

LEVEL II

(14)

AD A0 66375

TRAJECTORY RECONSTRUCTION AND ANALYSIS METHODOLOGY

Mathematics and Guidelines for Program Implementation

Dr. R. A. Brooks

DDC
RECEIVED
MAR 26 1979

DDC FILE COPY

PERFORMANCE ANALYSIS DEPARTMENT

FEDERAL ELECTRIC CORPORATION
WTR DIVISION, VANDENBERG AFB, CALIFORNIA 93437

NOVEMBER 1978
FINAL REPORT

APPROVED FOR PUBLIC RELEASE
UNLIMITED DISTRIBUTION

Prepared for
SPACE AND MISSILE TEST CENTER (AFSC)
VANDENBERG AFB, CALIFORNIA 93437


6 9 0 0 ~ 0 0 0 0

This final report was submitted by Federal Electric Corporation, Vandenberg AFB, CA 93437 under Contract F04703-77-C-0111 with the Space and Missile Test Center, Vandenberg AFB, CA 93437. Operations Research Analyst, Mr. J. McConnell, XRQR, was the Division Project Engineer-in-Charge.


This report has been reviewed by the Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.


John B. McConnell GS-13
Project Engineer


Eugene F. Flary GS-1
Chief, Requirements and
Evaluation Division

FOR THE COMMANDER


ROBERT E. FOSTER, Colonel, USAF
Director of Plans, Programs &
Resources

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER 18 SAMTEC TR-79-1	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER 9	
4. TITLE (and Subtitle) 6 TRAJECTORY RECONSTRUCTION AND ANALYSIS METHODOLOGY Mathematics and Guidelines For Program Implementation,		5. TYPE OF REPORT & PERIOD COVERED Final Report	
7. AUTHOR(s) 10 Dr. R. A. Brooks		8. CONTRACT OR GRANT NUMBER(s) 14 PA100-78-58	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Performance Analysis Department Federal Electric Corporation, WTR Division Vandenberg Air Force Base, CA 93437		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Space and Missile Test Center Contract # F04703-77-C-0111 78032F	
11. CONTROLLING OFFICE NAME AND ADDRESS Space and Missile Test Center (AFSC) Code XRXP Vandenberg Air Force Base, CA 93437		12. REPORT DATE 11 29 November 1978	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 12 193p.		13. NUMBER OF PAGES 189	
		15. SECURITY CLASS. (of this report) UNCLASSIFIED	
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Optimum estimation, trajectory reconstruction, square root filtering, error analysis.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Optimal estimation techniques are applied to certain fundamental problems associated with ballistic missile testing. These problems include multiple vehicle trajectory reconstruction and performance analysis of metric, telemetry and vehicle subsystems. The mathematical background and guidelines for program implementation are provided for a recursive filtering and smoothing realization of the optimum linear estimator. Considerable emphasis is placed on efficient

411 113

UNCLASSIFIED

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Block 20 (Continued)

computation with minimal introduction of processing error. System considerations are applied to the aggregate tracking and estimation functions, and the optimum interface between tracking (data collection) and estimation (data processing) is identified.

ACCESSION TO
NTIS
DDC
UNANNOUNCED
JUSTIFICATION
BY
DISTRIBUTION
Dist. A3
A

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

TABLE OF CONTENTS

	<u>Page</u>
LIST OF FIGURES	<i>v</i>
<u>Section</u>	
1.0 INTRODUCTION	1
2.0 SCOPE OF THE REPORT	3
3.0 SYSTEM CONSIDERATIONS APPLIED TO ESTIMATOR IMPLEMENTATION	4
4.0 OVERVIEW	6
4.1 Estimation	6
4.2 Error Analysis	8
4.3 Sequential Linear Estimation	9
4.4 Nonlinear Sequential Estimation	11
4.5 Recapitulation	12
5.0 MATHEMATICAL DEVELOPMENT	15
5.1 Estimation	16
5.2 Error Analysis	22
6.0 COMPUTATIONAL CONSIDERATIONS	31
6.1 Scalar Measurement Update	31
6.2 Square Root Filter	34
6.3 Computation of Nominal State Vector and Transition Matrix	45
6.4 Numerical Calculation of Partial Derivatives	51
6.5 Transit Time, Refraction, and Doppler Calculations	55

TABLE OF CONTENTS (Cont'd)

	<u>Page</u>
6.6 Measurement Processing	59
6.6.1 Dynamic Measurement Algorithm	61
6.6.2 Measurement Equation Partial Derivatives	66
6.6.3 Static Measurement Algorithm	67
6.6.4 Measurement Processing with Adjustable Estimation Times	68
6.6.5 Processing with Measurement Variation Average	70
6.7 Augmentation and Permutation of the State Vector	74
6.8 General Partitioned Structure of the State Vector, Transition Matrix, and Measurement Sensitivity Matrix	78
6.9 Suboptimal Processing in the Presence of IMU Noise	81
7.0 PROGRAM REQUIREMENTS	85
7.1 Pre-Processing Phase	86
7.1.1 Metric Data Pre-Processing	86
7.1.2 Telemetry Data Pre-Processing	88
7.2 Estimation Phase	89
7.2.1 Input Requirements	90
7.2.2 Initialization Requirements	92
7.2.3 Filter Requirements	95
7.2.4 Smoother Requirements	101
7.2.5 Convergence Test	103
7.2.6 Reset	104
7.3 Error Analysis Phase	104

TABLE OF CONTENTS (Cont'd)

	<u>Page</u>
REFERENCES	106
APPENDICES	
A. Linear Estimation	109
B. Variational Form of the Navigation Equations	123
C. Metric Sensor Systems	136
D. Miscellaneous Topics in Linear Algebra	162

LIST OF FIGURES

<u>FIGURE</u>		<u>PAGE</u>
4.1	Multiple RV Mission	7
4.2	Two Stage Optimal Linear Estimator Applied to Multiple RV Mission	10
4.3	Flow Diagram of Iterative Estimation and Error Analysis for Multiple RV Mission	13
5.1	Estimator Flow Diagram	23
5.2	Error Propagation Flow Diagram	29
6.1	Scalar Measurement Update	33
6.2	Error Propagation Modified for Scalar Measurement Update	35
6.3	Square Root Filter Update	40
6.4	Square Root Covariance Extrapolation	46
C.1	Metric Sensor System Flow Diagram	137
C.2	Spherical Coordinates with Respect to Topocentric Frame at Sensor	140
C.3	Linear Model for Single Channel Metric Sensor	156

1.0 INTRODUCTION

In recent years the techniques of estimation theory have been applied to many problems associated with ballistic missile flight testing. In particular, trajectory reconstruction, weapon system analysis, metric sensor evaluation, and geodetic and geopotential model refinement are examples of problem areas to which estimation theory has been successfully applied.

In practice, estimation theory is used in two ways; namely to estimate various trajectory and system parameters using available measurement data, and to perform an error analysis of the estimation process itself.

Test range activities, such as pre-mission planning and post-mission analysis, are strongly influenced by requirements imposed for trajectory reconstruction and system performance analysis. Planning for mission support usually involves error analyses based on alternative range instrumentation configurations. Post-mission activities include estimation of trajectory and system parameters and associated error analyses.

Future test range activity is projected to involve more accurate guidance systems, advanced reentry systems, and range instrumentation with more accuracy and precision. The ability to meet future range testing requirements will depend to a great extent on the fidelity and generality of the estimation techniques employed.

Computer programs currently in use at SAMTEC for estimation and error analysis are of the batch processing variety and, as such, have certain deficiencies. In the first place, whenever the trajectory constraints are noisy, such as when derived from noisy guidance data or during uninstrumented reentry, it is not practical to perform optimum trajectory reconstruction with batch processing techniques. Secondly, the types of estimation problems which arise in connection with range operations involve nonlinear equations, and the iterative methods of solution these problems require often converge slowly, or even diverge, when implemented in a batch processor.

The alternative to batch processing is recursive processing which not only admits optimum trajectory reconstruction when the trajectory constraints are noisy, but also provides excellent convergence properties which can result in considerable computational superiority over batch processing when the latter requires many iterations [1].

Because of the limitations of existing programs vis-a-vis projected future requirements, an effort has been initiated at SAMTEC to develop a new program (TRAM) to meet these requirements. The TRAM program employs recursive processing so that more general optimum estimation techniques can be implemented and better convergence properties achieved than are possible with batch processing.

