NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Memorandum 33-798

Triangular Covariance Factorizations for Kalman Filtering

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

October 15, 1976

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. 33-798	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle TRIANGULAR COVARIANCE FACTORIZATIONS FOR KALMAN FILTERING		5. Report Date October 15, 1976 6. Performing Organization Code
7. Author(s) Catherine L. Thornton		8. Performing Organization Report No.
9. Performing Organization Name and Address JET PROPULSION LABORATORY California Institute of Technology 4800 Oak Grove Drive Pasadena, California 91103		10. Work Unit No. 11. Contract or Grant No. NAS 7-100 13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address NATIONAL AERONAUTICS AND SPACE ADMINISTRATION Washington, D.C. 20546		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

In this report an improved computational form of the discrete Kalman filter is derived using an upper triangular factorization of the error covariance matrix. The covariance P is factored such that $P = UDU^T$ where U is unit upper triangular and D is diagonal. Recursions are developed for propagating the U-D covariance factors together with the corresponding state estimate. The resulting algorithm, referred to as the U-D filter, combines the superior numerical precision of square root filtering techniques with an efficiency comparable to that of Kalman's original formula. Moreover, this method is easily implemented and involves no more computer storage than the Kalman algorithm. These characteristics make the U-D method an attractive real-time filtering technique.

A new covariance error analysis technique is obtained from an estension of the U-D filter equations. This evaluation method is flexible and efficient and may provide significantly improved numerical results. Cost comparisons show that for a large class of problems the U-D evaluation algorithm is noticeably less expensive than conventional error analysis methods. The U-D method is shown to be especially attractive for problems involving large numbers of bias parameters since it yields accurate and efficient techniques for performing sensitivity analysis and reduced-order filtering.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Spacecraft Communications, Command and Tracking Computer Programming and Software Numerical Analysis Statistics and Probability	Unclassified -- Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	197	

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Memorandum 33-798

Triangular Covariance Factorizations for Kalman Filtering

Catherine L. Thornton

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

October 15, 1976

Preface

The work described in this report was performed by the Mission Analysis Division of the Jet Propulsion Laboratory.

Acknowledgments

This report was initially prepared as a Ph.D. dissertation for the School of Engineering at the University of California, Los Angeles. The author would like to thank the members of her doctoral committee, Professors L. W. Wiberg, J. K. Omura, K. Yao, S. C. Port and D. A. Sanchez, for their support and their interest in this research. Special appreciation is reserved for Professor Wiberg, who served as principal advisor and offered many valuable comments and criticisms throughout the doctoral research.

Much credit for the successful completion of this work goes to my associates at the Jet Propulsion Laboratory. A special debt is owed to Dr. W. G. Melbourne for his support of this research and to Dr. G. J. Bierman for stimulating the author's interest in the research topic and for generously sharing invaluable background information and useful insights as the research progressed. The numerous helpful suggestions made by Dr. J. P. McDanell and Dr. S. R. McReynolds are also greatly appreciated. Grateful acknowledgment is made to Mary Fran Buehler for her proficient technical editing of the report and to Susan G. Finley and Margaret Medina for their assistance in obtaining the numerical results.

Contents

Tables

Figures

General Notation

System Parameters and Variables

$x(i)$	n-dimensional state vector of discrete linear stochastic process
$z(i)$	m-dimensional observation vector
$w(i)$	k-dimensional zero mean white Gaussian process noise
$v(i)$	m-dimensional zero mean white Gaussian data noise, independent of $w(i)$
$\Phi(i)$	nxn state transition matrix
$B(i)$	nxk noise transition matrix
$A(i)$	mxn data coefficient matrix
$a^T(i)$	n-dimensional vector of data coefficients
$p(i)$	k-dimensional colored process noise vector
$M(i)$	kxk diagonal colored noise transition matrix
y	b-dimensional vector of bias parameters
$R(i)$	mxm diagonal covariance of data errors, $v(i)$
$r(i)$	covariance of scalar data error
$Q(i)$	kxk diagonal covariance of white process noise

Statistical and Filter-Related Quantities

$\hat{x}(i)$	minimum variance estimate of $x(i)$ given data $\{z(j)\}$, $j \leq i$
$\tilde{x}(i)$	a priori estimate of $x(i)$, based upon data $\{z(j)\}$, $j < i$
$K(i)$	nxm filter gain matrix
$\hat{P}(i)$	nxn error covariance matrix for $\hat{x}(i)$
$\tilde{P}(i)$	nxn error covariance matrix for $\tilde{x}(i)$
α	innovations error covariance
S	nxn square root of estimate error covariance matrix

Statistical and Filter-Related Quantities (contd)

U	nxn unit upper triangular factor of error covariance matrix
D	nxn diagonal matrix, related to error covariance matrix by the identity $P = UDU^T$ where U is defined above
U_x, D_x	those portions of the U-D covariance factors which correspond to the x parameters
σ_x	standard deviation of the parameter x
$E\{\cdot\}$	expectation operator
$x \sim N(0,\bar{P})$	x is a normally distributed random variable with zero mean and covariance P

Mathematical Symbols

ϵ	is an element of
R_n	the set consisting of n-tuples whose components are real numbers
Σ	summation
$\overset{r}{=}$	rounded to
$\neq$	not equal to
$\|\|\cdot\|\|$	Euclidian vector norm
$\|\|\cdot\|\|_D$	norm with metric D, i.e., $\|\|v\|\|_D = \left(v^T D v\right)^{1/2}$
$\{\}_{i=1}^{n}$	sequence of n quantities
:=	replace in computer storage
>>	much greater than

Abbreviations

WGS	weighted Gram-Schmidt
MWGS	modified weighted Gram-Schmidt
RSS	root sum of squares
GS	Gram-Schmidt

Abbreviations (contd)

MGS	modified Gram-Schmidt
sgn	sign
diag	diagonal
μsec	microsecond
km	kilometer
km/sec	kilometer per second
m/sec	meter per second
mm/sec	millimeter per second
GM	gravitational mass

Superscripts

T	matrix transpose
-1	matrix inverse
1/2	upper triangular Cholesky square root
-1/2	inverse of upper triangular Cholesky square root
^	after incorporating measurement
~	before incorporating measurement
(j)	after jth iteration (section 3.3)
(j)	after (n-j)th iteration (sections 3.4 and 3.5)

Abstract

In this report an improved computational form of the discrete Kalman filter is derived using an upper triangular factorization of the error covariance matrix. The covariance P is factored such that $P = UDU^T$ where U is unit upper triangular and D is diagonal. Recursions are developed for propagating the U-D covariance factors together with the corresponding state estimate. The resulting algorithm, referred to as the U-D filter, combines the superior numerical precision of square root filtering techniques with an efficiency comparable to that of Kalman's original formula. Moreover, this method is easily implemented and involves no more computer storage than the Kalman algorithm. These characteristics make the U-D method an attractive real-time filtering technique.

A new covariance error analysis technique is obtained from an extension of the U-D filter equations. This evaluation method is flexible and efficient and may provide significantly improved numerical results. Cost comparisons show that for a large class of problems the U-D evaluation algorithm is noticeably less expensive than conventional error analysis methods. The U-D method is shown to be especially attractive for problems involving large numbers of bias parameters since it yields accurate and efficient techniques for performing sensitivity analysis and reduced-order filtering.

Chapter I. Introduction

1.1 Background

The optimal estimation of stochastic linear dynamic processes from imperfect linear measurements is a key problem in the fields of communication and control. Usually estimators are linear functions of the data, and optimal solutions are those which minimize the mean square error. When estimates are based solely upon past and present measurements, this problem is termed a linear filtering problem.

This report addresses the discrete-time linear filtering problem. A solution to this problem, for general nonstationary processes, was first derived by Kalman [1960]. The discrete Kalman filter is a recursive algorithm consisting of two parts: a time update and a measurement update. Each part contains difference equations for propagation of a state estimate and its error covariance matrix. The efficiency and simplicity of Kalman's algorithm make it particularly attractive for use in real-time estimation problems involving small digital computers. These features are among the reasons that Kalman filtering techniques have been widely used in a variety of engineering applications such as spacecraft navigation, aircraft guidance and control, marine navigation and power systems control (cf Battin and Levine [1970], Huddle [1969], Holdsworth and Stolz [1970] or Miller and Lewis [1971]).

Although the discrete Kalman algorithm has been successfully employed in a number of filtering situations, practical applications of the method have often been plagued with numerical difficulties.

Instances of serious accuracy loss in the Kalman filter have been reported by Bellantoni and Dodge [1967], Schmidt et al. [1968], and Dyer and McReynolds [1969]. Computational problems with Kalman's method are often evident in the form of indefinite computed covariance matrices. Loss of covariance positivity is usually the result of computer round-off and cancellation† errors, aggravated by numerical ill-conditioning. Numerical problems may occur, for example, when very accurate measurements are processed in conjunction with large initial error covariances, or when a linear combination of parameters can be precisely estimated while others are relatively unobservable. In these cases computations involving the error covariance matrix are particularly susceptible to round-of and cancellation errors.

A number of schemes have been devised to prevent loss of covariance positivity with the hope that algorithm performance would thereby improve. For example, Schmidt [1967 and 1968] has used artificially large process noise and measurement noise covariances, while Kaminski [1971a] suggests coordinate rotation. Such problem-dependent techniques are nonoptimal, largely empirical, and often cumbersome. Moreover, they are usually inappropriate when precise parameter estimation is required.

The computational shortcomings of Kalman's formula have motivated researchers to derive alternative formulations of his solution. While

†Significant cancellation errors can occur when two nearly equal numbers are differenced. Suppose, for example, that two numbers agree to six digits, and each number is accurate to eight digits. Then the difference would have only two-digit accuracy.

these methods are algebraically equivalent to the Kalman algorithm, they represent different computational techniques designed for improved numerical accuracy. The alternative algorithms usually require more computation than Kalman's original formula, and some methods involve additional computer storage. Hence, many of these methods are not suitable for applications where computer time and storage are limited. This situation is common to many real-time applications, e.g., on-board navigation of aircraft or spacecraft. In the following paragraphs several Kalman filtering techniques are discussed with this kind of application in mind.

Alternative formulations of the Kalman algorithm generally fall into two categories: the information filters and the covariance filters. Information filters recursively compute either the information matrix or one of its square roots.† Covariance-related algorithms, like Kalman's original formula, deal directly with the error covariance or factorizations of this matrix. Square root formulations in each category are acknowledged to be numerically superior to their conventional counterparts. Kaminski [1971b] points out that square root factorization of a filter algorithm improves numerical conditioning and provides greater effective precision.

The square root information filter was introduced by Golub [1965] and Businger and Golub [1965] as a reliable solution to the linear

†The factorization $\Lambda = SS^T$ is not unique. See Appendix A.

least squares problem. This method, based upon Householder transformation techniques (cf Householder [1964]), was later extended by Dyer and McReynolds [1969] to include process noise. Their filter algorithm has been applied extensively in spacecraft navigation and has demonstrated superior numerical characteristics (cf Christensen [1976]).

Analysis by Bierman [1973] and Kaminski [1971a] has shown that square root information filters are particularly efficient for problems involving large batches of data and infrequent estimate calculations. However, they show that covariance-type filters are more appropriate for real-time applications or when measurements are sparse. In these situations measurements are most efficiently processed one at a time (cf Bierman [1973]), in which case the covariance filter is referred to as a point processing algorithm.

Recent research in covariance filtering methods has been stimulated by the need for fast, reliable point processing algorithms. A number of square root covariance methods have been investigated. The Chandrasekhar-type algorithms developed by Kailath [1974], Morf and Kailath [1975], and Lindquist [1974] appear to be efficient square root estimation schemes for stationary processes. These algorithms are not directly applicable, however, to the general nonstationary filtering problem considered in this report. For this reason Chandrasekhar methods are omitted from further discussion.

Several covariance factorization algorithms have been derived to solve the nonstationary filtering problem. Notable among these

is the square root covariance filter, originally introduced by Potter [1963] and later extended by Schmidt [1970]. The Potter-Schmidt algorithm relies on Householder techniques which are known for their numerical stability and accuracy. Although the Potter-Schmidt filter has produced reliable results, Bierman [1973] has shown that this method requires considerably more storage and computation than does the original Kalman algorithm. This inefficiency is related to the fact that Potter's square root is a general nxn matrix, while the Kalman formula involves a symmetric matrix.

Motivated by the need for a more efficient square root covariance filter, Carlson [1973] derived an algorithm which retains the square root in triangular form. Although sometimes less expensive than Potter's method, the n-dimensional Carlson filter requires n square root calculations each time a scalar measurement is processed. Square root calculations are usually time consuming compared to other arithmetic operations. Hence, for many applications Carlson's method is still noticeably more expensive than the Kalman formula.

A promising new approach to Kalman filtering involves a triangular covariance factorization which requires no square roots. The covariance P is factored such that $P = UDU^T$, where U is unit upper triangular and D is diagonal. Bierman [1976a] suggested this factorization and derived a U-D measurement update algorithm. The numerical integrity of this algorithm has been established by the work of Gentleman [1973] and [1975] which relates the U-D measurement update to the numerically stable Givens transformation methods. Moreover, the computational

requirements of the U-D algorithm are equivalent to those of Kalman's method. Because of these attributes the U-D factorization algorithm is ideally suited for real-time applications.

Bierman's results provided the starting point for this research. Attention was directed to the following problem areas:

(1) Extension of the U-D estimation method to allow for time propagation.

(2) Demonstration, by analysis and experimentation, that the U-D factorization filter is a reliable and efficient point-processing algorithm.

(3) Application of U-D filtering techniques to other areas of linear estimation theory.

1.2 Outline of the Contents

Chapters II and III contain a description of the various discrete-time covariance filter algorithms. In Chapter II attention is restricted to recursive data processing methods. The conventional Kalman measurement update formula is presented, and several alternatives to this method are described. These discussions address the computational aspects of each algorithm and conclude with a derivation of Bierman's U-D measurement update formula. Finally, a simple example problem is solved to illustrate the improved performance obtained with the U-D and square root covariance factorization methods.

Chapter III is devoted to time propagation. After the filtering problem is stated and the conventional Kalman solution is presented, several covariance propagation algorithms are described. These algorithms are designed to propagate covariance factors and generally involve orthogonal transformations. Modifications of the familiar Gram-Schmidt and Givens triangularization techniques are used to derive reliable U-D propagation algorithms. The general propagation methods are then adapted to problems involving bias parameters and colored process noise. Exploitation of system structure yields a particularly efficient U-D colored noise time update algorithm. Finally, an example problem is included to illustrate the superior numerical characteristics of the orthogonal transformation methods for covariance propagation.

Chapter IV contains analytical cost comparisons of the various algorithms studied in Chapters II and III. Comparisons are based upon arithmetic operation counts which are weighted to reflect the different execution times required for each calculation. The measurement update algorithms are compared first, followed by the time update cost comparisons. Based upon this analysis the most efficient U-D and square root covariance propagation algorithms are selected. Measurement and time update costs for each method are then combined to yield filter algorithm cost comparisons.

In Chapter V the U-D filtering method is extended to obtain flexible and concise algorithms for performing covariance error analysis. An efficient gain evaluation method, suitable for analyzing the effects of incorrect a priori statistics, is first derived. Analysis is then

extended to allow for evaluation of mismodeled data and state transition matrices. The general error analysis algorithm enjoys certain simplifications when mismodeled bias and process noise parameters are evaluated, and these effects are described. The analysis of neglected bias parameters is also given special attention. In this case the U-D evaluation method allows for efficient sensitivity analysis and variable dimension filtering.

In Chapter VI the various filter algorithms are applied to a realistic planetary navigation problem. Numerical accuracies of the different methods are compared by computing in both double and single precision arithmetic. Double precision results from all algorithms are in close agreement and are used as a reference for comparing the single precision results. Variations are introduced into the system model in order to evaluate algorithm sensitivity to a priori statistics and state dimensionality. Error analysis for this study is performed by applying the U-D gain evaluation method developed in Chapter V.

Chapter VII gives a summary of results and recommendations for further research.

Chapter II. Recursive Data Processing Algorithms

2.1 Problem Statement and Conventional Kalman Solution

In this chapter attention is restricted to the data processing aspects of discrete linear estimation. That is, given the finite sequence of observations, $\{z(i)\}$, we seek the minimum variance estimate of the parameter vector, x, where

$$z(i) = A(i)x + \nu(i) \qquad i = 0,1,2,... \tag{2.1}$$

and $$x \in R_n \qquad z(i) \in R_m$$

The random variables x and $\nu(i)$ are distributed such that

$$x \sim N(\bar{x},\bar{P}) \tag{2.2}$$

$$\nu(i) \sim N(0,R(i)) \qquad R(i) = \mathrm{diag}(r_1,...,r_m) \tag{2.3}$$

$$E\{x\nu(i)^T\} = 0 \tag{2.4}$$

$$E\{\nu(i)\nu(j)^T\} = 0 \qquad i \neq j \tag{2.5}$$

The notation $x \sim N(\bar{x},\bar{P})$ describes x as normally distributed with mean $\bar{x}$, and covariance $\bar{P}$. Without loss of generality we assume the m components of $\nu(i)$ to be uncorrelated. Correlated observations may be uncoupled by the "whitening" process suggested by Andrews [1968]. See Appendix A.

This thesis is primarily concerned with problems where m<<n and where estimates are required frequently during the data processing. Bierman [1973] and Kaminski [1971a] have shown that this kind of estimation problem is most efficiently solved by recursive covariance-type algorithms such as the following Kalman formula.

Let the variable $\hat{x}^{(i-1)}$ denote the minimum variance estimate of x given data $Z_{i-1} = \{z(0), z(1),\ldots,z(i-1)\}$. It has been proven† that $\hat{x}^{(i-1)} = E\{x/Z_{i-1}\}$, where E is the expectation operator and "/" denotes the conditioning. Kalman [1960] derived the following formula for computing $\hat{x}^{(i)}$, given $\hat{x}^{(i-1)}$ and z(i).

$$\hat{x}^{(i)} = \hat{x}^{(i-1)} + K(i)\left(z(i) - A(i)\hat{x}^{(i-1)}\right) \tag{2.6}$$

$$K(i) = \hat{P}(i-1)A(i)^T\left(A(i)\hat{P}(i-1)A(i)^T + R(i)\right)^{-1} \tag{2.7}$$

$$\hat{P}(i) = \hat{P}(i-1) - K(i)A(i)\hat{P}(i-1) \tag{2.8}$$

This recursion has the following initial values.

$$\hat{x}^{(-1)} = \bar{x} \tag{2.9}$$

$$\hat{P}(-1) = \bar{P} \tag{2.10}$$

---

†cf Sage and Melsa [1971].

The vector K(i) is referred to as the Kalman gain and $\hat{P}(i)$ is the error covariance matrix. Thus

$$\hat{P}(i) = E\{(\hat{x}^{(i)} - x)(\hat{x}^{(i)} - x)^T\} \tag{2.11}$$

Proof of this algorithm is given in numerous texts (cf Astrom [1970], or Sage and Melsa [1971]). We will not repeat these well-known proofs here but will, instead, consider the computational aspects of Kalman's algorithm. Bierman [1973] has shown that the Kalman formula is more efficient when the m components of z(i) are processed one at a time. In this case the mxm matrix inversion in Eq. (2.7) reduces to a trivial calculation, and the cost of updating is a linear function of m. Since the measurement errors in each batch are uncorrelated, the data vector z(i) may be included one component at a time by cycling through Eqs. (2.6)-(2.8) m times. This approach is emphasized and clarified by rewriting the Kalman formula as a scalar measurement update algorithm.

For convenience we adopt the notation

$$\tilde{x} = \hat{x}^{(i-1)} \quad \hat{x} = \hat{x}^{(i)} \tag{2.12}$$

$$\tilde{P} = \hat{P}(i-1) \quad \hat{P} = \hat{P}(i) \tag{2.13}$$

The Kalman measurement update may then be written as follows.

Conventional Kalman Scalar Measurement Update

$$\hat{x} = \tilde{x} + K(z - a^T\tilde{x}) \tag{2.14}$$

$$K = \tilde{P}a/(a^T\tilde{P}a + r) \tag{2.15}$$

$$\hat{P} = \tilde{P} - Ka^T\tilde{P} \tag{2.16}$$

Henceforth time dependence is suppressed for notational convenience unless required to avoid confusion. Where recursions are involved we will rely on the superscripts "~" and "^" to denote a priori and a posteriori quantities, respectively.

2.2 Stabilized Kalman Algorithm

Numerical difficulties with the conventional Kalman algorithm prompted Joseph to reformulate the covariance update (cf Bucy and Joseph [1968]). His method, referred to as the stabilized Kalman algorithm, computes $\hat{P}$ in the following way.

$$\hat{P} = (I-Ka^T)\,\tilde{P}\,(I-Ka^T)^T + KrK^T \tag{2.17}$$

This formula can be mechanized as follows.

$$W_1 = I-Ka^T \tag{2.18}$$

$$W_2 = W_1\,\tilde{P} \tag{2.19}$$

$$\hat{P} = W_2 W_1^T + KrK^T \tag{2.20}$$

A second method, due to Bierman [1973], computes $\hat{P}$ using vector outer products in the following way.

$$v_1 = \tilde{P}a \tag{2.21}$$

$$P_1 = \tilde{P} - Kv_1^T \tag{2.22}$$

$$v_2 = P_1 a \tag{2.23}$$

$$\hat{P} = (P_1 - v_2 K^T) + KrK^T \tag{2.24}$$

The first method is not necessarily more reliable than the second one even though $\hat{P}$ in Eq. (2.20) appears to be a positive definite matrix. Furthermore, the original Joseph arrangement requires nearly an order of magnitude more computation than does the Bierman method.† Because of this inefficiency and because there is no proof of improved stability, the first mechanization is omitted from further discussion. Even when the stabilized Kalman formula is implemented by Eqs. (2.21)-(2.24), it involves more than double the arithmetic operations required by the conventional Kalman method.

---

†For the most part, these mechanizations involve addition and multiplication operations. The first method requires $1.5n^3+2n^2+n$ multiplications, while the second involves only $4n^2+4n$ such calculations.

Although generally more reliable than the conventional Kalman algorithm, the stabilized Kalman formula is susceptible to numerical errors. This little publicized fact is demonstrated in Chapter VI.

2.3 Potter Square Root Covariance Algorithm

Potter [1963] observed that numerical problems associated with the Kalman algorithms were often evident in the form of indefinite computed error covariances. In order to avoid such degradation he factored the covariance $\hat{P}$ so that

$$\hat{P} = \hat{S}\hat{S}^T \tag{2.25}$$

and derived an algorithm for recursively computing $\hat{S}$ instead of $\hat{P}$.† Potter noted that the covariance update for scalar measurements can be written as follows

$$\hat{P} = \hat{S}\hat{S}^T = \tilde{P} - Ka^T\tilde{P} = \tilde{S}[I - \frac{1}{\alpha} ff^T]\tilde{S}^T \tag{2.26}$$

where

$$f = \tilde{S}^T a \tag{2.27}$$

$$\alpha = r + f^T f \tag{2.28}$$

---

†The initial factor, $\tilde{S}_0$, may be uniquely defined by applying a Cholesky square root decomposition to $\tilde{P}_0$ (see Appendix A).

By proper algebraic manipulation Potter found that if a constant γ is chosen such that

$$\gamma = \frac{-\lambda}{1 + \sqrt{r\lambda}} \qquad \lambda = 1/\alpha \tag{2.29}$$

then

$$[I-\lambda ff^T] = [I-\gamma\lambda ff^T]\,[I-\gamma\lambda ff^T]^T \tag{2.30}$$

Equations (2.26) and (2.30) imply that $\hat{S}$ may be computed as follows.

$$\hat{S} = \tilde{S}-\gamma Kf^T \tag{2.31}$$

$$K = \lambda\tilde{S}f \qquad \text{(Kalman gain)} \tag{2.32}$$

The following efficient mechanization of the Potter algorithm was suggested by Bierman [1976a].

Potter Square Root Measurement Update

$$f = \tilde{S}^T a \tag{2.33}$$

$$\lambda = 1/(r+f^T f) \tag{2.34}$$

$$\gamma = \lambda/(1+\sqrt{r\lambda}) \tag{2.35}$$

$$\bar{K} = \tilde{S} f \tag{2.36}$$

$$\hat{x} = \tilde{x} + \bar{K}[\lambda(z-a^T\tilde{x})] \tag{2.37}$$

$$\hat{S} = \tilde{S}-(\gamma\bar{K})f^T \tag{2.38}$$

This square root method guarantees positivity of the computed error covariance. It is also numerically better-conditioned than the Kalman algorithms. Bierman [1973] has shown that the Potter algorithm is equivalent to a particular Householder update (cf Appendix B). Householder methods are known for their accuracy and stability, and so this equivalence establishes the numerical integrity of the Potter formula.

Kaminski [1971a] suggests that when problems are ill-conditioned square root filters can provide twice the effective precision of covariance methods. In the case of the Potter algorithm, however, numerical stability is coupled with greater computational expense. Since the Potter update computes a general nxn matrix, $\hat{S}$, his method requires nearly twice as much storage and calculation as the conventional Kalman update.

Because of this inefficiency the Potter algorithm is not attractive for use in many real-time applications.

2.4 Carlson Triangular Square Root Algorithm

Motivated by the stability and accuracy characteristics of square root estimation techniques and by the need for a fast, reliable point processing algorithm, Carlson [1973] derived an alternative square root formula. His method recursively computes an upper triangular covariance square root as follows.†

Carlson Square Root Measurement Update Algorithm

$$f = \tilde{S}^T a \qquad f^T = (f_1, f_2, \ldots f_n) \tag{2.39}$$

$$\alpha_0 = r \qquad \bar{K}_0 = 0 \tag{2.40}$$

For $j = 1,2,\ldots,n$ cycle through Eqs. (2.41)-(2.45).

$$\alpha_j = \alpha_{j-1} + f_j^2 \tag{2.41}$$

$$\beta_j = \sqrt{\alpha_{j-1}/\alpha_j} \tag{2.42}$$

$$v_j = -f_j/\beta_j\alpha_j \tag{2.43}$$

---

†The algorithm given here is a modest rearrangement of Carlson's formula. Bierman [1976b] has observed that the calculation of $\sqrt{\alpha_{j-1}/\alpha_j}$ in Eq. (2.42) may be more accurate than the corresponding computation, $\sqrt{\alpha_{j-1}\alpha_j}$, recommended by Carlson.

$$\dot{S}_j = \beta_j \tilde{S}_j + v_j \bar{K}_{j-1} \tag{2.44}$$

$$\bar{K}_j = \bar{K}_{j-1} + f_j \tilde{S}_j \tag{2.45}$$

where

$$S = [S_1\ S_2 \ldots S_n] \text{ and } S_j = (S_j(1),\ldots,S_j(j),0,\ldots,0)^T$$

The Kalman gain is given by

$$K = \bar{K}_n/\alpha_n \tag{2.46}$$

Thus, the estimate update may be written as follows.

$$\hat{x} = \tilde{x} + \bar{K}_n[(\tilde{z} - a^T\tilde{x})/\alpha_n] \tag{2.47}$$

Derivation of this algorithm is deferred until the next section where it is shown to be an easy consequence of the U-D measurement update.

The Carlson algorithm has a good computational form and enjoys the traits of stability and accuracy generally attributed to square root filters. Although the Carlson formula requires considerably less storage and computation than the Potter square root method, it is still noticeably less efficient than the conventional Kalman algorithm. Unlike the conventional method, the Carlson algorithm requires n square root calculations for each scalar measurement update, and square roots are usually time-consuming operations. This is particularly true for small on-board computers such as the Litton 4516. This computer has 32 bit double

precision arithmetic and requires approximately 1000 μsec to calculate a square root. By contrast it performs an addition in only 4 μsec. A careful comparison of measurement update costs is included in Chapter IV where the Carlson method is shown to be unreasonably expensive for a significant class of problems.

2.5 U-D Covariance Factorization Algorithm

Bierman [1976a] recognized that square root calculations required by the Carlson algorithm are often costly and proposed a square-root-free measurement update scheme. His method employs the covariance factorization

$$P = UDU^T \tag{2.48}$$

where U is unit upper triangular[†] and D is a positive diagonal. It is well known that for symmetric positive definite matrices this factorization exists and is unique (cf Martin et al. [1965]). An algorithm for computing the U-D factors of P is described in Appendix A.

Measurement updating using the U-D factorization preserves the non-negative definite structure of P and is equivalent to the Carlson method without square root computations.

[†]The factor U could also be taken as lower triangular. However, upper triangular factorizations allow for variable dimensioned filtering as described in Chapter V.

The U-D data processing algorithm may be derived by factoring the Kalman update, Eq. (2.16). If $\tilde{P}$ and $\hat{P}$ are both factored according to Eq. (2.48), then

$$\hat{U}\,\hat{D}\,\hat{U}^T = \tilde{U}[\tilde{D} - \frac{1}{\alpha} vv^T]\tilde{U}^T \tag{2.49}$$

where

$$v = \tilde{D}f \qquad v_i = \tilde{d}_i f_i \tag{2.50}$$

$$f = U^T a \tag{2.51}$$

$$\alpha = r + f^T\tilde{D}f = r + \sum_{i=1}^{n} v_i f_i$$

The bracketed term in Eq. (2.49) is positive definite and hence may be factored as $\bar{U}\,\bar{D}\,\bar{U}^T$. Since the product of two unit upper triangular matrices is again unit upper triangular it follows that

$$\hat{U} = \tilde{U}\,\bar{U} \qquad \hat{D} = \bar{D} \tag{2.53}$$

Hence the U-D update rests on the factorization

$$\bar{U}\,\hat{D}\,\bar{U}^T = \tilde{D} - (1/\alpha)\,vv^T \tag{2.54}$$

Bierman has observed that the Agee-Turner matrix factorization in Appendix C may be applied to this problem and results in the following recursion for computing $\bar{U}$ and $\hat{D}$.

With $c_n = -1/\alpha$, cycle through Eqs. (2.55)-(2.57) for $j = n,\ldots,2$.

$$\hat{d}_j = \tilde{d}_j + c_j v_j^2 \tag{2.55}$$

$$\overline{U}_{kj} = c_j v_k v_j / \hat{d}_j \qquad k = 1,\ldots,j-1 \tag{2.56}$$

$$c_{j-1} = c_j \tilde{d}_j / \hat{d}_j \tag{2.57}$$

$$\hat{d}_1 = \tilde{d}_1 + c_1 v_1 \tag{2.58}$$

Equation (2.55) is subject to cancellation errors since it involves the differencing of two positive quantities. However this expression for $\hat{d}_j$ and, hence the entire recursion, can be rewritten as a numerically stable formula.

Let the partial sums, α_j, be defined as

$$\alpha_j = r + \sum_{i=1}^{j} \tilde{d}_i f_i^2 = \alpha_{j-1} + \tilde{d}_j f_j^2 \tag{2.59}$$

Then since $v_j = \tilde{d}_j f_j$, Eqs. (2.55) and (2.57) imply that

$$c_j = -1/\alpha_j \tag{2.60}$$

This value for c_j yields the following expression for $\hat{d}_j$.

$$\hat{d}_j = \tilde{d}_j \left(\frac{\alpha_{j-1}}{\alpha_j}\right) \qquad j = 1,2,\ldots,n \tag{2.61}$$

Eqs. (2.56)-(2.61) imply that

$$\overline{U} = I + [0 | \lambda_2 v^{(1)} | \lambda_3 v^{(2)} | \ldots | \lambda_n v^{(n-1)}] \tag{2.62}$$

where

$$\lambda_j = -f_j / a_{j-1} \tag{2.63}$$

$$\left(v^{(j)}\right)^T = (v_1, v_2, \ldots, v_j, 0, 0, 0) \tag{2.64}$$

The desired factor $\hat{U} = \tilde{U}\,\overline{U}$ may now be constructed from Eq. (2.62). Let $\hat{U}_j$ and $\tilde{U}_j$ denote the jth columns of $\hat{U}$ and $\tilde{U}$ respectively. Then

$$\hat{U}_j = \tilde{U}_j + \lambda_j \overline{K}_{j-1} \tag{2.65}$$

$$\overline{K}_j = \tilde{U} v^{(j)} = \overline{K}_{j-1} + v_j \tilde{U}_j \tag{2.66}$$

$$\overline{K}_0 = 0 \tag{2.67}$$

and

$$\overline{K}_n / a_n = K \quad \text{(Kalman gain)} \tag{2.68}$$

The U-D update is summarized as follows. Given a priori covariance factors $\tilde{U}$ and $\tilde{D}$ and the scalar measurement $z = a^T x + v$ $(E\{v^2\} = r)$, the updated covariance factors $\hat{U}$ and $\hat{D}$ and the Kalman gain, K, are obtained by evaluating the following sequence of equations.

Bierman U-D Measurement Update Algorithm

$$f = \tilde{U}^T a \qquad f^T = (f_1, f_2, \ldots, f_n) \tag{2.69}$$

$$v = \tilde{D} f \qquad v_j = \tilde{d}_j f_j \tag{2.70}$$

$$\bar{K}_1^T = (v_1, \overbrace{0, \ldots, 0}^{n-1}) \tag{2.71}$$

$$\alpha_1 = r + v_1 f_1 \tag{2.72}$$

$$\hat{d}_1 = \left(\frac{r}{\alpha_1}\right) d_1 \tag{2.73}$$

For $j = 2, \ldots, n$ cycle through Eqs. (2.74)-(2.78).

$$\alpha_j = \alpha_{j-1} + v_j f_j \tag{2.74}$$

$$\hat{d}_j = \left(\frac{\alpha_{j-1}}{\alpha_j}\right) \tilde{d}_j \tag{2.75}$$

$$\lambda_j = -f_j / \alpha_{j-1} \tag{2.76}$$

$$\hat{U}_j = \tilde{U}_j + \lambda_j \bar{K}_{j-1} \tag{2.77}$$

$$\bar{K}_j = \bar{K}_{j-1} + v_j \tilde{U}_j \tag{2.78}$$

where

$$\tilde{U} = [\tilde{U}_1, \tilde{U}_2, \ldots \tilde{U}_n] \text{ and } \hat{U} = [\hat{U}_1, \hat{U}_2, \ldots \hat{U}_n]$$

The Kalman gain is given by

$$K = \bar{K}_n / \alpha_n \tag{2.79}$$

The salient feature of Bierman's algorithm is the way in which the updated diagonal $\hat{D}$ is computed. Since the quantities, α_j, are calculated as positive sums (cf Eq. 2.74), cancellation-type errors are avoided and the positivity of $\hat{D}$, and hence, $\hat{P}$, is assured. Furthermore, the elements of $\hat{D}$ may diminish to near-zero without affecting the stability of the algorithm. The numerical integrity of this method is further established by the analysis of Gentleman [1973] and Fletcher and Powell [1974]. Their work relates the U-D algorithm to a numerically stable Givens orthogonal transformation. An efficient Fortran implementation of the U-D measurement update algorithm is included in Appendix D.

Proof of the Carlson algorithm (Section 2.4) is immediate from the U-D formula when the square roots $\tilde{S}$ and $\hat{S}$ are identified as

$$\tilde{S} = \tilde{U}\tilde{D}^{1/2} \qquad \hat{S} = \hat{U}\hat{D}^{1/2} \tag{2.80}$$

Equation (2.80) suggests that the U-D method shares the attributes of accuracy and stability generally associated with square root filtering techniques. This fact is demonstrated by the case study in Chapter VI.

2.6 Example Problem

The following example provided by Bierman [1976b], illustrates the numerical characteristics of the various data processing techniques

described in this chapter. In particular it demonstrates how the conventional Kalman method can produce nonpositive error covariances while the factorization methods yield accurate results.

Consider the problem of estimating x_1 and x_2 from scalar measurements z_1 and z_2 where

$$z_1 = x_1 + \varepsilon x_2 + v_1$$
$$z_2 = x_1 + x_2 + v_2 \tag{2.81}$$

Data errors, v_1 and v_2 are uncorrelated, zero mean random variables with unit variances. The a priori error covariance is assumed to be

$$\tilde{P} = \sigma^2 I$$

where $\sigma = 1/\varepsilon$ and $0 < \varepsilon << 1$. The quantity ε is assumed to be small enough such that computer round-off produces

$$1 + \varepsilon^2 \overset{r}{=} 1 \tag{2.82}$$

This estimation problem is certainly well posed. The observation z_1 provides an accurate measurement of x_1 which, when coupled with the observation z_2, should accurately determine x_2. However, when the various data processing algorithms are applied to this problem several diverse results are obtained.

Let the gain and error covariance associated with the measurement z_1 be denoted as K_1 and P_1 respectively. Similarly the measurement z_2 is associated with K_2 and P_2. Table 2.1 gives the exact solutions for K and P at each step and shows the rounded solutions computed by the various filter mechanizations. Notice how computer round-off errors are evident after the first measurement is processed. The first covariance matrix computed by the conventional Kalman formula has a zero variance. One might expect this result if the exact variance were on the order of ϵ^2 or even ϵ. However, in this case the correct value is approximately $P_{11} = 2$, and so the Kalman error is appreciable. Subsequent processing of data by the conventional formula results in even larger errors. The second computed covariance, for example, has one negative diagonal element and one that is zero. It is also interesting to note that this covariance matrix has off-diagonal elements equal to -1 if the P_{12} element is computed and set equal to P_{21}. On the other hand, if P_{21} is computed directly it has a value of +1. Yet computer implementations of this algorithm typically include calculation of only the upper (or lower) triangular elements.

