) \quad (i = 1, \ldots, n) \tag{5.9.31}$$

$$\dot{\overline{U}}_{ij} = (M_{ij} + T_{ij} - \overline{U}_{ij}\dot{\overline{d}}_{jj}/2)/\overline{d}_{jj} \quad (i = 1, \ldots, n; \quad j = 1, \ldots, i-1)\,. \tag{5.9.32}$$

The algorithm summarized in Eqs. (5.9.15) through (5.9.32) defines a complete sequential estimation algorithm in which the covariance matrices P and $\overline{P}$ are replaced by the factors (U, D) and $(\overline{U}, \overline{D})$, respectively. The algorithm given here assumes that only a single scalar observation is processed at each observation epoch; however, the algorithm is applicable to the case of multiple observations at a given epoch if the observation errors are uncorrelated. To obtain

$$A(t) = \frac{F(\mathbf{X}, t)}{\partial \mathbf{X}}\,,$$

the additional integration of the system of nonlinear differential equations, $\dot{\mathbf{X}} = F(\mathbf{X}, t)$, is required. At the initial point, the starting conditions $\overline{\mathbf{X}}_k = \hat{\mathbf{X}}_k$ are used.

5.10 THE SQUARE ROOT INFORMATION FILTER

In Sections 5.7, 5.8, and 5.9, attention has been directed to the problem of developing estimation algorithms based on the square root of the covariance matrix. Both measurement incorporation and time propagation algorithms were considered. In this section, we consider algorithms derived from the information equations, referred to as the normal equations in Chapter 4. Specifically, we consider algorithms that deal with factoring the information matrix. In general such an algorithm is referred to as a *square root information filter* or SRIF.

We will first consider the case where the state vector, $\mathbf{x}$, is independent of time; that is, a constant. Assume that *a priori* information $[\bar{\mathbf{x}}, \bar{\Lambda}]$ is given, where $\bar{\mathbf{x}}$ is the *a priori* value of $\mathbf{x}$ and $\bar{\Lambda}$ is the *a priori* information matrix, $\bar{\Lambda} = \bar{P}^{-1}$. The *a priori* information can be written in the form of a data equation by noting that

$$\bar{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta}\,, \tag{5.10.1}$$

where $\boldsymbol{\eta}$ is the error in $\bar{\mathbf{x}}$ and is assumed to have the following characteristics

$$E[\boldsymbol{\eta}] = 0, \quad E[\boldsymbol{\eta}\boldsymbol{\eta}^T] = \bar{P} = \bar{\Lambda}^{-1}. \tag{5.10.2}$$

Factoring the information matrix yields

$$\bar{\Lambda} = \bar{R}^T\bar{R}. \tag{5.10.3}$$

Multiplying Eq. (5.10.1) by $\bar{R}$ yields

$$\bar{R}\bar{\mathbf{x}} = \bar{R}\mathbf{x} + \bar{R}\boldsymbol{\eta}\,. \tag{5.10.4}$$

Define

$$\bar{\mathbf{b}} = \bar{R}\bar{\mathbf{x}} \quad \text{and} \quad \bar{\boldsymbol{\eta}} = \bar{R}\boldsymbol{\eta}\,. \tag{5.10.5}$$

Then Eq. (5.10.4) assumes the standard form of the data equation

$$\bar{\mathbf{b}} = \bar{R}\mathbf{x} + \bar{\boldsymbol{\eta}}\,. \tag{5.10.6}$$

Note that the error $\bar{\boldsymbol{\eta}}$ still has zero mean but now has unit variance,

$$\begin{aligned} E[\bar{\boldsymbol{\eta}}] &= \bar{R}E[\boldsymbol{\eta}] = 0 \\ E[\boldsymbol{\eta}\boldsymbol{\eta}^T] &= \bar{R}E[\boldsymbol{\eta}\boldsymbol{\eta}^T]\bar{R}^T = \bar{R}\,\bar{P}\,\bar{R}^T = I\,. \end{aligned} \tag{5.10.7}$$

We now seek to determine the "best" estimate of $\mathbf{x}$ given the *a priori* information in the form of Eq. (5.10.6), and additional observation data

$$\mathbf{y} = H\mathbf{x} + \boldsymbol{\epsilon}\,, \tag{5.10.8}$$

where $\epsilon \sim (O, I)$; the observations have been prewhitened as described in Section 5.7.1. This value of $\mathbf{x}$ will be defined as that which minimizes the least squares performance index,

$$\begin{aligned} J(\mathbf{x}) &= \|\epsilon\|^2 + \|\overline{\eta}\|^2 \\ &= \|H\mathbf{x} - \mathbf{y}\|^2 + \|\overline{R}\mathbf{x} - \overline{\mathbf{b}}\|^2 \\ &= \left\| \begin{bmatrix} \overline{R} \\ H \end{bmatrix} \mathbf{x} - \begin{bmatrix} \overline{\mathbf{b}} \\ \mathbf{y} \end{bmatrix} \right\|^2 . \end{aligned} \tag{5.10.9}$$

Following the procedure of Section 5.4, we apply a series of orthogonal transformations to Eq. (5.10.9) such that

$$T \begin{bmatrix} \overline{R} & \overline{\mathbf{b}} \\ H & \mathbf{y} \end{bmatrix} = \begin{bmatrix} R & \mathbf{b} \\ O & \mathbf{e} \end{bmatrix} \tag{5.10.10}$$

and the performance index becomes

$$\begin{aligned} J(\mathbf{x}) &= \left\| \begin{bmatrix} R \\ O \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b} \\ \mathbf{e} \end{bmatrix} \right\|^2 \\ &= \|R\mathbf{x} - \mathbf{b}\|^2 + \|\mathbf{e}\|^2 . \end{aligned} \tag{5.10.11}$$

The value of the $\mathbf{x}$ that minimizes the performance index is

$$\hat{\mathbf{x}} = R^{-1}\mathbf{b} . \tag{5.10.12}$$

Equation (5.10.12) is most easily solved by a backward substitution given by Eq. (5.2.8),

$$\hat{x}_i = \left(b_i - \sum_{j=i+1}^{n} R_{ij}\, \hat{x}_j \right) / R_{ii}, i = n \ldots 1 . \tag{5.10.13}$$

The data processing algorithm based on the orthogonal transformation can be summarized as follows. Assume we are given the *a priori* information $\overline{R}$, $\overline{\mathbf{b}}$ and the measurements $y_j = H_j\mathbf{x} + \epsilon_j$, $j = 1, \ldots, \ell$, where each y_j is a scalar and where $\epsilon_j \sim (O, 1)$. Then the least squares estimate can be generated recursively as follows:

(a) Compute

$$T_j \begin{bmatrix} \overline{R} & \overline{\mathbf{b}} \\ H_j & y_j \end{bmatrix} = \begin{bmatrix} R_j & \mathbf{b}_j \\ 0 & e_j \end{bmatrix} .$$

(b) $R_j\hat{\mathbf{x}}_j = \mathbf{b}_j$.

Solve for $\hat{\mathbf{x}}_j$ by a backward substitution as given by Eq. (5.10.13) at each stage j. If the estimation error covariance matrix is desired at any stage, it is computed from

$$P_j = R_j^{-1}R_j^{-T}. \tag{5.10.14}$$

(c) R_j and $\mathbf{b}_j$ become the *a priori* for the next stage.

(d) Repeat this process for $j = 1, \ldots, \ell$.

Note that the residual sum of squares of the *a priori* errors plus observation errors based on ℓ measurements is

$$J(\hat{\mathbf{x}}_\ell) = \sum_{i=1}^{\ell}(y_i - H_i\hat{\mathbf{x}}_\ell)^2 + (\hat{\mathbf{x}}_\ell - \overline{\mathbf{x}})^T\overline{R}^T\overline{R}(\hat{\mathbf{x}}_\ell - \overline{\mathbf{x}}) = \sum_{i=1}^{\ell} e_i^2 . \tag{5.10.15}$$

5.10.1 The Square Root Information Filter with Time-Dependent Effects

Consider the dynamic model of Eq. (4.9.46),

$$\mathbf{x}_k = \Phi(t_k, t_j)\mathbf{x}_j + \Gamma(t_k, t_j)\mathbf{u}_j \tag{5.10.16}$$

where

$$E[\mathbf{u}_j] = \overline{\mathbf{u}}, \quad E[(\mathbf{u}_j - \overline{\mathbf{u}})(\mathbf{u}_k - \overline{\mathbf{u}})^T] = Q\delta_{jk} .$$

Generally we assume that $\overline{\mathbf{u}} = 0$, but for this discussion we will assume that the random sequence $\mathbf{u}_j$ has a known mean, $\overline{\mathbf{u}}$. Q is assumed to be positive definite, and may vary with time, but we will assume that both Q and $\overline{\mathbf{u}}$ are constant here.

Assume that a sequence of scalar observation, y_i , is given such that

$$y_i = H_i\mathbf{x}_i + \epsilon_i, \quad i = 1, \ldots, \ell \tag{5.10.17}$$

and the observations have been prewhitened so that

$$E[\epsilon_i] = 0, \quad E(\epsilon_i^2) = 1 . \tag{5.10.18}$$

Assume that at the initial time, say t_0 , there is *a priori* information given in the form of the information array $[\overline{R}_0, \overline{\mathbf{b}}_0]$ from which $\overline{\mathbf{x}}_0$ and $\overline{P}_0$ are determined as (the *a priori* information may be $\overline{\mathbf{x}}_0$ and $\overline{P}_0$ from which $[\overline{R}_0, \overline{\mathbf{b}}_0]$ are computed)

$$\overline{\mathbf{x}}_0 = \overline{R}_0^{-1}\overline{\mathbf{b}}_0 \quad \overline{P}_0 = \overline{R}_0^{-1}\overline{R}_0^{-T} . \tag{5.10.19}$$

Because $E[\mathbf{u}(t_0)] = \overline{\mathbf{u}}$, the *a priori* quantities are mapped to t_1 by

$$\begin{aligned}
\overline{\mathbf{x}}_1 &= \Phi(t_1, t_0)\overline{\mathbf{x}}_0 + \Gamma(t_1, t_0)\overline{\mathbf{u}} \\
\mathbf{x}_1 &= \Phi(t_1, t_0)\mathbf{x}_0 + \Gamma(t_1, t_0)\mathbf{u}_0 \\
\overline{P}_1 &= E[(\overline{\mathbf{x}}_1 - \mathbf{x}_1)(\overline{\mathbf{x}}_1 - \mathbf{x}_1)^T] \\
&= \Phi(t_1, t_0)\overline{P}_0\Phi^T(t_1, t_0) + \Gamma(t_1, t_0)Q\Gamma^T(t_1, t_0)\,. \qquad (5.10.20)
\end{aligned}$$

For the case where there is no process noise, Q is assumed to be zero, as is $\mathbf{u}_j$, and the following solution is applicable.

Process Noise Absent

In this case both Q and $\overline{\mathbf{u}}$ are assumed to be zero and

$$\overline{P}_k = \Phi(t_k, t_j)P_j\Phi^T(t_k, t_j)\,.$$

In terms of square root notation

$$\overline{P}_k = \overline{R}_k^{-1}\overline{R}_k^{-T} = \Phi(t_k, t_j)\, R_j^{-1}R_j^{-T}\Phi^T(t_k, t_j)\,. \qquad (5.10.21)$$

Hence,

$$\overline{R}_k^{-1} = \Phi(t_k, t_j)R_j^{-1} \qquad (5.10.22)$$

and

$$\overline{R}_k = R_j\Phi^{-1}(t_k, t_j)\,. \qquad (5.10.23)$$

It follows then that the mapping relations are

$$\overline{\mathbf{x}}_k = \Phi(t_k, t_j)\hat{\mathbf{x}}_j \qquad (5.10.24)$$

$$\overline{R}_k = R_j\Phi^{-1}(t_k, t_j)\,. \qquad (5.10.25)$$

Note also that the compressed observation can be obtained as

$$\begin{aligned}
\overline{\mathbf{b}}_k &= \overline{R}_k\overline{\mathbf{x}}_k = R_j\Phi^{-1}(t_k, t_j)\Phi(t_k, t_j)\hat{\mathbf{x}}_j \\
&= R_j\hat{\mathbf{x}}_j = \mathbf{b}_j\,. \qquad (5.10.26)
\end{aligned}$$

With these mapping equations we can set up the equations for the SRIF algorithm. Assume that at time, t_j , we have *a priori* information

$$\overline{\mathbf{b}}_j = \overline{R}_j\overline{\mathbf{x}}_j$$

where

$$\overline{\mathbf{x}}_j = \mathbf{x}_j + \boldsymbol{\eta}_j \qquad (5.10.27)$$

and

$$E[\boldsymbol{\eta}_j] = 0$$
$$E[\boldsymbol{\eta}_j \boldsymbol{\eta}_j^T] = \overline{P}_j = \overline{R}_j^{-1}\, \overline{R}_j^{-T}.$$

By multiplying Eq. (5.10.27) by $\overline{R}_j$, we obtain

$$\overline{\mathbf{b}}_j = \overline{R}_j \mathbf{x}_j + \overline{\boldsymbol{\eta}}_j \tag{5.10.28}$$

where $\overline{\boldsymbol{\eta}}_j = \overline{R}_j \boldsymbol{\eta}_j$. Further, $\overline{\boldsymbol{\eta}}_j$ satisfies the condition $E(\overline{\boldsymbol{\eta}}_j) = 0$ and it is easily demonstrated that $E(\overline{\boldsymbol{\eta}}_j \overline{\boldsymbol{\eta}}_j^T) = I$. Now assume that we have the new observation, y_j, where

$$y_j = H_j\, \mathbf{x}_j + \epsilon_j$$

and ϵ_j satisfies the conditions given in Eq. (5.10.18). To obtain a best estimate of $\mathbf{x}_j$, the least squares performance index to be minimized is

$$J(\mathbf{x}_j) = \|\overline{\boldsymbol{\eta}}_j\|^2 + \epsilon_j^2, \tag{5.10.29}$$

or

$$J(\mathbf{x}_j) = \|\overline{R}_j \mathbf{x}_j - \overline{\mathbf{b}}_j\|^2 + (H_j \mathbf{x}_j - y_j)^2. \tag{5.10.30}$$