2.0 SCOPE OF THE REPORT

In Section 3.0 of the report, a system viewpoint is adopted for the aggregate functions of tracking and estimation. This viewpoint provides insight which is useful in estimator implementation.

In Section 4.0 an overview of TRAM operation and its applications is provided. The discussion, although purely qualitative, illustrates the processing techniques employed in TRAM and the capabilities therein achieved.

Section 5.0 provides the mathematical development on which TRAM is based. The fundamental estimation and error analysis equations are developed in this section.

A discussion of computational techniques and trades is given in Section 6.0. Included are specific algorithms and methods for computer implementation.

Section 7.0 is a discussion of program requirements which must be satisfied by TRAM. Guidelines for program development, rather than detailed specifications are given in this section.

The report also includes a set of appendices. In the main, the appendices provide support for the material in Sections 5.0 and 6.0. However, some discussion of vehicle and metric instrumentation systems, together with mathematical models, is also included.

3.0 SYSTEM CONSIDERATIONS APPLIED TO ESTIMATOR IMPLEMENTATION

The processing which is implemented in TRAM is based on the theory of optimal linear estimation. The fundamental assumptions required for optimality, together with the basic algorithms of linear estimation, are discussed in Appendix A. In order to implement these algorithms in a manner which most nearly satisfies the conditions required for optimality, it is advantageous to view the tracking and estimation functions as a composite system.

The utility of the system viewpoint is that it enables a clear distinction to be drawn between tracker and estimator functions. This in turn leads to the establishment of system interfaces which facilitate the implementation of the estimator in an optimum form.

In order for an estimator to be optimum, its mechanization must be based on models for all processes which have occurred in the generation of the measurements at its input. Thus, if the interfaces between the estimator and the functions which precede it are not carefully selected, the resulting estimator either will be overly complicated or it will perform sub-optimally.

Much of the discussion in Appendix C is directed to establishing the interfaces between the tracking and estimation functions by identifying the most useful outputs from tracking instrumentation. These outputs are shown to include data from both the encoder and the sensing element of each tracking channel.

In Appendix C.3, the effect of collecting and processing only the encoder data is analyzed. It is shown that this results both in suboptimum smoothing and in the introduction of tracking error, neither of which can be fully compensated in the estimator.

Since an optimum estimator inherently performs smoothing of noise process errors and compensation for modeled systematic error, pre-processing of

track data for either of these purposes is superfluous. In particular, pre-smoothing of track data may destroy information and result in degraded estimator performance.

The above considerations are reflected in the formulation of TRAM processing algorithms. Considerable emphasis has been placed on minimizing the introduction of processing error. Also, the algorithms have been devised to allow optimum processing of joint encoder and sensing element outputs whenever both are available.

4.0 OVERVIEW

The purpose of this section is to provide an overview of the principal TRAM operations, and develop a framework for subsequent mathematics and program requirements sections. To illustrate TRAM operation a multiple reentry vehicle (RV) mission will be considered. A typical mission is depicted in Figure 4.1. Illustrated there are the trajectory segments of the boost vehicle (BV) and the RVs from launch to impact. Also noted are the separation and pierce points of each RV.

During the mission, off-board data is collected by various metric sensor systems and on-board data is collected by telemetry systems.

Post mission processing of the metric and telemetry data is performed by TRAM. The objectives of TRAM processing are twofold:

1. Optimal estimation of selected trajectory, instrumentation, geodetic, geopotential, and aerodynamic parameters.
2. Error analysis of the estimated parameters.

4.1 Estimation

The parameters to be estimated are selected from a state vector composed of the following groups:

1. DYNAMIC (TIME VARYING) TRAJECTORY GROUP
 - a. position and velocity for each vehicle
 - b. time correlated IMU or aerodynamic parameters induced by random phenomena
2. METRIC SENSOR GROUP
 - a. sensor and pedestal
 - b. beacon
 - c. refraction
 - d. geodetic
 - e. timing

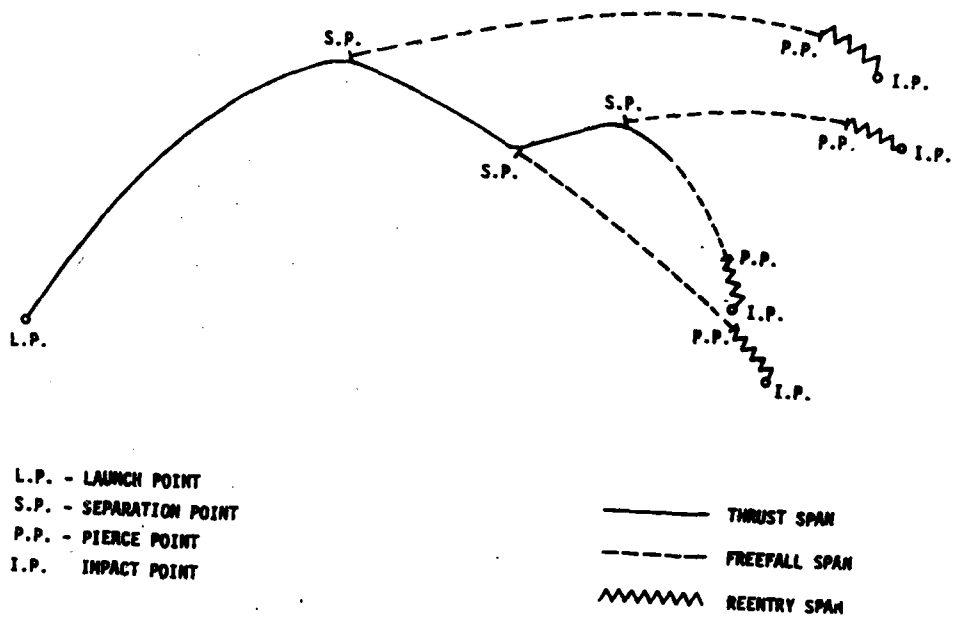


FIGURE 4.1 MULTIPLE RV MISSION

3. INERTIAL MEASURING UNIT (IMU) GROUP
 - a. timing
 - b. platform, gyro, and accelerometer
4. STATIC (CONSTANT) TRAJECTORY GROUP
 - a. geopotential
 - b. aerodynamic

In essence the state vector consists of all parameters which appear in either trajectory or instrumentation equations. The reasons for partitioning and ordering the state vector elements into the above groups will become clear in Section 6.0.

The subset of state vector elements which are estimated are called the estimated states. The remaining state vector elements are called the constrained states.

4.2 Error Analysis

The error in an estimated parameter is defined in general by

$$\text{error} = \text{estimate} - \text{true.}$$

The purpose of an error analysis is to quantify, to the extent possible, the errors which remain after estimation. Since the true values of the parameters are not generally known (except in simulations) the best that can be done is to provide a probabilistic description of estimation error. An example of such a description would be the means, variances, and cross correlations of the set of estimation errors.

An estimation error analysis is a two stage process. The first stage, which is performed concurrently with estimation, consists of calculating the sensitivities of the parameter estimates to each of the error sources to be considered. The second stage combines these sensitivities with an error budget (i.e., a statistical description of the errors for the sources being considered) to obtain a probabilistic characterization of the estimation errors. The second stage can either be performed concurrent with or subsequent to estimation.

The sources of error considered in the error analysis generally include sensor and trajectory noise errors and state vector initialization errors.

The subset of states whose initialization errors are included in the error analysis are called propagated states. The set of propagated states, which always includes the estimated states, may, in addition, include any subset of the constrained states.

4.3 Sequential Linear Estimation

In general, an optimal linear estimator can be implemented by means of a two stage sequential algorithm. The two stages are, respectively, filter and smoother. The application of a sequential two stage linear estimator to the multiple RV mission is illustrated in Figure 4.2. Examination of the flow diagram shown in this figure reveals that all trajectory segments for boost and reentry vehicles are first filtered, and then each segment is smoothed. The order in which the segments are filtered is somewhat arbitrary, but the order of smoothing is the exact reverse of the filtering order. Furthermore, the filter operates on each segment by processing the data in the direction of increasing time. The smoother, on the other hand, processes the filter outputs in the direction of decreasing time.

The filter provides estimates utilizing only the data processed up to and including estimation time. The smoother provides estimates utilizing all of the data, by adjusting the filter estimates. It should be noted, however, that only the dynamic parameter estimates (i.e., those in Group 1) require adjustment by smoothing. The estimates of the static (constant) parameters which are obtained at the end of the filter stage are unaffected by smoothing, since these estimates are already based on the entire data set.