Although the stabilized Kalman formula performs better than the conventional method, it too computes different values for P_{12} and P_{21} after the second update. The quantities $P_{12} = -1 - 3\epsilon$ and $P_{21} = -1 - 2\epsilon$ could be averaged to give $P_{12} = P_{21} = -1 - (5/2)\,\epsilon$. However, note that the exact answer rounds to $P_{12} = P_{21} = -1 - 3\epsilon$.

All the factorization algorithms discussed here produce accurate results. The U-D and Carlson methods yield nearly identical results and compute rounded covariances equal to the rounded exact answer.

This sample problem illustrates how the Kalman covariance algorithms are prone to numerical errors which can significantly affect filtering accuracy. The factorization methods, on the other hand, avoid critical round-off and cancellation errors and yield reliable results.

Table 2.1. Comparison of Solutions to Example Problem

Filter Mechanization	K_1 Gain	Covariance Factors	P_1* (Covariance)	K_2 (Gain)	Covariance Factors	P_2* (Covariance)
Exact Answer (Rounded)	$\begin{bmatrix}1\\ \epsilon\end{bmatrix}$	-	$\begin{bmatrix}2 & -\sigma\\ -\sigma & \sigma^2\end{bmatrix}$	$\begin{bmatrix}-\epsilon\\ 1+\epsilon\end{bmatrix}$	-	$\begin{bmatrix}1+2\epsilon & -1-3\epsilon\\ -1-3\epsilon & 2+4\epsilon\end{bmatrix}$
Conventional Kalman	$\begin{bmatrix}1\\ \epsilon\end{bmatrix}$	-	$\begin{bmatrix}0 & -\sigma\\ -\sigma & \sigma^2\end{bmatrix}$	$\begin{bmatrix}-\epsilon\\ 1+\epsilon\end{bmatrix}$	-	$\begin{bmatrix}-1 & -1\\ -1 & 0\end{bmatrix}$
Stabilized Kalman (Joseph)	$\begin{bmatrix}1\\ \epsilon\end{bmatrix}$	-	$\begin{bmatrix}2 & \sigma\\ -\sigma & \sigma^2\end{bmatrix}$	$\begin{bmatrix}-\epsilon\\ 1+\epsilon\end{bmatrix}$	-	$\begin{bmatrix}1+2\epsilon & -1-3\epsilon\\ -1-3\epsilon & 2+3\epsilon\end{bmatrix}$
Potter Square Root	$\begin{bmatrix}1\\ \epsilon\end{bmatrix}$	$S_1=\begin{bmatrix}1-\epsilon & -1+\epsilon\\ -1+\epsilon & \sigma\end{bmatrix}$	$\begin{bmatrix}2-4\epsilon & -\sigma\\ -\sigma & \sigma^2\end{bmatrix}$	$\begin{bmatrix}-\epsilon\\ 1+\epsilon\end{bmatrix}$	$S_2=\begin{bmatrix}1-\epsilon & -\epsilon\\ -1+\epsilon & 1\end{bmatrix}$	$\begin{bmatrix}1-2\epsilon & -1+\epsilon\\ -1+\epsilon & 2-2\epsilon\end{bmatrix}$
Carlson Square Root	$\begin{bmatrix}1\\ \epsilon\end{bmatrix}$	$S_1=\begin{bmatrix}1 & -1\\ 0 & \sigma\end{bmatrix}$	$\begin{bmatrix}2 & -\sigma\\ -\sigma & \sigma^2\end{bmatrix}$	$\begin{bmatrix}-\epsilon\\ 1+\epsilon\end{bmatrix}$	$S_2=\begin{bmatrix}\frac{1}{\sqrt{2}} & -\sqrt{2}(1/2+\epsilon)\\ 0 & \sqrt{2}(1+\epsilon)\end{bmatrix}$	$\begin{bmatrix}1+2\epsilon & -1-3\epsilon\\ -1-3\epsilon & 2+4\epsilon\end{bmatrix}$
U-D Factorization	$\begin{bmatrix}1\\ \epsilon\end{bmatrix}$	$U_1=\begin{bmatrix}1 & -\epsilon\\ 0 & 1\end{bmatrix}$ $D_1=(1,\sigma^2)$	$\begin{bmatrix}2 & -\sigma\\ -\sigma & \sigma^2\end{bmatrix}$	$\begin{bmatrix}-\epsilon\\ 1+\epsilon\end{bmatrix}$	$U_2=\begin{bmatrix}1 & -.5(1+\epsilon)\\ 0 & 1\end{bmatrix}$ $D_2=(\frac{1}{2},\ 2+4\epsilon)$	$\begin{bmatrix}1+2\epsilon & -1-3\epsilon\\ -1-3\epsilon & 2+4\epsilon\end{bmatrix}$

*Covariances are not a part of the Potter, Carlson and U-D algorithms.

Chapter III. Filtering for Discrete Linear Processes

3.1 Problem Statement and Conventional Solution

The discrete linear filtering problem is an extension of the parameter estimation problem discussed in Chapter II. Instead of being constant, the parameter x is described as the state of the following linear multistage process.

$$x(i+1) = \Phi(i)x(i) + B(i)w(i) \qquad i = 0,1,2,... \quad (3.1)$$

The array $\Phi(i)$ is a known nxn transition matrix, B(i) is a known nxk matrix and

$$x(0) \sim N(\bar{x},\bar{P}) \quad (3.2)$$

$$w(i) \sim N\Big(0,Q(i)\Big) \quad Q(i) = \text{diag}\,(q_1,..,q_k) \quad (3.3)^{\dagger}$$

$$E\{w(i)w^T(j)\} = 0 \qquad i \neq j \quad (3.4)$$

$$E\{x(0)w^T(i)\} = 0 \qquad \text{for all } i \quad (3.5)$$

At stage i scalar measurements are available and are linearly related to x(i) by the equation

---

†There is no loss of generality in assuming that Q(i) is diagonal.

$$z(i) = a^T(i)x(i) + v(i) \tag{3.6}$$

where

$$v(i) \sim N\Big(0, r(i)\Big) \tag{3.7}$$

$$E\{v(i)v^T(j)\} = 0 \qquad i \neq j \tag{3.8}$$

$$E\{v(i)x^T(j)\} = 0 \qquad \text{for all } i,j \tag{3.9}$$

$$E\{v(i)w^T(j)\} = 0 \qquad \text{for all } i,j \tag{3.10}$$

The optimum solution to this filtering problem is the minimum variance estimate of x(i) given the observations $Z_i = \{z(0),z(1),\ldots,z(i)\}$. Kalman [1960] derived a recursive solution to this problem by combining the data processing formula of Chapter II with an optimal time update algorithm. The complete Kalman filter is summarized as follows.

Conventional Kalman Filter

Time Update

$$\tilde{x}(i+1) = \phi(i)\,\hat{x}(i) \tag{3.11}$$

$$\tilde{P}(i+1) = \phi(i)\,\hat{P}(i)\,\phi^T(i) + B(i)\,Q(i)\,B^T(i) \tag{3.12}$$

Measurement Update

$$\hat{x}(i) = \tilde{x}(i) + K(i)\left(z(i) - a^T(i)\,\tilde{x}(i)\right) \tag{3.13}$$

$$\hat{P}(i) = \tilde{P}(i) - K(i)a^T(i)\tilde{P}(i) \tag{3.14}$$

$$K(i) = \tilde{P}(i)a(i)\Big/\left(a^T(i)\tilde{P}(i)a(i) + r(i)\right) \tag{3.15}$$

This recursion is initialized by the following quantities.

$$\tilde{x}(0) = \bar{x} \qquad \tilde{P}(0) = \bar{P} \tag{3.16}$$

In Chapter II several alternative formulations of Kalman's measurement update algorithm were presented. These data processing algorithms may be combined with appropriate time update schemes to provide alternative mechanizations of the Kalman filter. For example, the stabilized Kalman measurement update, Eq. (2.17), may be substituted for the conventional Kalman formula, Eq. (3.14), to yield the stabilized Kalman filter.

Measurement update algorithms involving covariance factorizations, such as the Potter square root or the Bierman U-D methods, require alternative formulations of the covariance propagation formula, Eq. (3.12). Several techniques for propagating covariance factors are described in the remainder of this chapter.

3.2 RSS Method for Propagating Covariance Factors

Suppose that at each stage of the filtering process the a posteriori covariance, $\hat{P}$, is factored such that

$$\hat{P} = \hat{S}\hat{S}^T \quad \text{or} \quad \hat{P} = \hat{U}\hat{D}\hat{U}^T \tag{3.17}$$

The RSS method for computing the a priori factors $\tilde{S}$ (or $\tilde{U}$-$\tilde{D}$) at the next stage involves "squaring up" $\hat{S}$ (or $\hat{U}$-$\hat{D}$) to obtain $\hat{P}$. The conventional propagation formula Eq. (3.12) is then applied to compute $\tilde{P}$, followed by an appropriate Cholesky decomposition to obtain $\tilde{S}$ (or $\tilde{U}$-$\tilde{D}$).

Carlson [1973] recommends the RSS method as an efficient square root propagation scheme. However, the analysis in Chapter IV demonstrates that this formula enjoys only a minimal savings in computation. Of greater significance is the potential loss of accuracy associated with this method. A major motivation for factoring the Kalman algorithm is to gain increased accuracy due to better numerical conditioning. This advantage can be eliminated by applying the RSS propagation scheme. The example problem in Section 3.7 illustrates why the RSS method is numerically hazardous.

3.3 Gram-Schmidt Propagation Algorithms

Suppose $\hat{P}$ is factored as $\hat{P} = \hat{U}\hat{D}\hat{U}^T$ and let

$$W = \underbrace{[B}_{k} \mid \underbrace{\Phi\hat{U}]}_{n} \Big\} n \tag{3.18}$$

$$\bar{D} = \text{diag}\ (\underbrace{Q}_{k},\ \underbrace{\hat{D}}_{n}) \tag{3.19}$$

The propagation equation $\tilde{P} = \Phi\hat{P}\Phi^T + BQB^T$ may then be written as

$$\tilde{P} = W\bar{D}W^T \tag{3.20}$$

The $\tilde{U}$-$\tilde{D}$ factors of $\tilde{P}$ may be obtained by applying a particular Gram Schmidt orthogonalization to the row vectors of W. This procedure involves the following inner product space.

Let V_n^D denote the vector space of n-tuples over the reals with inner product, $\langle .,. \rangle_D$ where

$$\langle v_i, v_j \rangle_D = v_i^T D v_j, \quad \text{for all } v \text{ in } V_n^D \tag{3.21}$$

$$||v||_D^2 = \langle v, v \rangle_D \tag{3.22}$$

Thus, the vectors v and w are "D-orthogonal" if $\langle v, w \rangle_D = 0$.

Suppose $\{w_i\}_{i=1}^n$ is an independent set of vectors in V_m^D where $n \leq m$. A D-orthogonal set $\{v_i\}$ may be obtained from $\{w_i\}$ by applying the following weighted Gram-Schmidt (WGS) algorithm.

Weighted Gram-Schmidt Orthogonalization

$$v_n = w_n$$

$$v_{n-1} = w_{n-1} - \frac{\langle w_{n-1}, v_n \rangle_D}{||v_n||_D^2} v_n$$

$$\vdots \qquad \vdots \qquad (3.23)$$

$$v_1 = w_1 - \sum_{i=2}^{n} \frac{\langle w_1, v_i \rangle_D}{||v_i||_D^2} v_i$$

This WGS algorithm may be used to compute the $\tilde{U}$-$\bar{D}$ factors of $\tilde{P}$ in the following way. Consider the n independent row vectors of W in Eq. (3.18) as vectors in $V^{\bar{D}}$ where $m = n + k$. If the $\bar{D}$-orthogonal set $\{v_i\}_{i=1}^{n}$ is computed via WGS applied to $\{w_i\}$, then Eqs. (3.23) imply

$$W = \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_n^T \end{bmatrix} = \tilde{U} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_n^T \end{bmatrix} \qquad (3.24)$$

where $\tilde{U}$ is unit upper triangular. Since the vectors $\{v_i\}$ are $\bar{D}$-orthogonal it follows that

$$\tilde{P} = W\bar{D}W^T = \tilde{U}\begin{bmatrix} \|v_1\|^2_{\bar{D}} & & 0 \\ & \|v_2\|^2_{\bar{D}} & \\ 0 & & \cdot \cdot \|v_n\|^2_{\bar{D}} \end{bmatrix}\tilde{U}^T \qquad (3.25)$$

Thus, the elements of $\tilde{U}$ and $\tilde{D}$ are given by

$$\tilde{d}_j = \|v_j\|^2_{\bar{D}} \qquad j = 1,\ldots,n \qquad (3.26)$$

$$\tilde{u}_{ij} = \frac{1}{\tilde{d}_j}\langle w_i, v_j\rangle_{\bar{D}} \qquad i = 1,\ldots,j-1 \qquad (3.27)$$

The orthogonalization used to compute $\tilde{U}$ and $\tilde{D}$ is a modest generalization of the familiar Gram-Schmidt (GS) procedure (cf Noble [1969]) in that inner products are weighted by the $\bar{D}$ matrix and the vectors $\{v_i\}$ are not normalized. The required orthogonalization may also be attained by applying a Modified Gram-Schmidt (MGS) procedure, (cf Lawson and Hanson [1974]). Bjorck [1967], Jordan [1968] and Rice [1966] have investigated the numerical characteristics of GS and MGS. Their studies establish that MGS is more accurate, takes no more arithmetic operations and requires less storage than does GS. Moreover, the works of Bjorck and Jordan show that triangularization using MGS has accuracy that is comparable with the reliable Givens and Householder methods to be described in this chapter. For these reasons the following Modified Weighted Gram-Schmidt (MWGS) algorithm is recommended for U-D time updating.

Modified Weighted Gram-Schmidt Factorization

Given the full rank nxm matrix W with row vectors $\{w_i\}$ and an m-dimensional positive diagonal matrix, $\bar{D}$. The $\tilde{U}$-$\tilde{D}$ factors of $\tilde{P} = W\bar{D}W^T$ are computed as follows.

Evaluate Eqs. (3.28)-(3.30) recursively for j = n, n - 1,...,1.

$$\tilde{d}_j = ||w_j^{(n-j)}||_{\bar{D}}^2 \tag{3.28}$$

$$\left.\begin{aligned} \tilde{u}_{ij} &= \frac{1}{\tilde{d}_j} < w_i^{(n-j)}, w_j^{(n-j)} >_{\bar{D}} \qquad (3.29) \\ w_i^{(n-j+1)} &= w_i^{(n-j)} - \tilde{u}_{ij} w_j^{(n-j)} \qquad (3.30) \end{aligned}\right\} \quad i = 1,\ldots,j-1$$

This recursion begins with the vectors $w_i^{(0)}$ where

$$w_i^{(0)} = w_i \qquad i = 1,\ldots,n$$

This algorithm may be obtained from the WGS results with the easily proven identity

$$v_j = w_j^{(n-j)} \qquad j = 1,\ldots,n \tag{3.31}$$

Although the inclusion of superscripts makes the MWGS algorithm appear rather complicated, this method is easily mechanized. Compactness,

efficiency and simplicity can be achieved by arranging the FORTRAN implementation so that successive w's in Eq. (3.30) are written over one another (see Appendix D).

Gram-Schmidt orthogonalization may also be used to propagate covariance square roots. Let $\tilde{P}$ be given by Eq. (3.12) where $\hat{P} = \hat{S}\hat{S}^T$. A triangular factorization $\tilde{P} = \tilde{S}\tilde{S}^T$ may be obtained by the following MGS algorithm.

Modified Gram-Schmidt Square Root Factorization

Let

$$W = \overbrace{[BQ^{1/2}}^{k} \; \overbrace{\vdots \; \hat{\Phi}\hat{S}]}^{n} \Big\} n \tag{3.32}$$

with row vectors denoted as $\{w_i\}_{i=1}^{n}$. An upper triangular factor, $\tilde{S}$, such that

$$\tilde{S}\tilde{S}^T = WW^T \tag{3.33}$$

may be obtained from the following recursion with inner products defined by

$$\langle w_i, w_j \rangle = w_i^T w_j \tag{3.34}$$

For $j = n,\ldots,1$ cycle through Eqs. (3.35)-(3.39).

$$\beta_j = ||w_j^{(n-j)}||^2 \tag{3.35}$$

$$\tilde{s}_{jj} = \sqrt{\beta_j} \tag{3.36}$$

$$v_j = w_j^{(n-j)}/\tilde{s}_{jj} \tag{3.37}$$

$$\left.\begin{aligned} \tilde{s}_{ij} &= \langle w_i^{(n-j)}, v_j \rangle && (3.38)\\ w_i^{(n-j+1)} &= w_i^{(n-j)} - \tilde{s}_{ij} v_j && (3.39) \end{aligned}\right\} i = 1,2,\ldots,j-1$$

This algorithm is an easy consequence of the MWGS factorization, Eqs. (3.28)-(3.30), and the identities

$$\tilde{D} = I \qquad \tilde{S} = \tilde{U}\tilde{D}^{1/2} \tag{3.40}$$

The MGS formula may be used in conjunction with both the Potter and the Carlson measurement updating methods.

The recursions Eqs. (3.35)-(3.39) may be used to compute a lower triangular $\tilde{S}$ if the indices, i and j, are reordered so that $j = 1,\ldots,n$ and $i = j,\ldots,n$. Since the Potter algorithm does not require $\tilde{S}$ to be upper triangular, this rearrangement of the MGS formula is also appropriate for propagating the Potter square root.

3.4 Givens Transformation Methods for Covariance Propagation

Consider the problem of constructing an mxm orthogonal transformation T such that

$$[W]T = n\left\{[\overbrace{0}^{m-n} \mid \overbrace{W'}^{n}]\right. \tag{3.41}$$

where W' is upper triangular. If W is defined to be

$$W = n\left\{[\overbrace{BQ^{1/2}}^{k} \mid \overbrace{\hat{S}}^{n}]\right. \tag{3.42}$$

then the transformation in Eq. (3.41) represents a covariance square root time update with $W' = \tilde{S}$.

Givens [1959] showed that T could be constructed as a product of n orthogonal transformations, T_j, where

$$T = T_n\, T_{n-1} \cdots T_1 \tag{3.43}$$

Each T_j is designed to zero out the subdiagonal elements of row j. Thus, T_n is constructed such that for $W^{(n)} = W$ we have

$$n\ \{\overbrace{[W^{(n)}]}^{m}T_n = W^{(n-1)} = \begin{matrix} \\ n-1 \\ 1 \end{matrix}\begin{matrix} \overbrace{}^{m-1} & \overbrace{}^{1} \\ \left[\begin{matrix} W' \\ 0\ \ 0\ \ 0\ \ 0 \end{matrix}\right. & \left.\begin{matrix} * \\ * \\ * \\ * \\ * \end{matrix}\right] \end{matrix} \qquad (3.44)^{\dagger}$$

This transformation consists of a sequence of two-dimensional column rotations called Givens transformations. These rotations pivot in succession the last column of the array W with each of the preceding columns. The rotation involving column i is designed to set $W_{ni}^{(n)} = 0$.

In a similar manner T_{n-1} is constructed to zero out the subdiagonal elements of row n-1 by successively pivoting column m-1 of $W^{(n-1)}$ with each of the preceding columns. Notice that this transformation does not alter the last row and column of $W^{(n-1)}$. This process is continued, and after T_{j+1} is applied the array has the form

$$W(T_n \cdots T_{j+1}) = \begin{matrix} \\ j \\ n-j \end{matrix}\begin{matrix} \overbrace{}^{m-n+j} & \overbrace{}^{n-j} \\ \left[\begin{matrix} W'(j) \\ 0 \end{matrix}\right. & \left.\begin{matrix} * & * \\ * & * \\ * & * \\ * & * \\ * & \\ 0 & \end{matrix}\right] \end{matrix} \qquad (3.45)$$

---

†An asterisk is used to denote nonzero, unnamed elements.

Thus, the process is concluded with the transformation T_1.

This triangularization technique involves a sequence of elementary Givens rotations of the following form.

$$\begin{array}{c} \\ j \\ \\ \\ \\ \\ \\ n-j \\ \\ \\ \end{array}\!\!\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_j & y_j \\ 0 & 0 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 0 & 0 \end{bmatrix} T = \begin{array}{c} \\ j \\ \\ \\ \\ \\ \\ n-j \\ \\ \\ \end{array}\!\!\begin{bmatrix} x_1' & y_1' \\ x_2' & y_2' \\ \cdot & \cdot \\ \cdot & \cdot \\ x_{j-1}' & y_{j-1}' \\ 0 & y_j' \\ 0 & 0 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 0 & 0 \end{bmatrix} \tag{3.46}$$

The vector y is the pivotal column and

$$y_j' = \sqrt{x_j^2 + y_j^2} \tag{3.47}$$

$$c = y_j / y_j' \tag{3.48}$$

$$s = x_j / y_j' \tag{3.49}$$

$$x_i' = cx_i - sy_i \tag{3.50}$$

$$y_i' = sx_i + cy_i \tag{3.51}$$

$$i = 1, \ldots, j-1$$

The Givens covariance square root propagation algorithm is summarized below. Superscripts have been suppressed to simplify the notation and to facilitate computer implementation. The symbol ":=" is used instead and denotes replacement in computer storage.

Givens Square Root Factorization Algorithm

Let W be a full rank nx(n+k) matrix with column vectors $\{w_i\}_{i=1}^{n+k}$. An upper triangular factor $\tilde{S}$ such that $\tilde{S}\tilde{S}^T = WW^T$ may be computed as follows.

For j = n,...,1 cycle through Eqs. (3.52)-(3.58).

$$m := k+j \tag{3.52}$$

For i = m-1,...,1 evaluate recursively Eqs. (3.53)-(3.58).

$$w_m'(j) = \sqrt{\left(w_i(j)\right)^2 + \left(w_m(j)\right)^2} \tag{3.53}$$

$$c := w_m(j)/w_m'(j) \tag{3.54}$$

$$s := w_i(j)/w_m'(j) \tag{3.55}$$

$$v := w_i \tag{3.56}$$

$$w_i := cw_i - sw_m \tag{3.57}$$

$$w_m := sv + cw_m \tag{3.58}$$

When this recursion is completed the W array has the form $W = \underbrace{[0}_{k} \,|\, \underbrace{\tilde{S}]}_{n} \Big\} n$ where $\tilde{S}$ is upper triangular.

A lower triangular factor $\tilde{S}$ may be obtained from the recursion Eqs. (3.52)-(3.58) if the indices, i and j, are reordered so that $j = 1,\ldots n$ and $i = j+1,\ldots,m$. At the conclusion of these calculations the W array has the form

$$W = \underbrace{[\tilde{S}}_{n} \,|\, \underbrace{0]}_{k} \Big\} n \tag{3.59}$$

where $\tilde{S}$ is lower triangular.

The Givens triangularization method is well known for its superior numerical characteristics (cf Wilkinson [1965]). Thus, the Givens square root factorization algorithm provides a reliable method for propagating Potter and Carlson covariance factors. However, this method is usually bypassed in favor of the more efficient Gram-Schmidt and Householder time update algorithms. The reader is referred to Chapter IV for detailed cost comparisons.

A modest generalization of the Givens triangularization technique yields a reliable and efficient U-D time update scheme. Recall that U-D factorizations involve the propagation equation $\tilde{P} = W\bar{D}W^T$ where

$$W = \underbrace{[B}_{k} \mid \underbrace{\hat{\Phi}\hat{U}]}_{n} \Big\} n \qquad \text{and} \qquad \bar{D} = \text{diag}\,\underbrace{(Q}_{k},\underbrace{\hat{D})}_{n}$$

Let $\{w_i\}$ denote the column vectors of W and let $\{\bar{d}_i\}$ represent the diagonals of $\bar{D}$. Note that $\tilde{P}$ can be written as

$$\tilde{P} = \bar{W}\bar{W}^T \tag{3.60}$$

where

$$\bar{W} = \left[\sqrt{\bar{d}_1}\, w_1 \mid \sqrt{\bar{d}_2}\, w_2 \mid \ldots \mid \sqrt{\bar{d}_m}\, w_m\right], \quad m=n+k \tag{3.61}$$

The Givens triangularization method may be applied to the array $\bar{W}$ in a way which explicitly accounts for column scaling. That is, an orthogonal transformation T may be constructed such that

$$\underbrace{n}_{m}\{[\bar{W}]T = [\underbrace{0}_{m-n} \mid \underbrace{\sqrt{\bar{d}_1}\,\tilde{u}_1 \mid \ldots \mid \sqrt{\bar{d}_n}\,\tilde{u}_n}_{n}] = [\underbrace{0}_{m-n} \mid \underbrace{\tilde{U}\bar{D}^{1/2}}_{n}] \tag{3.62}$$

where $\tilde{U}$ is unit upper triangular. Gentleman [1973] showed that an elementary Givens rotation, adapted for such column scaling, takes the following form.

$$
\begin{array}{c}
j \\
\\
n-j
\end{array}
\left\{
\begin{bmatrix}
\sqrt{\alpha}\, x_1 & \sqrt{\beta}\, y_1 \\
\vdots & \vdots \\
\sqrt{\alpha}\, x_j & \sqrt{\beta}\, y_j \\
\hline
0 & 0 \\
\vdots & \vdots \\
0 & 0
\end{bmatrix}
\right.
T =
\begin{array}{c}
j \\
\\
n-j
\end{array}
\left\{
\begin{bmatrix}
\sqrt{\alpha'}x'_1 & \sqrt{\beta'}y'_1 \\
\vdots & \vdots \\
\sqrt{\alpha'}x'_{j-1} & \sqrt{\beta'}y'_{j-1} \\
0 & \sqrt{\beta'} \\
\hline
0 & 0 \\
\vdots & \vdots \\
0 & 0
\end{bmatrix}
\right. \tag{3.63}
$$

where

$$\beta' = \alpha x_j^2 + \beta y_j^2 \tag{3.64}$$

$$\alpha' = \beta\alpha/\beta' \tag{3.65}$$

$$\bar{c} = \beta y_j/\beta' \tag{3.66}$$

$$\bar{s} = \alpha x_j/\beta' \tag{3.67}$$

$$x'_i = y_j x_i - x_j y_i \tag{3.68}$$

$$y'_i = \bar{c} y_i + \bar{s} x_i \tag{3.69}$$

$$i = 1, \ldots, j-1$$

These expressions for α', β', x' and y' are easily obtained by applying Eqs. (3.47)-(3.51) with x and y replaced by $\sqrt{\alpha}\, x$ and $\sqrt{\beta}\, y$ respectively.

Note that when the quantities α', β', x' and y' are desired, rather than the vectors $(\sqrt{\alpha'}x')$ and $(\sqrt{\beta'}y')$ this method involves no

square root calculations. Gentleman [1973] noted that when $y_j = 1$, Eqs. (3.64)-(3.69) can be rewritten to avoid a multiplication. Thus

$$\beta' = \beta + \alpha x_j^2 \tag{3.70}$$

$$\alpha' = \beta\alpha/\beta' \tag{3.71}$$

$$\bar{s} = \alpha x_j/\beta' \tag{3.72}$$

$$x_i' = x_i - x_j y_i \tag{3.73}$$

$$y_i' = y_i + \bar{s}x_i' \tag{3.74}$$

This algorithm is referred to as Method B while the formula described by Eqs. (3.64)-(3.69) is denoted as Method A. Both of these methods may be used to solve the U-D time update problem.

The $\tilde{U}$-$\tilde{D}$ factorization of $\tilde{P} = W\bar{D}W^T$ is initiated by denoting as pivotal elements the pair $(\bar{d}_m, w_m)$ where w_m represents the last column of W. The square-root-free Givens transformation defined as Method A is then applied to the pairs $(\bar{d}_{m-1}, w_{m-1})$ and $(\bar{d}_m, w_m)$. As a result of this initial pivot $w_{m-1}'(n) = 0$ and $w_m'(n) = 1$. Thus subsequent transformations involving the mth column may be accomplished by applying Method B. The pair (d_m', w_m') is then pivoted in turn with each of the remaining pairs (d_i, w_i), $i < m-1$. Implicit in this sequence of calculations is the transformation T_n where

$$\bar{W}\, T_n = \begin{array}{c} \\ n-1 \\ 1 \end{array} \overbrace{\begin{array}{|c|c|} \hline \bar{W}' & \begin{array}{c} \sqrt{\tilde{d}_n}\ \tilde{u}_{1n} \\ \sqrt{\tilde{d}_n}\ \tilde{u}_{2n} \\ \vdots \end{array} \\ \hline 0\ 0\ 0\ 0 & \sqrt{\tilde{d}_n} \\ \hline \end{array}}^{m-1 \qquad 1} \tag{3.75}$$

Explicitly, the procedure yields $W^{(n-1)}$ and $\bar{D}^{(n-1)}$ where

$$W^{(n-1)} = \begin{array}{c} \\ n-1 \\ 1 \end{array} \overbrace{\begin{array}{|c|c|} \hline W' & \begin{array}{c} \tilde{u}_{1n} \\ \tilde{u}_{2n} \\ \vdots \end{array} \\ \hline 0\ 0\ 0\ 0 & 1 \\ \hline \end{array}}^{m-1 \qquad 1} \tag{3.76}$$

$$\bar{D}^{(n-1)} = \text{diag}\,(d_1', d_2', \ldots, d_{m-1}', \tilde{d}_n) \tag{3.77}$$

This technique is repeated, with the pair (d_{m-1}', w_{m-1}') as the pivotal elements, to yield $W^{(n-2)}$ and $\bar{D}^{(n-2)}$. The process is thus continued until finally $W^{(0)}$ and $\bar{D}^{(0)}$ are obtained where

$$W^{(0)} = \overbrace{[\,\tilde{0}}^{m-n} \;\vdots\; \overbrace{\tilde{U}]}^{n}\Big\}\, n \tag{3.78}$$

$$\bar{D}^{(0)} = \text{diag}\,(\overbrace{* * * *}^{m-n}, \overbrace{\tilde{d}_1, \tilde{d}_2, \ldots, \tilde{d}_n}^{n}) \tag{3.79}$$

The modified Givens algorithm for propagating U-D covariance factors is summarized below.

Modified Givens Factorization

Suppose we are given the full rank $n\times(n+k)$ matrix W with column vectors $\{w_i\}$ and an $(n+k)$-dimensional positive diagonal matrix, $\bar{D}$. The $\tilde{U}$-$\tilde{D}$ factors of $\tilde{P} = W\bar{D}W^T$ may be computed as follows.

For $j = n,\ldots,1$ cycle through Eqs. (3.80)-(3.89).

$$m := j+k \tag{3.80}$$

For $i = m-1,\ldots,1$ evaluate Eqs. (3.81)-(3.89) as indicated.

$$d_m' := \bar{d}_m\, w_m^2(j) + \bar{d}_i\, w_i^2(j) \tag{3.81}^{\dagger}$$

$$\bar{s} := \bar{d}_i\, w_i(j)/d_m' \tag{3.82}$$

$$\bar{d}_i := \bar{d}_i \bar{d}_m/d_m' \tag{3.83}$$

---

†When $i < m-1$, $w_m(j) \equiv 1$.

$$\bar{c} := \bar{d}_m \, w_m(j)/d'_m \qquad \text{omit if } i < m-1 \tag{3.84}$$

$$v := w_i \tag{3.85}$$

$$w_i := w_m(j)\, w_i - w_i(j) w_m \tag{3.86}$$

Method A: $$w_m := \bar{c} w_m + \bar{s} v \qquad \text{if } i = m-1 \tag{3.87}$$

Method B: $$w_m := w_m + \bar{s} w_i \qquad \text{if } i < m-1 \tag{3.88}$$

$$\bar{d}_m = d'_m \tag{3.89}$$

Upon completion of this recursion the W array contains $\tilde{U}$. That is

$$W = \underbrace{[0}_{k} \mid \underbrace{\tilde{U}]}_{n} \}n \tag{3.90}$$

The diagonal elements of $\tilde{D}$ are given by

$$\tilde{d}_i = \bar{d}_{k+i} \qquad i = 1,\ldots,n \tag{3.91}$$

Analysis by Gentleman [1973] has established that when Method A, Eq. (3.87), is used exclusively in this recursion the algorithm is always numerically stable. However, use of Method B, as indicated in Eq. (3.88), saves approximately 1/3 n^3 multiplications (33%). Hence this formula is preferred when it can be relied upon. Gentleman [1973] has shown that

errors associated with Method B can be related to the ratio $d_m'/\bar{d}_m$. He recommends control of error growth by testing this ratio against some limiting value, c. Thus, Method B is applied only when $d_m'/\bar{d}_m < c$. Fletcher and Powell [1974] have experimented with this technique on a large number of problems using c = 4.† Their experience suggests that for most applications of this algorithm Method A is required so rarely that efficiency is near optimum. A numerically reliable computer implementation of the modified Givens update is included in Appendix D.

3.5 Householder Methods for Covariance Propagation

Recall that propagation of covariance square roots involves the triangularization problem,

$$\underset{\sim}{n}\left\{\overset{\sim}{[W]}\ T = \underset{\sim}{n}\left\{\overset{\sim}{[}\ \underset{\sim}{0}\ \overset{\sim}{|}\ \underset{\sim}{\tilde{S}}\overset{\sim}{]}\right.\right.$$

where $\tilde{S}$ is upper triangular and T is an mxm orthogonal transformation. This problem may be solved by applying a sequence of Householder transformations, T_j, such that $T = T_n T_{n-1} \ldots T_1$ (cf Householder [1964]). Each elementary transformation is a reflection operator and may be written as

$$T_j = I - \beta_j\, u^{(j)} u^{(j)T} \tag{3.92}$$

where

$$\beta_j = 2/||u^{(j)}||^2 \tag{3.93}$$

---

†Gentleman suggests c = 10 as a reasonable limit.

Note that for any vector y

$$y^T T_j = y^T - \gamma\left(u^{(j)}\right)^T \tag{3.94}$$

with γ given by

$$\gamma = \beta_j\left(y^T u^{(j)}\right) \tag{3.95}$$

Thus, explicit formation of the matrix T_j is not required if only the product $W^{(j)}\ T_j$ is desired.

The triangularization of W is begun by designing the transformation T_n to zero out the subdiagonal elements of row n. Thus,

$$[W]T_n = W^{(n-1)} = \begin{array}{c} \\ n-1 \\ \\ 1 \end{array} \overbrace{\left[\begin{array}{c|c} W' & \begin{array}{c} v_1 \\ v_2 \\ \vdots \\ v_{r-1} \end{array} \\ \hline 0\ 0\ 0\ 0 & -\sigma_n \end{array}\right]}^{m-1 \quad 1} \tag{3.96}$$

Equations (3.92)-(3.96) imply that T_n can be constructed from the vector

$$u^{(n)} = x^{(n)} + \sigma_n e_m \tag{3.97}$$

where $x^{(n)}$ is the nth row of the array W and

$$e_m^T = (\overbrace{0,0,0,0}^{m-1},1) \tag{3.98}$$

Since T_n is orthogonal, σ_n is constrained such that $\sigma_n^2 = ||x^{(n)}||^2$. Thus Eqs. (3.93) and (3.97) imply that

$$\beta_n = 1/(\sigma_n^2 + \sigma_n x_m^{(n)}) \tag{3.99}$$

In order to avoid cancellation errors when β_n is calculated we may define

$$\sigma_n = \text{sgn}\,(x_m^{(n)})||x^{(n)}|| \tag{3.100}$$

Once T_n has been applied to the array W via Eq. (3.94), the transformation T_{n-1} is similarly chosen to pack the (n-1)st row of the array $W^{(n-1)}$. Construction of $u^{(n-1)}$ is identical to the construction of $u^{(n)}$ with one exception. The mth component of $u^{(n-1)}$ is free to be chosen since there are only m-2 subdiagonal elements to be zeroed out. If the mth component of $u^{(n-1)}$ is set to zero then $(0\ 0\ldots0,-\sigma_n)\ u^{(n-1)} = 0$. Thus, Eqs. (3.94)-(3.95) imply that T_{n-1} will not disturb the nth row or the mth column of the array. Hence,

$$
\begin{array}{c}
\overbrace{}^{m-1}\ \overbrace{}^{1} \\
\left.\begin{array}{|c|c|}\hline W' & \begin{array}{c} v_1 \\ v_2 \\ \vdots \\ v_{n-1} \end{array} \\ \hline 0\ 0\ 0\ 0 & -\sigma_n \\ \hline \end{array}\right\} \begin{array}{l} n-1 \\ 1 \end{array}
\end{array}
\quad T_{n-1} = \quad
\begin{array}{c}
\overbrace{}^{m-2}\ \overbrace{}^{2} \\
\left.\begin{array}{|c|cc|}\hline & * & v_1 \\ W'' & * & v_2 \\ & * & \vdots \\ \hline 0\ 0\ 0\ 0 & -\sigma_{n-1} & v_{n-1} \\ 0\ 0\ 0\ 0 & 0 & -\sigma_n \\ \hline \end{array}\right\} \begin{array}{l} n-2 \\ 2 \end{array}
\end{array}
\qquad (3.101)
$$

This process is continued so that $u^{(j)}$ at step j is constructed according to Eq. (3.97) with n replaced by j and m replaced by (m-n+j). The m-dimensional vector $x^{(j)}$ is defined as follows. For j = n,...,1

$$
x_i^{(j)} = \begin{cases} w_{ji}^{(j)} & i = 1,\ldots,j+m-n \\ 0 & i > j+m-n \end{cases} \qquad (3.102)
$$

where $w_j^{(j)}$ is the jth row vector of $W^{(j)}$.