Following the procedure discussed in the previous section, $J(\mathbf{x}_j)$ can be written as

$$J(\mathbf{x}_j) = \left\| \begin{bmatrix} \overline{R}_j \\ H_j \end{bmatrix} \mathbf{x}_j - \begin{bmatrix} \overline{\mathbf{b}}_j \\ y_j \end{bmatrix} \right\|^2. \tag{5.10.31}$$

Multiplying by an orthogonal transformation, T, yields

$$J(\mathbf{x}_j) = \left[\begin{bmatrix} \overline{R}_j \\ H_j \end{bmatrix} \mathbf{x}_j - \begin{bmatrix} \overline{\mathbf{b}}_j \\ y_j \end{bmatrix} \right]^T T^T T \left[\begin{bmatrix} \overline{R}_j \\ H_j \end{bmatrix} \mathbf{x}_j - \begin{bmatrix} \overline{\mathbf{b}}_j \\ y_j \end{bmatrix} \right]. \tag{5.10.32}$$

We select T so that

$$T \begin{bmatrix} \overline{R}_j \\ H_j \end{bmatrix} = \begin{bmatrix} R_j \\ 0 \end{bmatrix}, T \begin{bmatrix} \overline{\mathbf{b}}_j \\ y_j \end{bmatrix} = \begin{bmatrix} \mathbf{b}_j \\ e_j \end{bmatrix} \tag{5.10.33}$$

where R_j is upper triangular. Eq. (5.10.30) can be expressed then as

$$J(\mathbf{x}_j) = \left\| \begin{bmatrix} R_j \\ 0 \end{bmatrix} \mathbf{x}_j - \begin{bmatrix} \mathbf{b}_j \\ e_j \end{bmatrix} \right\|^2 \tag{5.10.34}$$

or

$$J(\mathbf{x}_j) = (e_j)^2 + \|R_j\mathbf{x}_j - \mathbf{b}_j\|^2 . \tag{5.10.35}$$

It follows that to minimize $J(\mathbf{x}_j)$,

$$\hat{\mathbf{x}}_j = R_j^{-1}\mathbf{b}_j, P_j = R_j^{-1}R_j^{-T}, J(\hat{\mathbf{x}}_j) = e_j^2. \tag{5.10.36}$$

where the elements of the vector $\hat{\mathbf{x}}_j$ are obtained by evaluating Eq. (5.10.13). The previous steps are equivalent to the measurement update for the Kalman filter. The time update is obtained by mapping $\hat{\mathbf{x}}_j$ and R_j forward to time t_k as follows

$$\begin{aligned} \overline{\mathbf{x}}_k &= \Phi(t_k, t_j)\hat{\mathbf{x}}_j \\ \overline{R}_k &= R_j\Phi^{-1}(t_k, t_j) \\ \overline{\mathbf{b}}_k &= \overline{R}_k\overline{\mathbf{x}}_k = \mathbf{b}_j \end{aligned} \tag{5.10.37}$$

and the measurement update process is repeated to obtain $\hat{\mathbf{x}}_k$. Note that the sum of squares of the estimation errors would be stored as (assuming we have processed k scalar observations)[1]

$$J(\mathbf{x}_k) = \sum_{i=1}^{k} e_i^2.$$

An alternate equation to Eq. (5.10.23) can be obtained by noting that for any t,

$$\overline{R}(t) = R_j\Phi^{-1}(t, t_j)$$

and

$$\dot{\overline{R}}(t) = R_j\dot{\Phi}^{-1}(t, t_j). \tag{5.10.38}$$

Substituting for $\dot{\Phi}^{-1}(t, t_j)$ leads to

$$\dot{\overline{R}}(t) = -R_j\Phi^{-1}(t, t_j)A(t) = -\overline{R}(t)A(t).$$

Hence, as an alternate to Eq. (5.10.23) one can integrate the equation

$$\dot{\overline{R}}(t) = -\overline{R}A(t).$$

with initial conditions

$$\overline{R}(t_j) = R_j. \tag{5.10.39}$$

[1]Again it is noted that $J(\mathbf{x}_k)$ includes the effects of *a priori* information (see Eq. 5.4.36). Also, if there are different observation types the values of $\sum e_i^2$ for each type should be stored in separate arrays.

The propagation can be accomplished either by using Eq. (5.10.37) or by integrating Eq. (5.10.38) while propagating the state by integrating

$$\dot{\bar{\mathbf{x}}} = A(t)\bar{\mathbf{x}}, \tag{5.10.40}$$

with the initial conditions $\bar{\mathbf{x}}_j = \hat{\mathbf{x}}_j$. However, because $\bar{\mathbf{b}}_k = \mathbf{b}_j$ there is no need to map $\bar{\mathbf{x}}_j$. The *a priori* covariance $\bar{P}_k$ at t_k is obtained, in either case, as

$$\bar{P}_k = \bar{R}_k^{-1}\bar{R}_k^{-T}. \tag{5.10.41}$$

If needed, a differential equation for $\bar{R}^{-1}$ is easily developed. From Eq. (5.10.37)

$$\bar{R}_k^{-1} = \Phi(t_k, t_j)R_j^{-1}$$

and

$$\begin{aligned}\dot{\bar{R}}^{-1}(t) &= \dot{\Phi}(t, t_j)R_j^{-1} \\ &= A(t)\Phi(t, t_j)R_j^{-1} \\ \dot{\bar{R}}^{-1}(t) &= A(t)\bar{R}^{-1}(t), \quad \text{with I.C. } \bar{R}^{-1}(t_j) = R_j^{-1}.\end{aligned} \tag{5.10.42}$$

Finally, note that even though R_j is upper triangular, the propagated information matrix $\bar{R}_k$ will not be. An upper triangular time update for $\bar{R}_k$ can be developed by noting that

$$\bar{\Lambda}_k = \bar{P}_k^{-1} = \bar{R}_k^T\bar{R}_k.$$

From Eq. (5.10.37)

$$\bar{\Lambda}_k = \Phi^{-T}(t_k, t_j)R_j^T R_j \Phi^{-1}(t_k, t_j) \tag{5.10.43}$$

multiplying by an orthogonal transformation yields

$$\bar{\Lambda}_k = \Phi^{-T}(t_k, t_j)R_j^T T^T T R_j \Phi^{-1}(t_k, t_j) = \bar{R}_k^T\bar{R}_k$$

where T is chosen so that

$$\bar{R}_k = T R_j \Phi^{-1}(t_k, t_j) \tag{5.10.44}$$

is upper triangular. A Householder or Givens transformation may be used to accomplish this.

5.10.2 THE DYNAMIC CASE WITH PROCESS NOISE

The technique for including process noise in both the Potter and SRIF algorithms was developed by Dyer and McReynolds (1969). The state propagation equation for the case with process noise is given by Eq. (5.10.16),

$$\mathbf{x}_k = \Phi(t_k, t_{k-1})\mathbf{x}_{k-1} + \Gamma(t_k, t_{k-1})\mathbf{u}_{k-1}\,. \tag{5.10.45}$$

The SRIF for this case can be formulated as follows. Assume that at t_{k-1} *a priori* information in the form of an information array $[\overline{R}_{k-1}\ \overline{\mathbf{b}}_{k-1}]$ or equivalently $[\overline{P}_{k-1}\ \overline{\mathbf{x}}_{k-1}]$ is available,

$$\overline{\mathbf{b}}_{k-1} = \overline{R}_{k-1}\overline{\mathbf{x}}_{k-1} \tag{5.10.46}$$

but

$$\overline{\mathbf{x}}_{k-1} = \mathbf{x}_{k-1} + \boldsymbol{\eta}_{k-1} \tag{5.10.47}$$

where $\mathbf{x}_{k-1}$ is the true value. Also,

$$E[\boldsymbol{\eta}_{k-1}] = 0,\ E[\boldsymbol{\eta}_{k-1}\boldsymbol{\eta}_{k-1}^T] = \overline{P}_{k-1} = \overline{R}_{k-1}^{-1}\overline{R}_{k-1}^{-T}\,. \tag{5.10.48}$$

Substituting Eq. (5.10.47) into (5.10.46) yields the data equation for the *a priori* information

$$\overline{\mathbf{b}}_{k-1} = \overline{R}_{k-1}\mathbf{x}_{k-1} + \overline{\boldsymbol{\eta}}_{k-1} \tag{5.10.49}$$

where

$$E[\overline{\boldsymbol{\eta}}_{k-1}] = \overline{R}_{k-1}E[\boldsymbol{\eta}_{k-1}] = 0 \tag{5.10.50}$$

$$E[\overline{\boldsymbol{\eta}}_{k-1}\overline{\boldsymbol{\eta}}_{k-1}^T] = \overline{R}_{k-1}\overline{P}_{k-1}\overline{R}_{k-1}^T = I\,. \tag{5.10.51}$$

A scalar observation is given at t_{k-1}

$$y_{k-1} = H_{k-1}\mathbf{x}_{k-1} + \epsilon_{k-1} \tag{5.10.52}$$

where we assume that the observations have been prewhitened so that $\epsilon_{k-1} \sim [0, 1]$.

A priori information on $\mathbf{u}_{k-1}$ is given by covariance Q and by $\overline{\mathbf{u}}_{k-1}$, the mean value of $\mathbf{u}$. Generally it is assumed that $\mathbf{u}$ is a zero mean process so that the *a priori* value $\overline{\mathbf{u}} = 0$ at each stage. This information also may be written in the form of a data equation by noting that

$$\begin{aligned} \overline{\mathbf{u}} &= \overline{\mathbf{u}}_{k-1} \\ &= \mathbf{u}_{k-1} + \boldsymbol{\alpha}_{k-1} \end{aligned} \tag{5.10.53}$$

where $\overline{\mathbf{u}}_{k-1}$ is the *a priori* value and $\mathbf{u}_{k-1}$ is the true value. The error, $\boldsymbol{\alpha}_{k-1}$, has the properties

$$\begin{aligned} E[\boldsymbol{\alpha}_{k-1}] &= 0, \\ E[\boldsymbol{\alpha}_{k-1}\boldsymbol{\alpha}_{k-1}^T] &= Q. \end{aligned} \tag{5.10.54}$$

We will assume that the process noise is uncorrelated in time; that is, $E[\boldsymbol{\alpha}_i\boldsymbol{\alpha}_j^T] = 0$ for $i \neq j$. Although it is not necessary, we will assume that both $\mathbf{u}$ and Q are constant in time. Factor Q such that

$$R_u^{-1}R_u^{-T} = Q\,. \tag{5.10.55}$$

After multiplying Eq. (5.10.53) by R_u, we may write the data equation for $\overline{\mathbf{u}}_{k-1}$ as

$$\begin{aligned} R_u\overline{\mathbf{u}}_{k-1} &\equiv \overline{\mathbf{b}}_{u_{k-1}} \\ &= R_u\mathbf{u}_{k-1} + \overline{\boldsymbol{\alpha}}_{k-1} \end{aligned} \tag{5.10.56}$$

where

$$\overline{\boldsymbol{\alpha}}_{k-1} \sim [O, I]\,.$$

Now define a performance index for the measurement update at t_{k-1}. We choose as our least squares performance index the sum of the squared errors given by Eqs (5.10.49), (5.10.52), and (5.10.56),

$$\hat{J}_{k-1} = \|\overline{\boldsymbol{\eta}}_{k-1}\|^2 + (\epsilon_{k-1})^2 + \|\overline{\boldsymbol{\alpha}}_{k-1}\|^2\,. \tag{5.10.57}$$

Note that including $\|\overline{\boldsymbol{\alpha}}\|^2$ in the performance index allows us to estimate $\mathbf{u}$ at each stage. The values of $\mathbf{x}_{k-1}$ that minimizes $\hat{J}_{k-1}$ is the filter value, $\hat{\mathbf{x}}_{k-1}$. At this point the value of $\mathbf{u}_{k-1}$ that minimizes $\hat{J}_{k-1}$ is just the *a priori* value, $\overline{\mathbf{u}}_{k-1}$. After we perform a time and measurement update at t_k we will have the necessary information to compute $\hat{\mathbf{u}}_{k-1}$. Use the defining equations for the errors to write $\hat{J}_{k-1}$ as

$$\begin{aligned} \hat{J}_{k-1} = &\|\overline{R}_{k-1}\mathbf{x}_{k-1} - \overline{\mathbf{b}}_{k-1}\|^2 + (H_{k-1}\mathbf{x}_{k-1} - y_{k-1})^2 \\ &+\|R_u\mathbf{u}_{k-1} - \overline{\mathbf{b}}_{u_{k-1}}\|^2. \end{aligned} \tag{5.10.58}$$

Because $\mathbf{x}_{k-1}$ is independent of $\mathbf{u}_{k-1}$ it is convenient to write this as

$$\hat{J}_{k-1} = \left\| \begin{bmatrix} \overline{R}_{k-1} \\ H_{k-1} \end{bmatrix} \mathbf{x}_{k-1} - \begin{bmatrix} \overline{\mathbf{b}}_{k-1} \\ y_{k-1} \end{bmatrix} \right\|^2 + \|R_u\mathbf{u}_{k-1} - \overline{\mathbf{b}}_{u_{k-1}}\|^2\,. \tag{5.10.59}$$