For processing convenience and flexibility, each trajectory segment can be partitioned into regions of powered flight, freefall, and reentry and these regions can be further partitioned into intervals over which sensor coverage does not change.

In addition, in order to facilitate and enhance the efficiency of bulk storage (i.e., disk or tape) input/output (I/O) operations, the intervals

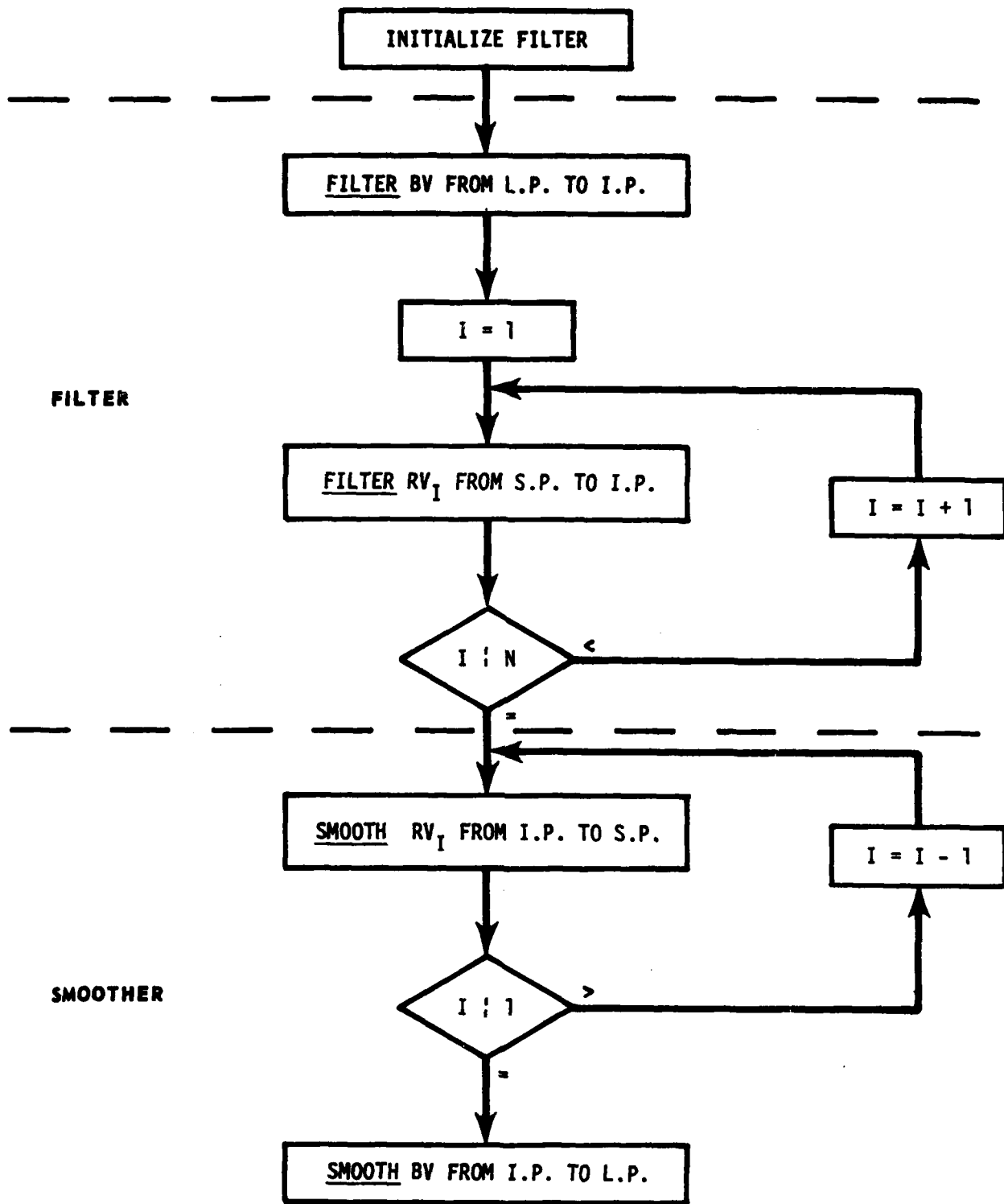


FIGURE 4.2 TWO STAGE OPTIMAL LINEAR ESTIMATOR APPLIED TO MULTIPLE RV MISSION

mentioned above can be further partitioned into subintervals at the junctures of which all I/O operations are performed. The size of these subintervals can be selected small enough such that the instantaneous storage capacity in the computer memory is not exceeded and large enough to maintain the number of I/O operations sufficiently small that computer efficiency does not suffer.

During filtering, the input operations entail reading metric and telemetry data, and the output operations consist of writing filter estimates of the whole state and the sensitivities of these estimates to the propagated states. During smoothing, the input operations read filter outputs, and the output operations write smoother estimates and the sensitivities of these estimates to the propagated states.

4.4 Nonlinear Sequential Estimation

In order to apply linear estimation to the multiple RV mission, a further refinement is necessary, because the equations governing such missions are, in fact, nonlinear. The refinement to be discussed is analogous to the use of the Newton-Raphson method for solving nonlinear algebraic equations. The algebraic method uses relinearization and iteration to obtain a solution. On each iteration a set of linear equations is solved, and the solution is used to relinearize the nonlinear equations to obtain the set of linear equations to be solved on the next iteration. The process is terminated when convergence occurs, i.e., when identical solutions are obtained on successive iterations.

In the estimation problem, the nonlinear equations are linearized about a nominal value of the state vector at each step in the sequential process. The nominal value of the state vector is an arbitrary approximation to the true state vector. The resulting equations are linear in the state vector variation which is defined component by component by

$$\text{variation} = \text{true} - \text{nominal}.$$

The linear estimation equations are applied to estimate the variation. The estimate of the whole state vector is then given component by component by

$$\text{whole estimate} = \text{nominal} + \text{variation estimate}.$$

For each component in the subset of constrained states, the whole estimate is by definition equal to the constrained value of the component.

The technique for nonlinear estimation consists of iterating the entire two stage sequential filter/smoothing operations. On each iteration, the variation estimate obtained upon completion of the filtering and smoothing of operations is added to the nominal to obtain an estimate of the whole state vector. The whole state estimate thus obtained is then used as the nominal state vector on the next iteration. This relinearization and iteration process continues until the variation estimates converge to zero, i.e., the whole state estimates on two successive iterations are identical.

On the first pass the nominal state vector is initialized with the best prior estimate available. Then during the filter stage the nominal state is reset periodically by equating the nominal state to the whole value estimate obtained by the filter.

The questions regarding convergence of the above procedure are not easily answered. While there are well known necessary and sufficient conditions for convergence, they are not easily verified for the class of trajectory estimation problems of interest here. However, experience has shown that, in well formulated problems of this nature, convergence generally occurs within several iterations provided the nominal state is initialized sufficiently close to the true state. The periodic nominal reset procedure described above is designed to maintain the variation sufficiently small on the first pass that convergence occurs quickly thereafter.

4.5 Recapitulation

The estimation procedure and the error analysis which can be performed concurrently with estimation are concisely represented by the flow diagram of Figure 4.3. The diagram illustrates an outer loop for relinearization and iteration, middle loops for interval processing, and inner loops for processing of subintervals. Counters I, J, and K control the outer, middle and inner loops, respectively.