The following algorithm, due originally to Schmidt [1970], applies the Householder triangularization method to propagate covariance square roots.

Schmidt Covariance Square Root Time Update

Let $\hat{P} = \hat{S}\hat{S}^T$ and define W to be the nx(n+k) array

$$
W = \left.\overbrace{[BQ^{1/2}}^{k} \;\vdots\; \overbrace{\Phi\hat{S}]}^{n}\right\} n
$$

with elements $\{w_{ij}\}$. An upper triangular factorization, $\tilde{S}\tilde{S}^T = WW^T$, may be obtained as follows.

For $j = n, n-1,\ldots,1$ compute recursively Eqs. (3.103)-(3.111).

$$\sigma_j = \left(\sum_{i=1}^{j+k} (w_{ji})^2\right)^{1/2} \operatorname{sgn}(w_{j,j+k}) \tag{3.103}$$

$$u_i := \begin{cases} w_{ji} & i = 1,\ldots,j+k-1 \quad (3.104) \\ w_{j,j+k} + \sigma_j & i = j+k \quad (3.105) \\ 0 & i = j+k+1,\ldots,n+k \quad (3.106) \end{cases}$$

$$\beta_j = 1/(\sigma_j u_{j+k}) \tag{3.107}$$

For $i=1,2,\ldots,j-1$ cycle through Eqs. (3.108)-(3.109).

$$\gamma_i := \beta_j \sum_{\ell=1}^{j+k} u_\ell w_{i\ell} \tag{3.108}$$

$$w_{i\ell} := w_{i\ell} - \gamma_i u_\ell \qquad \ell=1,\ldots,j+k \tag{3.109}$$

$$w_{j,j+k} := -\sigma_j \tag{3.110}$$

$$w_{j\ell} := 0 \qquad \ell = 1,\ldots j+k-1 \tag{3.111}$$

At the conclusion of this recursion the array W has the form

$$W = \underbrace{[0}_{k} \; \underbrace{\vdots \; \tilde{S}]}_{n} \Big\} \; n$$

where $\tilde{S}$ is upper triangular.

The Householder triangularization method can be modified to yield a U-D propagation algorithm by applying the identity $S = UD^{1/2}$ to Eqs. (3.103)-(3.111). However, this method is not pursued since it appears to have little advantage over the MWGS and modified Givens algorithms.

3.6 Propagation Algorithms for Systems with Colored Process Noise

Time propagation represents a major filtering expense, particularly in applications where measurements are sparse (cf Chapter IV). Hence, time updating methods warrant further consideration in terms of efficient computer implementation. It is often possible to significantly reduce propagation costs by exploiting special system structure. For example, one may trim computations considerably by taking advantage of sparse state transition matrices or triangular system equations. This section illustrates how triangular covariance factorizations lend themselves to efficient propagation schemes for systems with bias parameters and colored process noise. Suppose the state transition is given by

$$\begin{bmatrix} x \\ p \end{bmatrix}_{i+1} = \begin{bmatrix} \Phi_x & \Phi_{xp} \\ 0 & M \end{bmatrix} \begin{bmatrix} x \\ p \end{bmatrix}_i + \begin{bmatrix} 0 \\ w \end{bmatrix}_i \qquad i=0,1,\ldots \qquad (3.112)$$

where

$$M = \text{diag}\,(m_1,\ldots,m_k) \tag{3.113}$$

$$E\{w_i\, w_i^T\} = Q \qquad Q = \text{diag}\,(q_1,\ldots,q_k) \tag{3.114}$$

$$E\{x_0 w_i^T\} = 0 \qquad E\{p_0 w_i^T\} = 0$$

This system model closely approximates a large class of second-order processes and hence has wide application (cf Christensen [1976] or Maughmer and Byrd [1969]). The structure of Eq. (3.112) permits time propagation to be performed in two phases. Thus,

$$\begin{bmatrix}\bar{x}\\ \bar{p}\end{bmatrix}_{i+1} = \begin{bmatrix}\Phi_x & \Phi_{xp}\\ 0 & I\end{bmatrix}\begin{bmatrix}x\\ p\end{bmatrix}_i \tag{3.115}$$

$$\begin{bmatrix}x\\ p\end{bmatrix}_{i+1} = \begin{bmatrix}I & 0\\ 0 & M\end{bmatrix}\begin{bmatrix}\bar{x}\\ \bar{p}\end{bmatrix}_{i+1} + \begin{bmatrix}0\\ w\end{bmatrix}_i \tag{3.116}$$

Equation (3.115) defines a deterministic phase of the mapping. The error covariance associated with this phase is denoted as $\bar{P}$, while the final map, Eq. (3.116) yields the a priori error covariance, $\tilde{P}$.

---

†For notational brevity we use subscripts in Eqs. (3.112) - (3.116) to denote time dependence of system parameters. Notice further that the matrices Φ_x, Φ_{xp}, M and Q may also be time-varying.

The original problem is decomposed in this manner because each subproblem enjoys certain computational advantages, particularly when upper triangular covariance factorizations are involved.

3.6.1 U-D Colored Noise Time Update

Consider the covariance factorization $P = UDU^T$. The simplifications associated with a deterministic map of U and D via Eq. (3.115) are illustrated in the following algorithm.

Bias Partitioned U-D Factorization

Let $P = UDU^T$ where U and D are given by

$$U = \begin{bmatrix} \overset{n}{\sim} & \overset{k}{\sim} \\ U_x & U_{xp} \\ 0 & U_p \end{bmatrix} \begin{matrix} \}n \\ \}k \end{matrix} \qquad D = \operatorname{diag}\left(\overset{n}{\sim}\, D_x,\ \overset{k}{\sim}\, D_p\right) \tag{3.117}$$

Assume $\bar{P}$ is given by

$$\bar{P} = \Phi P \Phi^T \tag{3.118}$$

where

$$\Phi = \begin{bmatrix} \overset{n}{\sim} & \overset{k}{\sim} \\ \Phi_x & \Phi_{xp} \\ 0 & I \end{bmatrix} \begin{matrix} \}n \\ \}k \end{matrix} \tag{3.119}$$

Then the $\bar{U}$-$\bar{D}$ factors of $\bar{P}$, with similar partitioning assumed, are determined as follows.

$$[\bar{U}_p, \bar{D}_p] = [U_p, D_p] \tag{3.120}$$

$$\bar{U}_{xp} = \Phi_x U_{xp} + \Phi_{xp} U_p \tag{3.121}$$

$$\bar{U}_x \bar{D}_x \bar{U}_x^T = (\Phi_x U_x) D_x (\Phi_x U_x)^T \tag{3.122}$$

Equations (3.120)-(3.122) may be obtained by substituting the partitioned U-D and Φ factors into Eq. (3.118) and expanding the result. Note that U_x and D_x may be computed by applying either the MWGS algorithm, Eqs. (3.28)-(3.30), or the modified Givens algorithm, Eqs. (3.80)-(3.89), to the arrays $W = \Phi_x U_x$ and $\bar{D} = D_x$.

An easy generalization of this partition algorithm involves replacement of the identity matrix in Eq. (3.119) with a nonsingular, upper triangular matrix, Φ_p. The modifications required for Eqs. (3.120) and (3.121) are apparent.

The following features of the bias partition algorithm are notable.

1) The U-D factors corresponding to the bias portion of the state vector remain constant. This property is contingent upon having the bias parameters in the lower portion of the state vector.

2) The cross coupling $\bar{U}_{xp}$ is updated by a simple linear relation.

3) The updated factors $\bar{U}_x$ and $\bar{D}_x$ are obtained by considering only that portion of the problem which is independent of p.

4) All of the above comments also apply to problems where x is a stochastic process. In this case the right hand side of Eq. (3.122) would have an additional term corresponding to the process noise disturbance.

Once the deterministic time update, Eq. (3.115), is accomplished with updated covariance factors, $\bar{U}$ and $\bar{D}$, computed from Eqs. (3.120)-(3.122), the second phase of the mapping, Eq. (3.116), is performed. The diagonal structure of M and Q may be exploited by mapping the process noise, p, one component-at-a-time. The following algorithm indicates how the intermediate factors, $\bar{U}$ and $\bar{D}$, are updated as each component of p is mapped.

Single Component U-D Time Update

Let $\bar{P} = \bar{U}\,\bar{D}\,\bar{U}^T$ with $\bar{U}$ and $\bar{D}$ given by

$$\bar{U} = \begin{matrix} \\ \\ \\ \end{matrix} \begin{matrix} \underbrace{}_{n_a} & \underbrace{\phantom{U_{ab}}}_{1} & \underbrace{\phantom{U_{ac}}}_{n_c} \\ \end{matrix} \begin{bmatrix} U_a & U_{ab} & U_{ac} \\ 0 & 1 & U_{bc} \\ 0 & 0 & U_c \end{bmatrix} \begin{matrix} \} n_a \\ \} 1 \\ \} n_c \end{matrix} \qquad \bar{D} = \text{Diag}\,(\underbrace{D_a}_{n_a},\ \underbrace{d}_{1},\ \underbrace{D_c}_{n_c}) \tag{3.123}$$

Let Φ = Diag (I, m, I) and Q = diag (0, q, 0) be dimensioned compatibly with (3.123) and assume $\tilde{P}$ is given by the following expression.

$$\tilde{P} = \Phi \bar{P} \Phi^T + Q \tag{3.124}$$

Then the U-D factors of $\tilde{P}$, assumed to be partitioned consistently with (3.123), are determined as follows.

$$[\tilde{U}_{ac}, \tilde{U}_c, \tilde{D}_c] = [U_{ac}, U_c, D_c] \tag{3.125}$$

$$\tilde{d} = m^2 d + q \tag{3.126}$$

$$\tilde{U}_{bc} = m U_{bc} \tag{3.127}$$

$$\tilde{U}_{ab} = m \frac{d}{\tilde{d}} U_{ab} \tag{3.128}$$

The matrices $\tilde{U}_a$ and $\tilde{D}_a$ satisfy the following relation.

$$\tilde{U}_a \tilde{D}_a \tilde{U}_a^T = U_a D_a U_a^T + \left(\frac{d}{\tilde{d}} q\right) U_{ab} U_{ab}^T \tag{3.129}$$

Equations (3.125)-(3.128) are obtained by direct substitution of the partitioned $\bar{U}$-$\bar{D}$ factors into (3.124) and by equating this expression with $\tilde{U} \tilde{D} \tilde{U}^T$. Equation (3.129) is then easily derived with the aid of (3.126) and (3.128). The factorization required in Eq. (3.129) may be obtained by applying the Agee-Turner triangular factorization algorithm (Appendix C).

This single component update algorithm is the cornerstone of the U-D colored noise updating procedure.

U-D Colored Noise Updating Algorithm

Let $\bar{P} = \bar{U}\ \bar{D}\ \bar{U}^T$ where $\bar{U}$ and $\bar{D}$ are determined by equations (3.120)-(3.122) for the deterministic time update, (3.115). The U-D covariance factors after the final propagation (3.116) can be computed as follows.

Let n and k denote the dimensions of x and p respectively. For $\ell = 1,2,\ldots,k$, compute recursively equations (3.130)-(3.133).

$$\tilde{d}_{n+\ell} = m_\ell^2\ \bar{d}_{n+\ell} + q_\ell \tag{3.130}$$

$$\left.\begin{aligned} v_i &= \bar{u}_{i,n+\ell} \\ \tilde{u}_{i,n+\ell} &= m_\ell \frac{\bar{d}_{n+\ell}}{\tilde{d}_{n+\ell}}\ v_i \end{aligned}\right\} \quad i = 1,2,\ldots,n+\ell-1 \tag{3.131}$$

$$\tilde{u}_{n+\ell,j} = m_\ell\ \bar{u}_{n+\ell,j} \qquad j = n+\ell+1,\ldots,n+k \tag{3.132}$$

Use the Agee-Turner triangular factorization (Appendix C) to compute the U-D factors of (3.133).

$$U^{(\ell)}\ \tilde{D}^{(\ell)}\ \tilde{U}^{(\ell)T} := \tilde{U}^{(\ell)}\ \tilde{D}^{(\ell)}\ \tilde{U}^{(\ell)T} + c_\ell\ v\ v^T \tag{3.133}$$

where $\tilde{U}^{(\ell)}$ and $\tilde{D}^{(\ell)}$ denote the upper left $(n+\ell-1)$ rows and columns of the $\tilde{U}$, $\tilde{D}$ matrices and

$$c_\ell = \frac{\bar{d}_{n+\ell}}{\tilde{d}_{n+\ell}} q_\ell$$

This result is an immediate consequence of the single component time update, equations (3.125)-(3.129). Notice that (3.130)-(3.133) correspond to (3.126)-(3.129) respectively.

This one-at-a-time procedure also applies to the general class of problems involving bias parameters. If the state vector is partitioned as

$$X = \begin{bmatrix} x \\ p \\ y \end{bmatrix}$$

with bias parameters in the lowest portion, then the algorithms of this section may be applied mutatis-mutandis. If y has dimension b then Eq. (3.132) must be dimensioned so that all b columns corresponding to y are properly scaled; i.e., Eq. (3.132) would be replaced by

$$\tilde{u}_{n+\ell,j} = m_\ell \bar{u}_{n+\ell,j} \qquad j = n+\ell+1,\ldots,n+k+b$$

3.6.2 Square Root Covariance Colored Noise Time Update

Suppose the state error covariance for the system defined by Eq. (3.112) is factored as $P = SS^T$. If S is upper triangular it may

be propagated by applying a simple modification of the mapping procedure developed in the previous section. The triangular square root colored noise update is summarized in the following algorithm.

Triangular Square Root Colored Noise Updating Algorithm

Let $P = SS^T$ where S is partitioned such that

$$S = \begin{bmatrix} \overset{n}{S_x} & \overset{k}{S_{xp}} \\ 0 & S_p \end{bmatrix} \begin{matrix} \} n \\ \} k \end{matrix} \tag{3.134}$$

The propagation of S via Eq. (3.112) may be performed in two phases as follows. A deterministic map, Eq. (3.115), yields the intermediate covariance factor $\bar{S}$, assumed to be partitioned consistently with (3.134), where

$$\bar{S}_p = S_p \tag{3.135}$$

$$\bar{S}_{xp} = \Phi_x S_{xp} + \Phi_{xp} S_p \tag{3.136}$$

$$\bar{S}_x \bar{S}_x^T = (\Phi_x S_x)(\Phi_x S_x)^T \tag{3.137}$$

The upper triangular factor $\bar{S}_x$ may be computed by applying the Schmidt time update algorithm†, Eqs. (3.103)-(3.111), to the array $W = \Phi_x S_x$.

The final map, Eq. (3.116), yields the covariance square root, $\tilde{S}$, as follows.

For $\ell = 1,2,\ldots,k$ compute recursively Eqs. (3.138)-(3.144).

$$j = n+\ell \tag{3.138}$$

$$\tilde{s}_{jj} = \sqrt{(m_\ell \bar{s}_{jj})^2 + q_\ell} \tag{3.139}$$

$$v_i = \bar{s}_{ij} \tag{3.140}$$

$$\tilde{s}_{ij} = \left(\frac{m_\ell s_{jj}}{\tilde{s}_{jj}}\right) \bar{s}_{ij} \qquad \left.\begin{array}{l} \\ \\ \end{array}\right\} i = 1,2,\ldots,j-1 \tag{3.141}$$

$$\tilde{s}_{ji} = m_\ell \bar{s}_{ji} \qquad i = j+1,\ldots,n+k \tag{3.142}$$

$$c_\ell = q_\ell/(\tilde{s}_{jj})^2 \tag{3.143}$$

†The MGS algorithm, Section 3.3, or the Givens square root factorization, Section 3.4, could also be used to compute $\bar{S}_x$. However, the Schmidt formula is preferred because it is more efficient than the other methods (cf Chapter IV).

Apply the Agee-Turner square root factorization (Appendix C) to Eq. (3.144) where $\tilde{S}^{(\ell)}$ denotes the upper left $(n+\ell-1)$ rows and columns of the array $\tilde{S}$.

$$\tilde{S}^{(\ell)}\,\tilde{S}^{(\ell)T} := \tilde{S}^{(\ell)}\,\tilde{S}^{(\ell)T} + c_\ell\, vv^T \tag{3.144}$$

This algorithm follows directly from the U-D colored noise time update formula by applying the identity $S = UD^{1/2}$.

Consider the factorization $P = SS^T$ where S is the general matrix

$$S = \begin{bmatrix} \overbrace{S_x}^{n} & \overbrace{S_{xp}}^{k} \\ S_{px} & S_p \end{bmatrix} \begin{matrix} \} n \\ \} k \end{matrix} \tag{3.145}$$

Covariance factorizations of this type are computed, for example, by the Potter measurement update algorithm, Eqs. (2.33)-(2.38). Note that the presence of $S_{px} \neq 0$ introduces considerably more computation into the propagation of S via Eq. (3.112). In order to apply the one-at-a-time mapping procedure, Eqs. (3.138)-(3.144), it is first necessary to triangularize the full (n+k) x (n+k) array

$$S' = \left[\begin{array}{c|c} \overbrace{\Phi_x S_x + \Phi_{xp} S_{px}}^{n} & \overbrace{\Phi_x S_{xp} + \Phi_{xp} S_p}^{k} \\ \hline S_{px} & S_p \end{array}\right] \begin{matrix} \} n \\ \} k \end{matrix} \tag{3.146}$$

corresponding to the deterministic map in Eq. (3.115).† For this reason, the one-at-a-time procedure is not an efficient method for propagating general covariance factors.

3.7 Example Problem

Let the two-dimensional state transition be defined by

$$\Phi = I \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad \text{and} \qquad Q = 1 \tag{3.147}$$

If the a posteriori covariance $\hat{P}$ is propagated according to Eqs. (3.12) and (3.147) then

$$\tilde{P} = \hat{P} + \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.148}$$

Suppose $\hat{P}$ is factored such that

$$\hat{P} = \hat{S}\,\hat{S}^T \qquad \text{or} \qquad \hat{P} = \hat{U}\,\hat{D}\,\hat{U}^T \tag{3.149}$$

where

$$\hat{S} = \begin{bmatrix} 1 & \sigma \\ 0 & \sigma \end{bmatrix} \quad \text{or} \quad \hat{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \qquad \hat{D} = \text{diag}\,(1, \sigma^2)$$

Assume $\sigma \gg 1$ and $\sigma^2 + 1 \overset{r}{=} \sigma^2$.

---

†This observation implies that propagation of the Potter square root for noisy systems with large numbers of bias parameters is considerably more costly than the corresponding U-D and triangular square root bias-partitioned updates (cf Eqs. (3.120)-(3.122) and (3.135)-(3.137)).

Table 3.1 shows how the various propagation algorithms compute the a priori covariance factor $\tilde{S}$ (or $\tilde{U}$-$\tilde{D}$). Notice that all methods except the RSS formula yield an $\tilde{S}$ (or $\tilde{U}$-$\tilde{D}$) equivalent to the rounded exact answer. The RSS formula, however, suffers a loss of accuracy by applying Eq. (3.148) directly and computes a singular $\tilde{S}$ with $\tilde{s}_{11} = 0$. Similarly the U-D update yields $\tilde{d}_1 = 0$.

This simple example illustrates how use of the RSS propagation scheme can result in significant loss of filtering accuracy and possible failure of the algorithm. The other propagation algorithms, however, are known to be numerically reliable (cf sections 3.3-3.6) and thus are preferred over the RSS method.

Table 3.1. Comparison of Solutions to Example Propagation Problem

Exact Answer (Rounded)	$\tilde{S} = \begin{bmatrix} \sqrt{2} & \sigma \\ 0 & \sigma \end{bmatrix}$	$\tilde{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $\tilde{D} = (2,\sigma^2)$
RSS	$\tilde{S} = \begin{bmatrix} 0 & \sigma \\ 0 & \sigma \end{bmatrix}$	$\tilde{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $\tilde{D} = (0,\sigma^2)$
MGS	$\tilde{S} = \begin{bmatrix} \sqrt{2} & \sigma \\ 0 & \sigma \end{bmatrix}$	$\tilde{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $\tilde{D} = (2,\sigma^2)$
Givens	$\tilde{S} = \begin{bmatrix} \sqrt{2} & \sigma \\ 0 & \sigma \end{bmatrix}$	$\tilde{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $\tilde{D} = (2,\sigma^2)$
Schmidt (Householder)	$\tilde{S} = \begin{bmatrix} -\sqrt{2} & -\sigma \\ 0 & -\sigma \end{bmatrix}$	--

Chapter IV. Cost Comparisons of Filter Algorithms

4.1 Basis for Comparison

In this chapter the various algorithms described in Chapters II and III are compared in terms of computer costs. Comparisons are based upon the arithmetic operations required by each method. Computer-related costs, such as indexing, storage transfers and input-output operations, are difficult to quantify and are neglected. However, these costs are common to all the algorithms being considered.

Various arithmetic operations require different execution times, and these differences are computer dependent. The cost comparisons considered here are based primarily upon the approximate execution times of the Hewlett Packard HP21MX-20 computer. This computer is to be used in the NAVSTAR Global Positioning System (GPS) where it will be employed for on-board aircraft navigation (cf General Dynamics-Electronics [1975]). The HP computer, using single precision.† floating point arithmetic, has the following approximate operation times.

$\tau_+ = 34$ μsec $\qquad$ $\tau_\times = 57$ μsec

$\tau_\div = 60$ μsec $\qquad$ $\tau_{\surd} = 2400$ μsec

†Single precision arithmetic on the HP computer involves a 32 bit characteristic, or approximately 9 decimal digits.

Cost comparisons are simplified if these times are normalized by the add time to obtain $\tau_{\times} = 1.7\ \tau_{+}$, $\tau_{\div} = 1.8\ \tau_{+}$, and $\tau_{\surd} = 70\ \tau_{+}$. These factors are used to weight the operation counts of each algorithm being compared.† Thus, comparisons are based upon the relative execution times of the various algorithms.

Other computers may assign somewhat different weights to the arithmetic operations. In most cases, however, the algorithms will have the same efficiency ratings, although cost differentials may vary from one computer to another. This point is illustrated by including additional cost comparisons based upon the approximate execution times of the UNIVAC 1108 and Litton 4516 computers. The UNIVAC model is used in large, ground-based systems; while the Litton 4516 is a small, fixed-point computer, typical of those found on board many commercial and military aircraft. Hence, this cost analysis corresponds to a variety of computing situations. The weights associated with each computer involved in the study are compared in Table 4.1.

Optimum efficiency and accuracy were the main criteria for selecting the computer implementation of each algorithm to be compared. Itemized costs for most of the mechanizations are available in Bierman [1973] and [1976b] and are not repeated here. However, detailed operation counts for the U-D algorithms may be found in Appendix E, along with a discussion of the colored noise propagation schemes.

---

†This weighting scheme was originally suggested by Bierman [1976a].

Table 4.1. Comparison of Computer Arithmetic Operation Times

Computer	τ_+ (μsec)	$\tau_\times/\tau_+$	$\tau_\div/\tau_+$	$\tau_{\surd}/\tau_+$
HP21MX-20 (32 bit)	34	1.7	1.8	70
Litton 4516 (32 bit)	4	2.2	3.4	250
UNIVAC 1108 (27 bit)	1.9	1.4	4.5	21.4

Measurement and time updating costs are considered separately in order to carefully evaluate the efficiency of each individual algorithm. Several filter algorithms are then compared in terms of total costs at each stage of the filtering process.

4.2 Measurement Update Cost Comparisons

The arithmetic operations required by each of the measurement update algorithms studied in Chapter II are indicated in Table 4.2. In this table n denotes the filter dimension, while m represents the number of measurements included prior to variance calculations.

The relative execution times for the HP computer (cf Table 4.1) were used to weight the operations in Table 4.2. These weighted counts are included in Table 4.3 and indicate the relative execution times of each algorithm as a function of n and m.

Table 4.2. Operation Counts to Process m Scalar Measurements

Algorithm	Adds	Multiplies	Divides	Square Roots
Conventional Kalman $\hat{P} = \tilde{P} - K_L^T \tilde{P}$	$(1.5n^2 + 3.5n)m$	$(1.5n^2 + 4.5n)m$	m	0
U-D Covariance Factorization	$(1.5n^2 + 1.5n)m$ + $(0.5n^2 - 0.5n)^*$	$(1.5n^2 + 5.5n)m$ + $(n^2 - n)^*$	nm	0
Carlson Triangular Covariance Square Root	$(1.5n^2 + 2.5n)m$ + $(0.5n^2 + 0.5n)^*$	$(2n^2 + 5n)m$ + $(0.5n^2 + 0.5n)^*$	$(2n + 1)m$	nm
Potter Covariance Square Root	$(3n^2 + 3n)m$ + n^{2*}	$(3n^2 + 4n)m$ + n^{2*}	2m	m
Stabilized Kalman $\hat{P} = (I - Ka^T)\tilde{P}(I - Ka^T)^T + KrK^T$	$(4n^2 + 4n)m$	$(4n^2 + 6n)m$	m	0

*These operations involve computing estimate error variances.

Measurement update costs are further compared in Figure 4.1. Execution times have been normalized by the corresponding conventional Kalman times and are displayed as a function of n for m = 1 and m = 5. Hence, this figure gives the various algorithm costs as a percentage increase over the conventional updating costs. Notice that the U-D algorithm is considerably more efficient than any other alternative to the conventional Kalman method. Moreover, when variances are required infrequently, the U-D and conventional Kalman costs are nearly identical.

Table 4.3. Measurement Update Costs Weighted for the HP Computer

Algorithm	Execution Time/τ_+
Conventional Kalman	$(4.1n^2 + 11.2n)m$
U-D	$(4.1n^2 + 12.7n)m + (2.2n^2 - 2.2n)$*
Carlson	$(4.9n^2 + 84.6n + 1.8)m + (1.4n^2 + 1.4n)$*
Potter	$(8.1n^2 + 9.8n + 73.6)m + (2.7n^2)$*
Stabilized Kalman	$(10.8n^2 + 14.2n)m$

*Variances computed

In sharp contrast to this efficiency, the other algorithms generally require at least 100% more computation time than the conventional method. Note that the Carlson normalized costs increase as the state dimension

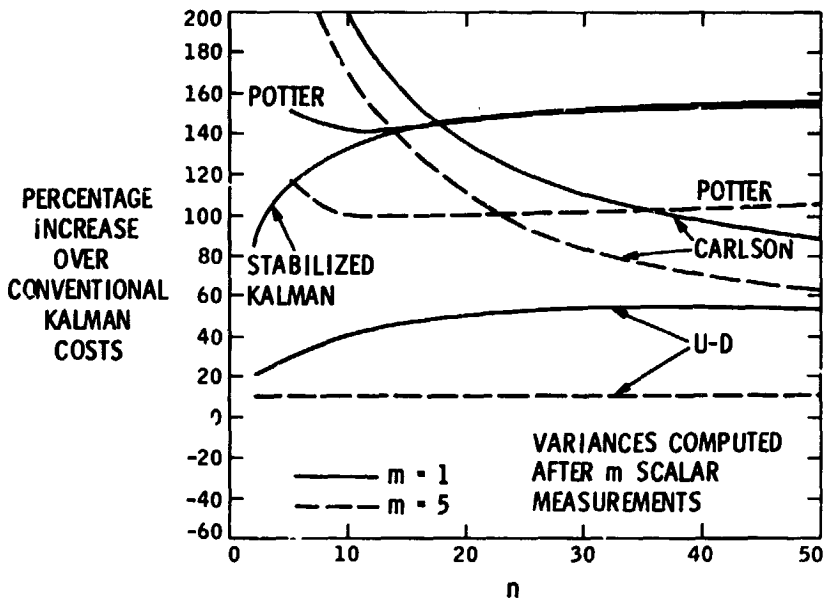

Figure 4.1. Measurement Update Cost Comparison

decreases. This trend is related to the n scalar square root calculations in the Carlson formula; these operations are relatively more expensive when n is small.

4.3 Time Update Cost Comparisons

4.3.1 Comparison of General Propagation Algorithms

Several algorithms for propagating covariance factors have been described in Chapter III. With the exception of the RSS methods, all of the algorithms are numerically reliable. Therefore, the most appropriate time update algorithm for each covariance factorization may be selected on the basis of efficiency.

Consider the covariance square root propagation schemes listed in Table 4.4. The operation counts given for each algorithm include only the calculations required to compute the updated square root $\tilde{S}$ from the $n \times (n+k)$ array $W = [BQ^{1/2} \; ; \; \Phi\hat{S}]$. Notice from Table 4.4 that the Schmidt and RSS methods require less computation than the other methods. Note, also, that these two algorithms involve equivalent amounts of computation. Therefore, no loss of efficiency is encountered when the more reliable Schmidt algorithm is selected for covariance square root propagation. The Schmidt time update may be coupled with the Carlson or the Potter square root data processing algorithms. The resulting filters are referred to as the Carlson-Schmidt

Table 4.4. Arithmetic Operations Required by Square Root Triangularization Algorithms

Algorithm	Adds	Multiplies	Divides	Square Roots
Givens (Eqs. (3.52)-(3.59))	$0.7n^3 + 1.5n^2 + 0.2n$ + n^2k	$1.3n + 4n^2 - 2.7n$ + $2(n^2 + n)k$	$0.5n^2 - 0.5n + nk$	$0.5n^2 - 0.5n + nk$
MGS (Eqs. (3.35)-(3.39))	$n^3 + n^2k$	$n^3 + 0.5n^2 - 0.5n$ + n^2k	$n - 1$	n
Schmidt (Eqs. (3.103)-(3.111))	$0.7n^3 + 1.3n + n^2k$	$0.7n^3 + n^2 + 0.3n$ + n^2k	$n - 1$	n
RSS (Section 3.2)	$0.7n^3 + n^2 + 0.3n$ + $(0.5n^2 + 0.5n)k$	$0.7n^3 + n^2 - 0.7n$ + $(0.5n^2 + 1.5n)k$	$n - 1$	n

Note: n and k denote system and process noise dimensions, respectively.

and Potter-Schmidt algorithms. These two square root covariance filters will be considered in the cost comparisons to follow.

A similar comparison of U-D propagation methods is given in Table 4.5. The calculations listed for each algorithm involve only the computation of $\tilde{U}$ and $\tilde{D}$ from the arrays $W = [B \mid \Phi\hat{U}]$ and $\bar{D} = \text{diag}(Q, \hat{D})$. The modified Givens operation counts in Table 4.5 represent the minimum attainable with this method; i.e., the more efficient formula, Method B in Eq. (3.87), is applied a maximum number of times. This assumption is justified by the final discussion in Section 3.4. The reader may wish to refer to Appendix E for a more detailed description of the U-D propagation costs.

It is not immediately apparent from Table 4.5 which U-D algorithm is the most efficient. However, it is clear that the RSS method has very little cost advantage over the modified Givens algorithm. Of the two reliable U-D methods, the Givens method is more efficient when n is large. However, when n is small and k is large the extra divide operations required by the Givens method are more apparent, and consequently the MWGS algorithm may be the more efficient method. Hence, in the cost comparisons to follow both of these U-D propagation schemes are considered.

Table 4.5. Arithmetic Operations Required By U-D Factorization Algorithms

Algorithm	Adds	Multiplies	Divides	Square Roots
Modified Givens (Eqs. (3.80)-(3.89))	$0.7n^3 - 0.5n^2 + 0.8n$ + n^2k	$0.7n^3 + 2.5n^2 - 0.2n$ + $(n^2 + 4n)k$	$0.5n^2 - 0.5n + nk$	—
MWGS (Eqs. (3.28)-(3.30))	$n^3 + n^2k$	$n^3 + 1.5n^2 - 0.5n$ + $(n^2 + n)k$	$n - 1$	—
RSS (Section 3.2)	$0.7n^3 + n^2 + 0.3n$ + $(0.5n^2 + 0.5n)k$	$0.7n^3 + 2.5n^2 - 1.2n$ + $(0.5n^2 + 1.5n)k$	$n - 1$	—

The arithmetic operations involved in a general time update via Eq. (3.1) are listed in Table 4.6 for each of the filter algorithms to be considered. These counts include the calculations required to compute estimates and the necessary updated covariance factors. The differences between the Potter-Schmidt and Carlson-Schmidt propagation costs are due entirely to the different structures of the covariance square roots involved. For example, computation of the product $\phi\hat{S}$ with a Potter square root requires $0.5n^3$ more adds and multiplies than the same calculation involving a Carlson triangular square root.

Table 4.7 contains the weighted propagation costs for each filter. These costs represent the relative execution times of the various time update schemes when the HP computer is used. Costs

are further compared in Figures 4.2 and 4.3 as a function of k/n where k denotes the number of process noise parameters. Execution times have been normalized by the corresponding execution time for the conventional method. Hence these figures show the percentage of conventional mapping costs which are required by the different algorithms. The costs associated with variance computations in the factorization filters are omitted from Figures 4.2, 4.3 and those to follow. However, the relative algorithm costs are not altered appreciably by including these computations (cf Table 4.7). In fact, when variance calculations are included, the cost curves in Figures 4.2 and 4.3 experience an upward shift of at most three percent.

Notice in Figures 4.2 and 4.3 that normalized propagation costs increase as a function of k/n. However, even when k/n = 2 all of the algorithms require less than 50% more computation than the conventional propagation formula. The Potter-Schmidt update is the most expensive time update algorithm while the other methods are generally competitive with one another. The modified Givens method is typically the more efficient U-D propagation scheme although for small dimensioned systems with many process noise parameters (i.e., k/n > 1), the MWGS algorithm may be less expensive (cf Figure 4.2). In either figure the cost differentials between the MWGS and the Givens methods are modest, and both schemes are generally competitive with the conventional propagation algorithm. Subsequent discussions of U-D propagation methods are therefore limited to the modified Givens algorithm.

Table 4.6. Operation Count for a General Time Update

Algorithm	Adds	Multiplies	Divides	Square Roots
Conventional Kalman	$1.5n^3 + 2n^2 + 0.5n$ + $(0.5n^2 + 0.5n)k$	$1.5n^3 + 1.5n^2$ + $(0.5n^2 + 1.5n)k$		
U-D (MWGS)	$1.5n^3 + 0.5n^2$ + n^2k + $(0.5n^2 - 0.5n)^*$	$1.5n^3 + 2n^2 - 0.5n$ + $(n^2 + n)k$ + $(n^2 - n)^*$	$n - 1$	
U-D (Givens)	$1.2n^3 + 0.8n$ + n^2k + $(0.5n^2 - 0.5n)^*$	$1.2n^3 + 3n^2 - 0.2n$ + $(n^2 + 4n)k$ + $(n^2 - n)^*$	$0.5n^2 - 0.5n$ + nk	
Carlson-Schmidt	$1.2n^3 + 1.5n^2 + 1.3n$ + n^2k + $(0.5n^2 + 0.5n)^*$	$1.2n^3 + 2.5n^2 + 0.3n$ + $(n^2 + n)k$ + $(0.5n^2 + 0.5n)^*$	$n - 1$	n
Potter-Schmidt	$1.7n^3 + n^2 + 1.3n$ + n^2k + $(n^2)^*$	$1.7n^3 + 2n^2 + 0.3n$ + $(n^2 + n)k$ + $(n^2)^*$	$n - 1$	n

*Variances computed.