Applying a series of orthogonal transformations to the first term of Eq. (5.10.59) results in

$$\hat{J}_{k-1} = \left\| \begin{bmatrix} \hat{R}_{k-1} \\ 0 \end{bmatrix} \mathbf{x}_{k-1} - \begin{bmatrix} \hat{\mathbf{b}}_{k-1} \\ e_{k-1} \end{bmatrix} \right\|^2 + \|R_u\mathbf{u}_{k-1} - \overline{\mathbf{b}}_{u_{k-1}}\|^2 \tag{5.10.60}$$

or

$$\hat{J}_{k-1} = (e_{k-1})^2 + \|\hat{R}_{k-1}\,\mathbf{x}_{k-1} - \hat{\mathbf{b}}_{k-1}\|^2 + \|R_u\mathbf{u}_{k-1} - \overline{\mathbf{b}}_{u_{k-1}}\|^2\,. \quad (5.10.61)$$

The minimum value of $\hat{J}_{k-1}$ is found by setting

$$\hat{R}_{k-1}\hat{\mathbf{x}}_{k-1} = \hat{\mathbf{b}}_{k-1} \quad (5.10.62)$$

$$R_u\overline{\mathbf{u}}_{k-1} = \overline{\mathbf{b}}_{u_{k-1}}\,. \quad (5.10.63)$$

As stated earlier, Eq. (5.10.63) returns the *a priori* value, $\overline{\mathbf{u}}_{k-1}$. The minimum value of $\hat{J}_{k-1}$ is given by

$$\hat{J}_{k-1} = (e_{k-1})^2\,. \quad (5.10.64)$$

Also,

$$P_{k-1} = \hat{R}_{k-1}^{-1}\hat{R}_{k-1}^{-T}\,. \quad (5.10.65)$$

Having completed the measurement update at t_{k-1}, we are ready to do the time update to t_k. In order to time update the performance index $\hat{J}_{k-1}$, we need to write Eq. (5.10.61) in terms of $\mathbf{x}_k$. Because $\mathbf{u}_{k-1}$ is not time dependent, the update of $\mathbf{u}$ to t_k will be handled in the measurement update. From Eq. (5.10.45) we may write $\mathbf{x}_{k-1}$ in terms of $\mathbf{x}_k$,

$$\mathbf{x}_{k-1} = \Phi^{-1}(t_k,\, t_{k-1})(\mathbf{x}_k - \Gamma(t_k,\, t_{k-1})\mathbf{u}_{k-1})\,. \quad (5.10.66)$$

Substituting Eq. (5.10.66) into Eq. (5.10.61) yields the time update

$$\overline{J}_k = (e_{k-1})^2 + \|\hat{R}_{k-1}\Phi^{-1}(t_k,\, t_{k-1})(\mathbf{x}_k - \Gamma(t_k, t_{k-1})\mathbf{u}_{k-1}) - \hat{\mathbf{b}}_{k-1}\|^2 + \|R_u\mathbf{u}_{k-1} - \overline{\mathbf{b}}_{u_{k-1}}\|^2\,, \quad (5.10.67)$$

which may be written as

$$\overline{J}_k = (e_{k-1})^2 + \left\| \begin{bmatrix} R_u & 0 \\ -\tilde{R}_k\Gamma(t_k,\, t_{k-1}) & \tilde{R}_k \end{bmatrix} \begin{bmatrix} \mathbf{u}_{k-1} \\ \mathbf{x}_k \end{bmatrix} - \begin{bmatrix} \overline{\mathbf{b}}_{u_{k-1}} \\ \hat{\mathbf{b}}_{k-1} \end{bmatrix} \right\|^2 \quad (5.10.68)$$

where

$$\tilde{R}_k \equiv \hat{R}_{k-1}\Phi^{-1}(t_k,\, t_{k-1})\,. \quad (5.10.69)$$

We now apply a series of q orthogonal transformations to the second term of Eq. (5.10.68), where q is the dimension of $\mathbf{u}_{k-1}$. This will partly upper triangularize Eq. (5.10.68) (for $q < n$) and will eliminate the explicit dependence of $\mathbf{x}_k$ on $\mathbf{u}_{k-1}$, i.e.

$$\overline{T}_k \begin{bmatrix} R_u & 0 & \overline{\mathbf{b}}_{u_{k-1}} \\ -\tilde{R}_k\Gamma(t_k, t_{k-1}) & \tilde{R}_k & \hat{\mathbf{b}}_{k-1} \end{bmatrix} = \begin{bmatrix} \overline{R}_{u_k} & \overline{R}_{ux_k} & \widetilde{\mathbf{b}}_{u_k} \\ 0 & \overline{R}_k & \overline{\mathbf{b}}_k \end{bmatrix}\,. \quad (5.10.70)$$

Then,

$$\overline{J}_k = (e_{k-1})^2 + \left\| \begin{bmatrix} \overline{R}_{u_k} & \overline{R}_{ux_k} \\ 0 & \overline{R}_k \end{bmatrix} \begin{bmatrix} \mathbf{u}_{k-1} \\ \mathbf{x}_k \end{bmatrix} - \begin{bmatrix} \widetilde{\mathbf{b}}_{u_k} \\ \overline{\mathbf{b}}_k \end{bmatrix} \right\|^2 . \tag{5.10.71}$$

The minimum value of J is obtained by setting

$$\overline{R}_{u_k}\mathbf{u}_{k-1} + \overline{R}_{ux_k}\mathbf{x}_k = \widetilde{\mathbf{b}}_{u_k} \tag{5.10.72}$$

$$\overline{R}_k\mathbf{x}_k = \overline{\mathbf{b}}_k . \tag{5.10.73}$$

Therefore,

$$\overline{\mathbf{x}}_k = \overline{R}_k^{-1}\overline{\mathbf{b}}_k . \tag{5.10.74}$$

Because $\overline{R}_{uk}$ is nonsingular we can find a value of $\mathbf{u}_{k-1}$ to satisfy Eq. (5.10.72) for any value of $\mathbf{x}_k$. We know that the value of $\overline{\mathbf{x}}_k$ that results from Eq. (5.10.74) is given by

$$\overline{\mathbf{x}}_k = \Phi(t_k, t_{k-1})\hat{\mathbf{x}}_{k-1} + \Gamma(t_k, t_{k-1})\overline{\mathbf{u}}_{k-1}$$

where $\overline{\mathbf{u}}_{k-1}$ is the mean or *a priori* value. Using this value of $\overline{\mathbf{x}}_k$ in Eq. (5.10.72) will yield $\overline{\mathbf{u}}_{k-1}$. This is to be expected since we get no information on $\overline{\mathbf{u}}_{k-1}$ until we process an observation at t_k.

Recall that the error covariance associated with $\bar{\mathbf{x}}_k$ is

$$\begin{aligned} \overline{P}_k &= E[(\overline{\mathbf{x}}_k - \mathbf{x}_k)(\overline{\mathbf{x}}_k - \mathbf{x}_k)^T] = \overline{R}_k^{-1}\overline{R}_k^{-T} \\ &= \Phi(t_k, t_{k-1})P_{k-1}\Phi^T(t_k, t_{k-1}) \\ &\quad + \Gamma(t_k, t_{k-1})Q\Gamma^T(t_k, t_{k-1}) . \end{aligned} \tag{5.10.75}$$

Equation (5.10.71) may be written as

$$\begin{aligned} \overline{J}_k &= (e_{k-1})^2 + \|\overline{R}_{u_k}\mathbf{u}_{k-1} + \overline{R}_{ux_k}\mathbf{x}_k - \widetilde{\mathbf{b}}_{u_k}\|^2 \\ &\quad + \|\overline{R}_k\mathbf{x}_k - \overline{\mathbf{b}}_k\|^2 . \end{aligned} \tag{5.10.76}$$

We may now do the measurement update at t_k. The least squares performance index for the measurement update is

$$\begin{aligned} \hat{J}_k &= \overline{J}_k + (\epsilon_k)^2 + \|\overline{\boldsymbol{\alpha}}_k\|^2 \\ &= \overline{J}_k + (H_k\mathbf{x}_k - y_k)^2 + \|R_u\mathbf{u}_k - \overline{\mathbf{b}}_{u_k}\|^2 . \end{aligned} \tag{5.10.77}$$

This may be written as

$$\begin{aligned} \hat{J}_k &= (e_{k-1})^2 + \|\overline{R}_{u_k}\mathbf{u}_{k-1} + \overline{R}_{ux_k}\mathbf{x}_k - \widetilde{\mathbf{b}}_{u_k}\|^2 \\ &\quad + \left\| \begin{bmatrix} \overline{R}_k \\ H_k \end{bmatrix} \mathbf{x}_k - \begin{bmatrix} \overline{\mathbf{b}}_k \\ y_k \end{bmatrix} \right\|^2 + \|R_u\mathbf{u}_k - \overline{\mathbf{b}}_{u_k}\|^2 . \end{aligned} \tag{5.10.78}$$

Applying orthogonal transformations to the third term yields

$$\hat{J}_k = (e_{k-1})^2 + \|\overline{R}_{u_k}\mathbf{u}_{k-1} + \overline{R}_{ux_k}\mathbf{x}_k - \tilde{\mathbf{b}}_{u_k}\|^2 + \|\hat{R}_k\mathbf{x}_k - \hat{\mathbf{b}}_k\|^2 + (e_k)^2 + \|R_u\mathbf{u}_k - \overline{\mathbf{b}}_{u_k}\|^2 . \tag{5.10.79}$$

Once again $\mathbf{u}_{k-1}$, $\mathbf{u}_k$, and $\mathbf{x}_k$ may be chosen to null all but the $(e)^2$ terms. Hence, the minimum value of $\hat{J}_k$ is

$$\hat{J}_k = (e_{k-1})^2 + (e_k)^2 . \tag{5.10.80}$$

The time update to obtain $\overline{J}_{k+1}$ may now be obtained by substituting

$$\mathbf{x}_k = \Phi^{-1}(t_{k+1}, t_k)(\mathbf{x}_{k+1} - \Gamma(t_{k+1}, t_k)\mathbf{u}_k) \tag{5.10.81}$$

for $\mathbf{x}_k$ in the term $\|\hat{R}_k\mathbf{x}_k - \hat{\mathbf{b}}_k\|^2$. Hence, the general expression for the time update at t_m after processing $m-1$ observations is

$$\overline{J}_m = \sum_{i=1}^{m-1}(e_i)^2 + \sum_{i=1}^{m-1}\|\overline{R}_{u_i}\mathbf{u}_{i-1} + \overline{R}_{ux_i}\mathbf{x}_i - \tilde{\mathbf{b}}_{u_i}\|^2 + \left\| \begin{bmatrix} R_u & 0 \\ -\tilde{R}_m\,\Gamma(t_m, t_{m-1}) & \tilde{R}_m \end{bmatrix} \begin{bmatrix} \mathbf{u}_{m-1} \\ \mathbf{x}_m \end{bmatrix} - \begin{bmatrix} \overline{\mathbf{b}}_{u_{m-1}} \\ \hat{\mathbf{b}}_{m-1} \end{bmatrix} \right\|^2 \tag{5.10.82}$$

and the corresponding measurement update for processing m observations is obtained by upper triangularizing the third term of Eq. (5.10.82) and adding the data equations for the m^{th} observation,

$$\begin{aligned} \hat{J}_m &= \overline{J}_m + (\epsilon_m)^2 + \|\overline{\boldsymbol{\alpha}}_m\|^2 \\ &= \sum_{i=1}^{m-1}(e_i)^2 + \sum_{i=1}^{m}\|\overline{R}_{u_i}\mathbf{u}_{i-1} + \overline{R}_{ux_i}\mathbf{x}_i - \tilde{\mathbf{b}}_{u_i}\|^2 + \|R_u\mathbf{u}_m - \overline{\mathbf{b}}_{u_m}\|^2 \\ &\quad + \left\| \begin{bmatrix} \overline{R}_m \\ H_m \end{bmatrix} \mathbf{x}_m - \begin{bmatrix} \overline{\mathbf{b}}_m \\ y_m \end{bmatrix} \right\|^2 . \end{aligned} \tag{5.10.83}$$

An orthogonal transformation is then applied to the last term of Eq. (5.10.83).

Finally, continuing the time and measurement update through stage m yields

$$\hat{J}_m = \sum_{i=1}^{m}(e_i)^2 + \sum_{i=1}^{m}\|\overline{R}_{u_i}\mathbf{u}_{i-1} + \overline{R}_{ux_i}\mathbf{x}_i - \tilde{\mathbf{b}}_{u_i}\|^2 + \|R_u\mathbf{u}_m - \overline{\mathbf{b}}_{u_m}\|^2 + \|\hat{R}_m\mathbf{x}_m - \hat{\mathbf{b}}_m\|^2 , \tag{5.10.84}$$

so that for the filtering problem $\hat{J}_m$ is minimized by choosing

$$\hat{\mathbf{x}}_m = \hat{R}_m^{-1}\,\hat{\mathbf{b}}_m \tag{5.10.85}$$

and

$$\overline{R}_{u_i}\hat{\mathbf{u}}_{i-1} = \widetilde{\mathbf{b}}_{u_i} - \overline{R}_{ux_i}\hat{\mathbf{x}}_i;\; i = m, m-1 \ldots 1\,. \tag{5.10.86}$$

Notice that the third term in Eq. (5.10.84) is simply the addition of *a priori* information on $\mathbf{u}_m$ and does not affect the performance index until we perform a time and measurement update at t_{m+1}. Then it yields the estimate for $\hat{\mathbf{u}}_m$ given by Eq. (5.10.86) with $i = m+1$.

If we were performing only a filtering operation and had a need for a filtered value of $\hat{\mathbf{u}}$, we would calculate it at each stage and not save the quantities $\overline{R}_u$, $\overline{\mathbf{b}}_u$, and $\overline{R}_{ux}$ needed to compute it. However, if we wish to perform smoothing, these quantities should be saved as described in Section 5.10.4.