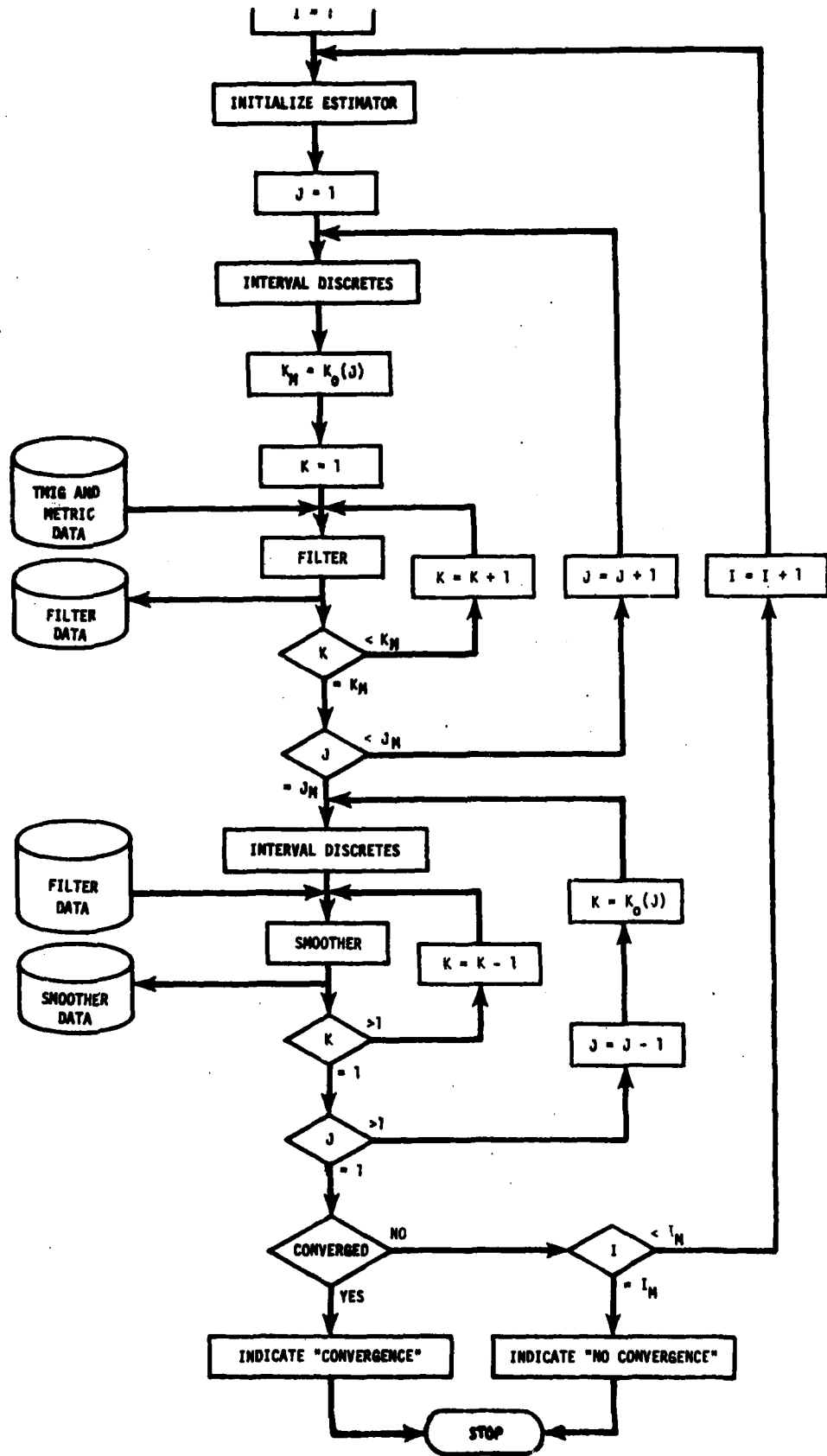


FIGURE 4.3 FLOW DIAGRAM OF ITERATIVE ESTIMATION AND ERROR ANALYSIS FOR MULTIPLE RV MISSION

The outer loop is repeated until convergence is achieved or the maximum allowed number of iterations has occurred. The middle and inner loops are traversed in a forward direction (counters increasing) during the filter stage and in the reverse direction (counters decreasing) during the smoother stage. Although it is not shown in the diagram, within each subinterval the filter processes input data sequentially with time increasing, and the smoother processes filter data sequentially in the reverse (i.e., time decreasing) direction.

Each interval, it will be recalled, consists of a trajectory segment portion, which is exclusively powered flight, freefall, or reentry, over which sensor coverage does not change. The interval discrete functions control filter/smoothing configuration over each interval by specifying sensors to be processed, parameters to be estimated, times of discrete events within the estimator, and so forth.

The subinterval discrete functions consist entirely of bulk storage I/O operations.

It was stated earlier that the error analysis and estimation procedures can be performed concurrently. This is accomplished by augmenting the filter and smoother with error propagation functions. However, because of the iterative feature of the estimator, it is more efficient to allow the estimator to converge before performing the error propagation functions. Thus, with reference to Figure 4.3, after convergence, one more pass through the outer loop is taken in which only the error propagation functions are performed.

5.0 MATHEMATICAL DEVELOPMENT

In Section 4.0, the major TRAM functions of estimation and error analysis were discussed in qualitative terms for a multiple RV mission. The concept of a state vector was introduced, and its composition by various groups was presented. The state vector consists of parameters which affect either trajectories or measurements. For purposes of estimation, two subvectors of the state vector are defined. The first contains the estimated states. The second contains the constrained states. For the error analysis, a subvector of propagated states is defined which contains all the estimated states and, in addition, some subset of the constrained states.

In this section, the basic equations for both estimation and error analysis will be developed using the state vector concept.

The state vector, denoted by x , satisfies a nonlinear differential equation

$$(1) \quad \dot{x}(t) = f[x(t), t] + w(t), \quad t \geq t_0,$$

called the state equation. The metric sensor measurements, denoted by y , are functions of state at discrete times and satisfy a nonlinear equation,

$$(2) \quad y(t_i) = h[x(t_i), t_i] + v(t_i), \quad i = 0, 1, 2, \dots,$$

called the measurement equation. The quantities w and v are called the state (or plant) and measurement noise processes, respectively.

While the TMIG measurements can also be expressed in the form given by (2), it is more convenient to express these measurements in a form suitable for introduction in the right hand side of (1) as a direct measurement of vehicle dynamics.

If the state vector is ordered and partitioned into the four groups defined in Section 4.0, equation (1) becomes

$$(3) \quad \begin{bmatrix} \dot{x}_a \\ \dot{x}_b \\ \dot{x}_c \\ \dot{x}_d \end{bmatrix} = \begin{bmatrix} f_a[x_a(t), x_c, x_d, t] \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} w_a(t) \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad t \geq t_0,$$

where the groups are identified as follows:

x_a , dynamic trajectory group

x_b , metric sensor group

x_c , inertial sensor group

x_d , static trajectory group

Similarly, equation (2) becomes

$$(4) \quad y(t_i) = h[x_a(t_i), x_b, t_i] + v(t_i), \quad i = 0, 1, 2, \dots$$

Equation (3) shows explicitly the dynamic behavior of the first group and the static nature of the remaining groups. Also shown is the fact that in general the dynamic behavior of x_a depends on x_a , x_c and x_d . However, the dependence of x_a on x_c occurs only in inertially instrumented trajectory spans. Equation (4) shows that the metric sensor measurements depend only on x_a and x_b .

The state noise component $w_a(t)$ represents the effect of random forces, such as aerodynamic forces, on those trajectory spans which are not inertially instrumented. On inertially instrumented spans, all inertial forces, including random forces, are sensed and incorporated into f_a ; and $w_a(t)$ represents noise in the inertial instruments.

5.1 Estimation

In this subsection, equations (1) and (2) will be used as a point of departure. However, explicit dependence on the subvector of constrained

states will be suppressed, and x will denote only the subvector of estimated states.

To apply linear estimation techniques, equations (1) and (2) must be linearized with respect to the state vector. This can be accomplished over any time interval of interest, say $[t', t'']$, in the following manner.

Let \tilde{x} denote the solution of the differential equation

$$(5) \quad \dot{\tilde{x}}(t) = f[\tilde{x}(t), t], \quad t' \leq t \leq t'' ,$$

with $\tilde{x}(t')$ arbitrarily specified. \tilde{x} is called the nominal state vector. Then, expansion of equations (1) and (2) to first order about \tilde{x} leads to

$$(6) \quad \dot{x}(t) = f[\tilde{x}(t), t] + F(t)[x(t) - \tilde{x}(t)] + w(t) ,$$

$$(7) \quad y(t_i) = h[\tilde{x}(t_i), t_i] + H(t_i)[x(t_i) - \tilde{x}(t_i)] + v(t_i) ,$$

where

$$(8) \quad F(t) \equiv \left(\frac{\partial f}{\partial x^T} \right)_{\tilde{x}(t), t} ,$$

and

$$(9) \quad H(t_i) \equiv \left(\frac{\partial h}{\partial x^T} \right)_{\tilde{x}(t_i), t_i} .$$

Now define the state and measurement variations, respectively, by

$$(10) \quad \delta x(t) \equiv x(t) - \tilde{x}(t) ,$$

and

$$(11) \quad \delta y(t_i) \equiv y(t_i) - h[\tilde{x}(t_i), t_i] .$$

From (5), (6), and (7) it then follows that

$$(12) \quad \dot{\delta x}(t) = F(t)\delta x(t) + w(t) ,$$

and

$$(13) \quad \delta y(t_i) = H(t_i)\delta x(t_i) + v(t_i) ,$$

for all t and t_i in the interval $[t', t'']$.