Table 4.7. General Time Update Costs Weighted for the HP Computer

Algorithm	Execution Time/τ_+
Conventional Kalman	$4.1n^3 + 4.6n^2 + 0.5n + (1.4n^2 + 3.1n)k$
U-D (MWGS)	$4.1n^3 + 4.8n^2 + 0.1n + (2.7n^2 + 1.7n)k$ + $(2.2n^2 - 2.2n)$*
U-D (Givens)	$3.2n^3 + 6n^2 - .4n + (2.7n^2 + 8.6n)k$ + $(2.2n^2 - 2.2n)$*
Carlson-Schmidt	$3.2n^3 + 5.8n^2 + 73.6n + (2.7n^2 + 1.7n)k$ + $(1.4n^2 + 1.4n)$*
Potter-Schmidt	$4.6n^3 + 4.4n^2 + 73.6n + (2.7n^2 + 1.7n)k$ + $(2.7n^2 + 2.7n)$*

*Variances computed

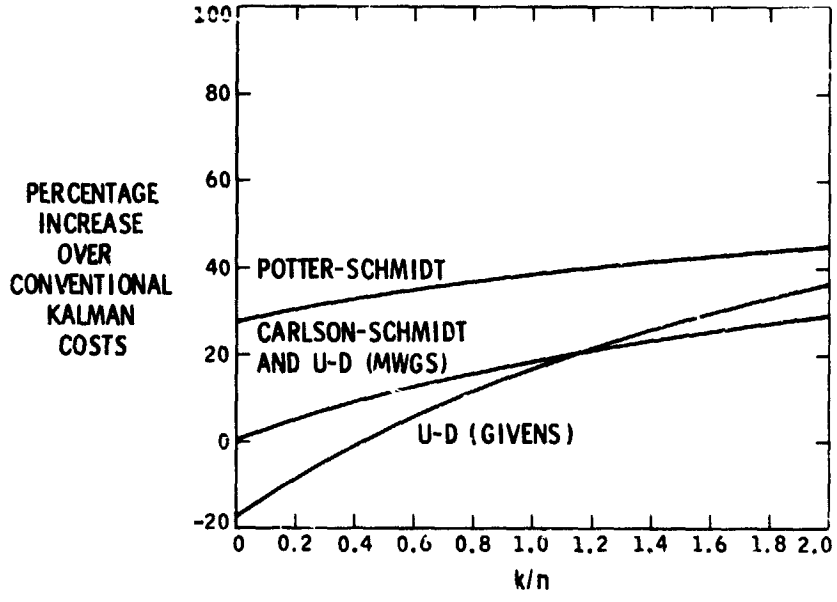

Figure 4.2. General Time Update Cost Comparison, n = 10

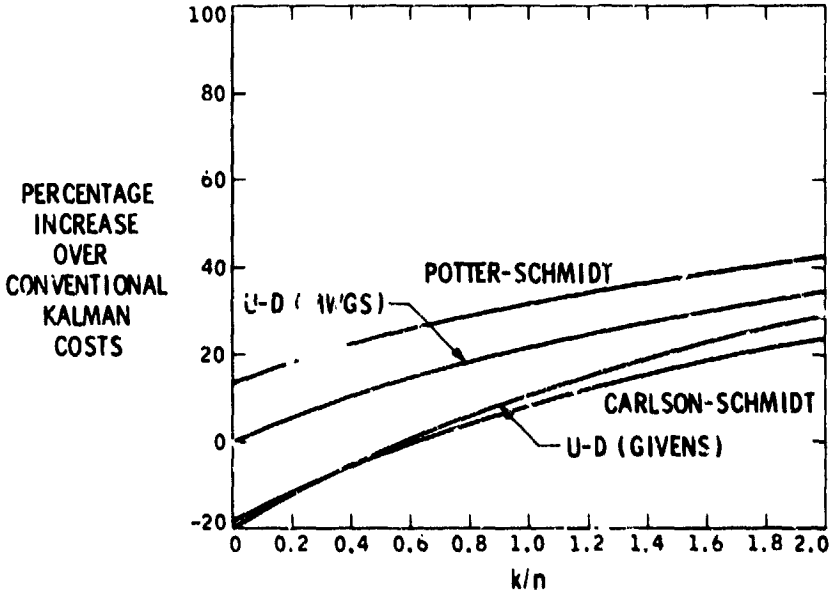

Figure 4.3. General Time Update Cost Comparison, n = 30

4.3.2 Comparison of Colored Noise Propagation Algorithms

The operation counts associated with a colored noise time update of the various covariance factors are compared in Table 4.8.† The U-D algorithm to be considered propagates process noise one component at a time and employs a modified Givens factorization for the deterministic phase of the update (cf Eqs. (3.120)-(3.122) and (3.130)-(3.133)). A similar algorithm based upon the MWGS factorization requires approximately the same computation time as the one considered here (see Appendix E).

Colored noise time updating of the Potter and Carlson covariance square roots is more efficiently accomplished by mapping the process noise in one step. Hence, the Potter-Schmidt and Carlson-Schmidt operation counts in Table 4.8 represent the cost of a Schmidt time update which exploits system structure. The reader is referred to Appendix E for further details of these mechanizations.

The weighted operation counts in Table 4.9 represent the relative costs of a colored noise update on the HP computer. These costs, normalized by the conventional propagation costs, are illustrated in Figures 4.4 and 4.5 as a function of k/n. Filter dimension in these figures is n+k where k represents the number of colored noise parameters. For all values of k and n the Potter-Schmidt colored noise update is considerably less efficient than the Carlson-Schmidt and U-D algorithms.

---

†The colored noise system is defined by Eq. (3.112).

Table 4.8. Operation Counts for Colored Noise Time Update

Algorithm	Adds	Multiplies	Divides	Square Roots
Conventional Kalman	$1.5n^3 + 1.5n^2 + nk^2$ $+ (2.5n^2 + 1.5n + 1)k$	$1.5n^3 + 1.5n^2 + (n + 1)k^2$ $+ (2.5n^2 + 2.5n + 1)k$		
U-D	$1.2n^3 + 0.8n + 0.3k^3$ $+ (1.5n - 0.5)k^2$ $+ (2n^2 - 0.5n + 1.2)k$ $+ (0.5(n + k)^2)^*$ $- 0.5(n + k))^*$	$1.2n^3 + 3n^2 - 0.2n$ $+ 0.3k^3 + (1.5n + 2)k^2$ $+ (2n^2 + 3n + 0.7)k$ $+ ((n + k)^2 - (n + k))^*$	$0.5n^2 - 0.5n - 1$ $+ 0.5k^2$ $+ (n + 0.5)k$	
Carlson-Schmidt	$1.2n^3 + 1.5n^2 + 1.3n$ $+ 0.3k^3$ $+ (1.5n + 1.5)k^2$ $+ (2n^2 + 4.5n + 1.2)k$ $+ (0.5(n + k)^2)^*$ $+ 0.5(n + k))^*$	$1.2n^3 + 2.5n^2 + 0.3n$ $+ 0.3k^3$ $+ (1.5n + 2.5)k^2$ $+ (2n^2 + 5.5n + 2.2)k$ $+ (0.5(n + k)^2)^*$ $+ 0.5(n + k))^*$	$n + k - 1$	$n + k$
Potter-Schmidt	$1.7n^3 + n^2 + 1.3n + k^3$ $+ (n + 2.5)k^2$ $+ (3n^2 + 3n + 2.5)k$ $+ (n + k)^{2*}$	$1.7n^3 + 2n^2 + 0.3n + k^3$ $+ (n + 4)k^2$ $+ (3n^2 + 4n + 3)k$ $+ (n + k)^{2*}$	$n + k - 1$	$n + k$
*Variances computed				

Table 4.9. Colored Noise Time Update Costs
Weighted for the HP Computer

Algorithm	Execution Time/τ_+
Conventional Kalman	$4.1n^3 + 4.1n^2 + (2.7n + 1.7)k^2 + (6.8n^2 + 5.8n + 4.4)k$
U-D	$3.2n^3 + 6n^2 - .4n + 0.8k^3 + (4.1n + 3.8)k^2 + (5.4n^2 + 6.4n + 3.3)k + (2.2(n + k)^2 - 2.2(n + k))$*
Carlson-Schmidt	$3.2n^3 + 5.8n^2 + 73.7n + 0.8k^3 + (4.1n + 5.8)k^2 + (5.4n^2 + 13.9n + 76.7)k + (1.4(n + k)^2 + 1.4(n + k))$*
Potter-Schmidt	$4.6n^3 + 4.4n^2 + 73.7n + 2.7k^3 + (2.7n + 9.3)k^2 + (8.1n^2 + 8.1n + 79.4)k + (2.7(n + k)^2 + 2.7(n + k))$*

*Variances computed

Unlike these triangular factorization methods, the Potter square root does not allow for full exploitation of the special system structure in Eq. (3.112). The Carlson-Schmidt and U-D colored noise algorithms are generally competitive with one another, although the U-D method is somewhat more efficient when n = 10.

Note that cost differentials of all the methods increase as a function of k/n. However, even when k/n = 2 the U-D costs are

within 30% of those related to the conventional propagation method. Moreover, when k/n < 1 the U-D method is the least expensive colored noise algorithm.

Of the two figures being discussed, Figure 4.4 is more representative of the costs usually incurred with small, on-board computers of the type considered here. For example, the GPS project expects to employ the HP computer for airborne filtering of a colored noise system which has dimensions n = k = 6 (cf General Dynamics-Electronics [1975]). Higher dimension filters, such as those represented by Figure 4.5, are generally designed for ground-based analysis on large computers such as the UNIVAC 1108 or IBM 360. Thus, further comparisons of colored noise costs will be restricted to the case n = 10.

4.4 Comparison of Total Filtering Costs

The measurement and time update costs given in the two previous sections may be combined to yield a rough comparison of total filtering costs. Most of these comparisons are based upon the assumption that a single scalar measurement is included at each step of the filtering process. Note that measurement updating is generally an order of magnitude less expensive than time updating (cf Tables 4.3 and 4.7). Hence, filtering costs primarily reflect time update expenses, especially when n is large. For small systems, however, measurement update costs are more visible, and the addition of several observations may noticeably alter the filter cost differentials. To illustrate these effects, additional comparisons involving multiple measurements are included for the case n = 10.

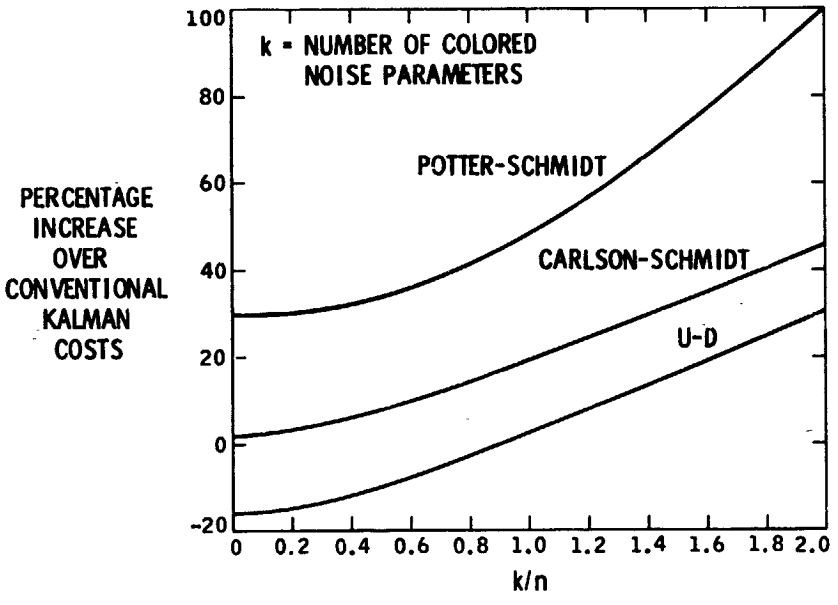

Figure 4.4. Colored Noise Time Update Cost Comparison, n = 10

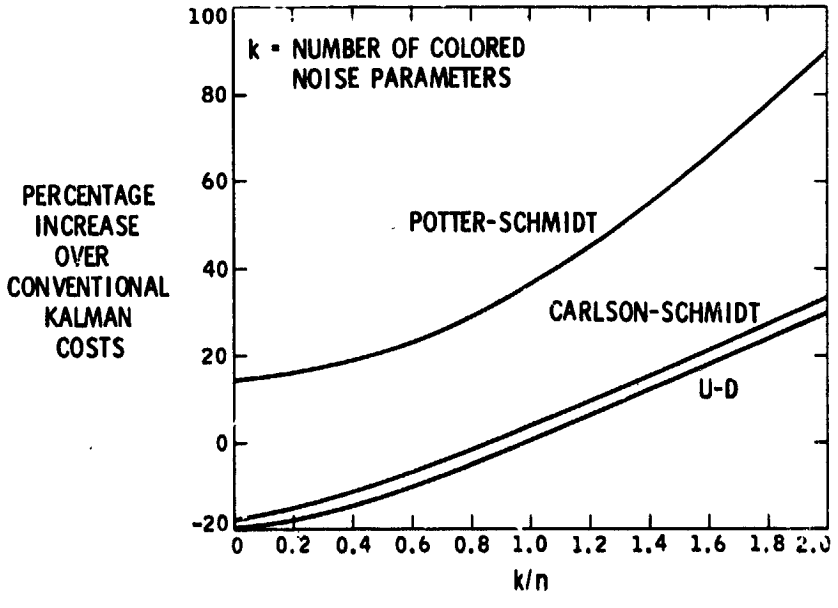

Figure 4.5. Colored Noise Time Update Cost Comparison, n = 30

Filter cost comparisons do not include the expenses related to variance calculations. As previously noted, these computations have a modest effect on relative time update costs[†] and, hence, on the normalized filter costs to be compared.

Filtering costs associated with the general system model, Eq. (3.1), are compared in Figures 4.6 and 4.7. These figures represent the relative computation times required by each filter algorithm[††] and were obtained by combining the weighted execution times in Tables 4.3 and 4.7. The costs of each algorithm have been normalized by the corresponding conventional Kalman costs and are displayed as a function of k/n for n = 10 and n = 30. In a similar manner, the colored noise filter algorithms are compared in Figure 4.8. Each colored noise filter has dimension n+k where n = 10.

Notice first that cost differentials associated with the covariance factorization methods increase as a function of k/n, while those related to the stabilized Kalman algorithm remain relatively constant. In each case the stabilized Kalman costs are within 20% of those related to the conventional Kalman formula. Of all the algorithms compared, the Potter-Schmidt method is the most expensive. The Potter-Schmidt colored noise algorithm is particularly inefficient when there are

---

[†] See the discussion of Figures 4.2 and 4.3.

[††] The U-D filter algorithm included in these comparisons employs the modified Givens time update method. Costs associated with this filter algorithm may be easily adjusted to obtain comparisons for a U-D filter based upon the MWGS time update algorithm (of Figures 4.2 and 4.3).

large numbers of colored noise parameters. For example, when k/n = 2 this algorithm is twice as expensive as the conventional Kalman method. By comparison, the Carlson-Schmidt and U-D colored noise algorithms are significantly more efficient and require less than 50% additional computation relative to the conventional Kalman formula.

Notice that the U-D method is the least expensive covariance factorization algorithm. When system dimension is small, this method has a noticeable cost advantage (cf Figures 4.6 and 4.8). Moreover, U-D costs are generally within 20% of those related to the conventional Kalman method, and when $k/n < 0.4$ the U-D algorithm is the least expensive method considered.

For large scale systems there is less diversity in filter algorithm costs than that observed in Figures 4.6 and 4.8 where n = 10. Consider the comparison in Figure 4.7 for the case n = 30. Notice that filter algorithm costs differ less in this comparison than they do in the corresponding lower dimension case in Figure 4.6. Cost differences are less apparent in Figure 4.7 because time updating expenses dominate the comparisons when n = 30. Compare Figures 4.3 and 4.7 and note that relative filtering costs are nearly identical to relative propagation costs.

Although filtering costs primarily reflect the expenses related to time updating, the cost effects of a single measurement update are apparent when n = 10. Compare, for example, Figures 4.2 and 4.6. Notice that the U-D results in each figure are nearly identical, but the Potter-Schmidt and Carlson-Schmidt cost curves are noticeably different. These

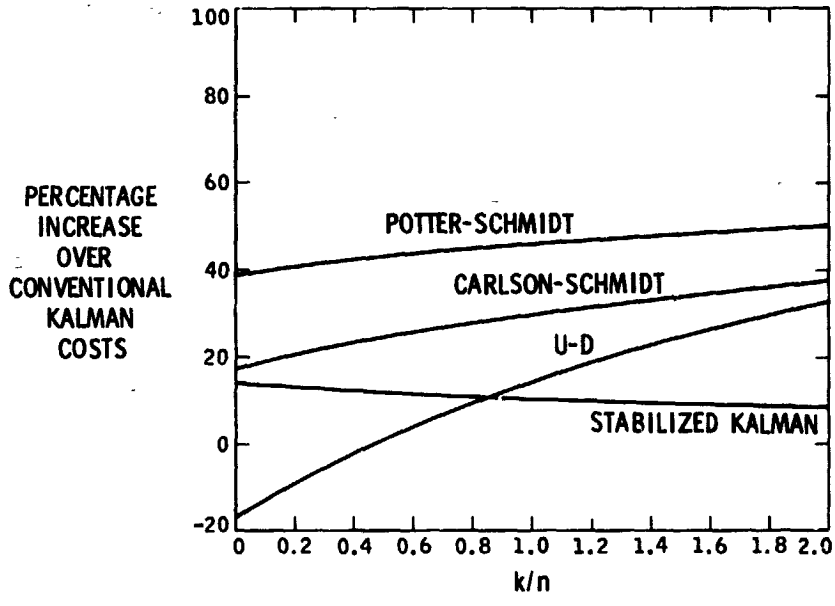

Figure 4.6. Cost Comparison of General Filter Algorithms, n = 10

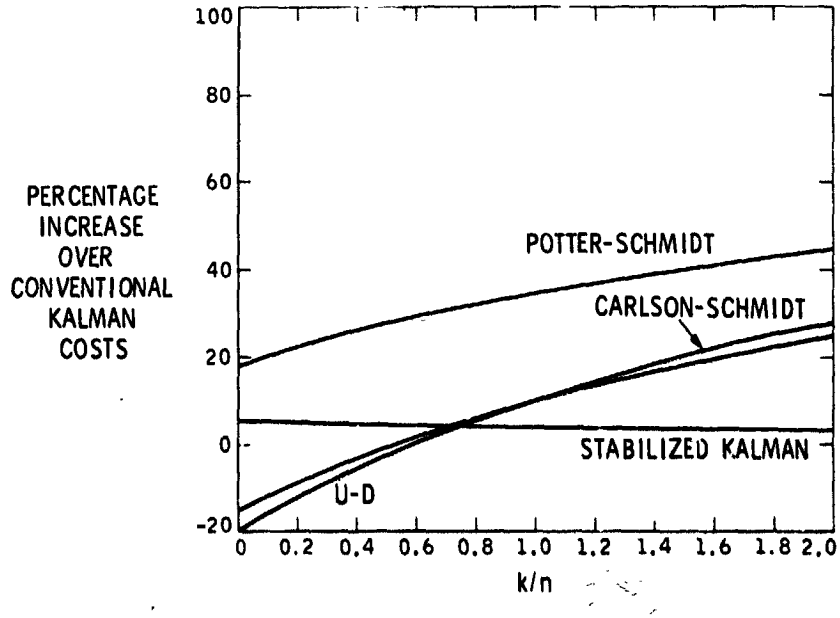

Figure 4.7. Cost Comparison of General Filter Algorithms, n = 30

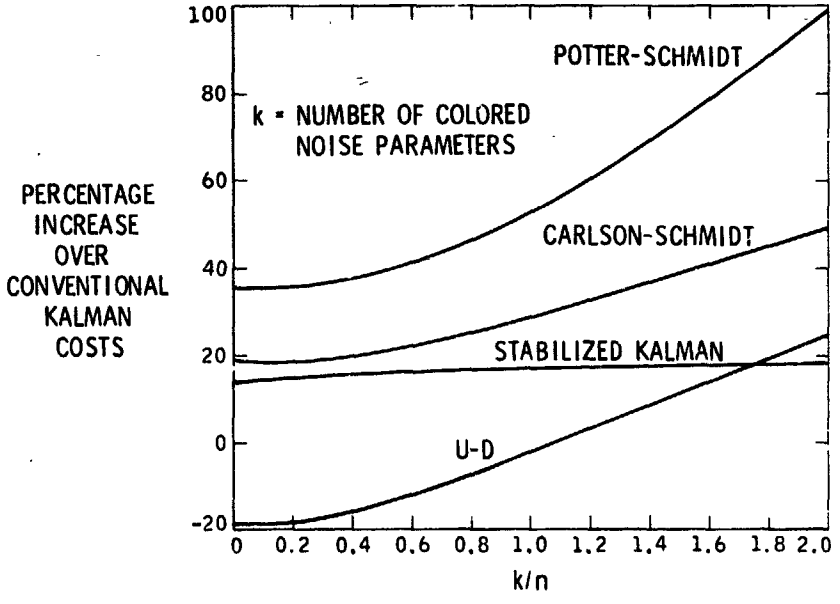

Figure 4.8. Cost Comparison of Colored Noise Filter Algorithms, n = 10

differences reflect the measurement cost differentials in Figure 4.1 where only the U-D algorithm has costs comparable to the conventional Kalman method. Thus, we would expect additional measurement updates to increase all of the normalized filter costs in Figure 4.6 except those corresponding to the U-D algorithm.

Consider the case where three measurements are incorporated after each time update. Relative filtering costs for this case are compared in Figures 4.9 and 4.10. The results in Figure 4.9 correspond to the general system model, while Figure 4.10 illustrates the colored noise filter costs. By comparing these figures with the corresponding results in Figures 4.6 and 4.8 we find that, except for the U-D method, each filter algorithm becomes relatively more expensive as additional measurements are incorporated. Figures 4.9 and 4.10 indicate that the Carlson-Schmidt and Potter-Schmidt lower-dimensioned filters can have equivalent costs when multiple measurements are involved. Further note from Figure 4.10 that whenever $k/n < 1$ the U-D method is the least expensive colored noise filter algorithm and requires at least 40% less computation than the other covariance factorization methods.

The cost comparisons discussed in this chapter have all been related to the HP computer. The relative costs of the various algorithms may vary considerably, depending upon which computer is used. Consider the different weights that the Litton and UNIVAC computers assign to the various arithmetic operations (see Table 4.1). Note that square root calculations are ten times more expensive on the Litton computer than on the UNIVAC model. Hence, we would expect the Carlson-Schmidt

and Potter-Schmidt covariance square-root methods to be noticeably less efficient on the Litton computer.

Cost comparisons related to the Litton and UNIVAC computers are illustrated in Figures 4.11-4.14 for the case where n = 10. These comparisons were obtained in the same manner used to generate the costs in Figures 4.6 and 4.8; only the weighting factors differ.

Notice from Figures 4.12 and 4.14 that when the UNIVAC computer is used, algorithm cost differences are generally modest. However, filtering costs related to the Litton model differ considerably. On the Litton computer the Carlson-Schmidt and Potter-Schmidt colored noise algorithms have nearly equivalent costs and are generally 60-80% more expensive than the corresponding U-D method. By comparing Figures 4.11-4.14 with the corresponding HP costs in Figures 4.6 and 4.8 we find that the relative costs of the U-D method are approximately the same on each computer involved in this analysis. In each case U-D costs are typically within 20% of those related to the conventional Kalman method, and when $k/n < 0.4$ the U-D algorithm is the most efficient method studied.

4.5 Conclusions

Cost analysis has demonstrated the relative efficiency of the U-D filtering technique. This method, employing either the modified Givens or MWGS propagation algorithms, is generally competitive with the conventional Kalman formula. The U-D filter algorithm based upon the modified Givens time update is particularly efficient and for problems involving

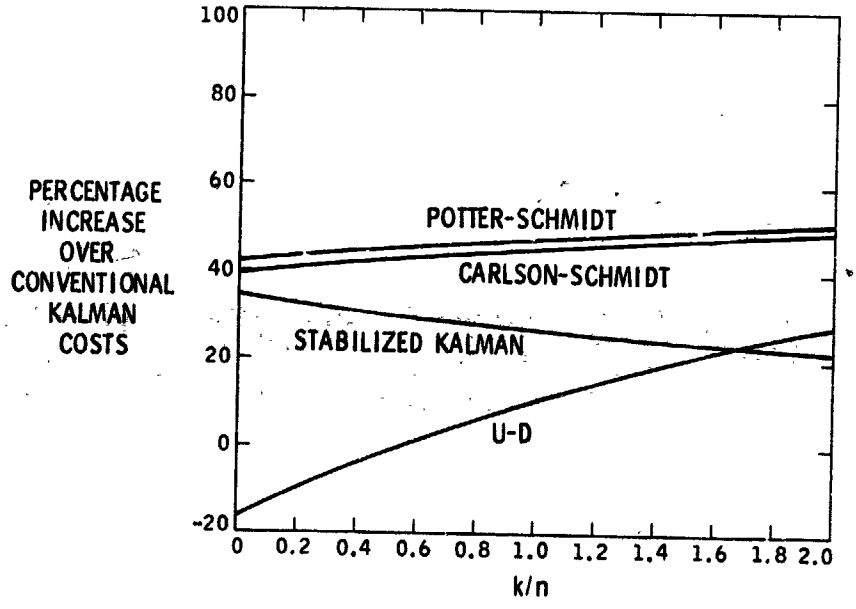

Figure 4.9. Relative Costs of General Filter Algorithms When Three Measurements are Included, n = 10

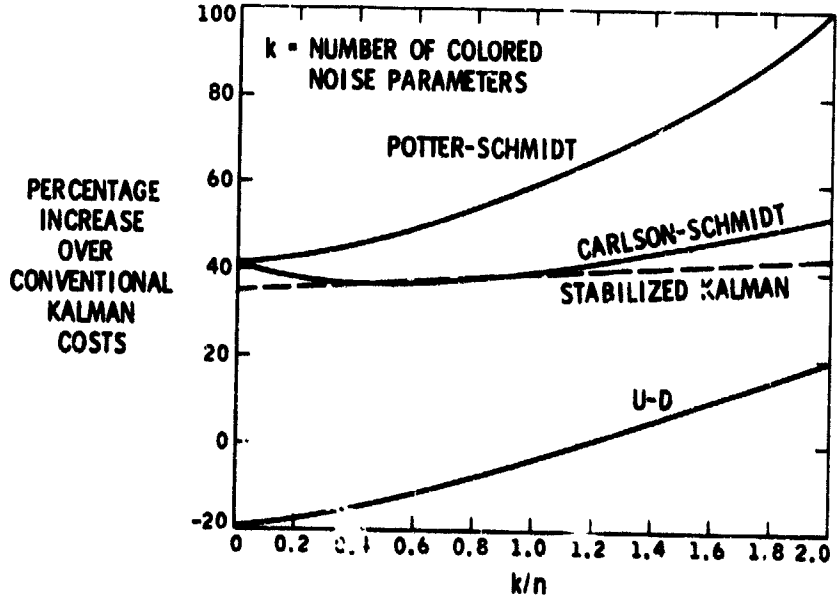

Figure 4.10. Relative Costs of Colored Noise Filter Algorithms When Three Measurements are Included, n = 10

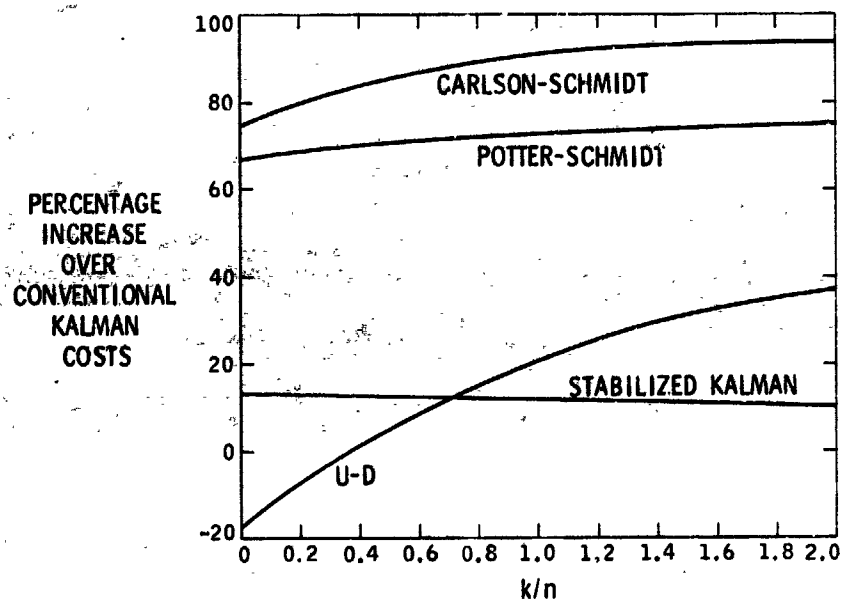

Figure 4.11. Relative Costs of General Filter Algorithms When Litton Computer is Used, n = 10

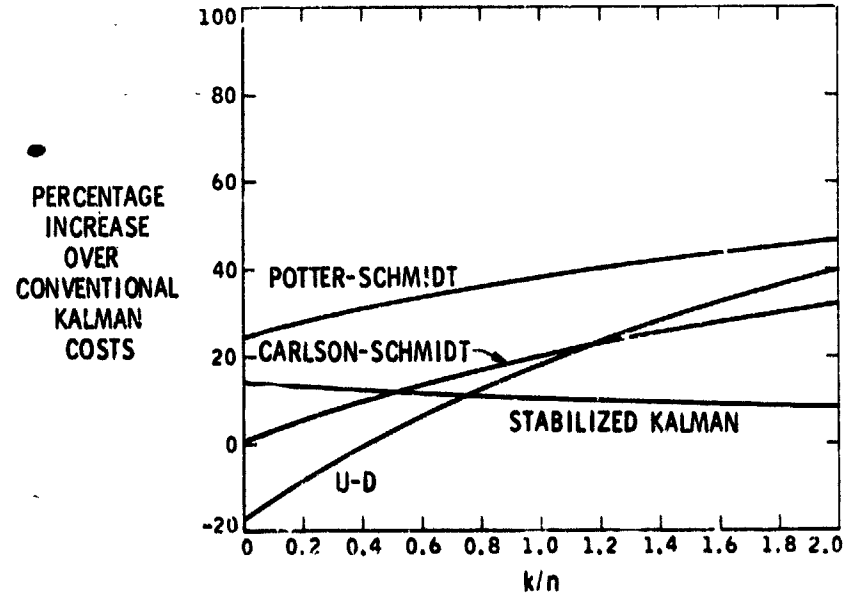

Figure 4.12. Relative Costs of General Filter Algorithms When UNIVAC Computer is Used, n = 10

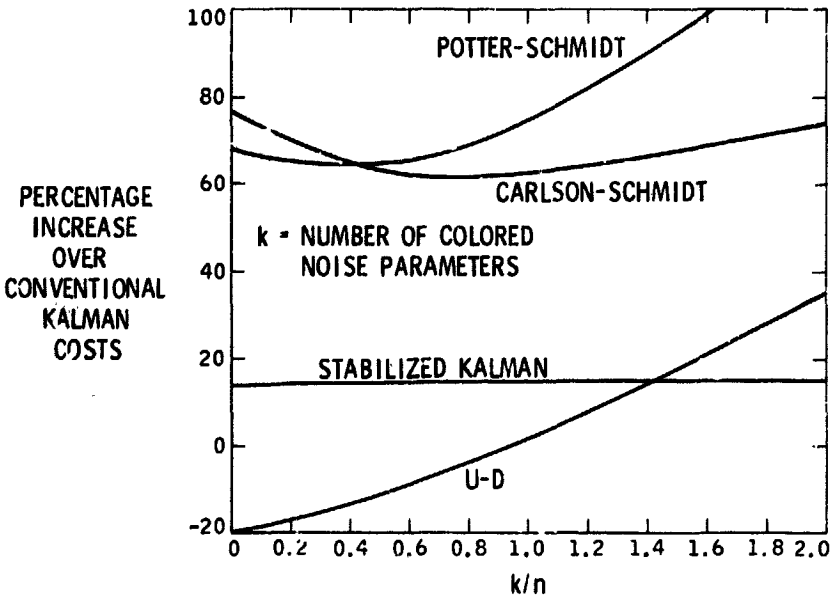

Figure 4.13. Relative Costs of Colored Noise Filter Algorithms When Litton Computer is Used, n = 10

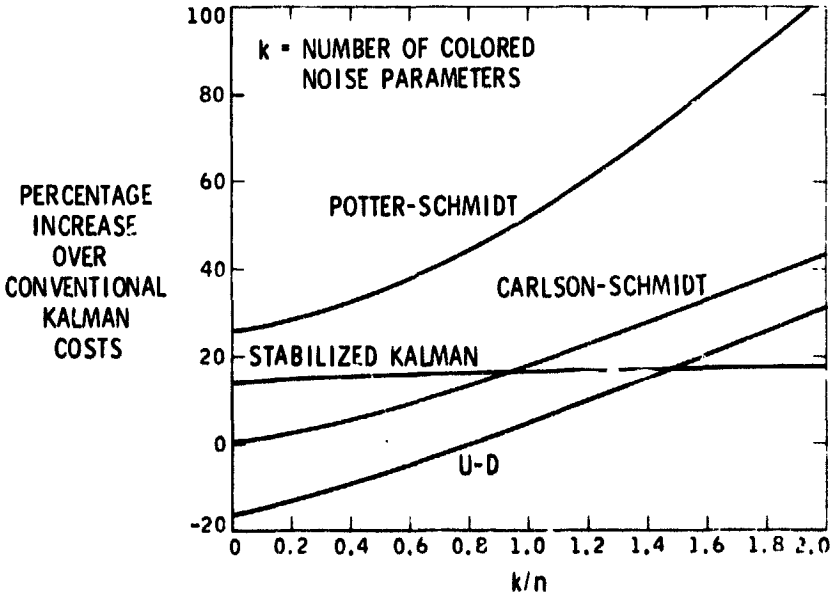

Figure 4.14. Relative Costs of Colored Noise Filter Algorithms When UNIVAC Computer is Used, n = 10

modest numbers of process noise parameters, this U-D algorithm is the least expensive of all the methods considered. By comparison, the Potter-Schmidt and Carlson-Schmidt square root algorithms are significantly more costly, particularly in real-time applications involving small computers. In these situations the Carlson-Schmidt method is generally no more efficient than the Potter-Schmidt algorithm and can be more expensive. Moreover, both methods can involve in excess of 60% more computation than the corresponding U-D algorithm.

Chapter V. Covariance Error Analysis Techniques

5.1 Introduction

Several difficulties are frequently encountered when filtering algorithms are applied to an actual estimation problem. For example, precise knowledge of the system model and complete a priori statistics are often unavailable. In addition, computational constraints may limit the dimensionality of the filter. Because of these problems suboptimal filtering is often inevitable. As an important aid in filter design, one may perform a sensitivity analysis of the estimate error covariance. Techniques for evaluating filtering sensitivity to incorrect a priori statistics have been derived for the discrete case by Fagin [1964], Nishimura [1966 and 1967] and Heffes [1966]. Their results were extended by Griffin and Sage [1968] to include the effects of incorrect data and state transition matrices. These error analysis methods all propagate an actual, or true, error covariance. The algorithms are often cumbersome and computationally expensive. Furthermore their reliability is questionable since they employ the numerically unstable Kalman formulas.

In this chapter a new approach to covariance error analysis is presented which is based upon the U-D covariance factorization. This method facilitates an accurate evaluation of general modeling errors, including a) incorrect a priori statistics, b) mismodeled system dynamics and data equations, and c) incomplete parameter sets. A general error

analysis algorithm which incorporates all of these effects is derived in two stages.

The case of incorrect a priori statistics is considered first. This case includes a large class of modeling errors which are easily analyzed. The basic evaluation algorithm developed for this important subproblem is then extended to include the effects of incorrect data and state transition matrices. The extended analysis provides a general evaluation algorithm capable of considering the effects of unmodeled or neglected parameters. However, the analysis of unmodeled bias parameters is included as a separate topic in order to highlight the simplicity and flexibility of the resulting algorithm.

5.2 Evaluation of Incorrect A Priori Statistics

Suppose a discrete linear filter is designed according to the following model

$$\left.\begin{aligned} x(i+1) &= \Phi(i)x(i) + B(i)w(i) \\ z(i) &= A(i)x(i) + v(i) \end{aligned}\right\} \quad i=0,1,2,\cdots \qquad (5.1),\ (5.2)^{\dagger}$$

where

$$x \in R_n \; ; \quad w \in R_k \; ; \quad z \in R_1$$

---

† Scalar measurement coefficient matrices are denoted as A instead of the "a^T" used in Chapter II. For the error analysis discussed in this chapter the symbol A is more appropriate, particularly in section 5.2 where the notation A_a is required.

$$x(0) \sim N(0,P)\ ;\ w(i) \sim N(0,Q(i));\ \nu(i) \sim N(0,r(i)) \qquad (5.3)$$

and

$$E\{x(0)w(i)^T\}=0;\ E\{x(0)\nu(i)\}=0;\ E\{w(i)\nu(j)\}=0;\quad i,j\geq 0 \qquad (5.4)$$

Let the correct process and data model be the following

$$\left.\begin{aligned} x_a(i+1) &= \Phi(i)x_a(i) + B_a(i)w_a(i) \qquad (5.5)\\ z(i) &= A(i)x_a(i) + \nu_a(i) \qquad (5.6)\end{aligned}\right\} \quad i=0,1,2,\ldots$$

where

$x_a(i)$, $w_a(i)$ and $\nu_a(i)$ satisfy (5.4) and

$$x_a(0) \sim N(0,P_a(0))\ ;\ w_a(i) \sim N(0,Q_a(i))\ ;\ \nu_a(i) \sim N(0,r_a(i)) \qquad (5.7)^{\dagger}$$

Let G(i) denote the computed (suboptimal) gain at stage i. Then the actual estimate errors associated with the model (5.5)-(5.7) satisfy the following recursions

$$\Delta\hat{x}(i) = \hat{x}(i)-x_a(i) = (I-G(i)A(i))\Delta\tilde{x}(i) + G(i)\nu_a(i) \qquad (5.8)$$

$$\Delta\tilde{x}(i+1) = \tilde{x}(i+1)-x_a(i+1) = \Phi(i)\Delta\hat{x}(i)-B_a(i)w_a(i) \qquad (5.9)$$

where

$$\Delta\tilde{x}(0) = -x_a(0) \qquad (5.10)$$

Equations (5.8)-(5.10) may be verified by applying (3.11), (3.13) and (3.16) together with (5.5)-(5.7).

---

†There is no loss of generality in assuming a zero mean process. This assumption is made in order to free the error analysis of superfluous algebra.

Thus, the actual a priori and a posteriori error covariance for this problem may be computed from the following recursions

$$\hat{P}_a = (I-GA)\tilde{P}_a(I-GA)^T + Gr_aG^T \tag{5.11}$$

$$\tilde{P}_a = \Phi\hat{P}_a\Phi^T + B_aQ_aB_a^T \tag{5.12}$$

where initially

$$\tilde{P}_a = P_a(0) \tag{5.13}$$

Equations (5.11)-(5.13) represent the conventional error analysis algorithm for mismodeled a priori statistics (cf Nishimura [1966] and Heffes [1966]). This evaluation method is simply an application of the stabilized Kalman covariance equations and for this reason it is numerically unreliable.†

A more reliable error analysis method may be obtained by employing the U-D factorization techniques studied in Chapters II and III. Time propagation of the actual U-D covariance factors, via (5.12), may be achieved by a direct application of the MWGS algorithm (or the modified Givens update) in Chapter III. The U-D suboptimal measurement update, corresponding to (5.11), is based upon the following covariance decomposition.