5.10.3 SRIF Computational Algorithm

The SRIF computational algorithm is summarized as follows. Assuming we have processed the observations at the $k-1^{st}$ stage, the time update for t_k is obtained by applying a series of orthogonal transformations, $\overline{T}_k$, to

$$\overline{T}_k \begin{array}{c} \overbrace{\qquad\qquad}^{q} \quad \overbrace{\quad}^{n} \quad \overbrace{\quad}^{1} \\ \left[\begin{array}{c|c|c} R_u & 0 & \overline{\mathbf{b}}_{u_{k-1}} \\ \hline -\widetilde{R}_k\Gamma(t_k, t_{k-1}) & \widetilde{R}_k & \hat{\mathbf{b}}_{k-1} \end{array}\right] \begin{array}{l} \}q \\ \}n \end{array} \\ = \begin{array}{c} \overbrace{\quad}^{q} \quad \overbrace{\quad}^{n} \quad \overbrace{\quad}^{1} \\ \left[\begin{array}{ccc} \overline{R}_{u_k} & \overline{R}_{ux_k} & \widetilde{\mathbf{b}}_{u_k} \\ 0 & \overline{R}_k & \overline{\mathbf{b}}_k \end{array}\right] \begin{array}{l} \}q \\ \}n \end{array} \end{array} \tag{5.10.87}$$

where $\tilde{R}_k$ is defined by Eq. (5.10.69) and q and n are the dimensions of the process noise and state vector, respectively. From Eqs. (5.10.70) and (5.10.71) we can write

$$\begin{bmatrix} \overline{R}_{u_k} & \overline{R}_{ux_k} \\ 0 & \overline{R}_k \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{u}}_{k-1} \\ \overline{\mathbf{x}}_k \end{bmatrix} = \begin{bmatrix} \widetilde{\mathbf{b}}_{u_k} \\ \overline{\mathbf{b}}_k \end{bmatrix} \tag{5.10.88}$$

from which $\tilde{\mathbf{u}}_{k-1}$, $\overline{\mathbf{x}}_k$ and $\overline{P}_k$ may be computed if desired. It is not necessary to compute these quantities. However, the quantities in the first row of Eq. (5.10.87) should be stored if smoothing is to take place.

The measurement update at t_k is obtained by applying a series of orthogonal transformations, $\hat{T}_k$, to

$$\hat{T}_k \begin{bmatrix} \overbrace{\overline{R}_k}^{n} & \overbrace{\overline{\mathbf{b}}_k}^{1} \\ H_k & y_k \end{bmatrix} \begin{matrix} \}n \\ \}1 \end{matrix} = \begin{bmatrix} \hat{R}_k & \hat{\mathbf{b}}_k \\ 0 & e_k \end{bmatrix} \tag{5.10.89}$$

and

$$\hat{R}_k \hat{\mathbf{x}}_k = \hat{\mathbf{b}}_k \tag{5.10.90}$$

$$P_k = \hat{R}_k^{-1} \hat{R}_k^{-T} \tag{5.10.91}$$

$$e = e + e_k^2 \tag{5.10.92}$$

where e is the sum of squares of the observation residuals.

We can compute the filter value, $\hat{\mathbf{u}}_{k-1}$, by substituting $\hat{\mathbf{x}}_k$ into Eq. (5.10.86),

$$\hat{\mathbf{u}}_{k-1} = \overline{R}_{u_k}^{-1} (\widetilde{\mathbf{b}}_{u_k} - \overline{R}_{ux_k} \hat{\mathbf{x}}_k) . \tag{5.10.93}$$

The time update at t_{k+1} may now be computed from Eq. (5.10.87) and the measurement update from Eq. (5.10.89) after changing the index from k to $k+1$. The procedure is continued until all observations have been processed.

5.10.4 SMOOTHING WITH THE SRIF

There are two approaches that may be taken to perform smoothing with the SRIF (Kaminski, 1971). The first of these uses the performance index given by Eq. (5.10.84). This performance index must be satisfied by the smooth states as well as the filter states. Hence, the equations needed for smoothing are Eqs. (5.10.85), (5.10.86), and (5.10.66). These are repeated using smoothing notation:

$$\hat{R}_m \hat{\mathbf{x}}_m^m = \hat{\mathbf{b}}_m \tag{5.10.94}$$

$$\overline{R}_{u_i} \hat{\mathbf{u}}_{i-1}^m = \widetilde{\mathbf{b}}_{u_i} - \overline{R}_{ux_i} \hat{\mathbf{x}}_i^m \qquad i = m, m-1, \ldots, 1. \tag{5.10.95}$$

$$\hat{\mathbf{x}}_{i-1}^m = \Phi^{-1}(t_i, t_{i-1})(\hat{\mathbf{x}}_i^m - \Gamma(t_i, t_{i-1}) \hat{\mathbf{u}}_{i-1}^m) \tag{5.10.96}$$

where the notation $()_i^m$ means the smoothed value of the quantity at t_i based on m observations. Note that this is the same notation used for the conventional smoother in Section 4.15.

Starting with $\hat{\mathbf{x}}_m^m$ from Eq. (5.10.94), then $\hat{\mathbf{u}}_{m-1}^m$ is computed from Eq. (5.10.95) and $\hat{\mathbf{x}}_{m-1}^m$ is determined from Eq. (5.10.96). With this value of $\hat{\mathbf{x}}_{m-1}^m$, compute $\hat{\mathbf{u}}_{m-2}^m$ from Eq. (5.10.95) and $\hat{\mathbf{x}}_{m-2}^m$ from Eq. (5.10.96), and so on.

This backward sweep strategy may be shown to be equivalent to the smoother described in Section 4.15, and hence it is also equivalent to the Rauch, Tung, and Striebel smoother (1995).

This equivalence may be used to derive an expression for the smoothed covariance for $\hat{\mathbf{x}}_k^m$ using the filter time update equation

$$\overline{\mathbf{x}}_{k+1} = \Phi(t_{k+1}, t_k)\hat{\mathbf{x}}_k + \Gamma(t_{k+1}, t_k)\overline{\mathbf{u}}_k \,. \tag{5.10.97}$$

The smoothed solution must also satisfy this equation; hence

$$\hat{\mathbf{x}}_{k+1}^m = \Phi(t_{k+1}, t_k)\hat{\mathbf{x}}_k^m + \Gamma(t_{k+1}, t_k)\hat{\mathbf{u}}_k^m. \tag{5.10.98}$$

Using Eqs. (5.10.97) and (5.10.98) we may write

$$\begin{aligned}\hat{\mathbf{x}}_k^m - \hat{\mathbf{x}}_k = \Phi^{-1}(t_{k+1}, t_k)\left[\hat{\mathbf{x}}_{k+1}^m - \overline{\mathbf{x}}_{k+1}\right] \\ +\Phi^{-1}(t_{k+1}, t_k)\Gamma(t_{k+1}, t_k)\left[\overline{\mathbf{u}}_k - \hat{\mathbf{u}}_k^m\right].\end{aligned} \tag{5.10.99}$$

From Eq. (5.10.95)

$$\hat{\mathbf{u}}_k^m = \overline{R}_{u_{k+1}}^{-1}(\widetilde{\mathbf{b}}_{u_{k+1}} - \overline{R}_{ux_{k+1}}\hat{\mathbf{x}}_{k+1}^m). \tag{5.10.100}$$

This equation also yields the filter value of $\overline{\mathbf{u}}_k$,

$$\overline{\mathbf{u}}_k = \overline{R}_{u_{k+1}}^{-1}(\widetilde{\mathbf{b}}_{u_{k+1}} - \overline{R}_{ux_{k+1}}\overline{\mathbf{x}}_{k+1}). \tag{5.10.101}$$

From Eq. (5.10.100) and (5.10.101) we may write

$$\overline{\mathbf{u}}_k - \hat{\mathbf{u}}_k^m = \overline{R}_{u_{k+1}}^{-1}\overline{R}_{ux_{k+1}}(\hat{\mathbf{x}}_{k+1}^m - \overline{\mathbf{x}}_{k+1}) \,. \tag{5.10.102}$$

Substituting Eq. (5.10.102) into (5.10.99) results in

$$\begin{aligned}\hat{\mathbf{x}}_k^m = \hat{\mathbf{x}}_k + \Phi^{-1}(t_{k+1}, t_k)\left[I + \Gamma(t_{k+1}, t_k)\overline{R}_{u_{k+1}}^{-1}\overline{R}_{ux_{k+1}}\right] \\ \times \left[\hat{\mathbf{x}}_{k+1}^m - \overline{\mathbf{x}}_{k+1}\right] \,.\end{aligned} \tag{5.10.103}$$

Comparing Eq. (5.10.103) with Eqs. (4.15.9) and (4.15.10) yields the identity

$$\begin{aligned}S_k &= P_k^k \Phi^T(t_{k+1}, t_k)\left(P_{k+1}^k\right)^{-1} \\ &= \Phi^{-1}(t_{k+1}, t_k)\left[I + \Gamma(t_{k+1}, t_k)\overline{R}_{u_{k+1}}^{-1}\overline{R}_{ux_{k+1}}\right] \,.\end{aligned} \tag{5.10.104}$$

Recall that $\overline{\mathbf{x}}_{k+1} = \Phi(t_{k+1}, t_k)\hat{\mathbf{x}}_k^k$ in Eq. (4.15.9) since $\overline{\mathbf{u}}_k = 0$ by assumption. Hence, the smoothed covariance for the estimation error in $\hat{\mathbf{x}}_k^m$ may be obtained

from Eq. (4.15.24),

$$P_k^m = P_k^k + S_k(P_{k+1}^m - P_{k+1}^k)S_k^T \tag{5.10.105}$$

where P_k^k is the measurement update of the filter covariance at t_k and P_{k+1}^k is the time-updated value of P_k^k at t_{k+1}.

Note that this approach requires storage of the time and measurement update of the filter covariance. The time update is given by Eq. (5.10.75), where $\overline{P} \equiv P_{k+1}^k$. The measurement update is given by Eq. (5.10.91), where $P_k \equiv P_k^k$.

The smoothing operation also may be carried out, as described by Kaminski (1971), using an approach that yields the smoothed covariance directly and does not require storage of the filter covariance. Use Eq. (5.10.66) in Eq. (5.10.84) to eliminate $\mathbf{x}_m$. Then the performance index in terms of $\mathbf{x}_{m-1}^m$ and $\mathbf{u}_{m-1}^m$ becomes

$$\begin{aligned}\hat{J}_{m-1}^m = e_m + \Sigma_{m-1} + \|\overline{R}_{u_m}^m \mathbf{u}_{m-1}^m + \overline{R}_{ux_m}\left[\Phi(t_m, t_{m-1})\mathbf{x}_{m-1}^m\right. \\ \left. + \Gamma(t_m, t_{m-1})\mathbf{u}_{m-1}^m\right] - \widetilde{\mathbf{b}}_{u_m}\|^2 + \|\hat{R}_m\left[\Phi(t_m, t_{m-1})\mathbf{x}_{m-1}^m\right. \\ \left. + \Gamma(t_m, t_{m-1})\mathbf{u}_{m-1}^m\right] - \hat{\mathbf{b}}_m\|^2 .\end{aligned} \tag{5.10.106}$$

The term $\|R_u\mathbf{u}_m - \overline{\mathbf{b}}_{u_m}\|^2$ has been dropped since its value is zero by definition; hence, it does not effect the value of $\hat{J}_{m-1}^m$. Also,

$$e_m \equiv \sum_{i=1}^{m}(e_i)^2 \tag{5.10.107}$$

$$\Sigma_{m-1} \equiv \sum_{i-1}^{m-1}\left\|\overline{R}_{u_i}\mathbf{u}_{i-1} + \overline{R}_{ux_i}\mathbf{x}_i - \widetilde{\mathbf{b}}_{u_i}\right\|^2 . \tag{5.10.108}$$

Equation (5.10.106) may be written as

$$\begin{aligned}\hat{J}_{m-1}^m = \left\| \left[\begin{array}{c|c} \overline{R}_{u_m} + \overline{R}_{ux_m}\Gamma(t_m, t_{m-1}) & \overline{R}_{ux_m}\Phi(t_m, t_{m-1}) \\ \hline \hat{R}_m\Gamma(t_m, t_{m-1}) & \hat{R}_m\Phi(t_m, t_{m-1}) \end{array}\right] \right. \\ \left. \begin{bmatrix} \mathbf{u}_{m-1}^m \\ \mathbf{x}_{m-1}^m \end{bmatrix} - \begin{bmatrix} \widetilde{\mathbf{b}}_{u_m} \\ \hat{\mathbf{b}}_m \end{bmatrix} \right\|^2 + e_m + \Sigma_{m-1}.\end{aligned} \tag{5.10.109}$$

A series of orthogonal transformation is used to upper triangularize the first term of Eq. (5.10.109),

$$T_{m-1}^* \quad \left[\begin{array}{c|c} \overline{R}_{u_m} + \overline{R}_{ux_m}\Gamma(t_m, t_{m-1}) & \overline{R}_{ux_m}\Phi(t_m, t_{m-1}) \\ \hline \hat{R}_m\Gamma(t_m, t_{m-1}) & \hat{R}_m\Phi(t_m, t_{m-1}) \end{array}\right]$$

$$= \left[\begin{array}{c|c|c} R^*_{u_{m-1}} & R^*_{ux_{m-1}} & \mathbf{b}^*_{u_{m-1}} \\ 0 & R^*_{m-1} & \mathbf{b}^*_{m-1} \end{array}\right]. \tag{5.10.110}$$

Hence,

$$\begin{aligned} \hat{J}^m_{m-1} &= \left\| R^*_{u_{m-1}} \mathbf{u}^m_{m-1} + R^*_{ux_{m-1}} \mathbf{x}^m_{m-1} - \mathbf{b}^*_{u_{m-1}} \right\|^2 \\ &\quad + \left\| R^*_{m-1}\, \mathbf{x}^m_{m-1} - \mathbf{b}^*_{m-1} \right\|^2 + e_m + \Sigma_{m-1}\,. \end{aligned} \tag{5.10.111}$$