The solution of (12) can be expressed in a convenient form using the transition matrix, ϕ , defined by

$$(14) \quad \phi(t, s) = \Psi(t)\Psi^{-1}(s)$$

for all t and s in $[t', t'']$, where Ψ is the so-called fundamental matrix defined by the differential equation

$$(15) \quad \dot{\Psi}(t) = F(t)\Psi(t), \Psi(t') = I, t' \leq t \leq t'' .$$

The existence of $\Psi^{-1}(t)$ is guaranteed for all $t' \leq t \leq t''$, and thus ϕ is well defined by (14).

In terms of ϕ , the solution of (12) is expressed by

$$(16) \quad \delta x(t) = \phi(t, s)\delta x(s) + \int_s^t \phi(t, r)w(r)dr$$

for all t and s in $[t', t'']$. In particular, for t_i and t_{i+1} in $[t', t'']$,

$$(17) \quad \delta x(t_{i+1}) = \phi(t_{i+1}, t_i)\delta x(t_i) + u(t_i) ,$$

where

$$(18) \quad u(t_i) = \int_{t_i}^{t_{i+1}} \phi(t_{i+1}, r)w(r)dr .$$

By specifying \tilde{x} at a finite set of times and using (5) to obtain \tilde{x} in the intervals between these times, the linear equations obtained above can be

extended to hold for all times of interest on the multiple vehicle trajectories. Thus the linear state and measurement equations become

$$(19) \quad \delta x(t_{i+1}) = \Phi(t_{i+1}, t_i) \delta x(t_i) + u(t_i),$$

$$(20) \quad \delta y(t_i) = H(t_i) \delta x(t_i) + v(t_i), \quad i = 0, 1, 2, \dots,$$

with the provision that whenever \tilde{x} is reset, δx must also be reset by the relation

$$(21) \quad \delta x(t^+) = \delta x(t^-) - [\tilde{x}(t^+) - \tilde{x}(t^-)],$$

where t^- and t^+ denote values before and after reset, respectively.

Equations (19) and (20) provide the model basis for application of the two stage (filter/smoothen) linear estimation procedure which will now be developed. With somewhat briefer notation, the estimation model is given by

$$(22) \quad \delta x_{i+1} = \Phi_i \delta x_i + u_i,$$

$$(23) \quad \delta y_i = H_i \delta x_i + v_i, \quad i = 0, 1, 2, \dots$$

The sequential filtering and smoothing algorithms of linear estimation, together with the assumptions on which they are based, are discussed fully in Appendix A.

The filter algorithm for (22) and (23) is given by

$$(24) \quad K_i = P_i^- H_i^T [H_i P_i^- H_i^T + R_i]^{-1},$$

$$(25) \quad \hat{\delta x}_i = \hat{\delta x}_i^- + K_i (\delta y_i - H_i \hat{\delta x}_i^-),$$

$$(26) \quad P_i = P_i^- - K_i H_i P_i^-,$$

$$(27) \quad \hat{\delta x}_{i+1} = \Phi_i \hat{\delta x}_i,$$

$$(28) \quad P_{i+1}^- = \Phi_i P_i \Phi_i^T + Q_i, \quad i = 0, 1, 2, \dots$$

Q_i and R_i are the covariances of u_i and v_i , respectively. $\hat{\delta x}_i^-$ is the estimate of δx_i based on the set $\{y_0, \dots, y_{i-1}\}$, while $\hat{\delta x}_i$ is the estimate of δx_i based on the set $\{y_0, \dots, y_i\}$. P_i^- and P_i are the respective filter covariances of $\hat{\delta x}_i^-$ and $\hat{\delta x}_i$. K_i is the filter gain matrix.

The estimates $\hat{\delta x}$ and the filter covariances P are computed sequentially. At each step $\hat{\delta x}_i^-$, based on measurements $\{y_0, \dots, y_{i-1}\}$, is updated using y_i to obtain $\hat{\delta x}_i$, and P_i^- is updated to obtain P_i . Then $\hat{\delta x}_i$ is extrapolated to obtain $\hat{\delta x}_{i+1}^-$, based on measurements $\{y_0, \dots, y_i\}$, and P_i is extrapolated to obtain P_{i+1}^- .

The smoother algorithm, to be considered here, has two forms, one of which can only be used in the special case in which there is no state noise. In either case the smoother is employed after the complete measurement set $\{y_0, \dots, y_N\}$ has been filtered.

The most general form of smoother, to be used here, is the fixed interval smoother, and the algorithm for this form is given by

$$(29) \quad A_i = P_i \Phi_i^T (P_{i+1}^-)^{-1},$$

$$(30) \quad \hat{x}_{i|N} = \hat{x}_i + A_i (\hat{x}_{i+1|N} - \hat{x}_{i+1}^-),$$

$$(31) \quad P_{i|N} = P_i - A_i (P_{i+1}^- - P_{i+1|N}) A_i^T, \quad i = N-1, \dots, 0.$$

A_i is the smoother gain matrix, $\hat{x}_{i|N}$ is the estimate of x_i based on the complete measurement set $\{y_0, \dots, y_N\}$, and $P_{i|N}$ is the smoother covariance of $\hat{x}_{i|N}$.

The fixed interval smoother is initialized by the final values of the filter. That is

$$\hat{x}_{N|N} = \hat{x}_N,$$

and

$$P_{N|N} = P_N.$$

The smoother quantities are computed sequentially but in reverse order to the filter. At each step $\hat{x}_i|N$ is obtained from $\hat{x}_{i+1}|N$ and filter outputs. Similarly $P_i|N$ is obtained from $P_{i+1}|N$ and filter outputs.

The filter estimates of the whole state, required in the smoother, are obtained during the filter operation by simply computing and storing the quantities

$$(32) \quad \hat{x}_i^- = \delta \hat{x}_i^- + \tilde{x}_i,$$

and

$$(33) \quad \hat{x}_i = \delta \hat{x}_i + \tilde{x}_i, \quad i = 0, 1, \dots, N.$$

It should be noted that, unlike the filter covariances, the smoother covariances are not required to obtain the smoothed estimates of the state vector. However, the smoother covariances are required for the error analysis. In fact, if all the conditions stated in Appendix A were satisfied, the smoother covariance would be equal to the covariance of estimation error.

The second form of the smoother which can be used only when state noise is zero is the retrograde integration smoother. This method of solution consists of simply solving equation (1) (where $w(t) = 0$) by reverse or retrograde integration from t_N to t_0 using the final value estimate from the filter to initialize the integrator. The smoother covariance is similarly obtained by retrograde integration using the final value of the filter covariance for initialization.

The retrograde integration algorithm is given by

$$(34) \quad \dot{\hat{x}}(t|t_N) = f[\hat{x}(t|t_N), t], \hat{x}(t_N|t_N) = \hat{x}_N,$$

$$(35) \quad \dot{P}(t|t_N) = F(t|t_N)P(t|t_N) + P(t|t_N)F^T(t|t_N), P(t_N|t_N) = P_N, t_0 \leq t \leq t_N,$$

where

$$(36) \quad \hat{x}_N = \delta x_N + \tilde{x}(t_N),$$

$$(37) \quad F(t|t_N) = \left(\frac{\partial f}{\partial x^T} \right)_{\tilde{x}(t|t_N), t}$$

$\hat{x}(t|t_N)$ is the whole value state estimate based on $\{y_0, \dots, y_N\}$, and $P(t|t_N)$ is the smoother covariance of $\hat{x}(t|t_N)$.

A flow diagram for the two stage estimator illustrating the basic functions of the filter and both smoother types is illustrated in Figure 5.1. The diagram is somewhat simplified, but serves the purpose of demonstrating the essential structure of the estimator.

There are two major filter functions, update and extrapolate, which are prepared by measurement processing and integration functions, respectively. Notice also, that in the filter stage the variation estimate must be reset whenever nominal reset occurs (Cf. equation (21)).