---

†The numerical deficiencies of the stabilized Kalman formula are demonstrated in Chapter VI where, in sharp contrast, the U-D and Potter methods are shown to possess superior numerical characteristics.

Suboptimal Error Covariance Decomposition

Given: $\tilde{P}$ - a priori error covariance

A, r_a - observation coefficients and scalar measurement error covariance

G - arbitrary gain

The a posteriori error covariance $\hat{P}$, where

$$\hat{P} = (I-GA)\tilde{P}(I-GA)^T + Gr_aG^T \tag{5.14}$$

may be written as follows

$$\hat{P} = \bar{P} + \alpha\lambda\lambda^T \tag{5.15}$$

where

$$\bar{P} = \tilde{P}-Kv^T \tag{5.16}$$

$$\lambda = K-G \tag{5.17}$$

$$v = \tilde{P}A^T \tag{5.18}$$

and

$$K = v/\alpha \qquad \alpha = Av + r_a \tag{5.19}$$

Consider the quadratic $\hat{P}(G)$ in Eq. (5.14). Note that the quantity K defined by (5.19) minimizes $\hat{P}(G)$ and that this minimum is $\hat{P}(K) = \bar{P}$. Since the difference $\hat{P}(G)-\bar{P}$ is non-negative definite and symmetric, we would expect $\hat{P}(G)$ to have the form (5.15). This decomposition is readily verified by substituting $(K-\lambda)$ for G in Eq. (5.14), regrouping terms and applying the identities (5.18) and (5.19). The covariance

$\bar{P}$ thus corresponds to the minimum quadratic, i.e., the optimum updated covariance associated with the Kalman gain K.†

Suppose the covariance $\tilde{P}$ in (5.14) is factored such that $\tilde{P} = \tilde{U}\tilde{D}\tilde{U}^T$. Equation (5.16) then implies that the $\bar{U}$-$\bar{D}$ factors of $\bar{P}$ may be computed from the Bierman optimum measurement update, Eqs. (2.69)-(2.79). Given the $\bar{U}$-$\bar{D}$ factors of $\bar{P}$, Eq. (5.15) suggests using the Agee-Turner triangular matrix factorization (Appendix C). Thus, the U-D suboptimal measurement update may be performed in two stages; an optimal update to a modified problem,† followed by an Agee-Turner rank one matrix update. This method is summarized in the following algorithm.

U-D Arbitrary Gain Update Algorithm

Given:
- $\tilde{U},\tilde{D}$ - factors of a priori covariance $\tilde{P}$
- A, r_a - observation coefficients and scalar measurement error covariance
- G - arbitrary gain

The a posteriori covariance factors $\hat{U}$ and $\hat{D}$ can be obtained as follows:

- Compute via the U-D optimum measurement update algorithm, Eqs. (2.69)-(2.79), the quantities:

†The gain K is optimum given a priori covariance $\tilde{P}$. However, $\bar{P}$ does not represent the optimum (or minimum) covariance attainable unless $\tilde{P}$ is also optimum.

K - Kalman gain

α - innovations covariance

$\bar{U},\bar{D}$ - the factors of $\bar{P} = (I-KA)\tilde{P}$

- Set $\lambda = K-G$

- Apply the Agee-Turner algorithm, Appendix C, to compute $\hat{U}$ and $\hat{D}$ where

$$\hat{U}\hat{D}\hat{U}^T = \bar{U}\bar{D}\bar{U}^T + \alpha\lambda\lambda^T \tag{5.20}$$

This evaluation may be represented symbolically as

$$[\tilde{U},\tilde{D},A,r_a] \rightarrow [\bar{U},\bar{D},\lambda,\alpha] \rightarrow [\hat{U},\hat{D}]$$

The U-D arbitrary gain update inherits the efficiency and reliability of the two algorithms it employs. The optimal U-D update was shown in Chapter IV to require $1.5n^2 + O(n)$ multiplications while the Agee-Turner factorization requires n^2 such operations (cf Appendix C). Hence the entire update is performed with $2.5n^2 + O(n)$ muitiplications. When variances are computed, total multiplications reach $3n^2 + O(n)$ as compared to the $4n^2 + O(n)$ such calculations required by the conventional evaluation formula, Eq. (5.11). A detailed cost comparison of gain evaluation algorithms is included in Appendix F.

This arbitrary gain update is the cornerstone of the U-D error analysis technique. Note that evaluation of incorrect a priori statistics

requires only a minor addendum to the optimal U-D filtering algorithm. Thus, both the gain computations and the subsequent error analysis can be performed by one program. The gains may be computed during a first pass through the filter, operating in what it believes to be an optimum mode. The same filter, tuned to account for suboptimal gains and given the correct system model, may then perform the gain evaluation.

A similar error analysis algorithm could also be developed for the factorization $P = SS^T$ where S is upper triangular. The suboptimal measurement update for this factorization closely parallels the U-D formula. First, $\bar{S}$, the square root of $\bar{P}$ in Eq. (5.15), is computed via Carlson's optimal update, Eqs. (2.39)-(2.46). Then an Agee-Turner square-root factorization (Appendix C) yields $\hat{S}$ where $\hat{S}\hat{S}^T = \bar{S}\bar{S}^T + \alpha\lambda\lambda^T$. This update may be coupled with a Schmidt (Householder) propagation scheme to obtain a complete square root error analysis algorithm. The square root method is not pursued further, however, because it lacks the efficiency of the U-D evaluation algorithm (see Appendix F).

5.3 Evaluation of Mismodeled Data and State Transition Matrices

5.3.1 General Error Analysis Technique

In this section the U-D error analysis method is extended to account for mismodeled data and state transition matrices. Suppose the correct process and data equations are the following

$$x_a(i+1) = \Phi_a(i)x_a(i) + B_a(i)w_a(i) \tag{5.21}$$

$$z(i) = A_a(i)x_a(i) + \nu_a(i) \tag{5.22}$$

where w_a, ν_a and $x_a(0)$ satisfy Eq. (5.7).

Suppose the filter applied to this problem is based upon the assumptions (5.1)-(5.4) and generates the gain sequence $\{G(i)\}$. Then the actual a posteriori error equation at each stage has the form

$$\Delta\hat{x} = \hat{x} - x_a = (I-GA)\Delta\tilde{x} + G\Delta A x_a + G\nu_a \tag{5.23}$$†

where

$$\Delta A = A_a - A \tag{5.24}$$

The a priori error, $\Delta\tilde{x}$, is related to previous errors as follows

$$\Delta\tilde{x} = \tilde{x} - x_a = \Phi\Delta\hat{x} - \Delta\Phi x_a - B_a w_a \tag{5.25}$$

where

$$\Delta\Phi = \Phi_a - \Phi \tag{5.26}$$

Due to the errors ΔA and $\Delta\Phi$, covariance expressions for $\Delta\hat{x}$ and $\Delta\tilde{x}$ contain the actual state covariance, $P_a = E\{x_a x_a^T\}$, and the cross-covariance, $P_{\Delta x x} = E\{\Delta x x_a^T\}$. Thus, covariance error analysis for this problem involves propagation of the 2n-dimensional covariance

$$\mathcal{P} = E\left\{\begin{bmatrix}\Delta x \\ x_a\end{bmatrix}\begin{bmatrix}\Delta x^T & x_a^T\end{bmatrix}\right\}$$

†Note that mismodeled data and state transition matrices produce biased estimates unless $E\{x_a\} = 0$.

The recursions required to propagate $\mathscr{P}$ by conventional methods were derived by Griffin and Sage [1968]. However, their algorithm appears unwieldy and, judging from the experimental results in Chapter VI, it is numerically unsound. The problem may be solved more accurately and efficiently by applying U-D covariance factorization methods. For this error analysis we will find it convenient to avoid covariance equations altogether and study, instead, the associated "weighted error" propagation problem.† To this end, we interpret the U-D covariance factorization as a particular "whitening" of the estimate errors. That is, in terms of second-order statistics the errors may be described as

$$\Delta x = U v_x \tag{5.27}$$

where the zero mean noise, v_x, has diagonal covariance D.

Suppose the U-D factors of the a priori covariance

$$\tilde{\mathscr{P}} = E\left\{\begin{bmatrix}\Delta\tilde{x}\\ x_a\end{bmatrix}\begin{bmatrix}\Delta\tilde{x}^T & x_a^T\end{bmatrix}\right\}$$

are partitioned as follows.

$$\tilde{U} = \begin{bmatrix}\overbrace{U_{\tilde{x}}}^{n} & \overbrace{U_{\tilde{x}x}}^{n}\\ 0 & U_x\end{bmatrix} \qquad \tilde{D} = \operatorname{diag}\left(\overbrace{D_{\tilde{x}}}^{n}, \overbrace{D_x}^{n}\right) \tag{5.28}$$

†This approach is analogous to the error analysis technique associated with square-root information filtering (cf Bierman [1976b]).

Then the "whitening" interpretation of Eq. (5.27) implies that a priori estimate errors may be written as

$$\begin{bmatrix} \Delta\tilde{x} \\ -- \\ x_a \end{bmatrix} = \left[\begin{array}{c|c} U_{\tilde{x}} & U_{\tilde{x}x} \\ \hline 0 & U_x \end{array}\right] \begin{bmatrix} \nu_{\tilde{x}} \\ -- \\ \nu_x \end{bmatrix} \tag{5.29}$$

where $\nu_{\tilde{x}}$ and ν_x are zero mean, uncorrelated and have diagonal covariances $D_{\tilde{x}}$ and D_x respectively. Equations (5.23) and (5.29) yield the following a posteriori error equation

$$\begin{bmatrix} \Delta\hat{x} \\ -- \\ x_a \end{bmatrix} = \left[\begin{array}{c|c|c} \overbrace{G}^{1} & \overbrace{(I-GA)U_{\tilde{x}}}^{n} & \overbrace{\bar{U}_{\tilde{x}x}}^{n} \\ \hline 0 & 0 & U_x \end{array}\right] \begin{bmatrix} \nu_a \\ -- \\ \nu_{\tilde{x}} \\ -- \\ \nu_x \end{bmatrix} \begin{matrix} \}1 \\ \\ \}n \\ \\ \}n \end{matrix} \tag{5.30}$$

where

$$\bar{U}_{\tilde{x}x} = (I-GA)U_{\tilde{x}x} + GAAU_x \tag{5.31}$$

The noise vector, ν, has zero mean and diagonal covariance

$$D = (\overbrace{r_a}^{1}, \overbrace{D_{\tilde{x}}}^{n}, \overbrace{D_x}^{n}) \tag{5.32}$$

Given the expressions (5.30)-(5.32), we desire the representation

$$\begin{bmatrix} \Delta\hat{x} \\ -- \\ x_a \end{bmatrix} = \left[\begin{array}{c|c} U_{\hat{x}} & U_{\hat{x}x} \\ \hline 0 & U'_x \end{array}\right] \begin{bmatrix} \nu_{\hat{x}} \\ -- \\ \nu'_x \end{bmatrix} \tag{5.33}$$

with

$$E\left\{\begin{bmatrix} v_{\hat{x}} \\ v_{x}' \end{bmatrix}\begin{bmatrix} v_{\hat{x}}^T & v_{x}'^T \end{bmatrix}\right\} = \hat{D} = \text{diag}\ (\underbrace{D_{\hat{x}}}_{n}, \underbrace{D_{x}'}_{n}) \tag{5.34}$$

The constraint that

$$\hat{\mathcal{P}} = E\left\{\begin{bmatrix} \Delta\hat{x} \\ x_a \end{bmatrix}\begin{bmatrix} \Delta\hat{x}^T & x_a^T \end{bmatrix}\right\}$$

be unchanged during the transformation from (5.30) to (5.33) yields the identities

$$[U_{x}', D_{x}'] = [U_{x}, D_{x}] \tag{5.35}$$

$$U_{\hat{x}x} = \bar{U}_{\tilde{x}x} = U_{\tilde{x}x} - G(AU_{\tilde{x}x} - \Delta AU_{x}) \tag{5.36}$$

$$U_{\hat{x}}D_{\hat{x}}U_{\hat{x}}^T = (I-GA)U_{\tilde{x}}D_{\tilde{x}}U_{\tilde{x}}^T(I-GA)^T + Gr_aG^T \tag{5.37}$$

Thus, $U_{\hat{x}x}$, U_{x}' and D_{x}' are obtained directly. The factors $U_{\hat{x}}$ and $D_{\hat{x}}$ may be computed by applying the U-D arbitrary gain algorithm (section 5.2) to Eq. (5.37).

Similarly, a U-D algorithm may be derived for the time update, Eq. (5.25). When the expression (5.33) is applied to (5.25), the a priori error equation takes the form

$$\begin{bmatrix} \Delta\tilde{x} \\ -- \\ x_a \end{bmatrix} = \left[\begin{array}{c|c|c} -B_a & \Phi U_{\tilde{x}} & \bar{U}_{\tilde{x}x} \\ \hline B_a & 0 & \Phi_a U_x \end{array}\right] \begin{bmatrix} w_a \\ -- \\ v_{\tilde{x}} \\ -- \\ v_x \end{bmatrix} \begin{matrix} \}k \\ \\ \}n \\ \\ \}n \end{matrix} \tag{5.38}$$

where

$$\bar{U}_{\tilde{x}x} = \Phi U_{\tilde{x}x} - \Delta\Phi U_x \tag{5.39}$$

The noise vector v in Eq. (5.38) has zero mean and covariance $\bar{D}$ where

$$\bar{D} = \text{diag}\,(\underbrace{Q_a}_{k}, \underbrace{D_{\tilde{x}}}_{n}, \underbrace{D_x}_{n}) \tag{5.40}$$

Note that one of the U-D time update algorithms in Chapter III may be used to triangularize (5.38). In this case the W and $\bar{D}$ arrays involved in the time update are given by Eqs. (5.40) and (5.41).

$$W = \left[\begin{array}{c|c|c} \overbrace{-B_a}^{k} & \overbrace{\Phi U_{\tilde{x}}}^{n} & \overbrace{\bar{U}_{\tilde{x}x}}^{n} \\ \hline B_a & 0 & \Phi_a U_x \end{array}\right] \begin{matrix} \\ \}n \end{matrix} \tag{5.41}$$

Computer storage and computation required for this update may be reduced by taking advantage of the nxn block of zeroes in the W array.

The general U-D error analysis method is summarized as follows.

General U-D Covariance Error Analysis Algorithm

Given: $\{\Phi(i)\}$ and $\{A(i)\}$ - the incorrect state transition and data matrices assumed by the filter

$\{G(i)\}$ - suboptimal gain profile computed by the filter

$\{\Phi_a(i)\}$ - correct state transition matrices

$\{B_a(i)\}$ - actual process noise coefficients

$\{A_a\}$ - actual data coefficients

$\{r_a(i)\}$ - actual data noise covariances

$\{Q_a(i)\}$ - actual process noise covariances

$\begin{bmatrix} U_{\tilde{x}} & U_{\tilde{x}x} \\ 0 & U_x \end{bmatrix}$ and $D = \text{diag}\,(D_{\tilde{x}}, D_x)$ - factors of the true initial covariance of $\tilde{x}$ and x_a.†

The true error covariance for this model is propagated according to the following recursions.

---

†Typically, the initial factors are such that

$[U_{\tilde{x}}, D_{\tilde{x}}] = [U_x, D_x]$ and $U_{\tilde{x}x} = 0$, where $U_x D_x U_x^T = P_a(0)$.

Covariance Measurement Update

$$U_{\tilde{x}x} := U_{\tilde{x}x} - G(AU_{\tilde{x}x} - \Delta A U_x); \quad \Delta A = A_a - A \tag{5.42}$$

Apply the Bierman algorithm, Eqs. (2.69)-(2.79) to obtain

$$[U_{\tilde{x}}, D_{\tilde{x}}, A, r_a] \rightarrow [\bar{U}_{\tilde{x}}, \bar{D}_{\tilde{x}}, \alpha, K] \tag{5.43}$$

Let

$$\lambda = K - G \tag{5.44}$$

Compute the updated factors $[U_{\tilde{x}}, D_{\tilde{x}}]$ by applying the Agee-Turner triangular factorization (Appendix C) to the identity

$$U_{\tilde{x}} D_{\tilde{x}} U_{\tilde{x}}^T = \bar{U}_{\tilde{x}} \bar{D}_{\tilde{x}} \bar{U}_{\tilde{x}}^T + \alpha \lambda \lambda^T \tag{5.45}$$

Covariance Time Update

$$U_{\tilde{x}x} := \Phi U_{\tilde{x}x} - \Delta\Phi U_x; \quad \Delta\Phi = \Phi_a - \Phi \tag{5.46}$$

Let

$$W = \left[\begin{array}{c|c|c} -Ba & \Phi U_{\tilde{x}} & U_{\tilde{x}x} \\ \hline B_a & 0 & \Phi_a U_x \end{array}\right] \tag{5.47}$$

$$\bar{D} = \text{diag}\,(Q_a, D_{\tilde{x}}, D_x) \tag{5.48}$$

Apply MWGS algorithm,† Eqs. (3.28)-(3.30), to the W and $\bar{D}$ arrays to obtain

$$[W, \bar{D}] \rightarrow [U_{\tilde{x}}, U_{\tilde{x}x}, U_x, D_{\tilde{x}}, D_x] \tag{5.49}$$

At any stage, the true estimate error covariance is given by

$$P_{\tilde{x}} = U_{\tilde{x}} D_{\tilde{x}} U_{\tilde{x}}^T + U_{\tilde{x}x} D_x U_{\tilde{x}x}^T \tag{5.50}$$

---

†The modified Givens update, Eqs. (3.80)-(3.89), could also be used here.

This general error analysis technique inherits the computational efficiency of the U-D filtering algorithms upon which it is based. The evaluation scheme is easily implemented, requiring only minor modifications in the U-D filtering equations. The following sections illustrate how this method is easily adapted to handle some important reductions of the general error analysis problem.

5.3.2 Evaluation of Mismodeled Biases and Process Noise Parameters

Consider the following application of the U-D error analysis technique. Suppose the parameter vector, X, is partitioned such that

$$X = \begin{bmatrix} x_1 \\ -- \\ x_2 \end{bmatrix} \begin{matrix} \}n_1 \\ \\ \}n_2 \end{matrix} \tag{5.51}$$

and let ΔA , $\Delta\Phi$, and Φ_a have the form

$$\Delta A = \overset{\overset{n_1}{\sim}}{[0} \mid \overset{\overset{n_2}{\sim}}{\Delta A_2}] \; \}1 \tag{5.52}$$

$$\Delta\Phi = \overset{\overset{n_1}{\sim}}{[0} \mid \overset{\overset{n_2}{\sim}}{\Delta\Phi_2}] \; \}n_1+n_2 \tag{5.53}$$

$$\Phi_a = \begin{bmatrix} \overset{\overset{n_1}{\sim}}{\Phi_1} & \overset{\overset{n_2}{\sim}}{\Phi_{12}} \\ \hline 0 & \Phi_2 \end{bmatrix} \begin{matrix} \}n_1 \\ \}n_2 \end{matrix} \tag{5.54}$$

For this problem the error equations corresponding to (5.23) and (5.25) reduce to the following.

$$\Delta\hat{X} = (I-GA)\Delta\tilde{X} + G\Delta A_2 x_2 + G\nu_a \tag{5.55}$$

$$\Delta\tilde{X} = \Phi\Delta\hat{X} - \Delta\Phi_2 x_2 - B_a w_a \tag{5.56}$$

Hence, estimate errors contain only the x_2 portion of the actual state. Notice also that x_2 is uncoupled from x_1 by Eq. (5.54). Thus the error analysis corresponding to Eqs. (5.29)-(5.40) simplifies and involves only the vectors ΔX and x_2. Let B_a be partitioned compatibly with Φ_a. In this case the error equations (5.30) and (5.38) take the form

$$\begin{matrix} n_1+n_2\{ \\ n_2\{ \end{matrix} \begin{bmatrix} \Delta\hat{X} \\ -- \\ x_2 \end{bmatrix} = \overbrace{\begin{bmatrix} G & (I-GA)U_{\tilde{X}} & \bar{U}_{\tilde{X}x} \\ --- & ---------- & ---- \\ 0 & 0 & U_x \end{bmatrix}}^{1 \quad n_1+n_2 \quad n_2} \begin{bmatrix} \nu_a \\ -- \\ \nu_{\tilde{X}} \\ -- \\ \nu_x \end{bmatrix} \begin{matrix} \}1 \\ \}n_1+n_2 \\ \}n_2 \end{matrix} \tag{5.57}$$

$$\begin{matrix} n_1+n_2\{ \\ n_2\{ \end{matrix} \begin{bmatrix} \Delta\tilde{X} \\ -- \\ x_2 \end{bmatrix} = \overbrace{\begin{bmatrix} -B_a & \Phi U_{\tilde{X}} & \bar{U}_{\tilde{X}x} \\ ---- & ------ & ------ \\ B_2 & 0 & \Phi_2 U_x \end{bmatrix}}^{k \quad n_1+n_2 \quad n_2} \begin{bmatrix} w_a \\ -- \\ \nu_{\tilde{X}} \\ -- \\ \nu_x \end{bmatrix} \begin{matrix} \}k \\ \}n_1+n_2 \\ \}n_2 \end{matrix} \tag{5.58}$$

where

$$\bar{U}_{\tilde{X}x} = (I-GA)U_{\tilde{X}x} + G\Delta A_2 U_x \tag{5.59}$$

$$\bar{U}_{\tilde{X}x} = \Phi U_{\tilde{X}x} - \Delta\Phi_2 U_x \tag{5.60}$$

Thus, in Eqs. (5.42)-(5.47), the matrices Φ_a, $\Delta\Phi$ and ΔA reduce to Φ_2, $\Delta\Phi_2$ and ΔA_2, respectively. The array B_a in the lower portion of W, Eq. (5.47), reduces to B_2 (cf 5.58).

Suppose that x_2 represents biases or colored noise parameters, so that Φ_2 is diagonal. In this case, each mismodeled component of x_2 introduces only one additional "true" state into the error analysis; i.e., one additional row and column onto the augmented U and D arrays in Eqs. (5.42)-(5.49). Thus, filter sensitivity to mismodeled biases and colored noise parameters may be easily and efficiently evaluated.

5.3.3 Sensitivity Analysis of Unestimated Process Noise Parameters

Another important reduction of the general error analysis problem involves the evaluation of unmodeled colored noise parameters. In this case the estimate error recursions (5.29)-(5.40) are written in terms of Δx and p, where x represents the parameters estimated by the filter and p denotes the unmodeled parameters to be evaluated. The Φ_a, ΔA and $\Delta\Phi$ arrays in Eqs. (5.42)-(5.47) for this case reduce to $\Phi_a=\Phi_p$, $\Delta A=A_p$ and $\Delta\Phi=\Phi_{xp}$. Further modifications may be required in the W and $\bar{D}$ arrays (Eqs. 5.47 and 5.48), depending on the white process noise model used in the evaluation.

The effects of unmodeled bias parameters could also be evaluated by this reduction of the general error analysis algorithm. However, bias effects are more efficiently evaluated by the sensitivity analysis technique described in the next section.

5.4 Variable Dimension Filtering and Sensitivity Analysis

5.4.1 Considering the Effects of Unestimated Bias Parameters

When dealing with higher dimensioned systems involving large numbers of bias-type parameters, it is often necessary or desirable to neglect certain parameters in the filter model. Reasons for studying a reduced order problem include practical considerations of computer time and storage and concern that high-order filters can be overly sensitive to numeric effects. The following error analysis technique is of prime importance in the design and evaluation of reduced-order filters.

Suppose the parameter estimation problem involves the vector

$$X = \begin{bmatrix} x \\ y \end{bmatrix} \begin{matrix} \}n \\ \}b \end{matrix}$$

The parameters x are to be estimated and suppose y are either a) estimated, b) neglected in the filter model, or c) neglected in the filter model but considered when the true covariance of x is computed. Let the optimum estimate of the complete problem be $\hat{X}$ with error covariance $\hat{P} = \hat{U}\hat{D}\hat{U}^T$ where $\hat{U}$ and $\hat{D}$ are partitioned consistently with $\hat{X}$.

Thus

$$\hat{U} = \begin{bmatrix} \overbrace{U_x}^{n} & \overbrace{U_{xy}}^{b} \\ 0 & U_y \end{bmatrix} \begin{matrix} \}n \\ \}b \end{matrix} \quad \text{and} \quad \hat{D} = \operatorname{diag}\left(\overbrace{D_x}^{n}, \overbrace{D_y}^{b}\right) \tag{5.61}$$

In terms of second-order statistics the U-D factorization of $\hat{P}$ implies that estimate errors can be written as

$$\begin{matrix} n\{ \\ b\{ \end{matrix} \begin{bmatrix} \Delta\hat{x} \\ \Delta\hat{y} \end{bmatrix} = \overset{\begin{matrix} n & b \end{matrix}}{\begin{bmatrix} U_x & U_{xy} \\ 0 & U_y \end{bmatrix}} \begin{matrix} \}n \\ \}b \end{matrix} \begin{bmatrix} \nu_x \\ \nu_y \end{bmatrix} \tag{5.62}$$

The uncorrelated random vectors ν_x and ν_y have covariances D_x and D_y respectively. Equation (5.62) implies that

$$\Delta\hat{x} = U_x\nu_x + U_{xy}U_y^{-1}\Delta\hat{y} \tag{5.63}$$

Let $\hat{x}^c$ represent the estimate of x that corresponds to a n-dimensional filter, i.e., $\Delta\hat{y} = 0$ with probability one. If $\hat{P}_x{}^c$ denotes the error covariance of $\hat{x}^c$ then Eq. (5.63) implies that

$$\hat{P}_x{}^c = U_xD_xU_x^T \tag{5.64}$$

Note further from Eq. (5.63) that the "sensitivity" matrix, Σ, which relates the x estimates to errors in the y parameters is given by

$$\Sigma = U_{xy}U_y^{-1} \tag{5.65}$$

Let $\hat{x}$ and $\hat{y}$ represent the optimal parameter estimates based on the complete model and let $\hat{P}_x$ and $\hat{P}_y$ denote their respective covariances. Denote $\hat{P}_x{}^{(con)}$ as the "consider" covariance of $\hat{x}^c$, i.e., the error covariance based on the complete model. Then Eqs. (5.63)-(5.65) yield

$$\hat{P}_x = \hat{P}_x^c + \Sigma \hat{P}_y \Sigma^T \tag{5.66}$$

$$\hat{P}_x(\text{con}) = \hat{P}_x^c + \Sigma P_y(o) \Sigma^T \tag{5.67}$$

where

$$\hat{P}_y = U_y D_y U_y^T \tag{5.68}$$

and $P_y(o)$ is the a priori covariance of the y parameters. The covariance relations in Eqs. (5.64)-(5.68) were first recognized by Bierman [1976a]. He showed that $\hat{x}$ and $\hat{x}^c$ are related as follows.

$$\hat{x}^c = \hat{x} - \Sigma \hat{y} \tag{5.69}$$

The reader may wish to verify from Eqs. (2.69)-(2.79) that U_x and D_x are computed independently and are not affected by the last m columns in the U and D arrays.

This "consider" error analysis is easily extended to the discrete linear filtering problem studied in Chapter III if the transition equation has the form

$$\begin{matrix} n\{ \\ b\{ \end{matrix} \begin{bmatrix} x \\ y \end{bmatrix}_{i+1} = \begin{bmatrix} \overbrace{\Phi_x}^{n} & \overbrace{\Phi_{xy}}^{b} \\ 0 & \Phi_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}_i + \begin{bmatrix} \overbrace{B}^{k} \\ 0 \end{bmatrix} w_i \tag{5.70}$$

where $E\{w_i\} = 0$ and $E\{w_i w_i^T\} = \text{diag}\,(q_1, \cdots, q_k)$.

The results of Chapter III imply that the U-D time update for this system may be accomplished by a square-root-free Givens triangularization of the array

$$W = \begin{matrix} \overset{k}{\sim} & \overset{n}{\sim} & \overset{b}{\sim} \\ \end{matrix} \begin{bmatrix} B & W_x & W_{xy} \\ 0 & 0 & W_y \end{bmatrix} \begin{matrix} \}n \\ \}b \end{matrix} \tag{5.71}$$

involving the diagonal $\bar{D} = (Q, D_x, D_y)$. The elements of W in Eq. (5.71) are given by $W_x = \Phi_x U_x$, $W_y = \Phi_y U_y$, and $W_{xy} = \Phi_x U_{xy} + \Phi_{xy} U_y$. From Eqs. (3.75)-(3.77) it is clear that after the last b columns of W and $\bar{D}$ are packed by the modified Givens algorithm we are left with

$$W^{(n)} = \begin{matrix} \overset{k}{\sim} & \overset{n}{\sim} & \overset{b}{\sim} \\ \end{matrix} \begin{bmatrix} B & W_x & \tilde{U}_{xy} \\ 0 & 0 & \tilde{U}_y \end{bmatrix} \begin{matrix} \}n \\ \}b \end{matrix} \tag{5.72}$$

$$\bar{D}^{(n)} = (Q, D_x, \tilde{D}_y) \tag{5.73}$$

The $\tilde{U}_x$ and $\tilde{D}_x$ factors are then obtained by applying the modified Givens algorithm further to the arrays $W' = [B \quad W_x]$ and $D' = (Q, D_x)$. Thus, $\tilde{U}_x$ and $\tilde{D}_x$ are computed in a way which is independent of the y parameters. For this reason the "consider" error analysis, Eqs. (5-64)-(5.69), applies directly to systems of the form (5.70). A separate analysis is required to evaluate the effects of neglected stochastic parameters such as colored process noise (cf section 5.2.3).

Bierman [1976a] has noted the following features of the "consider" analysis technique.

1) From the results of a single large filter which includes all of the relevant parameters it is possible to evaluate a variety of filter models. Thus, beginning with the complete solution one can recursively compute estimates for models with n_i parameters from estimates corresponding to the n_{i+1}-dimensional model where $n_i < n_{i+1}$.

2) Parameters may be rearranged for additional flexibility. That is, one may decide to estimate the last b parameters and consider the first n. In that case the error equation (5.62) must be rearranged and has the form

$$\begin{bmatrix} \Delta\hat{y} \\ \Delta\hat{x} \end{bmatrix} = \begin{bmatrix} \overbrace{U_y}^{b} & \overbrace{0}^{n} \\ U_{xy} & U_x \end{bmatrix} \begin{bmatrix} \nu_y \\ \nu_x \end{bmatrix} \tag{5.74}$$

This expression may be triangularized by applying one of the U-D factorization algorithms in Chapter III. For example, a MWGS factorization may be applied to the arrays

$$W = \begin{bmatrix} U_y & 0 \\ U_{xy} & U_x \end{bmatrix} \quad \text{and} \quad \bar{D} = \text{diag}\,(D_y, D_x)$$

to yield

$$\begin{bmatrix} \Delta\hat{y} \\ \Delta\hat{x} \end{bmatrix} = \begin{bmatrix} U'_y & U'_{yx} \\ 0 & U'_x \end{bmatrix} \begin{bmatrix} \nu'_y \\ \nu'_x \end{bmatrix} \tag{5.75}$$

where ν'_y and ν'_x are uncorrelated and have covariances D'_y and D'_x respectively.

3) When the a priori covariance $P_y(o)$ in Eq. (5.67) is diagonal, the columns of $\Sigma\, P_y(o)^{1/2}$ represent 1σ perturbations in the estimate errors due to the corresponding y parameter. This association is useful in determining filter sensitivity to various parameters which may be omitted from the final filter model.

4) The matrix inversion in Eq. (5.65) involves a triangular matrix and, thus, can be easily accomplished by back substitution methods.

5) The U-D variable-dimension filtering method is more flexible, more reliable and considerably more efficient than corresponding techniques employing partitioning of the covariance matrices (of Friedland [1969]).

5.4.2 Consider Filter

The consider filter to be discussed here is a generalization of the method described by Schmidt [1968]. The filter model includes

parameters x and y where only the x variables are estimated. The y parameters and their associated error covariance are retained at the a priori values. This method differs from the consider error analysis studied in the previous section since the gain, G_x, used to update the x parameters is computed from the full "true" covariance on x and y.

This type of suboptimal filter might be used, for example, in situations where the y parameters cannot be accurately estimated or when estimating them might lead to an overly optimistic y covariance. In the latter situation the filter is, in effect, reduced to an x filter. When it is believed that a reduced-order filter cannot accurately estimate x, the consider filter is sometimes proposed (cf Bierman [1976b]).

Error propagation for the consider filter takes the form

$$\begin{matrix} n\{ \\ b\{ \end{matrix} \begin{bmatrix} \Delta\hat{x} \\ -- \\ \Delta\hat{y} \end{bmatrix} = \begin{bmatrix} \Delta\tilde{x} \\ -- \\ \Delta\tilde{y} \end{bmatrix} + \begin{bmatrix} G_x \\ -- \\ 0 \end{bmatrix} [z - A_x \tilde{x}] \tag{5.76}$$

$$\begin{matrix} n\{ \\ b\{ \end{matrix} \begin{bmatrix} \Delta\tilde{x} \\ -- \\ \Delta y \end{bmatrix} = \begin{bmatrix} \overbrace{\Phi_x}^{n} & \overbrace{\Phi_{xy}}^{b} \\ 0 & \Phi_y \end{bmatrix} \begin{bmatrix} \Delta\hat{x} \\ -- \\ \Delta\hat{y} \end{bmatrix} - \begin{bmatrix} \overbrace{B_a}^{k} \\ -- \\ 0 \end{bmatrix} w_a \tag{5.77}$$

where

$$z = A_x x + A_y y + v_a \tag{5.78}$$

$$\tilde{y} = E\{y\} = 0 \tag{5.79}$$

If the U-D factorization is assumed for the true covariances $\hat{P}$ and $\tilde{P}$ associated with these errors, then Eq. (5.76) may be interpreted as follows

$$\begin{matrix} n\{ \\ b\{ \end{matrix} \begin{bmatrix} \Delta\hat{x} \\ \hline \Delta\hat{y} \end{bmatrix} = \overset{\begin{matrix} n & b \end{matrix}}{\left[\begin{array}{c|c} I-G_xA_x & -G_xA_y \\ \hline 0 & I \end{array}\right]} \overset{\begin{matrix} n & b \end{matrix}}{\left[\begin{array}{c|c} \tilde{U}_x & \tilde{U}_{xy} \\ \hline 0 & \tilde{U}_y \end{array}\right]} \begin{bmatrix} \nu_x \\ \nu_y \end{bmatrix} + \begin{bmatrix} G_x \\ \hline 0 \end{bmatrix} \nu_a \qquad (5.80)$$

where ν_x, ν_y and ν_a are uncorrelated with diagonal covariances D_x, D_y and r_a, respectively. Equation (5.80) may be rearranged to yield

$$\begin{bmatrix} \Delta\hat{x} \\ \hline \Delta\hat{y} \end{bmatrix} = \overset{\begin{matrix} 1 & n & b \end{matrix}}{\left[\begin{array}{c|c|c} G_x & (I-G_xA_x)\tilde{U}_x & \bar{U}_{xy} \\ \hline 0 & 0 & \tilde{U}_y \end{array}\right]} \begin{matrix} \\ \left[\begin{array}{c} \nu_a \\ \hline \nu_x \\ \hline \nu_y \end{array}\right] \end{matrix} \begin{matrix} \}1 \\ \}n \\ \}b \end{matrix} \qquad (5.81)$$

where

$$\bar{U}_{xy} = \tilde{U}_{xy} - G_x[A_x\tilde{U}_{xy} + A_y\tilde{U}_y] \qquad (5.82)$$

The covariance update corresponding to (5.76) may be interpreted as a triangularization of the error expression (5.81) such that

$$\begin{bmatrix} \Delta\hat{x} \\ \hline \Delta\hat{y} \end{bmatrix} = \overset{\begin{matrix} n & b \end{matrix}}{\left[\begin{array}{c|c} \hat{U}_x & \hat{U}_{xy} \\ \hline 0 & \hat{U}_y \end{array}\right]} \begin{matrix} \left[\begin{array}{c} \hat{\nu}_x \\ \hline \hat{\nu}_y \end{array}\right] \end{matrix} \begin{matrix} \}n \\ \}b \end{matrix} \qquad (5.83)$$

where $\hat{v}_x$ and $\hat{v}_y$ are uncorrelated with diagonal covariances $\hat{D}_x$ and $\hat{D}_y$ respectively. The constraint that

$$\mathcal{P} = E\left\{\begin{bmatrix}\Delta\hat{x}\\ \Delta\hat{y}\end{bmatrix}\begin{bmatrix}\Delta\hat{x}^T & \Delta\hat{y}^T\end{bmatrix}\right\}$$

be unchanged by the transformation from (5.81) to (5.83) yields

$$[\hat{U}_y, \hat{D}_y] = [\tilde{U}_y, \tilde{D}_y] \tag{5.84}$$

$$\hat{U}_{xy} = \tilde{U}_{xy} \tag{5.85}$$

$$\hat{U}_x\hat{D}_x\hat{U}_x^T = (I-G_xA_x)\tilde{U}_x\tilde{D}_x\tilde{U}_x^T(I-G_xA_x)^T + G_xr_aG_x^T \tag{5.86}$$

If G_x were the Kalman gain associated with the reduced-order problem having a priori covariance factors $\tilde{U}_x$ and $\tilde{D}_x$, then the factors $\hat{U}_x$ and $\hat{D}_x$ could be obtained by applying the Bierman U-D measurement update algorithm to Eq. (5.86). Note, however, that for the generalized Schmidt consider filter, the gain G_x is computed from the full consider covariance. Thus, G_x is obtained from the identity

$$\begin{matrix}n\{\\ b\{\end{matrix}\begin{bmatrix}G_x\\ G_y\end{bmatrix} = \begin{bmatrix}\tilde{U}_x & \tilde{U}_{xy}\\ 0 & \tilde{U}_y\end{bmatrix}\begin{bmatrix}\tilde{D}_x & 0\\ 0 & \tilde{D}_y\end{bmatrix}\begin{bmatrix}\tilde{U}_x^T & 0\\ \tilde{U}_{xy}^T & \tilde{U}_y^T\end{bmatrix}\begin{bmatrix}A_x^T\\ A_y^T\end{bmatrix}\left(\frac{1}{a}\right) \tag{5.87}$$

where

$$a = r_a + v^Tf \tag{5.88}$$

$$f^T = [A_x \quad A_y]\begin{bmatrix} \tilde{U}_x & \tilde{U}_{xy} \\ 0 & \tilde{U}_y \end{bmatrix} = (\overbrace{f_x^T}^{n}, \overbrace{f_y^T}^{b}) \tag{5.89}$$

$$v^T = (\overbrace{f_x^T\tilde{D}_x}^{n}, \overbrace{f_y^T\tilde{D}_y}^{b}) \tag{5.90}$$

For this reason, we compute $\hat{U}_x$ and $\hat{D}_x$ by applying the U-D arbitrary gain update algorithm (section 5.2) to Eq. (5.36). The first phase of this update applies the Bierman algorithm, Eqs. (2.69)-(2.79), to obtain the gain $K = \bar{K}_n/\alpha_n$, corresponding to the reduced-order problem with a priori factors $\tilde{U}_x$ and $\tilde{D}_x$. Note that the full-dimension gain in Eq. (5.87) is given by $G = \bar{K}_{n+b}/\alpha_{n+b}$ where $\bar{K}_{n+b}$ and α_{n+b} may be obtained by cycling further through Eqs. (2.74) and (2.78) for $j = n+1,\ldots,n+b$. In this recursion only the first n elements of the vector $\bar{K}_j$ need be computed since the gain G_y is not required.