To minimize $\hat{J}^m_{m-1}$, choose

$$R^*_{u_{m-1}}\, \hat{\mathbf{u}}^m_{m-1} = \mathbf{b}^*_{u_{m-1}} - R^*_{ux_{m-1}} \hat{\mathbf{x}}^m_{m-1} \tag{5.10.112}$$

$$R^*_{m-1}\, \hat{\mathbf{x}}^m_{m-1} = \mathbf{b}^*_{m-1}\,. \tag{5.10.113}$$

The elements of Σ_{m-1} can be nulled by the proper selection of $\mathbf{u}_{m-2} \ldots \mathbf{u}_0$. Therefore, we do not need to consider this term. The covariance for the estimation error in $\hat{\mathbf{u}}^m_{m-1}$ and $\hat{\mathbf{x}}^m_{m-1}$ is given by

$$\begin{aligned} P^m_{m-1} &\equiv \left[\begin{array}{cc} P_u & P_{ux} \\ P_{xu} & P_x \end{array}\right]^m_{m-1} \\ &= \left[\begin{array}{cc} R^*_{u_{m-1}} & R^*_{ux_{m-1}} \\ 0 & R^*_{m-1} \end{array}\right]^{-1} \left[\begin{array}{cc} R^*_{u_{m-1}} & R^*_{ux_{m-1}} \\ 0 & R^*_{m-1} \end{array}\right]^{-T}. \end{aligned} \tag{5.10.114}$$

Solving the equation

$$\left[\begin{array}{cc} R^*_{u_{m-1}} & R^*_{ux_{m-1}} \\ 0 & R^*_{m-1} \end{array}\right] \left[\begin{array}{cc} A & B \\ 0 & C \end{array}\right] = \left[\begin{array}{cc} I & 0 \\ 0 & I \end{array}\right]$$

for the matrices A, B, and C yields the inverse necessary to solve for P^m_{m-1}. Substituting the result into Eq. (5.10.114) results in

$$\begin{aligned} P^m_{u_{m-1}} &= R^{*-1}_{u_{m-1}} R^{*-T}_{u_{m-1}} \\ &\quad + R^{*-1}_{u_{m-1}} R^*_{ux_{m-1}} R^{*-1}_{m-1} R^{*-T}_{m-1} R^{*T}_{ux_{m-1}} R^{*-T}_{u_{m-1}} \end{aligned} \tag{5.10.115}$$

$$P^m_{ux_{m-1}} = -R^{*-1}_{u_{m-1}}\, R^*_{ux_{m-1}}\, R^{*-1}_{m-1} R^{*-T}_{m-1} \tag{5.10.116}$$

$$P^m_{x_{m-1}} = R^{*-1}_{m-1}\, R^{*-T}_{m-1}\,. \tag{5.10.117}$$

We are now ready to write J^m_{m-1} in terms of $\mathbf{x}^m_{m-2}$ and $\mathbf{u}^m_{m-2}$. Because $\mathbf{u}^m_{m-1}$ is independent of either of these quantities and we have already chosen its value

to null the first term in Eq. (5.10.111), we may drop this term when writing J^m_{m-2}; hence,

$$\hat{J}^m_{m-2} = \|R^*_{m-1}\mathbf{x}_{m-1} - \mathbf{b}^*_{m-1}\|^2 + \|\overline{R}_{u_{m-1}}\mathbf{u}_{m-2} + \overline{R}_{ux_{m-1}}\mathbf{x}_{m-1} - \widetilde{\mathbf{b}}_{u_{m-1}}\|^2 + e_m + \Sigma_{m-2}\,. \quad (5.10.118)$$

Now use Eq. (5.10.45) to write $\mathbf{x}_{m-1}$ in Eq. (5.10.118) in terms of $\mathbf{x}_{m-2}$,

$$\begin{aligned}\hat{J}^m_{m-2} &= \left\|R^*_{m-1}\left[\Phi(t_{m-2},t_{m-1})\mathbf{x}_{m-2} + \Gamma(t_{m-2},t_{m-1})\mathbf{u}_{m-2}\right] - \mathbf{b}^*_{m-1}\right\|^2 \\ &+ \left\|\overline{R}_{u_{m-1}}\mathbf{u}_{m-2} + \overline{R}_{ux_{m-1}}\left[\Phi(t_{m-2},t_{m-1})\mathbf{x}_{m-2}\right.\right. \\ &\left.\left. + \Gamma(t_{m-2},t_{m-1})\mathbf{u}_{m-2}\right] - \widetilde{\mathbf{b}}_{u_{m-1}}\right\|^2 + e_m + \Sigma_{m-2} \\ &= \left\| \left[\begin{array}{c|c} \overline{R}_{u_{m-1}} + \overline{R}_{ux_{m-1}}\Gamma(t_{m-2},t_{m-1}) & \overline{R}_{ux_{m-1}}\Phi(t_{m-2},t_{m-1}) \\ R^*_{m-1}\Gamma(t_{m-2},t_{m-1}) & R^*_{m-1}\Phi(t_{m-2},t_{m-1}) \end{array}\right] \right. \\ &\left. \begin{bmatrix}\mathbf{u}_{m-2}\\ \mathbf{x}_{m-2}\end{bmatrix} - \begin{bmatrix}\widetilde{\mathbf{b}}_{u_{m-1}}\\ \mathbf{b}^*_{m-1}\end{bmatrix} \right\|^2 + e_m + \Sigma_{m-2}. \end{aligned} \quad (5.10.119)$$

Next a series of orthogonal tranformations is applied to Eq. (5.10.119). This yields the solution for $\hat{\mathbf{x}}^m_{m-2}$ and $\hat{\mathbf{u}}^m_{m-2}$ and the associated covariance matrices. This procedure is repeated until the initial stage is reached.

Hence, the procedure is continued recursively by applying orthogonal transformations so that

$$T^*_{k-1}\left[\begin{array}{cc|c} \overline{R}_{u_k}\overline{R}_{ux_k}\Gamma(t_k,t_{k-1}) & \overline{R}_{ux_k}\Phi(t_k,t_{k-1}) & \widetilde{\mathbf{b}}_{u_k} \\ \hline R^*_k\Gamma(t_k,t_{k-1}) & R^*_k\Phi(t_k,t_{k-1}) & \mathbf{b}^*_k \end{array}\right] = \begin{bmatrix} R^*_{u_{k-1}} & R^*_{ux_{k-1}} & \mathbf{b}^*_{u_{k-1}} \\ 0 & R^*_{k-1} & \mathbf{b}^*_{k-1} \end{bmatrix} \quad (5.10.120)$$

where $R^*_{u_{k-1}}$ and R^*_{k-1} are upper triangular. The smoothed solution is given by

$$R^*_{k-1}\hat{\mathbf{x}}^m_{k-1} = \mathbf{b}^*_{k-1} \quad (5.10.121)$$

$$R^*_{u_{k-1}}\hat{\mathbf{u}}^m_{k-1} = \mathbf{b}^*_{u_{k-1}} - R^*_{ux_{k-1}}\hat{\mathbf{x}}^m_{k-1}\,. \quad (5.10.122)$$

The smoothed covariance is given by

$$P^m_{x_{k-1}} = R^{*-1}_{k-1}\,R^{*-T}_{k-1} \quad (5.10.123)$$

$$\begin{aligned} P^m_{u_{k-1}} &= R^{*-1}_{u_{k-1}}R^{*-T}_{u_{k-1}} \\ &+ R^{*-1}_{u_{k-1}}R^*_{ux_{k-1}}R^{*-1}_{k-1}R^{*-T}_{k-1}R^{*T}_{ux_{k-1}}R^{*-T}_{u_{k-1}} \end{aligned} \quad (5.10.124)$$

$$P^{m}_{ux_{k-1}} = -R^{*-1}_{u_{k-1}} R^{*}_{ux_{k-1}} R^{*-1}_{k-1} R^{*-T}_{k-1} \,. \tag{5.10.125}$$

The first row of the left-hand side of Eq. (5.10.120) is saved from the filtering solution and the second row is computed as part of the smoothing procedure.

5.11 PROCESS NOISE PARAMETER FILTERING/ SMOOTHING USING A SRIF

Along with the dynamic state parameters, it is often advantageous to include some other types of parameters in filtering satellite data to improve the solution. In this section we expand the results of the previous section to include in the state vector, bias parameters, and exponentially correlated process noise parameters. These additional parameters will be defined as

c : Bias parameters (constant acceleration parameters, ephemeris corrections, station coordinates, etc.)

p : Correlated process noise parameters; many random or unmodeled phenomena can be approximated quite well with first order exponentially correlated process noise, also referred to as a Gauss-Markov Process and sometimes as colored noise. Variables that commonly are modeled as a Gauss-Markov Process include

- — Solar radiation pressure
- — Mismodeled drag effects
- — Leaky attitude control systems
- — Moving station positions
- — Polar motion parameters
- — Clock errors
- — Atmospheric path delays
- — Earth rotation parameters

The recursive equation for mapping a discrete first order exponentially correlated process is (Tapley and Ingram, 1973; Bierman, 1977)

$$\mathbf{p}_{k+1} = M_{k+1}\, \mathbf{p}_k + \mathbf{w}_k \,. \tag{5.11.1}$$

M is the process noise parameter transition matrix and is assumed diagonal, with diagonals, m , given by

$$m = e^{-(t_{k+1}-t_k)/\tau} \tag{5.11.2}$$

where τ is the time constant of the process and represents how correlated a process noise parameter is from one time step to the next. The extremes for τ are related to m as follows:

$\tau \to 0$	$\to$	$m \to 0$	White noise (not correlated at all in time)
$\tau \to \infty$	$\to$	$m \to 1$	Random walk (no steady state, strongly correlated in time).

$\mathbf{w}$ is called the process noise (not the same as $\mathbf{p}$) with

$$E[\mathbf{w}_j] = \overline{\mathbf{w}}_j \,. \tag{5.11.3}$$

In almost all applications, the *a priori* estimate of $\overline{\mathbf{w}}_j$ is zero, but this is not a necessary assumption for this development.

Next

$$E[(\mathbf{w}_j - \overline{\mathbf{w}}_j)(\mathbf{w}_k - \overline{\mathbf{w}}_k)^T] = Q\delta_{jk} \tag{5.11.4}$$

and

$$Q = R_w^{-1} R_w^{-T} \tag{5.11.5}$$

where Q is the process noise covariance and is diagonal with elements

$$q_i = (1 - m_i^2)\sigma_i^2 \,. \tag{5.11.6}$$

The variance corresponding to the particular process noise parameter p_i is σ_i^2 . The SRIF formulation of Section 5.10 can be adapted to handle bias parameters and first order exponentially correlated noise (Bierman, 1977). The state propogation equations are represented by

$$\begin{bmatrix} \mathbf{p} \\ \mathbf{x} \\ \mathbf{c} \end{bmatrix}_{k+1} = \begin{bmatrix} M & 0 & 0 \\ \Phi_p & \Phi_x & \Phi_c \\ 0 & 0 & I \end{bmatrix}_{k+1} \begin{bmatrix} \mathbf{p} \\ \mathbf{x} \\ \mathbf{c} \end{bmatrix}_k + \begin{bmatrix} \mathbf{w}_k \\ 0 \\ 0 \end{bmatrix} \tag{5.11.7}$$

where Φ_p, Φ_x, and Φ_c are state transition matrices that map perturbations in $\mathbf{p}$, $\mathbf{x}$, and $\mathbf{c}$ at t_k into perturbations in $\mathbf{x}$ at t_{k+1}. If we define

$$\mathbf{X}_{k+1} \equiv \begin{bmatrix} \mathbf{p} \\ \mathbf{x} \\ \mathbf{c} \end{bmatrix}_{k+1} \tag{5.11.8}$$

$$\Phi(t_{k+1}, t_k) \equiv \begin{bmatrix} M & 0 & 0 \\ \Phi_p & \Phi_x & \Phi_c \\ 0 & 0 & I \end{bmatrix}_{k+1} \tag{5.11.9}$$

$$\Gamma(t_{k+1}, t_k) \equiv \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix}, \tag{5.11.10}$$

Eq. (5.11.7) may be written as

$$\mathbf{X}_{k+1} = \Phi(t_{k+1}, t_k)\,\mathbf{X}_k + \Gamma(t_{k+1}, t_k)\,\mathbf{w}_k \tag{5.11.11}$$

which is identical to Eq. (5.10.45). Hence the algorithms of Section 5.10.2 may be applied directly to this problem. In actual practice, however, it is customary to reduce computations by taking advantage of the fact that some parameters are constant and the process noise parameters are modeled as a Gauss-Markov Process (Bierman, 1977).

5.11.1 Exponentially Correlated Process Noise SRIF

Recall that for a SRIF, R (the square root of the information matrix, Λ) is operated on for the measurement and time update rather than the covariance matrix, P, where

$$P = \Lambda^{-1} = R^{-1} R^{-T}. \tag{5.11.12}$$

By keeping R upper triangular through orthogonal transformations, the state deviation estimate, $\hat{\mathbf{x}}$, in a standard SRIF is found by simple back substitution and is given by

$$\hat{\mathbf{x}} = R^{-1}\mathbf{b}.$$

We will derive the filter/smoother equations for the SRIF with bias parameters and process noise parameters shown explicitly. This will be accomplished by deriving and minimizing a least squares performance index.