5.2 Error Analysis

The estimation equations which have been developed in the preceding subsection deal exclusively with the set of estimated states. In the linearization procedure, it was tacitly assumed that the variation (i.e., true - nominal) was zero for each constrained state. Since this subsection deals with all error sources, including constrained states, nonzero variations of these states will be considered.

Let x^E denote the subvector of estimated states, and let x^C denote the subvector of constrained* states to be included in the error analysis.

*It is assumed that each of the constrained states is a static state.

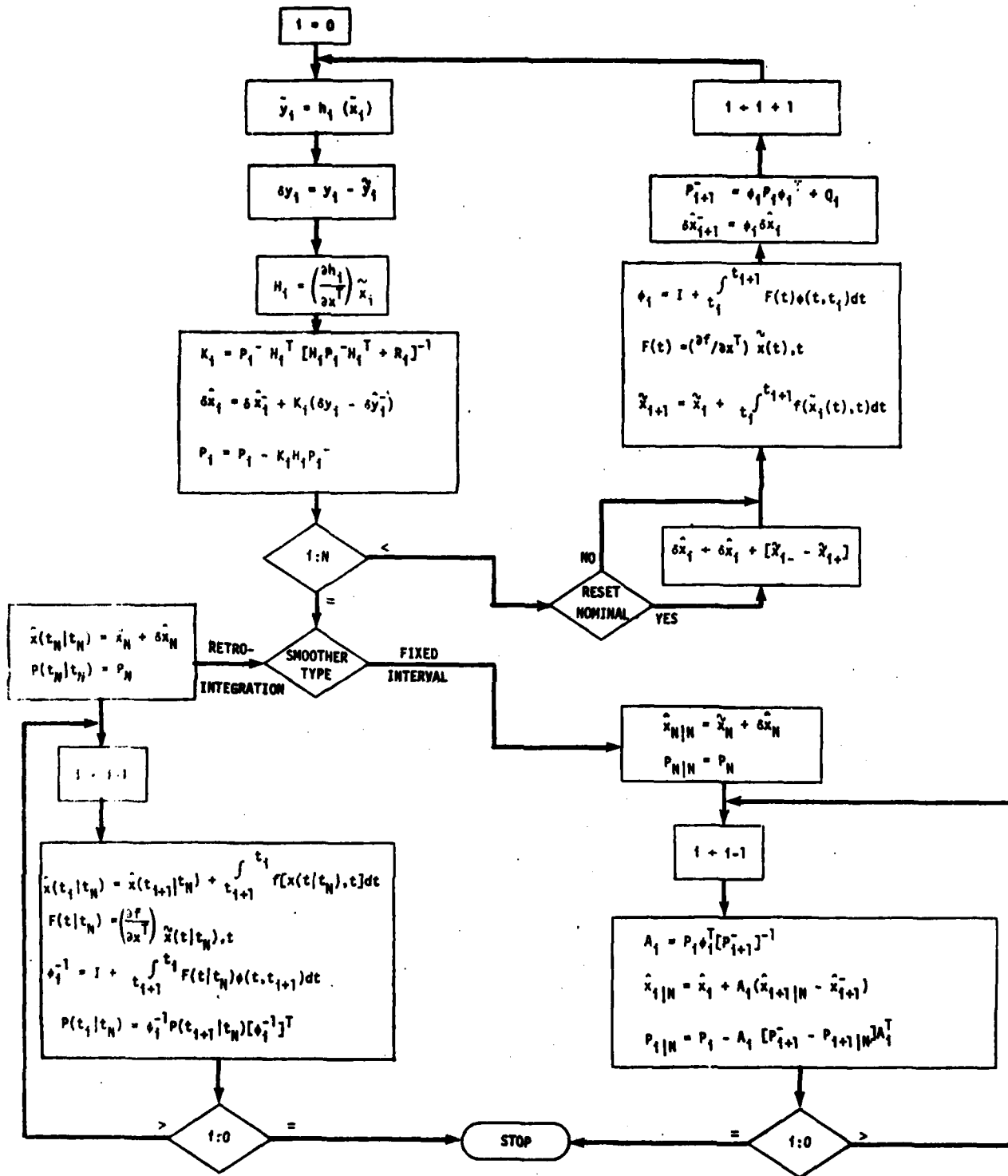


FIGURE 5.1 ESTIMATOR FLOW DIAGRAM

Then repetition of the linearization procedure used in the preceding subsection leads to analogous equations which include the variations in the constrained states. The analogues to equations (12) and (13) are

$$(38) \quad \dot{\delta x}^E(t) = F(t)\delta x^E(t) + G(t)\delta x_0^C + w(t),$$

$$(39) \quad \delta y(t_i) = H(t_i)\delta x^E(t_i) + J(t_i)\delta x_0^C + v(t_i),$$

where

$$(40) \quad G(t) \equiv \left(\frac{\partial f}{\partial (x_0^C)^T} \right)_{\tilde{x}(t), t},$$

and

$$(41) \quad J(t_i) \equiv \left(\frac{\partial h}{\partial (x_0^C)^T} \right)_{\tilde{x}(t_i), t_i}.$$

The analogues of equations (22) and (23) are

$$(42) \quad \delta x_{i+1}^E = \Phi_i \delta x_i^E + \Theta_i \delta x_0^C + u_i,$$

$$(43) \quad \delta y_i = H_i \delta x_i^E + J_i \delta x_0^C + v_i, \quad i = 0, 1, 2, \dots,$$

where

$$(44) \quad \Theta_i = \int_{t_i}^{t_{i+1}} \Phi(t_{i+1}, r) G(r) dr.$$

Now equations (42) and (43) can be used to develop the error propagation equations for the filter and smoother. The filter errors are defined by

$$(45) \quad e_i^- \equiv \hat{\delta x}_i^{E^-} - \delta x_i^E,$$

$$(46) \quad e_i \equiv \hat{\delta x}_i^E - \delta x_i^E, \quad i = 0, 1, 2, \dots, N.$$

The fixed interval smoother error is defined by

$$(47) \quad e_{i|N} = \hat{x}_{i|N}^E - x_i^E, \quad i = 0, 1, 2, \dots, N,$$

and the retrograde smoother error is defined by

$$(48) \quad e(t|t_N) = \hat{x}^E(t|t_N) - x^E(t), \quad t_0 \leq t \leq t_N.$$

From the error equations and the filter algorithm, it follows that

$$(49) \quad e_i = (I - K_i H_i) e_i^- + K_i J_i \delta x_0^C + K_i v_i,$$

$$(50) \quad e_{i+1}^- = \phi_i e_i - \theta_i \delta x_0^C - u_i.$$

Similarly for the two smoother types

$$(51) \quad e_{i|N} = e_i + A_i [e_{i+1|N} - e_{i+1}^-],$$

and

$$(52) \quad \dot{e}(t|t_N) = F(t|t_N) e(t|t_N) - G(t|t_N) \delta x_0^C,$$

where

$$F(t|t_N) = \left(\frac{\partial f}{\partial (x^E)^T} \right) \hat{x}(t|t_N), \quad t$$

and

$$G(t|t_N) = \left(\frac{\partial f}{\partial (x_0^C)^T} \right) \hat{x}(t|t_N), \quad t$$

The estimation error at any point in the estimation process is a linear combination of initialization errors in the estimated and constrained states and errors due to measurement and state noise. Define

$$(53) \quad D = \frac{\partial e}{\partial (x_0^E)^T}$$

$$(54) \quad E = \frac{\partial e}{\partial (x_0^C)^T}$$

Then the initialization error propagation equations are

$$(55) \quad D_i = (I - K_i H_i) D_i^-, D_0^- = I ,$$

$$(56) \quad E_i = (I - K_i H_i) E_i^- + K_i J_i, E_0^- = 0 ,$$

$$(57) \quad D_{i+1}^- = \Phi_i D_i ,$$

$$(58) \quad E_{i+1}^- = \Phi_i E_i - \Theta_i ,$$

$$(59) \quad D_{i|N} = D_i + A_i [D_{i+1|N} - D_{i+1}^-], D_{N|N} = D_N$$

$$(60) \quad E_{i|N} = E_i + A_i [E_{i+1|N} - E_{i+1}^-], E_{N|N} = E_N$$

$$(61) \quad \dot{D}(t|t_N) = F(t|t_N) D(t|t_N), D(t_N|t_N) = D_N ,$$

$$(62) \quad \dot{E}(t|t_N) = F(t|t_N) E(t|t_N) - G(t|t_N), E(t_N|t_N) = E_N .$$

A procedure for the error analysis using the above error propagation equations together with the filter and smoother error covariance equations will now be developed. The error analysis will provide the mean and covariance of the estimation error based on an error budget.