Time propagation via (5.77) of the full consider covariance U-D array is easily accomplished by a MWGS or modified Givens factorization applied to

$$W = \left[\begin{array}{c|c|c} \overbrace{B_a}^{k} & \overbrace{\Phi_x\hat{U}_x}^{n} & \overbrace{U'_{xy}}^{b} \\ \hline 0 & 0 & \Phi_y\hat{U}_y \end{array}\right]\begin{matrix}\}n \\ \}b\end{matrix} \tag{5.91}$$

and

$$\bar{D} = \text{diag}\,(Q_a,\ \hat{D}_x,\ \hat{D}_y) \tag{5.92}$$

where

$$U'_{xy} = \Phi_x \hat{U}_{xy} + \Phi_{xy} \hat{U}_y \tag{5.93}$$

Chapter VI. Numerical Comparison of Discrete Kalman Filtering Algorithms

6.1 Introduction

The numerical stability and accuracy of various discrete Kalman filtering algorithms have been carefully studied by applying these methods to a realistic interplanetary navigation problem. Included in the study were the conventional and stabilized Kalman algorithms, the Potter-Schmidt square root filter, and the U-D factorization filter. The Carlson-Schmidt algorithm was omitted from the comparison since this method clearly shares the numerical characteristics of the U-D filter (cf Chapters II and III).

Each of the algorithms was implemented on the UNIVAC 1108 computer which has a 60 bit characteristic (18 decimal digits) in double precision and 27 bit characteristic (8-9 decimal digits) in single precision. The complexity of the study problem prohibits closed form solutions. However, numerical solutions from all the algorithms, using double precision arithmetic, agreed to 8 digits or more. These double precision results were used as a reference for evaluating the accuracies of the same algorithms computing in single precision.

6.2 Problem Description

A portion of the 1977 Mariner Jupiter-Saturn (MJS) deep space mission was chosen for the filter comparison study. The problem involves spacecraft navigation for the 30 day period immediately preceding Saturn encounter (point of closest approach). For the initial 20 days

the spacecraft trajectory is nearly rectilinear, a characteristic which is typical of a major portion of most deep space missions. The last segment of the trajectory has a hyperbolic bend due to the gravitational effects of Saturn. Hence, tracking data from this phase of the mission is particularly useful for an accurate determination of the planetary mass and the spacecraft position and velocity. This trajectory is thus characteristic of a significant class of orbit determination problems.

The nominal trajectory and state transition matrices were obtained by integrating the equations of motion and variational equations (cf Moyer [1971]). Earth-based measurements of the spacecraft include both doppler and ranging data. Partial derivatives of the data with respect to the system parameters were evaluated analytically about the nominal trajectory using the orbit determination software described by Moyer. All of these calculations were performed in an earth-centered, cartesian coordinate system.

Perturbations from the nominal trajectory and simulated data were constructed from a discrete linear model. The complete model includes differential corrections in 19 parameters; the spacecraft position, velocity and acceleration components (3 each); the gravitational constant of Saturn (GM); and tracking station locations (3 cartesian components for each of 3 stations). The state transition for these perturbations is described by the following equation.

$$\begin{bmatrix} x \\ p \\ y \end{bmatrix}_{i+1} = \begin{bmatrix} \Phi_x & \Phi_{xp} & \Phi_{xy} \\ 0 & M & 0 \\ 0 & 0 & I \end{bmatrix}_i \begin{bmatrix} x \\ p \\ y \end{bmatrix}_i + \begin{bmatrix} 0 \\ w \\ 0 \end{bmatrix}_i \qquad (6.1)$$

The vector x denotes the 6 components of position and velocity, p represents the 3 acceleration components, and y represents the 10 bias parameters. Accelerations are modeled as piece-wise constant colored noise with time constants of 12 hours and standard deviations of 10^{-11} km/sec^2; and these define the variances of the white noise, w_i, in Eq. (6.1). The discrete times $\{t_i\}$ are assumed to be equally spaced with $t_{i+1} - t_i = \Delta t$ taken as 2 hours. Thus w_i has covariance $Q = qI_3$ where

$$q = (1 - m^2)\,\sigma_p^2 = (1 - e^{-1/3})\,10^{-22}\ (\text{km/sec}^2)^{\wedge} \qquad (6.2)$$

Let X represent the complete state vector in (6.1). The initial vector X_o has zero mean and diagonal covariance P_o. The nontrivial elements of P_o are defined by the standard deviations in Table 6.1. These a priori statistics are typical assumptions for the kind of orbit determination problem considered here (cf O'Neil, et al. [1973]).

The simulated state X_o is obtained from a Gaussian random variable generator with zero mean and covariance P_o.† The actual state is then propagated according to Eq. (6.1) where the components of w_i are obtained from the Gaussian random variable generator with covariance Q.

---

†Each component of X_o is obtained by using a N(0,1) random number generator followed by an appropriate scaling of the sampled value.

Table 6.1. A Priori Statistics Used to Generate Sample Path

Variable	Standard Deviation
Position	1000 km
Velocity	100 m/s
Acceleration	10^{-11} km/sec^2 (τ = 12 hrs)
Station location errors	spin axis - 1 meter Longitude - 2 meter Latitude - 5 meter
GM (Saturn)	.1%

Data for the linear simulation are generated from the equation

$$z_i = A_i X_i + \nu_i \tag{6.3}$$

where the elements of A_i are partial derivative coefficients evaluated along the nominal trajectory, and ν_i is white data noise obtained from a Gaussian random number generator using the appropriate measurement error covariance. The simulation includes two or three doppler measurements every two hours with 1 mm/sec accuracy (for 1 minute averaging time) and occasional range points with an accuracy of 3 meters. There were a total of 535 doppler and 72 range measurements in the 30 day tracking period.

This simulation is representative of a large number of interplanetary navigation problems. The a priori statistics are in no way contrived to produce poor results in the conventional algorithms. On the contrary, a priori state covariances were chosen on the small side for problems of this type in order to avoid initialization errors associated with the Kalman algorithms. Similarly, process noise uncertainties were assumed to be an order of magnitude higher than usual for this kind of mission† because the Kalman algorithms generally experience less numerical difficulties in high process noise environments.

This estimation problem is well posed in an engineering sense. The problem is observable; the transition matrix is not ill-conditioned; and the measurement coefficients and a priori error variances are not unusually large. Thus, the results of this study should be of interest to the entire estimation and control community.

6.3 Filter Implementations

The five covariance-type filter algorithms compared in this study are the conventional Kalman filter, Joseph's stabilized Kalman filter, a conventional Kalman filter with lower bounding, the Potter-Schmidt square root filter, and the U-D factorization filter with MWGS time propagation. A limited evaluation of the U-D filter using the modified Givens update is also included. Details of the various filter algorithms are given in Chapters II and III.

---

†See Christensen [1976] or O'Neil, et al. [1973].

Each algorithm involved in the study was mechanized for maximum accuracy and efficiency. Thus, the U-D filter employs the efficient one-at-a-time colored noise update, Eqs. (3.130)-(3.133); while the Potter-Schmidt filter performs time updating in one stage, taking advantage of special system structure and blocks of zeroes.

Computations are also reduced by using vector outer products whenever possible. In the Kalman algorithms, symmetry of the covariance matrix is preserved by computing only the upper triangular elements. The stabilized Kalman filter contains a single exception to this rule.

Significantly improved results were obtained with the stabilized Kalman filter when off-diagonal covariance elements were computed and averaged during a portion of the measurement updating. This averaging takes place during the computation of the first term in Eq. (2.24). Thus all n^2 elements of the array $P = P_1 - v_2 K^T$ are computed, and the off-diagonal elements of $\hat{P}$ are obtained as follows.

$$\hat{P}_{ij} = .5(P_{ij} + P_{ji}) + rK_iK_j \qquad i \neq j \tag{6.4}$$

The fact that numerical results are sensitive to such mechanization details is indicative of the algorithm's instability. Even with the averaging of off-diagonal elements, the stabilized Kalman algorithm performed poorly.

The conventional Kalman filter with lower bounding is designed to perform a covariance measurement update in the following way. The

conventional update formula is first applied to obtain an intermediate array $\bar{P}$. Thus,

$$\bar{P} = \tilde{P} - KA\tilde{P} \tag{6.5}$$

The updated covariance $\hat{P}$ is then computed from Eqs. (6.6)-(6.8).

$$\hat{P}_{jj} = \max(\bar{P}_{jj}, \sigma^2_{min}(j)) \qquad j = 1,\ldots,n \tag{6.6}$$

$$\hat{P}_{ij} = \begin{cases} \bar{P}_{ij} & \text{if } \bar{P}^2_{ij} < M_{ij} \\ \operatorname{sgn}(\bar{P}_{ij})\, M_{ij} & \text{otherwise} \end{cases} \tag{6.7}$$

where

$$M_{ij} = \rho^2_{min}\, \hat{P}_{ii}\, \hat{P}_{jj} \qquad i = 1,2,\ldots,j-1 \tag{6.8}$$

The n components of σ_{min} and the correlation ρ_{min} are chosen a priori. This mechanization is typical of the techniques that are frequently used to prevent the computed covariance from becoming indefinite (cf Schmidt et al. [1968]). The mechanization is no optimal and the computed $\hat{P}$ is generally not the actual error covariance. Choosing the bounds σ_{min} and ρ_{min} is something of an art, and appropriate values are usually determined from lengthy simulation studies. This lower bound filter algorithm is included in the comparison study in order to illustrate that ad hoc "patching" techniques can compensate to some extent for the numerical inadequacies of the conventional algorithm.

However, the results of the remaining sections demonstrate that such measures are not necessary when factorization algorithms are used.

All of the filter algorithms were coded and carefully checked using double precision arithmetic. These programs were then converted to single precision by removing the FORTRAN IV "implicit double precision" statement. However, in the single precision programs, inner products are accumulated in double precision and then rounded to single precision. In addition, estimates are retained in double precision because a significant accuracy loss is incurred when single precision arithmetic is used in the estimate propagation, Eqs. (3.11) and (3.13). This accuracy loss would not be attributable to a particular filtering method but would mask the desired comparison.

Since all of the filter programs compute estimates in double precision and use the same (single precision) inputs,† only the gain computations differ. Thus, filtering accuracies may be analyzed by applying the gain evaluation algorithm described in Section 5.2. This evaluation program, implemented in double precision and operating with the same inputs used by the filter, computes the actual, or true, error covariance associated with each gain profile; i.e., each filter algorithm. Hence, two complimentary methods are used to compare filter accuracies. The error analysis program provides a statistical means of comparing

†The state transition and data matrices used in the simulation are computed in double precision and rounded to single precision.

filter performance, while the single simulation results yield an illustration of these statistical predictions.

6.4 Numerical Results

A comparison of filtering accuracies was first obtained for the full 19-state model described in section 6.2. The filter model was then varied in order to evaluate sensitivities to a priori statistics. Reduced-dimension problems were also studied as a means of assessing how numerical stability is affected by model dimensionality. In every case studied, the double precision programs computed estimates and standard deviations that agreed to 8 digits or more. In fact, the factorization algorithms agreed to at least 10 digits. The single precision programs, however, produced widely varying estimates and covariances. The results to be described indicate that, while the single precision Kalman algorithms experience serious numerical deterioration as computer word length decreases, the factorization methods have excellent precision and stability.

6.4.1 The Complete State Model

For this portion of the study, all of the filters assumed the 19-state model described in section 6.2. Thus, the actual and assumed models coincide. For this case and the others to follow, the single precision factorization algorithms computed gains and variances which agreed with the double precision reference results to about 5-6

digits.[†] The single precision Kalman algorithms, however, produced markedly different gain profiles and variances. Numerical deterioration of these algorithms was evident in the form of negative computed variances which appeared at inexplicable times. Although this numerical instabilty is related to the choice of a priori statistics (cf section 6.4.2) negative variances do not appear until after four days of filtering with the conventional algorithm, and after ten days when the stabilized formula is used. This behavior is particularly interesting when we note that in the first four days there are 48 time and 80 measurement updates.

Both the conventional and the stabilized algorithms compute intermittent negative variances. Even the bias parameter variances are intermittently negative. Furthermore, these negative elements are not related to filter variances which are approaching zero. The erratic behavior of the stabilized algorithm illustrates this point. For example, at 9.75 days the stabilized formula computes $\sigma_{GM}^2 \approx -1.8 \times 10^9$ $(km^3/sec^2)^2$ and at ten days adjusts this to 1.7×10^4. The correct (double precision) value, however, is $\sigma_{GM}^2 \approx 5 \times 10^3$ $(km^3/sec^2)^2$.

The extent of the numerical deterioration in the Kalman algorithms is apparent in Figures 6.1 and 6.2. These figures compare the actual

---

[†]The U-D filtering results reported in this section were obtained with both time update methods; the MWGS and the modified Givens algorithms (cf sections 3.3 and 3.4). The latter propagation scheme employed Method B exclusively (cf Eqs. (3.80)-(3.89)) and experienced no numerical degradation; i.e., both filter implementations produced results which agreed to about 5-6 digits with the double precision standard.

position and velocity uncertainties† associated with each filter and were obtained by applying the gain evaluation program.

In Figures 6.1 and 6.2 the U-D and Potter single precision filtering accuracies are shown to agree with the double precision references. Notice, however, the excursions of the conventional and stabilized Kalman single precision accuracy curves. The conventional Kalman results are seen to be particularly inaccurate at 4 days, while the stabilized algorithm deteriorates rapidly at 10 days.

Simulated data, generated using the same model assumed in the error analysis, was filtered by each of the algorithms. Actual estimation errors for this single simulation are compared in Figures 6.3 and 6.4. Notice how closely the gain evaluation results in Figures 6.1 and 6.2 predict the error curves of the sample path. As anticipated, the conventional algorithm in single precision produces large errors at 4 days. The single precision stabilized algorithm also performs poorly, especially after 10 days. However, the estimates and computed standard deviations obtained from the stabilized filter, when monitored at one day intervals, show few signs of numeric deterioration. Except for the times (3) that negative variances are printed, these estimates and statistics appear reasonable and consistent. Only when the results are compared

†The root-sum-square of position and velocity errors are chosen as a measure of estimation accuracy because these parameters are of primary interest in navigation and are representative of the general filtering results obtained in this study.

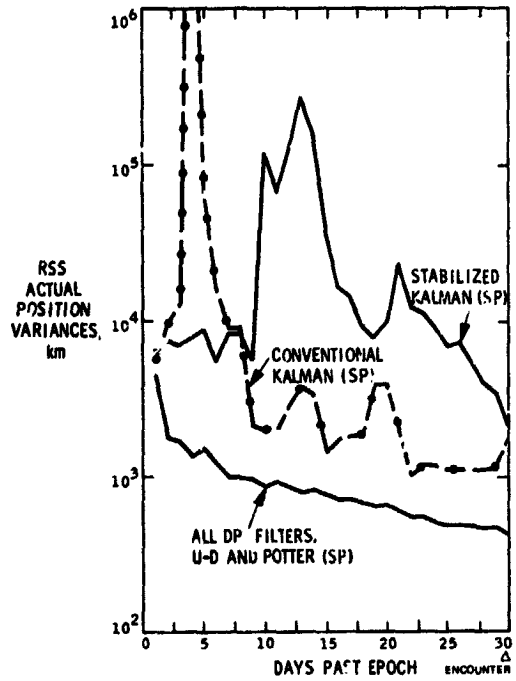

Fig. 6-1. Comparison of Actual Position Uncertainties, Correct 19-State Model

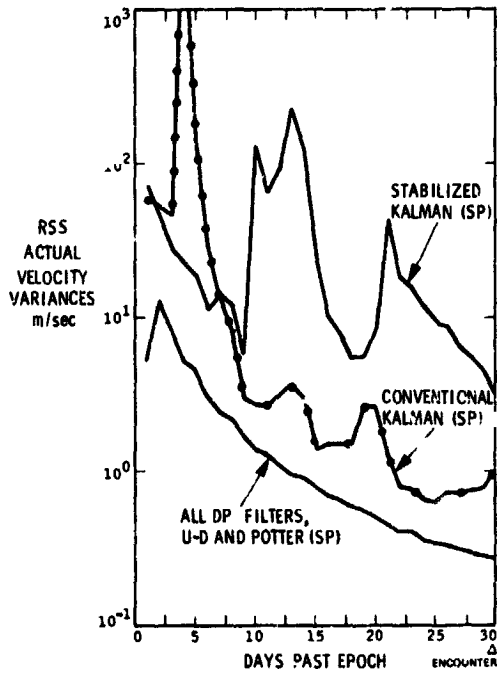

Fig. 6.2. Comparison of Actual Velocity Uncertainties, Correct 19-State Model

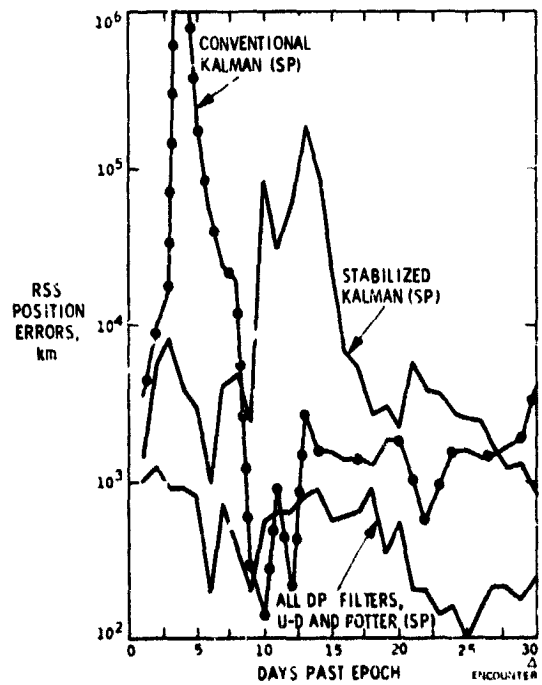

Fig. 6-3. Comparison of Position Errors, 19-State Single Simulation

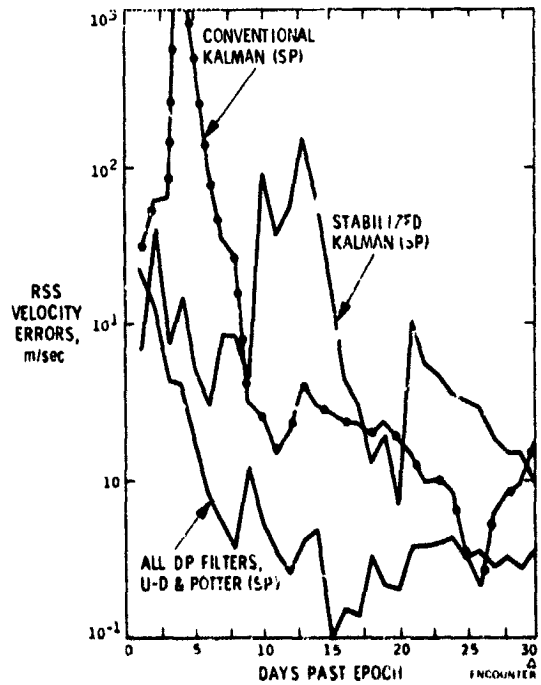

Fig. 6.4. Comparison of Velocity Errors, 19-State Single Simulation

with the double precision references is it apparent that the stabilized Kalman estimates are far from optimum.

Recall that the stabilized formula was introduced as a computational improvement to the conventional method. It is, therefore, surprising to note that after 10 days the conventional algorithm yields significantly more accurate results than the stabilized formula. This phenomenon was not observed in all of the cases studied, however. The point to be made here is that, although the stabilized algorithm does generally give improved performance over the conventional formula, neither method can be considered reliable.

By comparison, estimates computed using the factorization algorithms agree to about 4 or 5 digits with the double precision values. This agreement corresponds to better than 1 km in position and 50 mm/sec in velocity. These single precision accuracies are particularly impressive when it is noted that estimation uncertainties are two orders of magnitude greater than these differences (cf Figures 6.1 and 6.2). In other words, the differences in the single and double precision results are in the "noise" level.

In every case studied, the relative position and velocity accuracies displayed the same general agreement illustrated in Figures 6.1-6.4. In view of this, subsequent discussions are restricted to the comparison of position errors and uncertainties. Thus, unnecessary discussions of velocity results are omitted. For similar reasons, the conventional Kalman algorithm is also omitted from subsequent discussions. The

numerical instability of the conventional algorithm is already well documented (cf Bellantoni and Dodge [1967] or Schmidt [1968]). This method is dismissed from further comparisons with the observation that in nearly every case studied the single precision conventional formula suffered numerical degradations similar to those displayed in Figures 6.1-6.4.

Numerical divergence of the Kalman filter is generally associated with indefinite computed covariance matrices. Hence, it is common practice to attempt to preserve positivity by bounding the diagonal elements from below and to limit the correlations between pairs of variables (cf Eqs. (6.6)-(6.8)). Any such attempt to stabilize the conventional Kalman algorithm introduces a myriad of filtering alternatives. For example, should the lower bound on the velocity uncertainties be 1.0 m/sec or 0.1 m/sec, and should the maximum correlation be .99 or .98? The choice of such patch factors is often problem dependent and may require lengthy simulations.

In this study filtering results are indeed sensitive to the choice of bounds as Figure 6.5 illustrates. This figure displays the RSS position error profiles produced by the single precision patched algorithm for various bounding schemes. By comparing Figures 6.3 and 6.5 one can conclude that patching yields a marked improvement over the stabilized Kalman results. However, all of the error curves associated with the patched algorithm are far above the optimal double precision results in Figure 6.3. Actual filtering uncertainties are compared in

Figure 6.6, and the patched algorithm is shown to be significantly less accurate than the factorization filters.

In all of the cases studied, the patched Kalman filter displayed the same poor performance observed in Figures 6.5 and 6.6. This study demonstrates that the practice of introducing ad hoc patch factors to combat Kalman filter numerical divergence results in algorithms that are significantly less efficient and accurate than the factorization methods. Patching techniques are thus omitted from further consideration.

6.4.2 Effects of Incorrect A Priori Statistics

Numerical difficulties with the Kalman algorithm can often be attributed to initial ill-conditioning caused by large a priori state uncertainties and relatively small data covariances. These problems can be reduced by scaling the a priori statistics. However, any improvement in numerical conditioning is offset somewhat by the effects of suboptimal modeling. Consider, for example, the case where initial velocity uncertainties are reduced by an order of magnitude to 10 m/sec, and range uncertainty is increased to 10 meters (from 3 meters). This combination of a priori statistics, selected by numerical experimentation, allows the stabilized Kalman algorithm to appear "stable." In fact, for this choice of filter statistics neither the conventional nor the stabilized algorithm computes negative variances. Moreover, for this example simulation estimate errors are consistent with the filter formal statistics. This consistency creates the false impression that the Kalman algorithms are performing well when, in fact, they are grossly inaccurate. Actual Kalman accuracies, computed by the gain evaluation

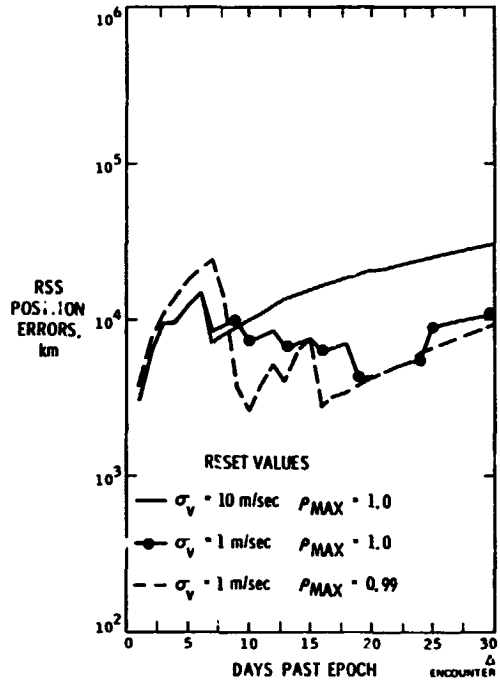

Fig. 6.5. Comparison of Patched Kalman Filter Errors for Different Reset Values, 19-State Single Simulation Results

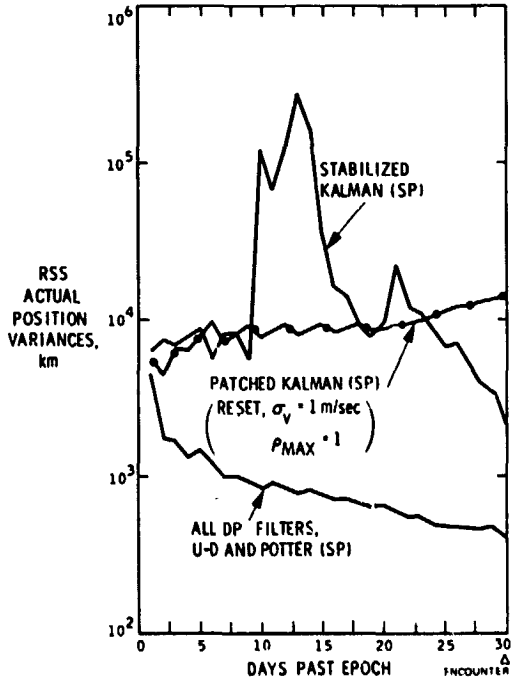

Fig. 6.6. Performance Comparison of Patched and Stabilized Kalman Filters, Complete 19-State Model

program using the correct statistics, are displayed in Figure 6.7. This figure shows that for most of the filtering period the stabilized Kalman uncertainties (using scaled a priori statistics) are an order of magnitude greater than achievable filter accuracies.

The results in Figure 6.7 also show that the Kalman filter is more accurate when suboptimal (σ_v = 10 m/sec, σ_R = 10 meters) rather than optimal (σ_v = 100 m/sec, σ_R = 3 meters) statistics are assumed. This theoretical impossibility is, of course, due to numerical errors which can cause unpredictable results.

When only one of the a priori uncertainties (σ_v or σ_R) is scaled, the single precision Kalman filters continue to produce negative computed variances and unreliable gain profiles. Actual filtering accuracies corresponding to scaled a priori velocity variances are illustrated in Figure 6.8. In this case, σ_v is scaled down by an order of magnitude in the filter model, and gain profiles are evaluated using the correct statistical model. Note that, for this example, initial velocity variances are scaled down by two orders of magnitude. However, instead of improving Kalman filter accuracies, this scaling produced greater errors. Figure 6.8 demonstrates that the stabilized filter (with σ_v = 10 m/sec) yields large uncertainties at 5 days which peak again at 22 days. By comparing Figures 6.1 and 6.8 we can conclude that numerical errors in the Kalman results occur at completely arbitrary times and are unrelated to any physical phenomena peculiar to the problem.

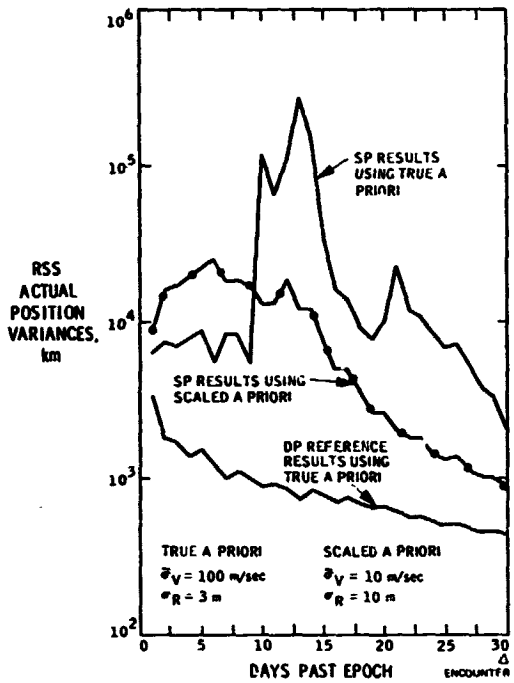

Fig. 6.7. Performance Comparison of Stabilized Kalman Filters Using Scaled A Priori Statistics, 19-State Filter Model

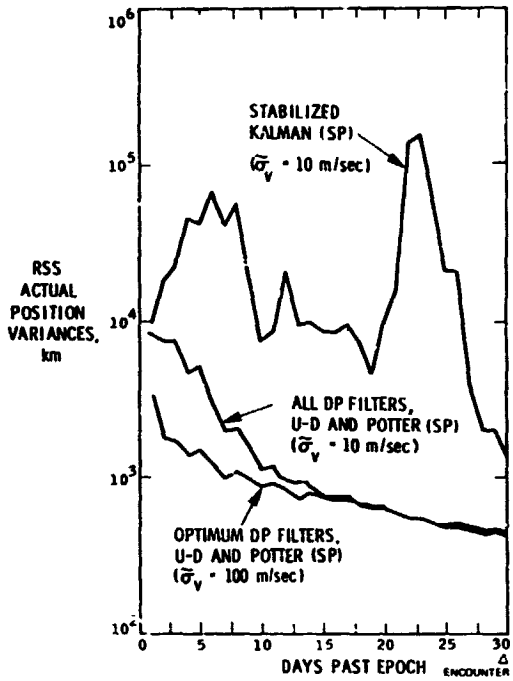

Fig. 6.8. Comparison of Actual Position Uncertainties for Scaled Velocity A Priori, 19-State Filter Model

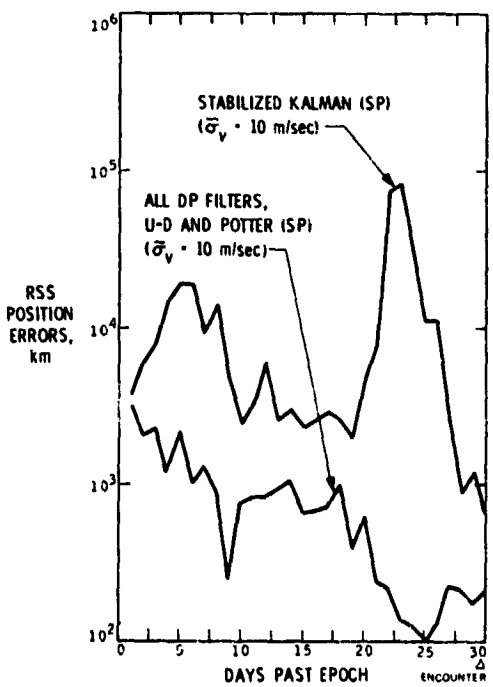

Fig. 6.9. Comparison of Position Errors for Scaled Velocity A Priori, 19-State Single Simulation Results

In a filtering problem with observability and significant amounts of process noise, one would expect estimates to depend primarily on data in the recent past. Thus, estimate error profiles corresponding to different a priori covariance assumptions should, except for initial transient effects, look quite similar. The factorization filters illustrate this effect in the bottom two curves of Figure 6-8. In every case involving scaled a priori statistics, the factorization filters demonstrate the same stability evident in Figure 6.8. The Kalman algorithm, on the other hand, is quite sensitive to the choice of a priori statistics as the topmost curves in Figures 6.7 and 6.8 illustrate.

The single simulation results for this case yield estimate error profiles close to those predicted by the error analysis. Position errors are compared in Figure 6.9. Notice the striking resemblance between the factorization filter error curves in Figures 6.3 and 6.9, particularly after 10 days. As expected from the error analysis, the Kalman results in these two figures appear totally unrelated.

This analysis illustrates how numerical instability can cause unpredictable results which violate established estimation principles.

6.4.3 Reduced-Dimension Problems

The discussions in the previous sections were based upon the complete 19-state model. In this section models of smaller dimension are examined. This study shows, among other things, that the numerical instability of the Kalman algorithms is not caused by the dimensionality of the model; and that the inclusion of process noise improves the

appearance of the computed covariances but not the accuracy of the estimates.

The smallest, physically meaningful model corresponding to the planetary approach problem involves the six position and velocity variables. This six-state system represents a parameter estimation problem since there is no process noise. The Kalman algorithms are known to be numerically unstable for parameter estimation problems. Hence, it is not surprising that the single precision Kalman filters compute intermittent negative variances when the six dimensional model is assumed. However, these 6-state filters, when applied to the simulated data, manage to at least partially track the spacecraft. Actual filtering accuracies for this case are displayed in Figure 6.10. The top two curves in this figure represent the accuracies obtained with the six dimensional filters operating in the complete 19-state environment. Notice that the stabilized Kalman algorithm suffers a severe accuracy degradation in this case. Position uncertainties are two orders of magnitude greater than those obtained with the correct model (cf Figure 6.1). By comparing the factorization results in Figure 6.10 (bottom two curves) one can see that the accuracy loss due to mismodeling is considerable. The stabilized Kalman results in this figure suggest that either this algorithm compounds the effects of mismodeling or numerics represent the dominant error source.

Numerical effects may be further separated from modeling errors by evaluating filtering accuracies for an actual six-dimensional model. Hence, no effects due to mismodeling are present in the accuracy

comparisons of Figure 6.11. Notice that position uncertainties corresponding to the stabilized algorithm are nearly identical in Figures 6.10 and 6.11. These figures demonstrate that numerical errors associated with the stabilized algorithm are so large that they completely obscure the effects of mismodeling. By contrast, the factorization curves in Figures 6.10 and 6.11 clearly indicate how 6-state filtering accuracies are degraded by the presence of unmodeled parameters.

The second model selected for careful study included the three colored noise accelerations in addition to the six position and velocity parameters. This 9-state system was chosen because it includes a significant amount of process noise, and it is generally assumed that high levels of process noise will stabilize the Kalman filter computations.

The stabilized Kalman results appeared to support this theory. That is, the stabilized algorithm computed covariances, gains, and estimates (based on the simulation sample) which looked reasonable. No hint of numerical difficulty was evident. The results, however, differed from those obtained with the U-D and Potter-Schmidt filters. As usual the factorization algorithms agreed closely with the double precision reference results.