As shown by Eq. (5.11.1), the recursive equation for a first-order exponentially correlated process is

$$\mathbf{p}_k = M_k \mathbf{p}_{k-1} + \mathbf{w}_{k-1}. \tag{5.11.13}$$

Assume that *a priori* information for $\mathbf{w}_{k-1}$ is given by $\overline{\mathbf{w}}_{k-1}$. Generally it is assumed that $\overline{\mathbf{w}} = 0$ at each stage. Assume that the error in $\overline{\mathbf{w}}_{k-1}$ has zero mean (i.e., $\overline{\mathbf{w}}_{k-1}$ is the mean value of $\mathbf{w}_{k-1}$) and covariance Q. Thus,

$$\overline{\mathbf{w}}_{k-1} = \mathbf{w}_{k-1} + \gamma_{k-1} \tag{5.11.14}$$

where $\mathbf{w}_{k-1}$ is the true value and $\boldsymbol{\gamma}_{k-1}$ is the error in $\overline{\mathbf{w}}_{k-1}$. Hence,

$$E[\boldsymbol{\gamma}_{k-1}] = 0,\ E[\boldsymbol{\gamma}_{k-1}\boldsymbol{\gamma}_{k-1}^T] = Q\,. \tag{5.11.15}$$

We will assume that $\boldsymbol{\gamma}$ is uncorrelated in time so that

$$E[\boldsymbol{\gamma}_k\boldsymbol{\gamma}_j^T] = Q\delta_{kj}\,. \tag{5.11.16}$$

As stated in Eq. (5.11.5), define the square root of Q as

$$R_w^{-1}R_w^{-T} = Q\,. \tag{5.11.17}$$

Substituting $\mathbf{w}_{k-1}$ from Eq. (5.11.13) into Eq. (5.11.14) and multiplying by R_w yields a data equation for $\mathbf{w}_{k-1}$,

$$R_w\overline{\mathbf{w}}_{k-1} \equiv \overline{\mathbf{b}}_{w_{k-1}} = R_w(\mathbf{p}_k - M_k\mathbf{p}_{k-1}) + R_w\boldsymbol{\gamma}_{k-1}$$

or

$$\overline{\mathbf{b}}_{w_{k-1}} = R_w(\mathbf{p}_k - M_k\ \mathbf{p}_{k-1}) + \overline{\boldsymbol{\gamma}}_{k-1} \tag{5.11.18}$$

where

$$E\left[\overline{\boldsymbol{\gamma}}_{k-1}\right] = R_wE\left[\boldsymbol{\gamma}_{k-1}\right] = 0 \text{ and } E\left[\overline{\boldsymbol{\gamma}}_{k-1}\overline{\boldsymbol{\gamma}}_{k-1}^T\right] = I\,. \tag{5.11.19}$$

Assume further that at t_{k-1} we have *a priori* information arrays $[\overline{R}_p\ \overline{\mathbf{b}}_p]_{k-1}$, $[\overline{R}_x\ \overline{\mathbf{b}}_x]_{k-1}$, and $[\overline{R}_c\ \overline{\mathbf{b}}_c]_{k-1}$ for $\mathbf{p}$, $\mathbf{x}$, and $\mathbf{c}$, respectively. Assume that an observation is available at t_{k-1} given by

$$y_{k-1} = [H_p\ \ H_x\ \ H_c]_{k-1}\begin{bmatrix}\mathbf{p}\\ \mathbf{x}\\ \mathbf{c}\end{bmatrix}_{k-1} + \epsilon_{k-1} \tag{5.11.20}$$

where the observations have been prewhitened so that ϵ has zero means and unit variance.

Recall that the *a priori* information on $\mathbf{p}$, $\mathbf{x}$, and $\mathbf{c}$ may be written in the form of a data equation. For example, the *a priori* value, $\overline{\mathbf{p}}_{k-1}$, may be written in terms of the true value, $\mathbf{p}_{k-1}$, and the error, $\boldsymbol{\eta}_{p_{k-1}}$, as

$$\overline{\mathbf{p}}_{k-1} = \mathbf{p}_{k-1} + \boldsymbol{\eta}_{p_{k-1}} \tag{5.11.21}$$

where

$$E\left[\boldsymbol{\eta}_{p_{k-1}}\right] = 0 \text{ and } E\left[\boldsymbol{\eta}_p\boldsymbol{\eta}_p^T\right]_{k-1} = \overline{P}_{p_{k-1}} \tag{5.11.22}$$

and

$$\overline{P}_{p_{k-1}} = (\overline{R}_p^{-1}\,\overline{R}_p^{-T})_{k-1}\,. \tag{5.11.23}$$

Then the desired data equation is given by multiplying Eq. (5.11.21) by $\overline{R}_{p_{k-1}}$,

$$\left(\overline{R}_p\,\overline{\mathbf{p}}\right)_{k-1} \equiv \overline{\mathbf{b}}_{p_{k-1}} = \left(\overline{R}_p\,\mathbf{p}\right)_{k-1} + \overline{\boldsymbol{\eta}}_{p_{k-1}} \tag{5.11.24}$$

where $\overline{\boldsymbol{\eta}}_{p_{k-1}}$ has zero mean and unit covariance. Similar data equations can be written to represent the *a priori* information for $\mathbf{x}$ and $\mathbf{c}$ at t_{k-1}.

Given the *a priori* information and the observation at t_{k-1} we wish to determine the corresponding filter values of $\mathbf{p}$, $\mathbf{x}$, and $\mathbf{c}$. The desired algorithm can be developed by minimizing the least squares performance index given by

$$\hat{J}_{k-1} \equiv \left\|\overline{\boldsymbol{\eta}}_{p_{k-1}}\right\|^2 + \left\|\overline{\boldsymbol{\eta}}_{x_{k-1}}\right\|^2 + \left\|\overline{\boldsymbol{\eta}}_{c_{k-1}}\right\|^2 + \left\|\overline{\boldsymbol{\gamma}}_{k-1}\right\|^2 + \left(\epsilon_{k-1}\right)^2\,. \tag{5.11.25}$$

Equation (5.11.25), which corresponds to the measurement update at t_{k-1}, may be written as (see Eqs. (5.11.18), (5.11.20), (5.11.24))

$$\begin{aligned}\hat{J}_{k-1} = {} & \left\|\overline{R}_p\,\mathbf{p} - \overline{\mathbf{b}}_p\right\|_{k-1}^2 + \left\|\overline{R}_x\,\mathbf{x} - \overline{\mathbf{b}}_x\right\|_{k-1}^2 + \left\|\overline{R}_c\,\mathbf{c} - \overline{\mathbf{b}}_c\right\|_{k-1}^2 \\ & + \left\|R_w(\mathbf{p}_k - M_k\mathbf{p}_{k-1}) - \overline{\mathbf{b}}_{w_{k-1}}\right\|^2 \\ & + \left(\begin{bmatrix} H_p & H_x & H_c \end{bmatrix}\begin{bmatrix}\mathbf{p}\\ \mathbf{x}\\ \mathbf{c}\end{bmatrix} - y\right)_{k-1}^2\,.\end{aligned} \tag{5.11.26}$$

Because we may choose $\mathbf{p}_k$ to zero the next-to-last term in Eq. (5.11.26), we do not have to deal with it until we do the time update to t_k. We may write $\hat{J}_{k-1}$ as

$$\begin{aligned}\hat{J}_{k-1} = {} & \left\|\begin{bmatrix} \overline{R}_p & 0 & 0 \\ 0 & \overline{R}_x & 0 \\ 0 & 0 & \overline{R}_c \\ H_p & H_x & H_c \end{bmatrix}\begin{bmatrix}\mathbf{p}\\ \mathbf{x}\\ \mathbf{c}\end{bmatrix} - \begin{bmatrix}\overline{\mathbf{b}}_p\\ \overline{\mathbf{b}}_x\\ \overline{\mathbf{b}}_c\\ y\end{bmatrix}\right\|_{k-1}^2 \\ & + \left\|R_w(\mathbf{p}_k - M_k\,\mathbf{p}_{k-1}) - \overline{\mathbf{b}}_{w_{k-1}}\right\|^2\,.\end{aligned} \tag{5.11.27}$$

Applying a series of orthogonal transformations to the first term of Eq. (5.11.27) yields

$$T_{k-1}\begin{bmatrix} \overline{R}_p & 0 & 0 & \overline{\mathbf{b}}_p \\ 0 & \overline{R}_x & 0 & \overline{\mathbf{b}}_x \\ 0 & 0 & \overline{R}_c & \overline{\mathbf{b}}_c \\ H_p & H_x & H_c & y \end{bmatrix}_{k-1} = \begin{bmatrix} \hat{R}_p & \hat{R}_{px} & \hat{R}_{pc} & \hat{\mathbf{b}}_p \\ 0 & \hat{R}_x & \hat{R}_{xc} & \hat{\mathbf{b}}_x \\ 0 & 0 & \hat{R}_c & \hat{\mathbf{b}}_c \\ 0 & 0 & 0 & e \end{bmatrix}_{k-1} \tag{5.11.28}$$

and Eq. (5.11.27) becomes

$$\hat{J}_{k-1} = \left\| \begin{bmatrix} \hat{R}_p & \hat{R}_{px} & \hat{R}_{pc} \\ 0 & \hat{R}_x & \hat{R}_{xc} \\ 0 & 0 & \hat{R}_c \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \mathbf{x} \\ \mathbf{c} \end{bmatrix} - \begin{bmatrix} \hat{\mathbf{b}}_p \\ \hat{\mathbf{b}}_x \\ \hat{\mathbf{b}}_c \end{bmatrix} \right\|_{k-1}^2 + \left\| R_w(\mathbf{p}_k - M_k\, \mathbf{p}_{k-1}) - \overline{\mathbf{b}}_{w_{k-1}} \right\|^2 + (e)_{k-1}^2 \,. \tag{5.11.29}$$

From Eq. (5.11.29) we could solve for $\hat{\mathbf{c}}$, $\hat{\mathbf{x}}$, and $\hat{\mathbf{p}}$ at the $k-1^{\text{th}}$ stage.

We now perform the time update of $\hat{J}_{k-1}$ in order to obtain $\overline{J}_k$. To accomplish this we need to replace $\mathbf{x}_{k-1}$ with $\mathbf{x}_k$. Notice that our performance index already contains $\mathbf{p}_k$ and we will leave $\mathbf{p}_{k-1}$ in the performance index. This will conveniently yield the smoothed value of $\mathbf{p}_{k-1}$.

From Eq. (5.11.7)

$$\mathbf{x}_k = \Phi_p(k)\mathbf{p}_{k-1} + \Phi_x(k)\mathbf{x}_{k-1} + \Phi_c(k)\mathbf{c}_{k-1} \tag{5.11.30}$$

or

$$\mathbf{x}_{k-1} = \Phi_x^{-1}(k)\left[\mathbf{x}_k - \Phi_p(k)\mathbf{p}_{k-1} - \Phi_c(k)\mathbf{c}_{k-1}\right] \tag{5.11.31}$$

where

$$\Phi(k) \equiv \Phi(t_k, t_{k-1}) \,.$$

Also, $\mathbf{c}_{k-1} = \mathbf{c}_k$, since $\mathbf{c}$ is a constant. Substituting Eq. (5.11.31) into Eq. (5.11.29) yields the following equations

$$\begin{aligned} &\hat{R}_p\mathbf{p}_{k-1} + \overline{R}_{px}\left[\mathbf{x}_k - \Phi_p(k)\mathbf{p}_{k-1} - \Phi_c(k)\mathbf{c}_{k-1}\right] + \hat{R}_{pc}\mathbf{c}_{k-1} = \hat{\mathbf{b}}_{p_{k-1}} \\ &\overline{R}_x\left[\mathbf{x}_k - \Phi_p(k)\mathbf{p}_{k-1} - \Phi_c(k)\mathbf{c}_{k-1}\right] + \hat{R}_{xc}\mathbf{c}_{k-1} = \hat{\mathbf{b}}_{x_{k-1}} \\ &\hat{R}_c\mathbf{c}_{k-1} = \hat{\mathbf{b}}_{c_{k-1}} \,. \end{aligned} \tag{5.11.32}$$

Also,

$$R_w(\mathbf{p}_k - M_k\, \mathbf{p}_{k-1}) = \overline{\mathbf{b}}_{w_{k-1}}$$

where

$$\begin{aligned} \overline{R}_{px} &\equiv \hat{R}_{px}\Phi_x^{-1}(k) \\ \overline{R}_x &\equiv \hat{R}_x\Phi_x^{-1}(k) \,. \end{aligned}$$

By regrouping and writing in matrix form (while noting that $\mathbf{c}_{k-1} = \mathbf{c}_k$), we may write Eq. (5.11.29) as

$$\hat{J}_{k-1} = \left\| \begin{bmatrix} -R_w M_k & R_w & 0 & 0 \\ (\hat{R}_p - \overline{R}_{px}\Phi_p(k)) & 0 & \overline{R}_{px} & (\hat{R}_{pc} - \overline{R}_{px}\Phi_c(k)) \\ -\overline{R}_x\Phi_p(k) & 0 & \overline{R}_x & (\hat{R}_{xc} - \overline{R}_x\Phi_c(k)) \\ 0 & 0 & 0 & \hat{R}_c \end{bmatrix}_{k-1} \right.$$
$$\left. \begin{bmatrix} \mathbf{p}_{k-1} \\ \mathbf{p}_k \\ \mathbf{x}_k \\ \mathbf{c}_k \end{bmatrix} - \begin{bmatrix} \overline{\mathbf{b}}_w \\ \hat{\mathbf{b}}_p \\ \hat{\mathbf{b}}_x \\ \hat{\mathbf{b}}_c \end{bmatrix}_{k-1} \right\|^2 + (e)^2_{k-1} . \tag{5.11.33}$$

Applying a series of orthogonal transformations to upper triangularize the first term in Eq. (5.11.33) yields the time update

$$\overline{J}_k = \left\| \begin{bmatrix} R^*_p & R^*_{pp} & R^*_{px} & R^*_{pc} \\ 0 & \overline{R}_p & \overline{R}_{px} & \overline{R}_{pc} \\ 0 & 0 & \overline{R}_x & \overline{R}_{xc} \\ 0 & 0 & 0 & \overline{R}_c \end{bmatrix}_k \begin{bmatrix} \mathbf{p}_{k-1} \\ \mathbf{p}_k \\ \mathbf{x}_k \\ \mathbf{c}_k \end{bmatrix} \right.$$
$$\left. - \begin{bmatrix} \mathbf{b}^*_p \\ \overline{\mathbf{b}}_p \\ \overline{\mathbf{b}}_x \\ \overline{\mathbf{b}}_c \end{bmatrix}_k \right\|^2 + (e)^2_{k-1} . \tag{5.11.34}$$

The $(\,)^*$ quantities are not used for filtering but are necessary if smoothing is to be done following the filtering. Hence, these quantities must be saved at each stage in the forward (filtering) sweep. Also, because $\mathbf{c}$ is a constant, it is unaffected by the mapping from t_{k-1} to t_k and need not be included in the time update procedure,

$$\left[\hat{R}_c \; \hat{\mathbf{b}}_c\right]_{k-1} = \left[\overline{R}_c \; \overline{\mathbf{b}}_c\right]_k . \tag{5.11.35}$$

We may now perform the measurement update on $\overline{J}_k$ to obtain $\hat{J}_k$. This is accomplished by adding $\|\overline{\gamma}_k\|^2$ (see Eq. (5.11.18)) and $(\epsilon_k)^2$ to $\overline{J}_k$ given by Eq.