The error budget specifies the mean and covariance of initialization error:

$$\begin{bmatrix} b_0^E \\ b_0^C \end{bmatrix} , \quad \begin{bmatrix} U_0^{EE} & U_0^{EC} \\ U_0^{CE} & U_0^{CC} \end{bmatrix} .$$

It also specifies the state and measurement noise process covariances Q_i , $i = 0, 1, 2, \dots$, and R_i , $i = 0, 1, 2, \dots$, and it is assumed that these noise processes are sequentially uncorrelated and mutually independent with each other and the initial value of the state.

The mean or bias estimation error is given by

$$(63) \quad b_{i|N}^E = D_{i|N} b_0^E + E_{i|N} b_0^C,$$

in the fixed interval smoother, and by

$$(64) \quad b(t|t_N) = D(t|t_N) b_0^E + E(t|t_N) b_0^C$$

in the retrograde integration smoother.

The covariance of the estimation error due to initialization error is given by

$$(65) \quad U_{i|N} = [D_{i|N} \quad E_{i|N}] \begin{bmatrix} U_0^{EE} & U_0^{EC} \\ U_0^{CE} & U_0^{CC} \end{bmatrix} \begin{bmatrix} D_{i|N}^T \\ E_{i|N}^T \end{bmatrix}$$

in the fixed interval smoother, and by

$$(66) \quad U(t|t_N) = [D(t|t_N) \quad E(t|t_N)] \begin{bmatrix} U_0^{EE} & U_0^{EC} \\ U_0^{CE} & U_0^{CC} \end{bmatrix} \begin{bmatrix} D^T(t|t_N) \\ E^T(t|t_N) \end{bmatrix}$$

in the retrograde integration smoother.

The covariance of estimation error due to the noise processes is most easily computed by an indirect method using the smoother covariance. The method is valid only if the error budget values of the noise process covariances, Q_i , $i = 0, 1, 2, \dots$, and R_i , $i = 0, 1, 2, \dots$, are used in the filter algorithm (Cf. equations (24) and (28)). Under this assumption, the estimation error covariance due to noise processes is given by

$$(67) \quad V_{i|N} = P_{i|N} - D_{i|N} P_0^{-D^T} |_{i|N},$$

or

$$(68) \quad V(t|t_N) = P(t|t_N) - D(t|t_N) P_0^{-D^T}(t|t_N),$$

where P_0^- is the initial covariance used in the filter.

In the special case in which state noise is zero and the retrograde integration smoother is used, an alternate direct method of calculating the estimation error covariance due to measurement noise may be used. This method is valid regardless of whether the error budget values of measurement noise are used in the filter. The first step is to compute V_i and V_i^- by

$$(69) \quad V_i = (I - K_i H_i) V_i^- (I - K_i H_i)^T + K_i R_i K_i^T, \quad V_0^- = 0,$$

$$(70) \quad V_{i+1}^- = \phi_i V_i \phi_i^T, \quad i = 0, 1, 2, \dots$$

Then the estimation error covariance due to measurement noise is given by

$$(71) \quad \dot{V}(t|t_N) = F(t|t_N) V(t|t_N) + V(t|t_N) F^T(t|t_N), \quad V(t_N|t_N) = V_N.$$

Finally, the total estimation error covariance due to combined initialization and noise process error is given by either

$$(72) \quad W_i|N = U_i|N + V_i|N, \quad i = 0, 1, 2, \dots,$$

or

$$(73) \quad W(t|t_N) = U(t|t_N) + V(t|t_N), \quad t_0 \leq t \leq t_N.$$

A flow diagram for the computation of the error propagation quantities D , E , and V is presented in Figure 5.2. The alternative calculation of V is shown as an option to be exercised only when state noise is zero and the measurement noise error budget values differ from those used in the filter algorithm.

The sequential nature of the error propagation equations is in one to one correspondence with the filter and smoother equations. Consequently, the error propagation equations could be easily computed in parallel with the estimation equations. However, because of the iterative process used to minimize error due to nonlinearity, it is most efficient to defer the

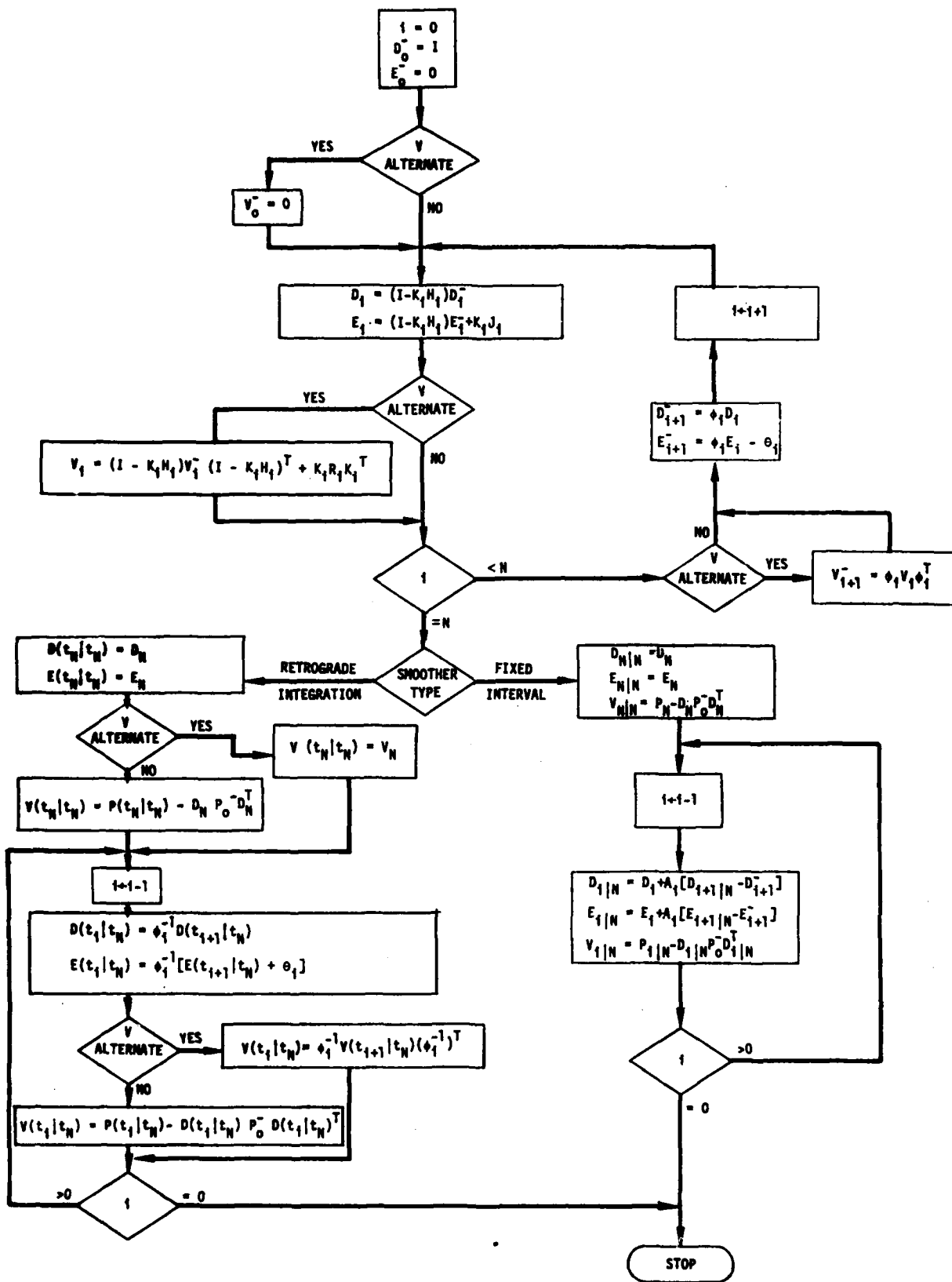


FIGURE 5.2 ERROR PROPAGATION FLOW DIAGRAM

calculation of the error propagation equations until the estimation process converges.

Once the error propagation quantities have been calculated and the initialization error budget has been specified, the error analysis is completed using equations (63) or (64) to compute the estimation bias error, and (65) and (72) or (66) and (73) to compute the covariance of estimation error.