Error analysis for this case produced results similar to those observed in the six-dimensional problem. That is, numerical errors in the Kalman calculations were again substantial and completely obscured any effects due to unmodeled parameters. Figure 6.12 shows how severely the Kalman results are degraded by numerical errors. This figure displays

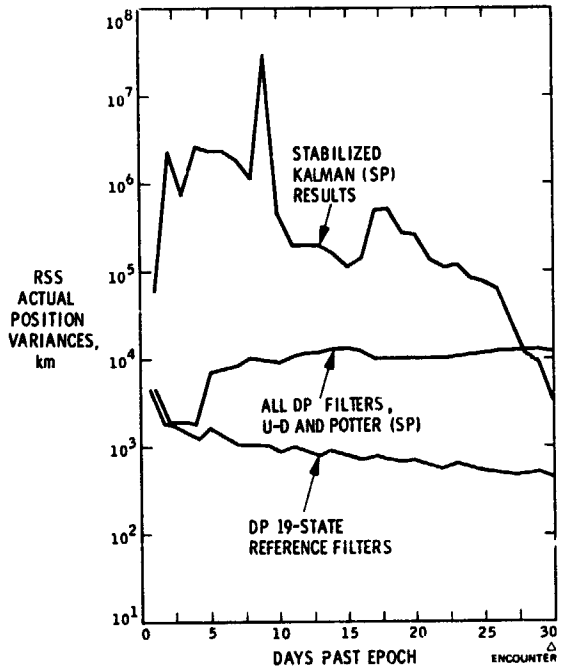

Fig. 6.10. Comparison of Actual Position Uncertainties, 6-State Filter Evaluated for 19-State Model

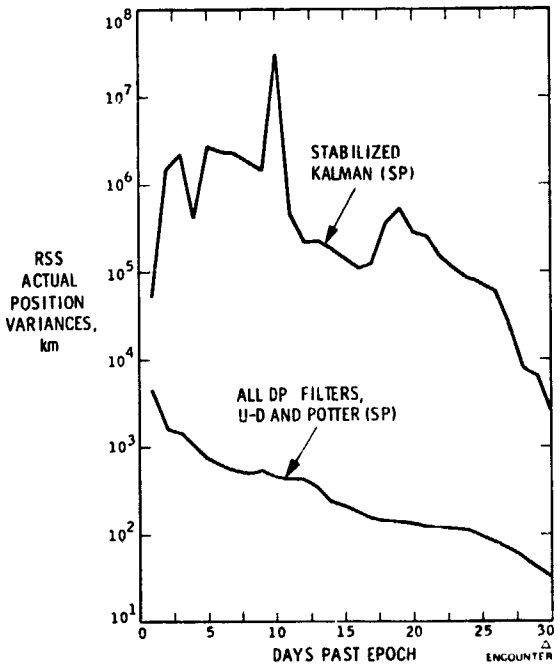

Fig. 6.11. Comparison of Actual Position Uncertainties, 6-State Filter Evaluated for 6-State Model

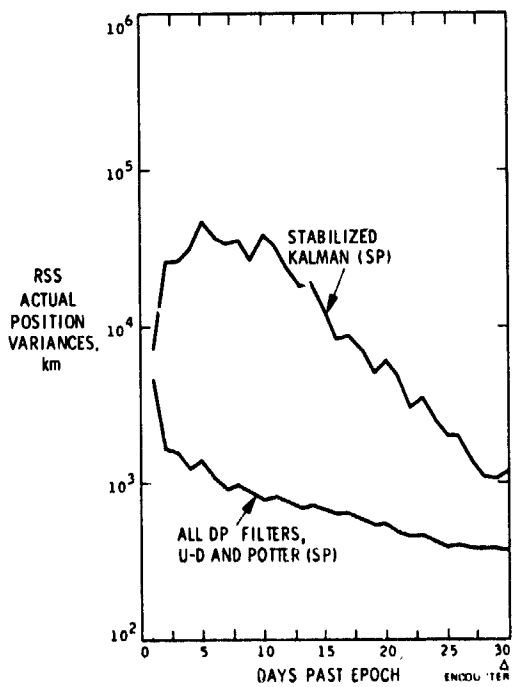

Fig. 6.12. Comparison of Actual Position Uncertainties, 9-State Filter Evaluated for 9-State Model

actual position accuracies associated with the 9-state filters operating in an actual 9-state environment. These results demonstrate that numerical deterioration in the Kalman filter translates into position errors which are an order of magnitude higher than they need be.

This example illustrates that, while the inclusion of process noise improves the performance of the Kalman algorithms (cf Figures 6.11 and 6.12), they still lack the accuracy achievable with the factorization methods.

6.5 Conclusions

This study demonstrates the excellent numerical stability and precision of the U-D and Potter-Schmidt factorization algorithms. Both methods, implemented in single precision, produced results which were close to the double precision references. The numerical stability of these algorithms was further demonstrated by their lack of sensitivity to the choice of a priori variances and process noise levels.

The Kalman filters, on the other hand, were acutely affected by the use of single precision arithmetic and scaled a priori statistics. Wampler [1970] observes that these are sufficient reasons to declare an algorithm numerically unstable and to abandon it. The results of this study support his conclusion. Both the conventional and the stabilized algorithms experienced severe numerical deterioration in nearly every case examined. Covariance matrices with negative diagonal elements were a common occurrence, and gain profiles were often erratic and inaccurate.

Even when the assumed statistics were modified to stabilize the calculations, the single precision Kalman algorithms performed poorly. Accuracy degradation was often not apparent but had to be detected with a double precision gain evaluation program.

In every case studied the U-D and Potter-Schmidt algorithms outperformed the Kalman methods. Accuracy improvements were generally substantial, often reaching orders of magnitude. Error analysis showed numerical degradation to be the dominant source of error in the Kalman algorithms. In fact, numerical errors completely obscured the effects of mismodeling. This result is of special interest because numeric effects are rarely considered when mission design requirements are constructed. These effects appear to be inconsequential, however, when the factorization algorithms are employed.

The cost comparisons of Chapter IV show that for most filtering problems the factorization algorithms are not unreasonably costly. The Potter-Schmidt costs are generally within 80% of the conventional Kalman costs. The U-D algorithm, however, is competitive with the conventional method, and for some applications is somewhat more efficient. For problems involving a few colored noise parameters and a large number of bias parameters, the U-D algorithm is particularly efficient relative to the other methods. The complete 19-state model studied in section 6.4.1 is an illustrative example. The CPU times for this case are compared in Table 6.2. These times include the costs associated with indexing, logic and other overhead costs not included in the analytic comparisons of Chapter IV. Table 6.2 shows the Potter-Schmidt algorithm

to be the most expensive filtering scheme, requiring considerably more time than the other methods. In sharp contrast, the U-D algorithm is seen to be the most efficient method, faster than even the conventional Kalman algorithm.†

This analysis of a meaningful engineering problem has demonstrated how the U-D and Potter-Schmidt factorization algorithms can dramatically reduce the effects of numerical errors. Moreover, the U-D method combines numerical stability with an efficiency comparable to that of the origin Kalman algorithm.

Table 6.2. Comparison of Filter Execution Times* for the Complete 19-State Model

Filter	Single Precision	Double Precision
Conventional Kalman	39	49
Stabilized Kalman	45	59
U-D (MWGS)	38	46
Potter-Schmidt	63	80

*CPU time in seconds.

†These times do not include the costs of variance calculations in the factorization filters. However, even if variances were computed at every stage, the total CPU time would be increased by no more than ten percent.

Chapter VII. Summary of Contributions and Areas for Future Research

7.1 Summary of Contributions

A new computational form of the discrete Kalman filter has been developed. This method is based upon the U-D factorization of the error covariance matrix and has the characteristics of improved accuracy and stability generally associated with square root filters. The U-D filter was obtained by extending the Bierman [1976a] data processing method to allow for time propagation. Two numerically reliable propagation algorithms were derived in Chapter III. These algorithms were obtained from modifications of the Givens and Gram-Schmidt matrix triangularization methods which are known for their numerical stability. The general U-D propagation results were then extended to derive an efficient U-D time update algorithm for systems with bias parameters and colored process noise.

Cost comparisons in Chapter IV have demonstrated that both of the basic U-D propagation methods are efficient and usually require equivalent amounts of computation. These comparisons further show that the U-D filter algorithm, using either the MWGS or the modified Givens propagation scheme, has efficiency nearly equal to that of Kalman's original formula. Moreover, for problems involving modest numbers of process noise parameters, the U-D method is somewhat faster than the Kalman algorithm. By comparison, the square root covariance filters were shown to be significantly less efficient. In typical real-time applications involving small computers and low-order systems, the square

root algorithms are at least 40% more time consuming than the conventional Kalman method and may require in excess of 100% more computation time.

An extension of the U-D filtering results has produced a new covariance error analysis technique. This method was derived in Chapter V as a means of evaluating general modeling errors, such as incorrect a priori statistics, incomplete parameter sets, and mismodeled system dynamics and data equations. The general error analysis algorithm requires only a modest alteration to the U-D filter equations. Efficiency is competitive with that of conventional error analysis methods, and numerical accuracy of the results is believed to be improved. The U-D evaluation method is easily adapted to analyze a variety of important special problems, such as the mismodeling of colored process noise and bias parameters. In addition, this method has produced a new consider filter algorithm which shares the simplicity and efficiency of the optimal U-D filter equations.

The orbit determination case study in Chapter VI provides a thorough and extensive examination of the numerical characteristics of the various Kalman filtering methods. This study demonstrates with a meaningful engineering problem the improved precision of the U-D and square root covariance methods. These methods consistently outperformed the conventional and stabilized Kalman algorithms. Accuracy improvements were usually substantial and often involved orders of magnitude. While the Kalman algorithms consistently experienced severe accuracy degradations and were generally unreliable, the covariance factorization methods exhibited

excellent numerical stability and precision throughout the study. A comparison of actual CPU times for this analysis has demonstrated that the U-D and conventional Kalman filters require equivalent amounts of computation. Thus, this study illustrates how the U-D filter combines superior numerical precision with exceptional computational efficiency.

7.2 Areas for Future Research

The modified Givens algorithm which applies the efficient formula, Method B, has been recommended for U-D propagation. This algorithm is known to be numerically stable if the ratio d_m'/d_m is tested at each step and if the more stable formula, Method A, is used whenever d_m'/d_m is sufficiently large (cf section 3.4). However, for most time updating applications of this algorithm, such testing is probably not necessary. Further analysis is needed to determine which problems, if any, are actually susceptible to error growth when Method B is used exclusively.

Another reliable method for propagating U-D covariance factors can be obtained by appropriately modifying the Householder triangularization technique. Preliminary analysis indicates that square root calculations are inherent to Householder techniques, and so this method may have little advantage over the MWGS or modified Givens propagation algorithms. However, a square-root-free Householder triangularization method could yield a U-D time update algorithm which requires fewer multiplications and divisions than the modified Givens formula. Hence this Householder method might be noticeably more efficient than either of the currently available U-D propagation techniques, particularly on computers where divide operations are relatively time consuming (cf section 4.3.1).

The U-D covariance error analysis method is based upon a number of algorithms which have been established as numerically reliable formulas. While the error analysis technique appears to inherit the stability of these individual algorithms, the integrity of this method should be confirmed by further research.

There is another area of covariance error analysis where research might prove fruitful. Although much effort has been expended to develop various gain evaluation techniques, very little attention has been given to the analysis of innovation computations. The case study in Chapter VI of this thesis has shown that parameter estimates are particularly sensitive to innovation errors. Recall that in the single precision U-D filter it was necessary to compute estimates and innovations in double precision arithmetic, although gain calculations were reasonably accurate in single precision. Further evaluation of innovation errors could produce some interesting and useful results.

Another extension of the U-D filtering method which would be useful, and might prove to be challenging, is the development of algorithms to allow for cross-correlations between measurement errors and process noise parameters. At first glance this cross-coupling appears to make the U-D time update unreasonably complicated. However, the "weighted error" propagation technique used in Chapter V could be applied to this problem and should make it apparent which orthogonal transformations are required to accomplish the update.

The covariance factorization techniques presented in this report might e beneficially applied to problems involving continuous parameter systems and continuous data. These methods could also prove valuable in distributed parameter problems and in the areas of identification and control.

Appendix A

Cholesky Decomposition and Data Whitening

A.1 Square-Root-Free Cholesky Decomposition

It has been shown (cf Martin et al. [1965]) that any positive definite symmetric matrix R can be uniquely factored as

$$R = UDU^T \tag{A.1}$$

where U is unit upper triangular† and D is a positive diagonal. The U-D factors of R may be computed as follows.

For j = n,...,1 evaluate recursively Eqs. (A.2)-(A.5).

$$d_j = r_{jj} - \sum_{k=j+1}^{n} d_k u_{jk}^2 \tag{A.2}$$

$$u_{jj} = 1 \tag{A.3}$$

$$u_{ij} = 1/d_j \left(r_{ij} - \sum_{k=j+1}^{n} d_k u_{ik} u_{jk}\right) \tag{A.4}$$

$$u_{ji} = 0 \tag{A.5}$$

$$\Big\} \; i = 1,\ldots j-1$$

†The factorization given here is actually a modification of the algorithm by Martin et al. [1965] since their formula computes a lower triangular factor.

The arithmetic operations required by this algorithm are summarized in Table A.1.

Table A.1. Arithmetic Operations Required for a U-D Decomposition

Computation	Adds	Multiplies	Divides
$j = n,\ldots,1$			
$v_k = u_{jk}d_k;\ k = j+1,\ldots,n$		$.5n^2 - .5n$	
$d_j = r_{jj} - \sum_{k=j+1}^{n} u_{jk}v_k$	$.5n^2 + .5n$	$.5n^2 - .5n$	
$\lambda_j = 1/d_j$			$n-1$
$i = 1,\ldots,j-1$			
$u_{ij} = \lambda_j(r_{ij} - \sum_{k=j+1}^{n} u_{ik}v_k)$	$.2n^3-1.5n^2+1.5n$	$.2n^3-1.5n^2+1.5n$	
Totals	$.2n^3 - n^2 + 2n$	$.2n^3 - n^2 + n$	$n-1$

A.2 Cholesky Square Root Decomposition

Let an upper triangular matrix S be defined as

$$S = UD^{1/2} \tag{A.6}$$

where $D^{1/2}$ is the positive square root of D in Eq. (A.1). Equations (A.1) and (A.6) then imply that any positive definite symmetric matrix R may be uniquely factored as

$$R = SS^T \tag{A.7}$$

where S is upper triangular. Historically, this decomposition preceded the U-D algorithm and was derived by Cholesky. The following algorithm is a modest rearrangement of the Cholesky square root factorization given in Fox [1954].

For j = n,...,1 compute recursively Eqs. (A.8)-(A.10).

$$s_{jj} = \left(r_{jj} - \sum_{k=j+1}^{n} s_{jk}^2\right)^{1/2} \tag{A.8}$$

$$s_{ij} = 1/s_{jj}\left(r_{ij} - \sum_{k=j+1}^{n} s_{ik}\, s_{jk}\right) \qquad i = 1,\ldots j-1 \tag{A.9}$$

$$s_{ji} = 0 \tag{A.10}$$

Table A.2 contains a summary of the arithmetic operations required by this algorithm.

Table A.2. Arithmetic Operations Required for a Square Root Decomposition

Computation	Adds	Multiplies	Divides	Square Roots
$j = n,\ldots,2$				
$s_{jj} = \left(r_{jj} - \sum_{k=j+1}^{n} s_{jk}^2\right)^{1/2}$*	$.5n^2+.5n$	$.5n^2-.5n$		n
$\lambda_j = 1/s_{jj}$			n-1	
$i = 1,2,\ldots,j-1$				
$s_{ij} = \lambda_j\left(r_{ij} - \sum_{k=j+1}^{n} s_{ik}s_{jk}\right)$	$.2n^3-.2n$	$.2n^3-.2n$		
Totals	$.2n^3+.5n^2+.3n$	$2n^3+.5n^2-.7n$	n-1	n

*This computation is also required for j=1.

A.3 Data Whitening

Suppose m measurements, $z = (z_1, z_2, \ldots, z_m)^T$, are made where

$$z = Ax + \nu \tag{A.10}$$

and $x \epsilon R_n$, $E(\nu) = 0$, $E\{\nu\ \nu^T\} = R$, $E\{x\ \nu^T\} = 0$. The data noise covariance, R, is a symmetric, positive definite matrix and, hence, can be factored by the square-root-free Cholesky algorithm, Eqs. (A.2)-(A.5). Thus,

$$R = J R' U^T \tag{A.11}$$

where U is unit upper triangular and R′ is diagonal. A new set of uncorrelated measurements, z′, may then be generated by computing

$$z' = U^{-1} z \tag{A.12}^{\dagger}$$

The data equation for z′ is given by

$$z' = A'x + \nu' \tag{A.13}$$

where

$$A' = U^{-1} A \tag{A.14}$$

$$E\{\nu' \nu'^T\} = R' = \text{diag}(r_1', r_2', \ldots, r_m') \tag{A.15}$$

This process of uncoupling observations is referred to as "data whitening." A precursor of this method generates uncoupled data, z′, with an identity noise covariance (cf Andrews [1968]). In that case R is factored by the square root Cholesky decomposition, Eqs. (A.8)-(A.10), to obtain

$$R = SS^T \tag{A.16}$$

The data $z' = S^{-1}z$ is then represented by Eq. (A.13) where A′ and ν′ are given by

$$A' = S^{-1}A \tag{A.17}$$

$$E\{\nu' \nu'^T\} = I_m \tag{A.18}$$

---

†The inversion of U is easily accomplished by back substitution.

Appendix B

The Potter Update Using Householder Transformations

The relationship between the Potter algorithm (Chapter II) and Householder transformation techniques was first noted by Bierman [1973]. He observed that

$$\begin{bmatrix} \tilde{S} & 0 \\ a^T\tilde{S} & \sqrt{r} \end{bmatrix} \begin{bmatrix} \tilde{S}^T & \tilde{S}^T a \\ 0 & \sqrt{r} \end{bmatrix} = \begin{bmatrix} \tilde{P} & \tilde{P}a \\ a^T\tilde{P} & \alpha \end{bmatrix} \tag{B.1}$$

where $\alpha = a^T\tilde{P}a + r$. If this array is equated with the product

$$\begin{bmatrix} W & G \\ 0 & c \end{bmatrix} \begin{bmatrix} W^T & 0 \\ G^T & c \end{bmatrix} = \begin{bmatrix} WW^T + GG^T & cG \\ cG^T & c^2 \end{bmatrix} \tag{B.2}$$

then the following identities are obtained.

$$c = \pm\sqrt{\alpha} \tag{B.3}$$

$$G = \left(1/\pm\sqrt{\alpha}\right)\tilde{P}a = \pm\sqrt{\alpha}\,K \quad (K = \text{Kalman gain}) \tag{B.4}$$

$$WW^T = \tilde{P} - GG^T \tag{B.5}$$

Note that

$$\tilde{P} - GG^T = \hat{P} \tag{B.6}$$

Equations (B.5)-(B.6) imply that

$$W = \hat{S} \tag{B.7}$$

where $\hat{S}$ is the updated covariance square root. Since the matrix products in Eqs. (B.1) and (B.2) are equal, the factors must be related by an orthogonal transformation. Thus, a Householder transformation, T, may be chosen such that

$$\begin{matrix} n\{ \\ 1\{ \end{matrix}\begin{bmatrix} \tilde{S} & 0 \\ - & - \\ f^T & \sqrt{r} \end{bmatrix} T = \begin{bmatrix} \hat{S} & G \\ - & - \\ 0 & -\sqrt{\alpha} \end{bmatrix} \tag{B.8}$$

where $f^T = a^T\tilde{S}$. If the transformation T is chosen to zero out only the subdiagonal elements in row n+1, then

$$T = I - 1/\beta\, uu^T \tag{B.9}^{\dagger}$$

where

$$u^T = (f^T, \quad \sqrt{r} + \sqrt{\alpha}) \tag{B.10}$$

$$\beta = 2/u^Tu = 1/(\alpha + \sqrt{\alpha r}) \tag{B.11}$$

When this T is substituted into Eq. (B.8) and terms are equated, the following results are obtained.

---

†Elementary Householder transformations are discussed in Chapter III.

$$\hat{S} = \tilde{S} - \gamma \bar{K} f^T \tag{B.12}$$

$$\gamma = \frac{\lambda}{1 + \sqrt{r\lambda}}, \qquad \lambda = 1/\alpha = 1/(r + f^T f) \tag{B.13}$$

$$\bar{K} = \tilde{S} f \tag{B.14}$$

The Kalman gain is computed as follows.

$$K = \lambda \bar{K} \tag{B.15}$$

Equations (B.12)-(B.15) correspond to the Potter measurement update.

Appendix C

Agee-Turner Matrix Factorizations

The two algorithms presented in this Appendix yield triangular factorizations of the matrix $\bar{P} = P + c\lambda\lambda^T$. The matrix P is assumed to be positive definite symmetric and λ represents a vector. The first algorithm requires that P be factored such that $P = UDU^T$ (U is unit upper triangular and D is diagonal) and computes the $\bar{U}$-$\bar{D}$ factors of $\bar{P}$. Similarly, the second algorithm yields the upper triangular matrix square root $\bar{S}$ where $\bar{P} = \bar{S}\bar{S}^T$. Cholesky has shown that when $\bar{P}$ is positive definite symmetric such factorizations exist and are unique (see Appendix A).

The matrix decompositions described below seem to have been first derived by Agee and Turner [1972a]. Their algorithms have been appropriately rearranged to compute the upper triangular factors used in this report. These algorithms are efficient and easy to mechanize, and when the scalar, c, is positive they are also numerically reliable. However, Agee and Turner have noted that when c is negative these methods are subject to large cancellation errors and can yield erroneous results. For this reason the following algorithms are recommended only for problems involving positive scalars.

C.1 U-D Triangular Factorization Algorithm

Suppose that the symmetric matrix P is factored such that

$$P = UDU^T \tag{C.1}$$

where the matrix U is upper triangular with ones on the diagonal and D is a positive diagonal. The corresponding $\bar{U}$-$\bar{D}$ factors of $\bar{P}$ where

$$\bar{P} = P + c\lambda\lambda^T \qquad c > 0 \tag{C.2}$$

may be obtained in the following way.

Evaluate Eqs. (C.3)-(C.7) recursively for j = n, n-1, ..., 2.

$$\bar{d}_j = d_j + c_j \lambda_j^2 \qquad (c_n = c) \tag{C.3}$$

$$v_j = c_j \lambda_j / \bar{d}_j \tag{C.4}$$

$$\left.\begin{aligned} \lambda_i &:= \lambda_i - \lambda_j u_{ij} && (C.5)^{\dagger} \\ \bar{u}_{ij} &= u_{ij} + \lambda_i v_j && (C.6) \end{aligned}\right\} \quad i = 1, 2, \ldots, j-1$$

$$c_{j-1} = c_j (d_j / \bar{d}_j) \tag{C.7}$$

$$\bar{d}_1 = d_1 + c_1 \lambda_1^2 \tag{C.8}$$

Table C.1 contains a summary of the arithmetic operations required by this algorithm.

---

†The symbol ":=" denotes replacement in computer storage.

Table C.1. Arithmetic Operation Counts For Triangular Factorization

Computation	Adds	Multiplies	Divides	Square Roots
$j = n, \ldots, 2$				
$\bar{d}_j = d_j + c_j\lambda_j^2$*	n	$2n$		
$\beta_j = c_j/\bar{d}_j$			$n - 1$	
$v_j = \beta_j\lambda_j$		$n - 1$		
$i = 1, \ldots, j-1$				
$\lambda_i := \lambda_i - \lambda_j u_{ij}$	$.5n^2-.5n$	$.5n^2-.5n$		
$u_{ij} := u_{ij} + v_j\lambda_i$	$.5n^2-.5n$	$.5n^2-.5n$		
$c_{j-1} = \beta_j d_j$		$n - 1$		
Totals	n^2	$n^2 + 3n - 2$	$n - 1$	0

*This calculation is also required for $j = 1$.

C.2 Triangular Square Root Factorization Algorithm

Let the positive definite symmetric matrix P be factored such that

$$P = SS^T \tag{C.9}$$

where S is upper triangular. Consider the problem of computing a similar matrix $\bar{S}$ where

$$\bar{S}\bar{S}^T = SS^T + c\lambda\lambda^T \qquad c > 0 \tag{C.10}$$

The following algorithm yields the nontrivial elements of $\bar{S}$.

Evaluate recursively Eqs. (C.11)-(C.17) for j = n, n-1, ..., 2.

$$\bar{s}_{jj} = (s_{jj}^2 + c_j\lambda_j^2)^{1/2}, \qquad c_n = c \tag{C.11}$$

$$b_j = \lambda_j/s_{jj} \tag{C.12}$$

$$a_j = s_{jj}/\bar{s}_{jj} \tag{C.13}$$

$$v_j = c_j\lambda_j/\bar{s}_{jj} \tag{C.14}$$

$$\left.\begin{aligned} \lambda_i &:= \lambda_i - b_j s_{ij} \qquad (C.15)\\ \bar{s}_{ij} &= a_j s_{ij} + v_j\lambda_i \qquad (C.16) \end{aligned}\right\} i = 1, 2, \ldots, j-1$$

$$c_{j-1} = c_j a_j^2 \tag{C.17}$$

$$\bar{s}_{11} = (s_{11}^2 + c_1 \lambda_1^2)^{1/2} \tag{C.18}$$

The matrix $\bar{S}$ obtained from this recursion has positive diagonals. Table C.2 lists the arithmetic operations required by this algorithm when it is arranged for maximum efficiency.

Table C.2. Arithmetic Operation Counts For Square Root Factorization

Computation	Adds	Multiplies	Divides	Square Roots
$j = n, \ldots, 2$				
$\bar{s}_{jj} = (s_{jj}^2 + c_j\lambda_j^2)^{1/2}$*	n	2n		n
$a_j = 1/\bar{s}_{jj}$			n - 1	
$b_j = \lambda_j/s_{jj}$			n - 1	
$v_j = a_j c_j \lambda_j$		2(n-1)		
$a_j := a_j s_{jj}$		n - 1		
$i = 1, 2, \ldots, j-1$				
$\lambda_i := \lambda_i - b_j s_{ij}$	$.5n^2-.5n$	$.5n^2-.5n$		
$s_{ij} := a_j s_{ij} + v_j \lambda_i$	$.5n^2-.5n$	n^2-n		
$c_{j-1} = c_j a_j^2$		2(n-1)		
Total	n^2	$1.5n^2 + 5.5n - 5$	$2n - 2$	n

*This calculation is also required for j=1.

Appendix D

FORTRAN Implementation of the U-D Algorithms

The computer implementations included in this Appendix are designed to minimize storage and computation. Storage could be conserved further, however, by mechanizing the algorithms to store the U and D arrays as vectors. Instead, the mechanizations described below assume the nontrivial elements of U to be stored in the upper triangular portion of an n×n array which contains D along the diagonal.

These computer mechanizations are described in a semi-FORTRAN style in the sense that Greek characters are used and DO loops are defined with algebraic functions. This style is adopted so that each portion of the code may be easily identified with the corresponding algebraic equation in Chapters II and III. The symbol "@@" is used to denote operations which can be omitted when estimates are not computed.

D.1 U-D Optimal Measurement Update

The following mechanization was suggested by Bierman [1976a].

Inputs: U - upper triangular matrix containing $\tilde{U}$-$\tilde{D}$ factors of a priori covariance, with U(I,I) = D(I)

Z,A,r - observation, observation coefficients and error covariance, respectively

X - a priori estimates (N-vector)

Outputs: U - updated array containing a posteriori U-D factors with updated D(I) located in U(I,I) position

α - innovation covariance

γ - 1/α

G - the unweighted Kalman gain (K = Gγ).

X - updated estimates

```
      DO 10  J = N, 2, -1
      Z = Z - A(J)*X(J)                                  @@
      DO 5   K = 1, J-1
5     A(J) = A(J) + A(K)*U(K,J)                          @ Eq. (2.69)
      G(J) = U(J,J)*A(J)                                 @ Eq. (2.70)

      Z = Z - A(1)*X(1)                                  @@
      G(1) = A(1)*U(1,1)
```

Comment: The quantities $Z := Z - A^TX$, $G = DU^TA$ and $A := U^TA$ have been computed.

```
      α = r + A(1)*V(1)                                  @ Eq. (2.72)
      γ = 1./α
      U(1,1) = U(1,1)*r*γ                                @ Eq. (2.73)

      DO 20  J = 2,N
      β = α
      α = α + V(J)*A(J)                                  @ Eq. (2.74)
      λ = -A(J)*γ                                        @ Eq. (2.76)
      γ = 1./α
      U(J,J) = U(J,J)*β*γ                                @ Eq. (2.75)
```

```
      DO 20  I = 1,J-1
      β = U(I,J)
      U(I,J) = β + G(I)*λ                                  @Eq. (2.77)
20    G(I) = G(I) + G(J)*β                                 @Eq. (2.78)

      Z = Z*Y                                              @@
      DO 30 J = 1,N
30    X(J) = X(J) + G(J)*Z                                 @@
```

D.2 U-D Time Update Algorithms

Each of the time update methods involves the following arrays and calculations.

Input: Φ - N×N state transition matrix

B - N×K matrix of process noise coefficients

U - upper triangular matrix containing U-D factors of error covariance with D stored along the diagonal

X - N-vector of estimates

V - N+K-vector with process noise variances stored in first K locations

Outputs: U - updated array containing a priori U-D factors with updated D(I) stored in the U(I,I) location

X - updated estimates

Comment: Define the N×(N+K) working array W by an EQUIVALENCE statement so that the first K columns of W are identified with the matrix B and the last N columns contain the matrix Φ.

```
      DO 40  I = 1,N
      F(I) = 0.
      DO 40  J = 1,N
40    F(I) = F(I) + Φ(I,J)*X(J)                                    @@

      DO 50  J = N, 1, -1
      X(J) = F(J)
      V(J+K) = U(J,J)
      DO 50  I = 1,N
      DO 50  L = 1,J-1
50    Φ(I,J) = Φ(I,J) + Φ(I,L)*U(L,J)
```

Comment: The array W now contains the matrix B in the first K columns and the product ΦU in the last N columns. The (N+K) array $\bar{D}$ = diag(Q,D) is in V, and X contains the updated estimates. At this point the updated U-D factors may be computed by either of the following algorithms.

Modified Weighted Gram-Schmidt Triangularization

```
      M = N+K
      DO 90  J = N, 1, -1
      D = 0.

      DO 60  I = 1,M
      F(I) = W(J,I)*V(I)
60    D = D + F(I)*W(J,I)                                          @ Eq. (3.28)

      U(J,J) = D
      IF(J = 1) GO TO 90
      IF(D = 0.) GO TO 80
      λ = 1./D
```

```
      DO 75  I = 1, J-1
      β = 0.
      DO 65  L = 1,M
65    β = β + W(I,L)*F(L)
      β = λ*β
      U(I,J) = β                                        @ Eq. (3.29)
      DO 75  L = 1,M
75    W(I,L) = W(I,L) - β*W(J,L)                        @ Eq. (3.30)

      GO TO 90

80    DO 85  I = 1,J-1
85    U(I,J) = 0.

90    CONTINUE
```

Modified Givens Triangularization

```
      DO 80  J=N,2,-1
      M = J+K
      YJ = W(J,M)
      R = V(M)
      D = R*YJ**2

      DO 60  I = M-1,1,-1
      XI = W(J,I)
      S = XI*V(I)
      D = D + S*XI                                      @ Eq. (3.81)
      λ = 1./D
      R = R*λ
      S = S*λ                                           @ Eq. (3.82)
      V(I) = R*V(I)                                     @ Eq. (3.83)

      IF(I < M-1) GO TO 30
      C = YJ*R                                          @ Eq. (3.84)

      DO 40  L = 1, J-1
      F = W(L,I)                                        @ Eq. (3.85)
      W(L,I) = YJ*W(L,I) - XI*W(L,M)                    @ Eq. (3.86)
40    W(L,M) = C*W(L,M) + S*F                           @ Eq. (3.87)
      GO TO 60

30    IF(R >.25) GO TO 45

      DO 25 L = 1, J-1
      F = W(L,I)
      W(L,I) = W(L,I) - XI*W(L,M)                       @ Eq. (3.86)
25    W(L,M) = R*W(L,M) + S*F                           @ Eq. (3.87)
      GO TO 60

45    DO 50  L = 1, J-1
      W(L,I) = W(L,I) - XI*W(L,M)                       @ Eq. (3.86)
```

```
50      W(L,M) = W(L,M) + S*W(L,I)                    @ Eq. (3.88)
60      R = D

        DO 75  I = 1,J-1
75      U(I,J) = W(I,M)

80      U(J,J) = D
        D = 0.

        DO 85  I = 1,K+1
85      D = D + V(I)*W(1,I)**2
        U(1,1) = D
```

Comment: The updated U-D factors are now in U.

Appendix E

Arithmetic Operation Counts

E.1 Itemized Costs of U-D Algorithms

The arithmetic operations required for a U-D optimal measurement update are listed in Table E.1. This list includes the operations needed to compute updated estimates and the U-D covariance factors via Eqs. (2.69-2.78).

The costs associated with the MWGS and modified Givens matrix factorizations (cf Chapter III) are itemized in Tables E.2 and E.3. Note that these costs do not include the operations required to compute the products $\Phi\hat{U}$ and $\Phi\hat{x}$. Hence the total U-D time update costs for each method may be obtained by including an additional $.5n^3 + .5n^2$ adds and multiplies to the totals in Tables E.2 and E.3.

E.2 Propagation Costs for Systems with Colored Process Noise

Propagation schemes for systems with colored process noise are discussed in the last section of Chapter III. The U-D covariance factorization is shown to be particularly well-suited for one-at-a-time propagation of colored noise parameters. This propagation method is derived by exploiting special system structure and permits savings in both computation and computer storage. The arithmetic operations involved in the U-D colored noise update are given in Table E.4. These counts represent the costs sustained when a modified Givens factorization is used for the deterministic phase of the mapping. The calculations required

for a MWGS colored noise update may be obtained by replacing the modified Givens counts in Table E.4 with the appropriate totals in Table E.2.

The costs associated with a similar one-at-a-time propagation of triangular covariance square roots are listed in Table E.5. Note that when k is large this square root algorithm is less efficient than a carefully structured Schmidt time update (cf Table E.6). The extra calculations involved in the one-at-a-time algorithm are related to the Agee-Turner matrix factorization.† Although the one-at-a-time algorithm is slower than the Schmidt update, it does reduce computer storage requirements and, thus, may be preferred in situations where storage is limited.

A colored noise time update of the Potter covariance square root is also most efficiently accomplished by applying a carefully constructed Schmidt algorithm. Notice, however, that the Potter-Schmidt update does not allow for full exploitation of special system structure. Since the square root $\hat{S}$ is a general nxn matrix, the product $\bar{S} = \Phi\hat{S}$ contains no large block of zeroes to simplify the update. However, some computational savings are possible if one is careful to order the parameter vector appropriately. Note that when an upper triangular a priori square root $\tilde{S}$ is desired, propagation costs are minimized if the state vector is defined as follows.

---

†The Agee-Turner square-root algorithm is considerably more expensive than its U-D counterpart. The reader may refer to Appendix C for details of these methods.

$$X = \begin{bmatrix} p \\ x \end{bmatrix} \begin{matrix} \}k \\ \}n \end{matrix} \qquad \text{(E.1)}$$

In this case a Schmidt time update may be performed in two stages as indicated in Eqs. (E.2) and E.3).

$$\begin{matrix} & \overbrace{\quad}^{k} & \overbrace{\quad}^{k} & \overbrace{\quad}^{n} \\ k\left\{\right. & \left[\begin{array}{ccc|c|c} \sqrt{q}_1 & & & & \\ & \ddots & & \bar{S}_p & \bar{S}_{px} \\ & & \sqrt{q}_k & & \end{array}\right] \\ n\left\{\right. & \left[\begin{array}{ccc|c|c} 0 & & & \bar{S}_{xp} & \bar{S}_x \end{array}\right] \end{matrix} \quad T = \quad \begin{matrix} & \overbrace{\quad}^{k} & \overbrace{\quad}^{k} & \overbrace{\quad}^{n} \\ & \left[\begin{array}{ccc|c|c} \sqrt{q}_1 & & 0 & & \\ & \ddots & & \bar{S}_p' & \tilde{S}_{px} \\ 0 & & \sqrt{q}_k & & \end{array}\right] \\ & \left[\begin{array}{ccc|c|c} 0 & & & 0 & \tilde{S}_x \end{array}\right] \end{matrix} \qquad \text{(E.2)}$$

$$\begin{matrix} & \overbrace{\quad}^{k} & \overbrace{\quad}^{k} & \overbrace{\quad}^{n} \\ k\left\{\right. & \left[\begin{array}{ccc|c|c} \sqrt{q}_1 & & 0 & & \\ & \ddots & & \bar{S}_p & \tilde{S}_{px} \\ & & \sqrt{q}_k & & \end{array}\right] \\ n\left\{\right. & \left[\begin{array}{ccc|c|c} 0 & & & 0 & \tilde{S}_x \end{array}\right] \end{matrix} \quad \bar{T} = \quad \begin{matrix} & \overbrace{\quad}^{k} & \overbrace{\quad}^{k} & \overbrace{\quad}^{n} \\ & \left[\begin{array}{ccc|c|c} & & & & \\ & 0 & & \tilde{S}_p & \tilde{S}_{px} \\ & & & & \end{array}\right] \\ & \left[\begin{array}{ccc|c|c} 0 & & & 0 & \tilde{S}_x \end{array}\right] \end{matrix} \qquad \text{(E.3)}$$

Notice that the Householder transformation T zeroes out the subdiagonal elements of the last n rows without disturbing the first k columns of the array. Similarly the second transformation $\bar{T}$ triangularizes the first k rows of the array without altering the last n rows and columns.

If the state vector were reordered with the colored noise parameters last, the nxk block of zeroes in the left hand side of Eq. (E.2)

could no longer be fully exploited. In this case an initial transformation T′ would be required to operate on all (n+2k) columns of the array in the following way.

$$
\begin{array}{c}
\begin{matrix} \overbrace{}^{k} & \overbrace{}^{n} & \overbrace{}^{k} \end{matrix} \\
\left.\begin{matrix} n \\ k \end{matrix}\right\{
\left[\begin{array}{cc|c|c}
\multicolumn{2}{c|}{0} & \bar{S}_x & \bar{S}_{xp} \\
\hline
\sqrt{q_1} & 0 & & \\
\vdots & & \bar{S}_{px} & \bar{S}_p \\
0 & \sqrt{q_k} & &
\end{array}\right]
\end{array}
\quad T = \quad
\begin{array}{c}
\begin{matrix} \overbrace{}^{k} & \overbrace{}^{n} & \overbrace{}^{k} \end{matrix} \\
\left[\begin{array}{c|c|c}
\bar{B} & \bar{S}'_x & \bar{S}_{xp} \\
\hline
0 & 0 & \bar{S}_p
\end{array}\right]
\end{array}
\qquad (E.4)
$$

Hence, the complete update for this parameter arrangement involves more computation than an update associated with the state vector in Eq. (E.1).