(5.11.34). Hence,

$$\hat{J}_k = \left\| \begin{bmatrix} \overline{R}_p & \overline{R}_{px} & \overline{R}_{pc} \\ 0 & \overline{R}_x & \overline{R}_{xc} \\ 0 & 0 & \overline{R}_c \\ H_p & H_x & H_c \end{bmatrix}_k \begin{bmatrix} \mathbf{p} \\ \mathbf{x} \\ \mathbf{c} \end{bmatrix}_k - \begin{bmatrix} \overline{\mathbf{b}}_p \\ \overline{\mathbf{b}}_x \\ \overline{\mathbf{b}}_c \\ y \end{bmatrix}_k \right\|^2 \tag{5.11.36}$$
$$+ \left\| R^*_{p_k} \mathbf{p}_{k-1} + R^*_{pp_k} \mathbf{p}_k + R^*_{px_k} \mathbf{x}_k + R^*_{pc_k} \mathbf{c}_k - \mathbf{b}^*_{p_k} \right\|^2$$
$$+ \left\| R_w(\mathbf{p}_{k+1} - M_{k+1}\mathbf{p}_k) - \overline{\mathbf{b}}_{w_k} \right\|^2 + (e)^2_{k-1} .$$

Once again we can null the third term, $\|\overline{\gamma}_k\|$, by proper choice of $\mathbf{p}_{k+1}$; hence, we do not have to deal with this term until we perform the time update to t_{k+1}. Notice that we can choose $\mathbf{p}_{k-1}$ to null the second term. Later we will see that this is the smoothed value of $\mathbf{p}_{k-1}$.

A series of orthogonal transformations is now applied to Eq. (5.11.36) to yield

$$\hat{J}_k = \left\| \begin{bmatrix} \hat{R}_p & \hat{R}_{px} & \hat{R}_{pc} \\ 0 & \hat{R}_x & \hat{R}_{xc} \\ 0 & 0 & \hat{R}_c \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \mathbf{x} \\ \mathbf{c} \end{bmatrix} - \begin{bmatrix} \hat{\mathbf{b}}_p \\ \hat{\mathbf{b}}_x \\ \hat{\mathbf{b}}_c \end{bmatrix} \right\|^2_k \tag{5.11.37}$$
$$+ \| R^*_{p_k} \mathbf{p}_{k-1} + R^*_{pp_k} \mathbf{p}_k + R^*_{px_k} \mathbf{x}_k + R^*_{pc_k} \mathbf{c}_k - \mathbf{b}^*_{p_k} \|^2$$
$$+ \| R_w(\mathbf{p}_{k+1} - M_{k+1}\mathbf{p}_k) - \overline{\mathbf{b}}_{w_k} \|^2 + (e)^2_{k-1} + (e)^2_k .$$

This procedure of time and measurement updates is carried out until the desired number of observations has been processed.

After the N^{th} measurement update the performance index is given by

$$\hat{J}_N = \Sigma_{N-1} + \left\| R^*_{p_N} \mathbf{p}_{N-1} + R^*_{pp_N} \mathbf{p}_N + R^*_{px_N} \mathbf{x}_N + R^*_{pc_N} \mathbf{c}_N - \mathbf{b}^*_{p_N} \right\|^2$$
$$+ \left\| \begin{bmatrix} \hat{R}_p & \hat{R}_{px} & \hat{R}_{pc} \\ 0 & \hat{R}_x & \hat{R}_{xc} \\ 0 & 0 & \hat{R}_c \end{bmatrix}_N \begin{bmatrix} \mathbf{p}_N \\ \mathbf{x}_N \\ \mathbf{c}_N \end{bmatrix} - \begin{bmatrix} \hat{\mathbf{b}}_p \\ \hat{\mathbf{b}}_x \\ \hat{\mathbf{b}}_c \end{bmatrix}_N \right\|^2$$
$$+ \sum_{i=1}^{N} (e_i)^2 \tag{5.11.38}$$

where

$$\Sigma_{N-1} \equiv \sum_{i=1}^{N-1} \| R^*_{p_i} \mathbf{p}_{i-1} + R^*_{pp_i} \mathbf{p}_i + R^*_{px_i} \mathbf{x}_i + R^*_{pc_i} \mathbf{c}_i - \mathbf{b}^*_{p_i} \|^2 . \quad (5.11.39)$$

The best estimate of the state at t_N is obtained from

$$\hat{\mathbf{X}}_N = \hat{R}_N^{-1} \hat{\mathbf{b}}_N \quad (5.11.40)$$

where

$$\hat{\mathbf{X}}_N \equiv \begin{bmatrix} \hat{\mathbf{p}} \\ \hat{\mathbf{x}} \\ \hat{\mathbf{c}} \end{bmatrix}_N , \quad \hat{R}_N \equiv \begin{bmatrix} \hat{R}_p & \hat{R}_{px} & \hat{R}_{pc} \\ 0 & \hat{R}_x & \hat{R}_{xc} \\ 0 & 0 & \hat{R}_c \end{bmatrix}_N \quad (5.11.41)$$

$$\hat{\mathbf{b}}_N \equiv \begin{bmatrix} \hat{\mathbf{b}}_p \\ \hat{\mathbf{b}}_x \\ \hat{\mathbf{b}}_c \end{bmatrix}_N .$$

The filter covariance is given by

$$P_N = R_N^{-1} R_N^{-T} . \quad (5.11.42)$$

Since $\hat{R}_N$ is upper triangular, $\hat{\mathbf{X}}_N$ is obtained directly from a back substitution described by Eq. (5.2.8). If $\hat{P}_N$, $\hat{\mathbf{x}}_N$, and $\hat{\mathbf{c}}_N$ are used in the second term of Eq. (5.11.38), the smoothed value of P_{N-1} may be obtained. Note that it is not necessary to retain the first two terms of $\hat{J}_N$ if smoothing is not used.

5.11.2 SMOOTHING WITH A SRIF

Smoothing can now be done using Eq. (5.11.38) and the values of $\hat{\mathbf{X}}_N$ given by Eq. (5.11.40). From the second term of Eq. (5.11.38) we have the smoothed value of $\mathbf{p}_{N-1}$ based on N measurements

$$\hat{\mathbf{p}}^N_{N-1} = R^{*-1}_{p_N} \left[\mathbf{b}^*_p - R^*_{pp} \hat{\mathbf{p}}_N - R^*_{px} \hat{\mathbf{x}}_N - R^*_{pc} \hat{\mathbf{c}}_N \right]_N . \quad (5.11.43)$$

The smoothed value of $\mathbf{x}_{N-1}$ is obtained from Eq. (5.11.31)

$$\hat{\mathbf{x}}^N_{N-1} = \Phi_x^{-1}(N) \left[\hat{\mathbf{x}}_N - \Phi_p(N) \hat{\mathbf{p}}^N_{N-1} - \Phi_c(N) \hat{\mathbf{c}}^N_{N-1} \right] \quad (5.11.44)$$

and $\mathbf{c}$ smooths as a constant,

$$\hat{\mathbf{c}}_N = \hat{\mathbf{c}}^N_{N-1} = \hat{\mathbf{c}}^N_i \; i = 1, \ldots, N . \quad (5.11.45)$$

Hence, the general expression for smoothing is given by

$$\hat{\mathbf{p}}_i^N = R_{pi+1}^{*\,-1}\left[\mathbf{b}_p^* - R_{pp}^*\,\hat{\mathbf{p}}_{i+1}^N - R_{px}^*\,\hat{\mathbf{x}}_{i+1}^N - R_{pc}^*\,\hat{\mathbf{c}}_N\right]_{i+1} \qquad i = N-1,\ldots,1 \tag{5.11.46}$$

where the $(\quad)^*$ quantities have been saved at each value of t_i during the filtering process. Also,

$$\hat{\mathbf{x}}_i^N = \Phi_x^{-1}(i+1)\left[\hat{\mathbf{x}}_{i+1}^N - \Phi_p(i+1)\hat{\mathbf{p}}_i^N - \Phi_c(i+1)\hat{\mathbf{c}}_N\right] \qquad i = N-1,\ldots,1\,. \tag{5.11.47}$$

The state transition matrices also are saved during filtering.

Although the procedure just outlined yields the smoothed solutions, it does not yield a smoothed covariance. To obtain the covariance we use the procedure for the filter time update and substitute for $\mathbf{x}_N$ in terms of parameters at the $N-1^{\text{st}}$ stage; from Eq. (5.11.30),

$$\mathbf{x}_N = \Phi_x(N)\mathbf{x}_{N-1} + \Phi_p(N)\mathbf{p}_{N-1} + \Phi_c(N)\mathbf{c}_{N-1}\,. \tag{5.11.48}$$

There is no need to substitute for $\mathbf{p}_N$ because we already have $\mathbf{p}_{N-1}$ available in Eq. (5.11.38) through the use of the data equation for the process noise parameters in the filter sweep.

Substituting Eq. (5.11.48) into Eq. (5.11.38) yields

$$\hat{J}_{N-1}^N = \sum_{i=1}^{N}(e_i)^2 + \Sigma_{N-1} + \left\| \begin{bmatrix} R_{pp}^* & R_p^* + R_{px}^*\Phi_p(N) & R_{px}^*\Phi_x(N) & R_{px}^*\Phi_c(N) + R_{pc}^* \\ \hat{R}_p & \hat{R}_{px}\Phi_p(N) & \hat{R}_{px}\Phi_x(N) & \hat{R}_{px}\Phi_c(N) + \hat{R}_{pc} \\ 0 & \hat{R}_x\Phi_p(N) & \hat{R}_x\Phi_x(N) & \hat{R}_x\Phi_c(N) + \hat{R}_{xc} \\ 0 & 0 & 0 & \hat{R}_c \end{bmatrix}_N \begin{bmatrix} \mathbf{p}_N \\ \mathbf{p}_{N-1} \\ \mathbf{x}_{N-1} \\ \mathbf{c}_{N-1} \end{bmatrix} - \begin{bmatrix} \mathbf{b}_p^* \\ \hat{\mathbf{b}}_p \\ \hat{\mathbf{b}}_x \\ \hat{\mathbf{b}}_c \end{bmatrix}_N \right\|^2 . \tag{5.11.49}$$

Because the smoothed value of $\mathbf{c}$ maps as a constant, we may drop the last row in the preceding matrix and use $\hat{R}_c$ and $\hat{\mathbf{c}}_N$ when needed. Applying a series of orthogonal transformations to the third term yields

$$\hat{J}_{N-1}^N = \sum_{i=1}^{N} (e_i)^2 + \Sigma_{N-1} \tag{5.11.50}$$
$$+ \left\| \begin{bmatrix} R'_{pp} & R'_p & R'_{px} & R'_{pc} \\ 0 & \tilde{R}_p & \tilde{R}_{px} & \tilde{R}_{pc} \\ 0 & 0 & \tilde{R}_x & \tilde{R}_{xc} \end{bmatrix}_{N-1} \begin{bmatrix} \hat{\mathbf{p}}_N^N \\ \hat{\mathbf{p}}_{N-1}^N \\ \hat{\mathbf{x}}_{N-1}^N \\ \hat{\mathbf{c}}_N \end{bmatrix} - \begin{bmatrix} \mathbf{b}^{*\prime} \\ \mathbf{b}'_p \\ \tilde{\mathbf{b}}_p \\ \tilde{\mathbf{b}}_x \end{bmatrix}_{N-1} \right\|^2 .$$

Solving for the state vector that minimizes $\hat{J}_{N-1}^N$ yields the smoothes solution for the state. We may ignore the $(\,)'$ quantities because we already know the value of $\hat{\mathbf{p}}_N^N = \hat{\mathbf{p}}_N$. The desired solution is

$$\tilde{R}_{N-1} \hat{\mathbf{X}}_{N-1}^N = \tilde{\mathbf{b}}_{N-1} \tag{5.11.51}$$

where

$$\tilde{R}_{N-1} = \begin{bmatrix} \tilde{R}_p & \tilde{R}_{px} & \tilde{R}_{pc} \\ 0 & \tilde{R}_x & \tilde{R}_{xc} \\ 0 & 0 & \hat{R}_c \end{bmatrix}$$
$$\hat{\mathbf{X}}_{N-1}^N = \begin{bmatrix} \hat{\mathbf{p}}_{N-1}^N \\ \hat{\mathbf{x}}_{N-1}^N \\ \hat{\mathbf{c}}_N \end{bmatrix}, \qquad \tilde{\mathbf{b}}_{N-1} = \begin{bmatrix} \mathbf{b}'_p \\ \tilde{\mathbf{b}}_p \\ \tilde{\mathbf{b}}_x \end{bmatrix} . \tag{5.11.52}$$

The smoothed covariance at t_{N-1} is given by

$$P_{N-1}^N = \tilde{R}_{N-1}^{-1} \tilde{R}_{N-1}^{-T} . \tag{5.11.53}$$

Recall that the smoothed value of $\mathbf{c}$ never changes and is always the final filter value. Because the smoothed value of the state must also satisfy Eq. (5.11.48), we may substitute this into Eq. (5.11.50) in order to determine the smoothed state at t_{N-2}. Keep in mind that while we drop the $()'$ terms, we must now deal with the $N - 1^{\text{st}}$ term in Σ_{N-1} because it depends on the state at t_{N-1} and contains

$\mathbf{p}_{N-2}$. Hence, the performance index $\hat{J}_{N-2}^{N}$ becomes

$$\hat{J}_{N-2}^{N} = \Sigma_{N-2} + \left\| \begin{bmatrix} R_p^* & R_{pp}^* & R_{px}^* & R_{pc}^* \\ 0 & \tilde{R}_p & \tilde{R}_{px} & \tilde{R}_{pc} \\ 0 & 0 & \hat{R}_x & \hat{R}_{xc} \end{bmatrix}_{N-1} \begin{bmatrix} \mathbf{p}_{N-2} \\ \mathbf{p}_{N-1} \\ \mathbf{x}_{N-1} \\ \mathbf{c}_{N-1} \end{bmatrix} - \begin{bmatrix} \mathbf{b}_p^* \\ \hat{\mathbf{b}}_p \\ \tilde{\mathbf{b}}_x \end{bmatrix}_{N-1} \right\|^2 . \tag{5.11.54}$$

Substituting

$$\mathbf{x}_{N-1} = \Phi_x(N-1)\mathbf{x}_{N-2} + \Phi_p(N-1)\mathbf{p}_{N-2} + \Phi_c(N-1)\mathbf{c}_N \tag{5.11.55}$$

into Eq. (5.11.54) we obtain the $N - 2^{\text{nd}}$ stage of Eq. (5.11.49). Next we apply a series of orthogonal transformations to this equation to obtain the $N-2^{\text{nd}}$ stage of Eq. (5.11.50). From this we obtain $\hat{\mathbf{X}}_{N-2}^{N}$ and the associated covariance matrix. This procedure is repeated until the initial stage is reached. Notice that we have dropped the term

$$\sum_{i=1}^{N} (e_i)^2$$

in Eq. (5.11.54). This is the sum of squares of residuals from the filter sweep and is not affected by smoothing. We do not obtain a smoothed sum of squares of residuals during the backward sweep.

5.12 REFERENCES

Battin, R., *An Introduction to the Mathematics and Methods of Astrodynamics*, American Institute of Aeronautics and Astronautics, Reston, VA, 1999.

Bierman, G. J., *Factorization Methods for Discrete Sequential Estimation*, Academic Press, New York, 1977.

Carlson, N. A., "Fast triangular formulation of the square root filter", *AIAA J.*, Vol. 11, No. 9, pp. 1239–1265, September 1973.

Dyer, P., and S. McReynolds, "Extension of square-root filtering to include process noise", *J. Optim. Theory Appl.*, Vol. 3, No. 6, pp. 444–458, 1969.

Gentleman, W. M., "Least squares computations by Givens transformations without square roots", *J. Inst. Math. Applic.*, Vol. 12, pp. 329–336, 1973.

Givens, W., "Computation of plane unitary rotations transforming a general matrix to triangular form," *J. Appl. Math.*, Vol. 6, pp. 26–50, 1958.

Golub, G. H., and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 1996.

Graybill, F. A., *An Introduction to Linear Statistical Models, Volume I, McGraw-Hill Series in Probability and Statistics*, McGraw-Hill, New York, 1961.

Householder, A. S., "Unitary triangularization of a nonsymmetric matrix," *J. Assoc. Comput. Mach.*, Vol. 5, No. 4, pp. 339–342, October 1958.

Kaminski, P. G., A. E. Bryson, and S. F. Schmidt, "Discrete square root filtering: A survey of current techniques", *Trans. Auto. Cont.*, Vol. AC-16, No. 6, pp. 727–735, 1971.

Lawson, C. L., and R. J. Hanson, *Solving Least Squares Problems*, Prentice-Hall, Inc. Englewood Cliffs, NJ, 1974 (republished by SIAM, Philadelphia, PA, 1995).

Tapley, B. D., and C. Y. Choe, "An algorithm for propagating the square root covariance matrix in triangular form," *Trans. Auto. Cont.*, Vol. AC-21, pp. 122–123, 1976.

Tapley, B. D., and D. S. Ingram, "Orbit determination in the presence of unmodeled accelerations," *Trans. Auto. Cont.*, Vol. AC-18, No. 4, pp. 369–373, August, 1973.

Tapley, B. D., and J. G. Peters, "A sequential estimation algorithm using a continuous UDU^T covariance factorization," *J. Guid. Cont.*, Vol. 3, No. 4, pp. 326–331, July–August 1980.

Thornton, Thornton, C. L., *Triangular covariance factorizations for Kalman filtering*, Technical Memorandum, 33–798, Jet Propulsion Laboratory, Pasadena, CA, October 15, 1976.

5.13 EXERCISES

1. In Section 5.2 the algorithm for $\hat{x}$ is derived assuming the factorization

$$M = R^T R$$

(a) Rederive the algorithm assuming the factorization

$$M = RR^T$$

where R is $n \times n$ upper triangular and

$$\begin{aligned} M\hat{x} &= N \\ RR^T\hat{x} &= N \\ z &\equiv R^T\hat{x} \\ Rz &= N\,. \end{aligned}$$

Answer:

$$z_i = \frac{(N_i - \sum_{j=i+1}^{n} R_{ij} z_j}{R_{ii}} \quad i = n,\ n-1 \ldots 1$$

$$\hat{x}_i = \frac{(z_i - \sum_{j=1}^{i-1} R_{ji}\hat{x}_j)}{R_{ii}} \quad i = 1,\ 2 \ldots n.$$

2. Verify that the algorithm for R in Exercise 1 is given by

$$R_{jj} = \left(M_{jj} - \sum_{k=j+1}^{n} R_{jk}^2 \right)^{1/2} \quad j = n,\ n-1, \ldots 1$$

$$R_{ij} = \frac{\left(M_{ij} - \sum_{k=j+1}^{n} R_{ik} R_{jk} \right)}{R_{jj}} \quad i = j-1, \ldots 1.$$

3. Using Eq. (5.2.6), find R for

(a) The matrix M given by

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 8 & 2 \\ 3 & 2 & 14 \end{bmatrix}.$$

(b) Compute R using the algorithm derived in Exercise 2. Note that these will be different matrices, thus illustrating the nonuniqueness of the matrix square root.

Answers:

Part (a)

$$R = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

Part (b)

$$R = \begin{bmatrix} \frac{1}{3\sqrt{3}} & \frac{11}{3\sqrt{42}} & \frac{3}{\sqrt{14}} \\ 0 & 3\sqrt{\frac{6}{7}} & \frac{2}{\sqrt{14}} \\ 0 & 0 & \sqrt{14} \end{bmatrix}$$

4. Use the square root free Cholesky algorithm (Eqs. (5.2.11) and (5.2.12)) to determine the U and D factors of M for problem 3,

$$M = UDU^T.$$

Answer:

$$U = \begin{bmatrix} 1 & \frac{11}{54} & \frac{3}{14} \\ 0 & 1 & \frac{1}{7} \\ 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} \frac{1}{27} & 0 & 0 \\ 0 & \frac{54}{7} & 0 \\ 0 & 0 & 14 \end{bmatrix}$$

5. Given the function

$$f(t) = \sum_{i=0}^{4} a_i t^i + \sum_{i=1}^{4} A_i \cos(\omega_i t) + B_i \sin(\omega_i t)$$

where

$$\omega_1 = \frac{2\pi}{709}, \quad \omega_2 = \frac{2\pi}{383}, \quad \omega_3 = \frac{2\pi}{107}, \quad \omega_4 = \frac{2\pi}{13}$$

and given measurements of f for $t = 0, 1, 2, \ldots, 1000$, do the following:

(a) Estimate the constants a_0, and a_i, A_i, B_i for $i = 1, 2, 3, 4$.

(b) Compare execution times and accuracy of the following four algorithms.

i. Cholesky decomposition
ii. Givens transformation
iii. Givens square root free transformation
iv. Householder transformation

Generate your own perfect observations using the coefficients given:

Exact Coefficients:

$$\begin{array}{lll} a_0 = -50 & & \\ a_1 = 0.25 & A_1 = -50 & B_1 = 101 \\ a_2 = -0.625 \times 10^{-3} & A_2 = 1 & B_2 = -0.5 \\ a_3 = -0.4 \times 10^{-6} & A_3 = -27 & B_3 = -27 \\ a_4 = 0.9 \times 10^{-9} & A_4 = 4 & B_4 = -3 \end{array}$$

(c) Redo (a) after adding Gaussian random noise with mean zero and variance = 2 to the data.

6. From Eq. (5.10.120), derive Eqs. (5.10.123), (5.10.124), and (5.10.125), which define the covariance for the smoothed solution including process noise. (Hint: see Eqs. (5.10.114) through Eq. (5.10.117)).

7. Generate one cycle of a sine wave with an amplitude of unity. Add white noise, $N(0, 0.05)$. Generate observation data by sampling the noisy sine wave 1000 times at equal intervals.

Using one process noise parameter, recover the sine wave using a SRIF both as a filter and a smoother. Try various combinations of τ and σ in Eqs. (5.11.2) and (5.11.6). For example try a large τ and small σ and estimate the amplitude as a constant (case 1). Try $\tau = 0$ and a very large σ to simulate the sine wave as a white noise process (case 2). Try a random walk process to describe the process noise parameter (i.e., $m = 1$, choose q (case 3)). Finally find a value of τ and σ that does a good job of replicating the sine wave (e.g., one for which the RMS of fit is near to the RMS of the noise on the data (case 4)). Assume *a priori* information, $\bar{p} = 0$ and $\overline{P}_0 = \infty$ and that the process noise, $\overline{w} = 0$, at each stage. Use the algorithms described in Sections 5.11.1 and 5.11.2 for filtering and smoothing, respectively.

Generate the following figures for these four cases:

(a) Plot the observations and the filter solutions. Compute the RMS difference.

(b) Plot the observations and the smoothed solutions. Compute the RMS difference.

(c) Compute the RMS differences between the true sine wave (without noise) and the filter and smoothed solutions.

(d) Plot the observations minus the smoothed solution.

(e) Plot w, the process noise.

For the optimal τ and σ, the plots in (d) and (e) should look similar. The optimal τ and σ result in the correlated signal in the observations being absorbed by p and the random noise by w.

* Solution hints for Exercise 7.

The measurement update at the i^{th} stage is given by performing an orthogonal transformation (Householder or Givens),

$$T_i \begin{bmatrix} \overline{R}_{p_i} & \bar{b}_{p_i} \\ H & y_i \end{bmatrix}_{2\times 2} = \begin{bmatrix} \hat{R}_{p_i} & \hat{b}_{p_i} \\ 0 & e_i \end{bmatrix}_{2\times 2} \tag{5.11.28}$$

where the equation number refers to the corresponding equation in the text. At the epoch time $\overline{R}_{p_0} = 0$ and $\bar{b}_{p_0} = 0$ are given.

Next, a time update to the $i+1^{\text{st}}$ stage results from a series of orthogonal transformations on

$$T_{i+1} \begin{bmatrix} -R_w m & R_w & \bar{b}_{w_i} \\ \hat{R}_{p_i} & 0 & \hat{b}_{p_i} \end{bmatrix}_{2\times 3} = \begin{bmatrix} R^*_{p_i} & R^*_{p_{i+1}} & b^*_{p_i} \\ 0 & \overline{R}_{p_{i+1}} & \bar{b}_{p_{i+1}} \end{bmatrix}_{2\times 3.} \tag{5.11.34}$$

The $(\ )^*$ values must be saved for smoothing. Also,

$$\begin{aligned} R_w &= \frac{1}{\sqrt{q}} \\ m &= e^{-(t_{i+1}-t_i)/\tau} && (5.11.2) \\ q &= (1-m^2)\sigma^2 && (5.11.6) \\ H &= 1 \quad (\text{the observation is } p) \\ \overline{P}_0 &= \infty; \end{aligned}$$

hence, $\overline{R}_{p_0} = 0$ and $\bar{b}_{p_0} = \overline{R}_{p_0}\bar{p}_0 = 0$. The filter value of p at each stage is

computed from

$$\hat{p}_i = \frac{\hat{b}_{p_i}}{\hat{R}_{p_i}}. \tag{5.11.40}$$

After the final data point has been processed we may begin smoothing. The first smoothed value of p is given by (assuming N data points)

$$p_{N-1}^N = \frac{b_{p_N}^* - R_{p_N}^* \hat{p}_N}{R_{p_{N-1}}^*}. \tag{5.11.43}$$

The smoothed value of p at the i^{th} stage is given by

$$\begin{aligned} p_i^N &= \frac{b_{p_{i+1}}^* - R_{p_{i+1}}^* \hat{P}_{i+1}^N}{R_{p_i}^*} \\ i &= N-1,\ N-2 \ldots 0. \end{aligned} \tag{5.11.46}$$

A value of $\hat{w}_i^N$ may be determined from Eq. (5.11.13) by using the smoothed values of p,

$$\hat{w}_i^N = p_{i+1}^N - m p_i^N \quad i = N-1,\ N-2, \ldots 0.$$

8. Work Example 4.8.2, the spring mass problem, using the square root free Givens method to solve for $\hat{\mathbf{x}}_0$ and the associated estimation error covariance matrix. You should get results almost identical to those given in Example 4.8.2.