By similar arguments one can show that when a lower triangular a priori square root is desired, the Schmidt update is less expensive if the colored noise parameters are positioned in the lower portion of the state vector. The operations required for a Potter-Schmidt colored noise update are listed in Table E.7.

Table E.1. Operation Count for U-D Measurement Update

Computations	Adds	Multiplies	Divides
$f = \tilde{U}^T a$	$.5n^2 - .5n$	$.5n^2 - .5n$	
$v = \tilde{D} f$		n	
$\alpha_0 = r$			
For $j=1,\ldots,n$			
$\alpha_j = \alpha_{j-1} + v_j f_j$	n	n	
$\beta_j = 1/\alpha_j$			n
$\hat{d}_j = \alpha_{j-1}\, \beta_j\, \tilde{d}_j$		$2n$	
Omit the following when $j=1$			
$\lambda_j = -f_j \beta_{j-1}$		$n-1$	
For $i=1,\ldots,j-1$			
$\hat{u}_{ij} = \tilde{u}_{ij} + \lambda_j \bar{K}_i$	$.5n^2 - .5n$	$.5n^2 - .5n$	
$\bar{K}_i := \bar{K}_i + v_j \tilde{u}_{ij}$	$.5n^2 - .5n$	$.5n^2 - .5n$	
$\bar{K}_j = v_j$			
$\tilde{z} = (z - a^T \tilde{x}) \beta_n$	n	n	
$\hat{x} = \tilde{x} + \bar{K}\tilde{z}$	n	n	
Totals	$1.5n^2 + 1.5n$	$1.5n^2 + 5.5n$	n

C-3

Table E.2. Operation Count for MWGS Factorization

Computations	Adds	Multiplies	Divides
For $j=n, n-1,\ldots,2$			
$f_\ell := d_\ell w_{j\ell}$; $\ell=1,\ldots,n+k$		$n^2 + nk$	
$\tilde{d}_j = \sum_{\ell=1}^{n+k} f_\ell w_{j\ell}$*	$n^2 + nk$	$n^2 + nk$	
$\lambda_j = 1/\tilde{d}_j$			$n - 1$
$i = 1,2,\ldots,j-1$			
$\tilde{u}_{ij} = \lambda_j \sum_{\ell=1}^{n+k} w_{i\ell} f_\ell$	$.5(n^2 - n)(n + k)$	$.5(n^2 - n)(n + k + 1)$	
$\ell=1,\ldots,n+k$			
$w_{i\ell} := w_{i\ell} - \tilde{u}_{ij} w_{j\ell}$	$.5(n^2 - n)(n + k)$	$.5(n^2 - n)(n + k)$	
Totals	$n^3 + n^2 k$	$n^3 + 1.5n^2 - .5n + (n^2 + n)k$	$n - 1$

*This computation also performed when $j=1$.

Table E.3. Operation Count for Modified Givens Factorization

Computations	Adds	Multiplies	Divides
$j=n,\ldots,1$			
$m=j+k$	n		
$i=m-1,\ldots,1$			
$s = w_{ji}\bar{d}_i \quad r = d_m$		$.5n^2 - .5n + nk$	
$d_m := d_m w_{jm}^2 + sw_{ji}$*	$.5n^2 - .5n + nk$	$.5n^2 + 1.5n + nk$	
$\delta = 1/d_m \quad r := r\delta$		$.5n^2 - .5n + nk$	$.5n^2 - .5n + nk$
$s := s\delta \quad d_i := rd_i$		$n^2 - n + 2nk$	
$\ell=1,2,\ldots,j-1 \quad v_\ell = w_{\ell i}$			
$w_{\ell i} := w_{jm}w_{\ell i} - w_{ji}w_{\ell m}$*	$1/3\ n^3 - .5n^2 + 1/6\ n + .5(n^2 - n)k$	$1/3\ n^3 - 1/3\ n + .5(n^2 - n)k$	
If $i = m-1$			
$c = rw_{jm}$		n	
$w_{\ell m} := cw_{\ell m} + sv_\ell$	$.5n^2 - .5n$	$n^2 - n$	
If $i < m-1$			
$w_{\ell m} := w_{\ell m} + sw_{\ell i}$	$1/3\ n^3 - n^2 + 2/3\ n + .5(n^2 - n)k$	$1/3\ n^3 - n^2 + 2/3\ n + .5(n^2 - n)k$	
Totals	$.7n^3 - .5n^2 + .8n + n^2k$	$.7n^3 + 2.5n^2 - .2n + (n^2 + 4n)k$	$.5n^2 - .5n + nk$

*When $i < m-1$, $w_{jm} \equiv 1$.

Table E.4. Operation Count for U-D Colored Noise Update

Computations	Adds	Multiplies	Divides
$\begin{bmatrix}\tilde{x}\\ \tilde{p}\end{bmatrix} = \begin{bmatrix}\Phi_x & \Phi_{xp}\\ 0 & M\end{bmatrix}\begin{bmatrix}x\\ p\end{bmatrix}$	$n^2 + nk$	$n^2 + nk + k$	
$[W \mid \bar{U}_{xp}] = [\Phi_x \; \Phi_{xp}]\left[\begin{array}{c\|c} U_x & U_{xp}\\ 0 & U_p\end{array}\right]$	$.5n^3 - .5n^2 + .5nk^2$ + $(n^2 - .5n)k$	$.5n^3 - .5n^2 + .5nk^2$ + $(n^2 - .5n)k$	
$[W, D_x] \leftarrow [\bar{U}_x, \bar{D}_x]$ (Givens update*)	$.7n^3 - .5n^2 + .8n$	$.7n^3 + 2.5n^2 - .2n$	$.5n^2 - .5n$
$j = 1,2,\ldots,k$			
$\tilde{d}_{n+j} = m_j^2 \bar{d}_{n+j} + q_j$	k	$2k$	
$\tilde{u}_{n+j,\ell} = m_j \bar{u}_{n+j,\ell};\ \ell = n+j+1,\ldots,n+k$		$.5k^2 - .5k$	
$i = 1,2,\ldots,n+j-1$			
$\tilde{u}_{i,n+j} = m_j\left(\frac{\bar{d}_{n+j}}{\tilde{d}_{n+j}}\bar{u}_{i,n+j}\right)$		$nk + .5k^2 + .5k$	k
$c_j = \left(\frac{\bar{d}_{n+j}}{\tilde{d}_{n+j}}\right)q_j$		k	
Agee-Turner Algorithm $\tilde{U}^{(j)}\tilde{D}^{(j)}\tilde{U}^{(j)T} := \tilde{U}^{(j)}\tilde{D}^{(j)}\tilde{U}^{(j)T} + c_j vv^T$	$.3k^3 + (n - .5)k^2$ + $(n^2 + n + .2)k$	$.3k^3 + (n + 1)k^2$ + $(n^2 + 2n - 3.3)k$	$.5k^2$ + $(n - .5)k$
Totals	$1.2n^3 + .8n + .3k^3$ + $(1.5n - .5)k^2$ + $(2n^2 - .5n + 1.2)k$	$1.2n^3 + 3n^2 - .2n$ + $.3k^3 + (1.5n + 2)k^2$ + $(2n^2 + 3.5n + .7)k$	$.5n^2 - .5n$ + $.5k^2 + (n - .5)k$

*Refer to Table E.3 and let k = 0.

Table E.5. One-at-a-Time Propagation Costs for Triangular Covariance Square Root

Computations	Adds	Multiplies	Divides	Square Roots
$\begin{bmatrix}\tilde{x}\\ \tilde{p}\end{bmatrix} = \begin{bmatrix}\Phi_x & \Phi_{xp}\\ 0 & M\end{bmatrix}\begin{bmatrix}x\\ p\end{bmatrix}$	$n^2 + nk$	$n^2 + nk + k$		
$[\ W \mid \bar{S}_{xp}\ ] = [\Phi_x \ \Phi_{xp}]\left[\begin{array}{c\|c} S_x & S_{xp}\\ 0 & S_p\end{array}\right]$	$.5n^3 + .5n^2$ + $.5nk^2 + (n^2 + .5n)k$	$.5n^3 + .5n^2$ + $.5nk^2 + (n^2 + .5n)k$		
$[W]T = \bar{S}_x$ (Schmidt update)	$.7n^3 + 1.3n$	$.7n^3 + n^2 + .3n$	$n - 1$	n
$j = 1,2,\ldots,k$				
$\bar{s}_{n+j,\ell} := m_j \bar{s}_{n+j,\ell};\ \ell = n+j+1,\ldots,n+k$		$.5k^2 - .5k$		
$a_j = m_j \bar{s}_{n+j,n+j}$		k		
$\delta_j = a_j \bar{s}_{n+j,n+j} + q_j$	k	k		
$\tilde{s}_{n+j,n+j} = \sqrt{\delta_j}$				k
$r_j = a_j / \tilde{s}_{n+j,n+j}$			k	
$i = 1,2,\ldots,n+j-1$				
$\tilde{s}_{i,n+j} = r_j \bar{s}_{i,n+j};\ v_i = \bar{s}_{i,n+j}$		$.5k^2 + (n - .5)k$		
$c_j = q_j / \delta_j$			k	
Agee-Turner Algorithm	$.3k^3 + (n - .5)k^2$	$.5k^3 + (1.5n + 2)k^2$	$k^2 + (2n - 3)k$	$.5k^2 + (n - .5)k$
$\tilde{S}(j)\tilde{S}(j)T := \tilde{S}(j)\tilde{S}(j)T + c_j v v^T$	+ $(n^2 - n + .2)k$	+ $(1.5n^2 + 4n - 7.5)k$		
Totals	$1.2n^3 + 1.5n^2 + 1.3n$ + $.3k^3 + (1.5n - .5)k^2$ + $(2n^2 + .5n + 1.2)k$	$1.2n^3 + 2.5n^2 + .3n$ + $.5k^3 + (2n + 3)k^2$ + $(2.5n^2 + 6.5n - 5.5)k$	$k^2 + (2n - 1)k$ + $n - 1$	$n + .5k^2$ + $(n + .5)k$

Table E.6. Schmidt Colored Noise Propagation Costs for Triangular Covariance Square Root

Computations	Adds	Multiplies	Divides	Square Roots
$\begin{bmatrix}\bar{x}\\ \bar{p}\end{bmatrix} = \begin{bmatrix}\Phi_x & \Phi_{xp}\\ 0 & \Phi_p\end{bmatrix}\begin{bmatrix}\hat{x}\\ \hat{p}\end{bmatrix}$	$n^2 + nk$	$n^2 + nk + k$		
$\begin{bmatrix}\bar{S}_x & \bar{S}_{xp}\\ 0 & \bar{S}_p\end{bmatrix} = \begin{bmatrix}\Phi_x & \Phi_{xp}\\ 0 & M\end{bmatrix}\begin{bmatrix}S_x & S_{xp}\\ 0 & S_p\end{bmatrix}$	$.5n^3 + .5n^2 + .5nk^2$ + $(n^2 + .5n)k$	$.5n^3 + .5n^2$ + $(.5n + .5)k^2$ + $(n^2 + .5n + .5)k$		
$\overbrace{\begin{bmatrix}0 & \bar{S}_x & \bar{S}_{xp}\\ \sqrt{q_1}\ 0 & 0 & \bar{S}_p\\ 0\ \sqrt{q_k} & 0 & \end{bmatrix}}^{k\quad n\quad k} T = \begin{bmatrix}\bar{B} & \bar{S}_x & \bar{S}_{xp}\\ 0 & 0 & \bar{S}_p\end{bmatrix}$ *	$.3k^3 + (n + 1.5)k^2$ + $(3n + 1.2)k$	$.3k^3 + (n + 2)k^2$ + $(4n + .7)k$	k	k
$n\{\overbrace{[\ \bar{B}\ \vdots\ \bar{S}_x\ ]}^{k\quad n}\bar{T} = \begin{bmatrix}0 & \bar{S}_x\\ & 0\end{bmatrix}$	$.7n^3 + 1.3n + n^2k$	$.7n^3 + n^2 + .3n + n^2k$	$n - 1$	n
Totals	$1.2n^3 + 1.5n^2 + 1.3n$ + $.3k^3 + (1.5n + 1.5)k^2$ + $(2n^2 + 4.5n + 1.2)k$	$1.2n^3 + 2.5n^2 + .3n$ + $.3k^3 + (1.5n + 2.5)k^2$ + $(2n^2 + 5.5n + 2.2)k$	$n + k - 1$	$n + k$

*A sequence of k elementary Householder transformations is constructed to pack the last k rows. The ith transformation is defined by a u vector with i+1 nonzero components and thus alters only i+1 columns of the array (cf section 3.5).

Table E.7. Potter-Schmidt Colored Noise Update Costs

Computations	Adds	Multiplies	Divides	Square Roots
$\begin{bmatrix}\tilde{p}\\ \tilde{x}\end{bmatrix} = \begin{bmatrix}M & 0\\ \Phi_{xp} & \Phi_x\end{bmatrix}\begin{bmatrix}\hat{p}\\ \hat{x}\end{bmatrix}$	$n^2 + nk$	$n^2 + nk + k$		
$\begin{bmatrix}\bar{S}_p & \bar{S}_{px}\\ \bar{S}_{xp} & \bar{S}_x\end{bmatrix} = \begin{bmatrix}M & 0\\ \Phi_{xp} & \Phi_x\end{bmatrix}\begin{bmatrix}\hat{S}_p & \hat{S}_{px}\\ \hat{S}_{xp} & \hat{S}_x\end{bmatrix}$	$n^3 + nk^2 + 2n^2k$	$n^3 + (n+1)k^2 + (2n^2 + n)k$	n	n
$\left[\begin{array}{c\|cc}\sqrt{q_1} & & \\ & \ddots & \\ & & \sqrt{q_k}\end{array}\right.$ $\left.\begin{array}{cc}\bar{S}_p & \bar{S}_{px}\\ \hline \bar{S}_{xp} & \bar{S}_x\end{array}\right]$ $T = \left[\begin{array}{c\|cc}\sqrt{q_1} & & \\ & \ddots & \\ & & \sqrt{q_k}\end{array}\right.$ $\left.\begin{array}{cc}S'_p & \tilde{S}_{px}\\ \hline 0 & \tilde{S}_x\end{array}\right]$	$.7n^3 + n^2k + 1.3n + 2nk + 2k^2 + k$	$.7n^3 + n^2(k+1) + .3n + 2nk + 2k^2 + k$		
$\left[\begin{array}{cc\|c}\sqrt{q_1} & & \\ & \ddots & \\ & \sqrt{q_k} & \end{array}\right.$ $\left.\begin{array}{c}S'_p\\ \hline \end{array}\right]$ $\bar{T} = \left[\begin{array}{cc\|c} & & \\ & 0 & \\ & & \end{array}\right.$ $\left.\begin{array}{c}\tilde{S}_p\\ \hline \end{array}\right]$	$k^3 + .5k^2 + 1.5k$	$k^3 + k^2 + k$	$k - 1$	k
Totals	$1.7n^3 + n^2 + 1.3n + k^3 + (n + 2.5)k^2 + (3n^2 + 3n + 2.5)k$	$1.7n^3 + 2n^2 + .3n + k^3 + (n + 4)k^2 + (3n^2 + 4n + 3)k$	$n + k - 1$	$n + k$

Appendix F

Cost Comparison of Gain Evaluation Algorithms

Triangular covariance factorizations are employed in Chapter V to develop new algorithms for evaluating suboptimal measurement updates. These algorithms consist of two parts: an "optimal" update followed by an Agee-Turner matrix factorization (see section 5.1).

The arithmetic operations required to perform a gain evaluation using the UDU^T and SS^T triangular factorizations are summarized in Table F.1. Notice that the operation counts given here for the Bierman and Carlson optimal updates are different from those listed in Chapter IV. The differences are related to the costs of estimate calculations which are omitted from the error analysis counts.

From the total counts in Table F.1 it is apparent that the U-D factorization yields the more efficient evaluation algorithm. The significance of this difference in efficiency is illustrated by applying the UNIVAC 1108 weights[†] to the total counts in Table F.1. The weighted counts for each factorization method and the conventional method are included in Table F.2.

[†]Error analysis is usually performed on large, ground-based computers. Hence the UNIVAC weights (of Chapter IV) are appropriate for this cost comparison.

Table F.1 Arithmetic Operations Required By Gain Evaluation Algorithms

Computation	Adds	Multiplies	Divides	Square Roots
Bierman U-D measurement update	$1.5n^2 - .5n$	$1.5n^2 + 4.5n$	n	0
Agee-Turner U-D Factorization (Appendix C)	n^2	$n^2 + 3n - 2$	$n - 1$	0
TOTALS for U-D Algorithm	$2.5n^2 - .5n$ + $(.5n^2 + .5n)$*	$2.5n^2 + 7.5n - 2$ + $(n^2 - n)$*	$2n - 1$	0
Carlson Square Root Update	$1.5n^2 + .5n$	$2n^2 + 4n$	$2n + 1$	n
Agee-Turner SS^T Factorization (Appendix C)	n^2	$1.5n^2 + 5.5n - 5$	$2n - 2$	n
TOTALS for S Algorithm	$2.5n^2 + .5n$ + $(.5n^2 + .5n)$*	$3.5n^2 + 9.5n - 5$ + $(.5n^2 + .5n)$*	$4n - 1$	$2n$

Table F.2 Gain Evaluation Operation Counts Weighted For UNIVAC 1108

Algorithm	Execution Time/τ_+
U-D	$6n^2 + 19n - 7.3 + (1.9n^2 - 1.9n)$*
S	$7.4n^2 + 74.6n - 11.5 + (1.2n^2 + 1.2n)$*
Conventional (Stabilized Kalman)	$9.6n^2 + 7.6n + 4.5$

*variances computed

Figure F.1 displays the weighted counts from Table F.2 as a function of n and m where m represents the number of measurements included before variances are computed. Each cost has been normalized by the corresponding conventional evaluation cost. Hence the curves for S and U-D represent the percentage cost increase (or decrease) relative to the conventional evaluation method.

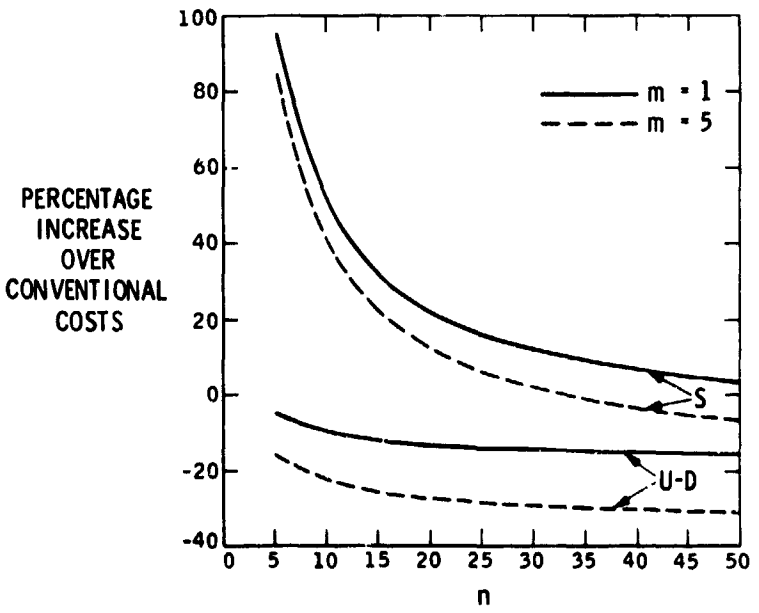

Fig. F.1. Cost Comparison of Arbitrary Gain Update Algorithms

Notice that for all values of n and m the U-D algorithm is more efficient than either of the other methods. Note also that the relative expense of the square root algorithm increases as n decreases. This increase is related to the costs of square root calculations which are more apparent when n is small. Even when n is large, however, the

U-D method is noticeably less costly than the square root scheme. When variances are required infrequently, the U-D evaluation is approximately 35% less expensive than the conventional method.

From the cost comparisons in Chapter IV we know that time propagation usually requires an order of magnitude more computation than does measurement updating. Hence Figures F.2 - F.5 give a more realistic comparison of error analysis costs. These figures display the UNIVAC costs associated with a U-D time update using the modified Givens algorithm, followed by m suboptimal measurement updates. The U-D costs have been normalized by the conventional error analysis costs and are given as a function of k/n for n = 10 or n = 30. Notice that for general systems the U-D algorithm usually requires less than 40% more computation than the conventional error analysis method, and when m = 5, less than 25% additional computation is required. For systems with colored process noise the U-D algorithm is particularly efficient, and when $k/n < 1.2$ this U-D method is faster than the conventional algorithm. In fact, when m = 5 the U-D method may require half as much calculation as the conven ional error analysis formula.

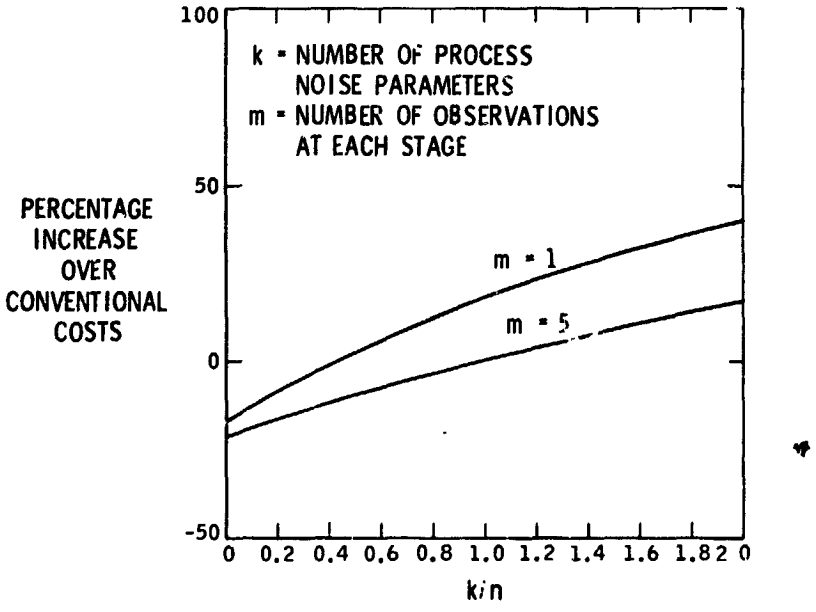

Fig. F.2. U-D Error Analysis Costs for General System, n = 10

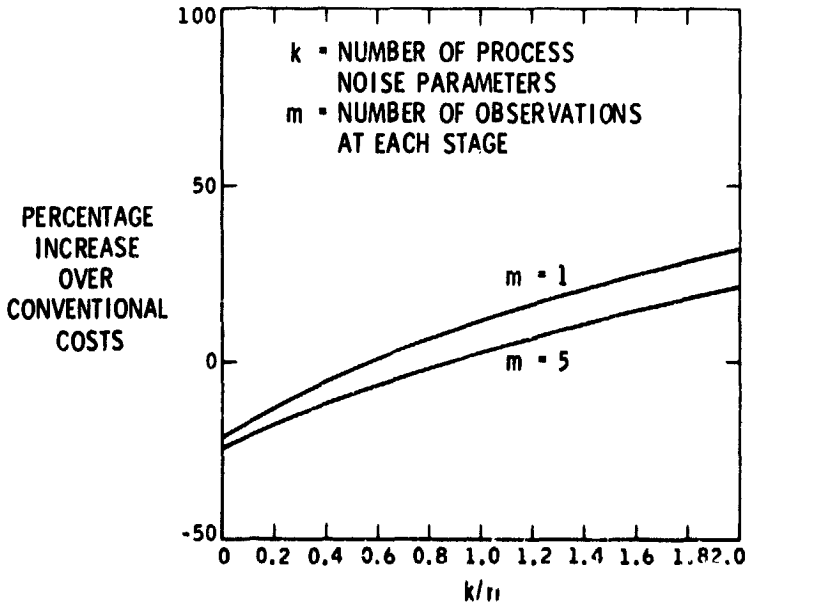

Fig. F.3. U-D Error Analysis Costs for General System, n = 30

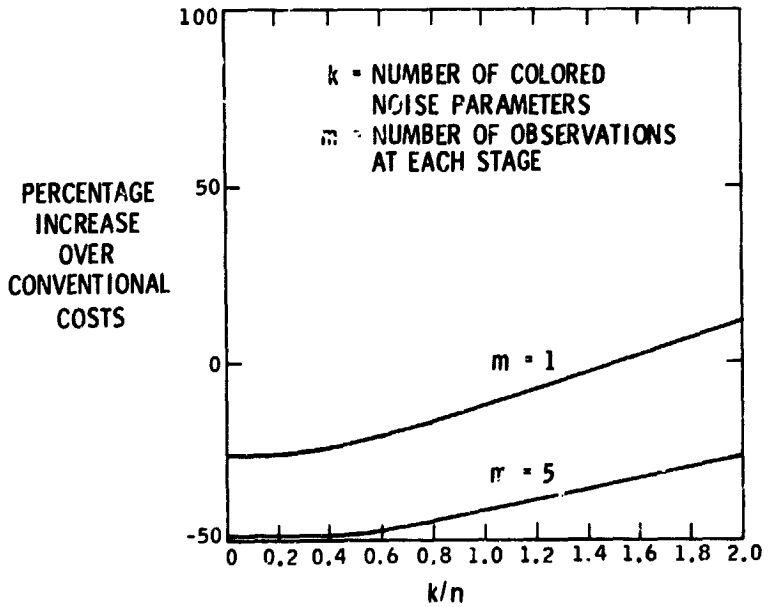

Fig. F.4. U-D Error Analysis Costs for Colored Noise System, n = 10

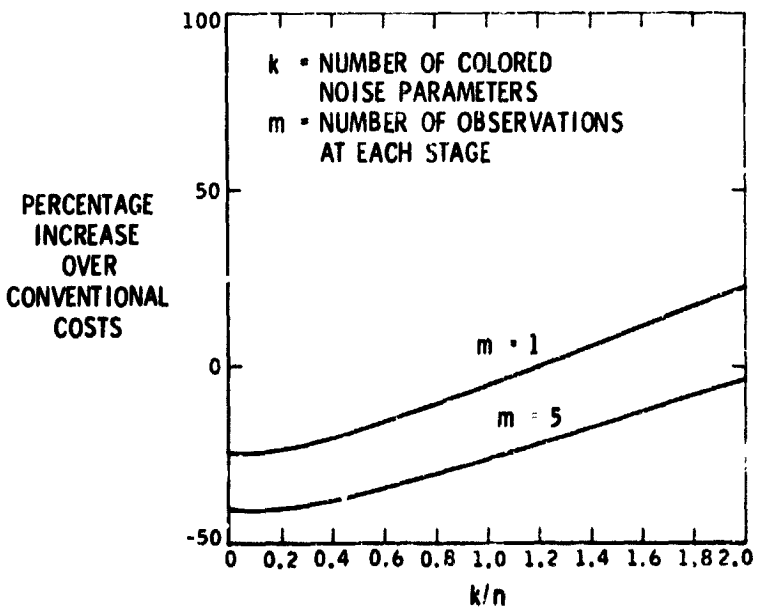

Fig. F.5. U-D Error Analysis Costs for Colored Noise System, n = 30

BIBLIOGRAPHY

[1] Agee, W. S. and R. H. Turner. "Triangular Decomposition of a Positive Definite Matrix Plus a Symmetric Dyad with Application to Kalman Filtering," Mathematical Services Branch, Analysis and Computation Division, White Sands Missile Range, New Mexico, Technical Report 38, 1972a.

[2] Agee, W. S. and R. H. Turner. "Optimal Estimation of Measurement Bias," Mathematical Services Branch, Analysis and Computation Division, White Sands Missile Range, New Mexico, Technical Report 41, 1972b.

[3] Andrews, A. "A Square Root Formulation of the Kalman Covariance Equations," AIAA J., 6(6):1165-1166, June 1968.

[4] Astrom, K. J. "Prediction and Filtering Theory," Introduction to Stochastic Control Theory, Academic Press, London, 1970.

[5] Athans, M. Ed. Special Issue on Linear Quadratic Gaussian Problem, IEEE Trans. Auto. Control, AC-16, December 1971.

[6] Bellantoni, J. F. and K. W. Dodge. "A Square Root Formulation of the Kalman-Schmidt Filter," AIAA J., 5:1309-1314, July 1967.

[7] Battin, R. H. and G. M. Levine. "Application of Kalman Filtering Techniques in the Apollo Program," Theory and Applications of Kalman Filtering, NATO Advisory Group for Aerospace Research and Development, AGARDograph 139, Technical Editing and Reproduction Limited, Harford House, London, February 1970.

[8] Bierman, G. J. "A Comparison of Discrete Linear Filtering Algorithms," IEEE Trans. on Aerospace and Electronic Systems, AE9-9(1):28-37, June 1973.

[9] Bierman, G. J. "Measurement Updating Using the U-D Factorization," to appear in Automatica, 1976a.

[10] Bierman, G. J. Factorization Methods For Discrete Sequential Estimation, Academic Press, New York, 1976b.

[11] Bjorck, A. "Solving Linear Least Squares Problems by Gram-Schmidt Orthogonalization," BIT 7:1-21, 1967.

[12] Businger, P. and G. H. Golub. "Linear Least Squares Problems by Householder Transformations," Numer. Math., 7:269-276, 1965.

[13] Bucy S. and P. D. Joseph. "Errors in the Filtering Process," Filtering for Stochastic Processes with Applications to Guidance, Interscience Tracts in Pure and Applied Mathematics, No. 23, John Wiley and Sons, New York, 1968.

[14] Carlson, N. A. "Fast Triangular Formulation of the Square Root Filter," AIAA J., 11(9):1259-1265, September 1973.

[15] Christensen, C. S. "Performance of the Square Root Information Filter for Navigation of the Mariner 10 Spacecraft," Jet Propulsion Laboratory, Pasadena, California, Technical Memorandum 33-757, January 1976.

[16] Curkendall, D. W., et al. "The Effects of Random Accelerations on Estimation Accuracy with Applications to the Mariner '69 Relativity Experiment," Jet Propulsion Laboratory, Pasadena, California, Technical Memorandum 158, January 1971 (JPL Internal Document).

[17] Dyer, P. and S. R. McReynolds. "Extension of Square Root Filtering to Include Process Noise," J. Optimization Theory and Applications 3(6):444-459, 1969.

[18] Fagin, S. L. "Recursive Linear Regression Theory, Optimal Filter Theory and Error Analysis of Optimal Systems," 1964 IEEE Intern. Conv. Record, New York, March, 1964, pp. 216-240.

[19] Fletcher, R. and M. Powell. "On the Modification of LDL^T Factorizations," Mathematics of Computation, 28(128):1067-1087, October 1974.

[20] Fox, L. "Practical Solution of Linear Equations and Inversion of Matrices," Nat. Bur. Standards Appl. Math., Ser. 39:1-54, 1954.

[21] Friedland, B. "Treatment of Bias in Recursive Filtering," IEEE Trans. Auto. Control, AC 14(4):359-367, 1969.

[22] General Dynamics-Electronics, San Diego, California, "Prime Item Development Specification for the GPS X User Set (Computer Program) of the NAVSTAR Space GPS User Equipment Segment Phase I," Specification CP-US-301, Part I, July 1975.

[23] Gentleman, W. M. "Least Squares Computations by Givens Transformations Without Square Roots," J. Inst. Math. Appl., 12:329-336, 1973.

[24] Gentleman, W. M. "Error Analysis of QR Decompositions by Givens Transformations," J. Linear Algebra and Its Applications, 10:189-197, 1975.

[25] Gill, P. E., G. H. Golub, W. Murray and M. A. Saunders. "Methods for Modifying Matrix Factorizations," Math. of Comp., 28(126):505-535, 1974.

[26] Givens, J. W. "The Linear Equations Problem," Applied Mathematics and Statistics Laboratories, Stanford University, Technical Report 3, 1959.

[27] Golub, G. H. "Numerical Methods for Solving Linear Least Squares Problems," Numer. Math., 7:206-216, 1965.

[28] Griffin, R. E. and A. P. Sage. "Sensitivity Analysis of Discrete Filtering and Smoothing Algorithms," AIAA J., 7(10):1890-1897, October 1967.

[29] Gura, I. A. and A. B. Bierman. "On Computational Efficiency of Linear Filtering Algorithms," Automatica, 7:299-314, 1971.

[30] Heffes, J. "The Effects of Erroneous Models on the Kalman Filter Response," IEEE Trans. Auto. Control, 11(3):541-543, July 1966.

[31] Holdsworth, J. and J. Stolz. "Marine Applications of Kalman Filtering," Theory and Applications of Kalman Filtering, NATO Advisory Group for Aerospace Research and Development, AGARDograph 139, Technical Editing and Reproductions Limited, Harford House, London, February 1970.

[32] Householder, A. S. "Direct Methods of Inversion," The Theory of Matrices in Numerical Analysis, Blaisdell, Waltham, Massachusetts, 1964.

[33] Huddle, J. R. "Application of Kalman Filtering Theory to Augmented Inertial Navigation Systems," Litton Systems, Incorporated, Guidance and Control Systems Division, Woodland Hills, California, Publication 8903, March 1969.

[34] Jazwinski, A. H. Stochastic Processes and Filtering Theory, Academic Press, New York, 1970.

[35] Jordan, T. L. "Experiments on Error Growth Associated with Some Linear Least-Squares Procedures," Math. of Comp., 22:579-588, 1968.

[36] Kailath, T. "A View of Three Decades of Linear Filtering Theory," IEEE Trans. on Information Theory, IT-20(2):146-181, March 1974.

[37] Kailath, T. "Some New Algorithms for Recursive Estimation in Constant Linear Systems," IEEE Trans. on Information Theory, IT-19(6):750-760, November 1973.

[38] Kalman, R. E. "A New Approach to Linear Filtering and Prediction Problems," ASME Trans., 82D:35-50, 1960.

[39] Kaminski, P. G., Square Root Filtering and Smoothing for Discrete Processes, PH.D in Engineering, Aeronautics and Astronautics, Stanford University, Stanford, California, July 1971a.

[40] Kaminski, P. G., A. E. Bryson and S. F. Schmidt. "Discrete Square Root Filtering: A Survey of Current Techniques," IEEE Trans., AC-16 (6):727-736, 1971b.

[41] Lawson, C. L. and R. J. Hanson. "Other Methods for Least Squares Problems," Solving Least Squares Problems, Prentice Hall, Englewood Cliffs, New Jersey, 1974.

[42] Lindquist, A. "A New Algorithm for Optimal Filtering of Discrete-Time Stationary Processes," SIAM J. Control, 12(4):736-746, 1974.

[43] Martin, P. S., G. Peters, and J. H. Wilkinson. "Symmetric Decomposition of a Positive Definite Matrix," Numer. Math., 7:362-383, 1965.

[44] Maughmer, R. W. and D. M. Byrd. "Supplement to Final Report Advanced Study of a Position and Azimuth Determining System," Advanced Systems Laboratory, Litton Systems, Incorporated, Guidance and Control Systems Division, Woodland Hills, California, Engineering Publication Document AQ51073J685, September 1969.

[45] Morf, M. and T. Kailath. "Square Root Algorithms for Least Squares Estimation," IEEE Trans. Auto. Control, AC-20(4):487-497, August 1975.

[46] Moyer, T. D. "Mathematical Formulation of the Double Precision Orbit Determination Program (DPODP)," Jet Propulsion Laboratory, Pasadena, California, Technical Report 32-1527, 1971.

[47] Nishimura, T. "On the A Priori Information in Sequential Estimation Problems," IEEE Trans. Auto. Control, AC-11(2):197-204, April 1966.

[48] Nishimura, T. "Correction to, and Extension of 'On the A Priori Information in Sequential Estimation Problems'," IEEE Trans. Auto. Control, AC-12(1):123, February 1967.

[49] Noble, B. Applied Linear Algebra, Prentice Hall, Englewood Cliffs, New Jersey, 1969.

[50] O'Neill, W. J., et al. "Mariner 9 Navigation," Jet Propulsion Laboratory, Pasadena, California, Technical Report 32-1586, November 1973.

[51] Potter, J. E. "New Statistical Formulas," Instrumentation Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, Space Guidance Analysis Memo 40, April 1963.

[52] Rice, J. R. "Experiments on Error Growth Associated with Some Linear Least Squares Procedures," Math. of Comp., 22:579-588, 1968.

[53] Sage, A. P. and J. L. Melsa. Estimation Theory with Applications to Communications and Control, McGraw-Hill, New York, 1971.

[54] Schmidt, S. F. "Computational Techniques in Kalman Filtering," Theory and Application of Kalman Filtering, NATO Advisory Group for Aerospace Research and Development, AGARDograph 139, Technical Editing and Reproductions Limited, Harford House, London, February 1970.

[55] Schmidt, S. F. "Estimation of State with Acceptable Accuracy Constraints," Analytical Mechanics Associated, Palo Alto, California, Technical Report 67-16, 1967.

[56] Schmidt, S. F., J. D., Weinburg and J. S. Lukesh. "Case Study of Kalman Filtering in the C-5 Aircraft Navigation System," Theory and Application of Kalman Filtering, NATO Advisory Group for Aerospace Research and Development, AGARDograph 139, Technical Editing and Reproductions Limited, Harford House, London, February 1970.

[57] Wilkinson, J. H. "Solution of Linear Algebraic Equations," The Algebraic Eigenvalue Problem, Oxford University Press, London, 1965.

[58] Wampler, R. H. "A Report on the Accuracy of Some Widely Used Least Squares Computer Programs," J. American Statistical Association, 65(330):549-565, 1970.