6.0 COMPUTATIONAL CONSIDERATIONS

In order to effectively select and implement an algorithm in a computer program, due consideration of several factors is required. These factors include the numerical precision of the computer and the accuracy, stability, and efficiency of the algorithm.

The numerical precision of a computer is determined by the number of digits allocated to the mantissae of numbers represented in the machine. The error which occurs because of limited numerical precision is called roundoff error.

All other computational errors are inherently due to the algorithm itself. The most common of these is truncation error. Truncation error results when higher order terms in a Taylor series expansion are neglected.

The stability of the algorithm is determined by how roundoff and truncation errors propagate. If these errors propagate in an unbounded or oscillatory manner the algorithm is unstable.

Efficiency of an algorithm is a relative concept based on execution time and storage requirements. Gross estimates of execution time and storage requirements must be considered in selecting candidate algorithms, and relative efficiency can be used as a tradeoff basis for otherwise comparable algorithms.

In the remainder of this section, the computer implementation necessary to satisfy the functional requirements, developed in Section 5.0, for estimation and error analysis will be discussed based on consideration of the computational factors mentioned above.

6.1 Scalar Measurement Update

The update equations of the filter algorithm developed in Section 5.0 are given by

$$K = P^{-1}H^T[HP^{-1}H^T + R]^{-1},$$

$$\hat{\delta x} = \hat{\delta x}^- + K(\delta y - H\hat{\delta x}^-),$$

$$P = P^- - KHP^-$$

where the time index i has been dropped. The function of these equations is to process the measurement variation δy and thereby update $\hat{\delta x}$ and P^- to obtain $\hat{\delta x}$ and P respectively. As written these equations are valid for a measurement vector of arbitrary dimension. For example, if δy is an m -vector and P is $n \times n$, H is $m \times n$, R is $m \times m$, and K is $n \times m$.

The update equations are based in part on the assumption that the measurement variation is given by

$$\delta y = H\delta x + v,$$

where the covariance of v is the matrix R .

In the particular case where R is diagonal*, the vector update procedure can be replaced with an entirely equivalent scalar update procedure which is more efficient.

To illustrate the scalar update procedure, assume

$$\delta y = \begin{bmatrix} \delta y_1 \\ \delta y_2 \\ \vdots \\ \delta y_m \end{bmatrix}, \quad H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_m \end{bmatrix}, \quad \text{and } R = \begin{bmatrix} R_{11} & & & \\ & R_{22} & & \\ & & \ddots & \\ & & & R_{mm} \end{bmatrix}.$$

Thus for each $j = 1, \dots, m$, δy_j is the j -th scalar element of δy , $H_{j \cdot}$ is the j -th row of H , and R_{jj} is the j -th diagonal element of R . Now the scalar update procedure consists simply of applying the update equations sequentially to the elements of δy as illustrated in Figure 6.1.

*If R is, in fact, the covariance matrix of the measurement noise vector v , then R is diagonal if and only if the elements of v are mutually uncorrelated.

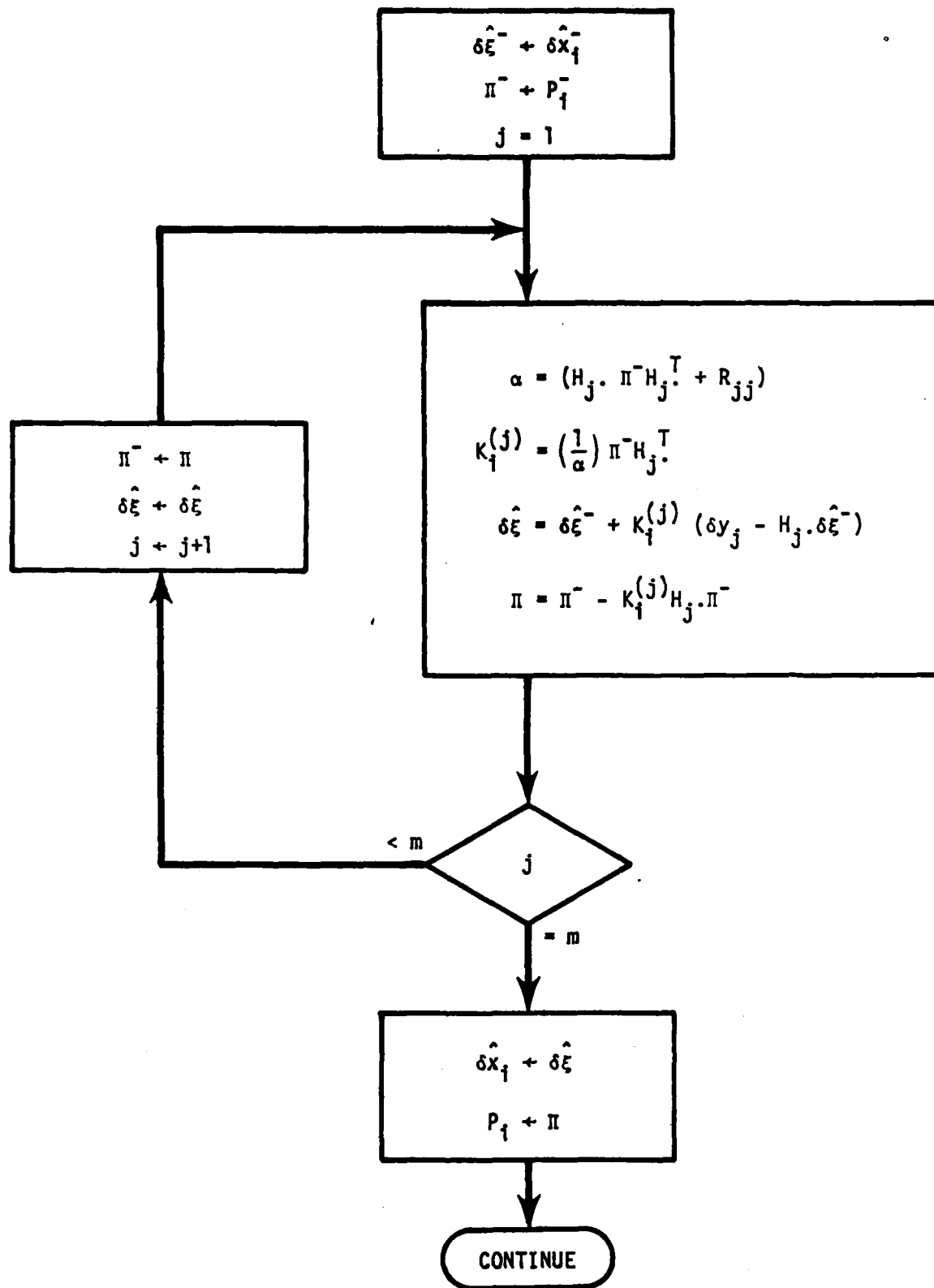


FIGURE 6.1 SCALAR MEASUREMENT UPDATE

In the figure, dummy variables $\hat{\delta\xi}$ and Π were used for immediate results which need not be saved. The final values of these variables yield the same values of δx_i and P_i that would have been obtained by the vector update procedure.

The gain matrices $K_i^{(j)}$, $j = 1, \dots, m$, must all be saved for use in the error propagation equations. Furthermore, equations (55), (56), and (69) in Section 5.0 must also be processed sequentially for each fixed i at which the scalar update method is employed. The reason for this is that

$$K_i \neq [K_i^{(1)} \quad \vdots \quad \dots \quad \vdots \quad K_i^{(m)}] .$$

The sequential error propagation equations are illustrated in Figure 6.2. Dummy variables are once again used for intermediate results which need not be saved.

6.2 Square Root Filter

The standard filter algorithm which was developed in Section 5.0 includes sequential calculation of the error covariance matrix, P . A necessary condition for an $n \times n$ matrix P to be a covariance matrix is that it be nonnegative, i.e., $P^T = P$ and

$$a^T P a \geq 0$$

for all n -vectors a . However, careless computation of P in the standard filter can result in violation of the nonnegativity condition because of roundoff error. When this occurs, filter instability can result.

There are a number of methods typically employed to preserve the nonnegativity of P in filter implementations. However, some of these methods increase execution time by as much as 100%. Worse, they sometimes fail to preserve nonnegativity, especially if P becomes ill-conditioned, i.e., nearly singular. An alternative to employing any of these methods is provided by the square root filter.

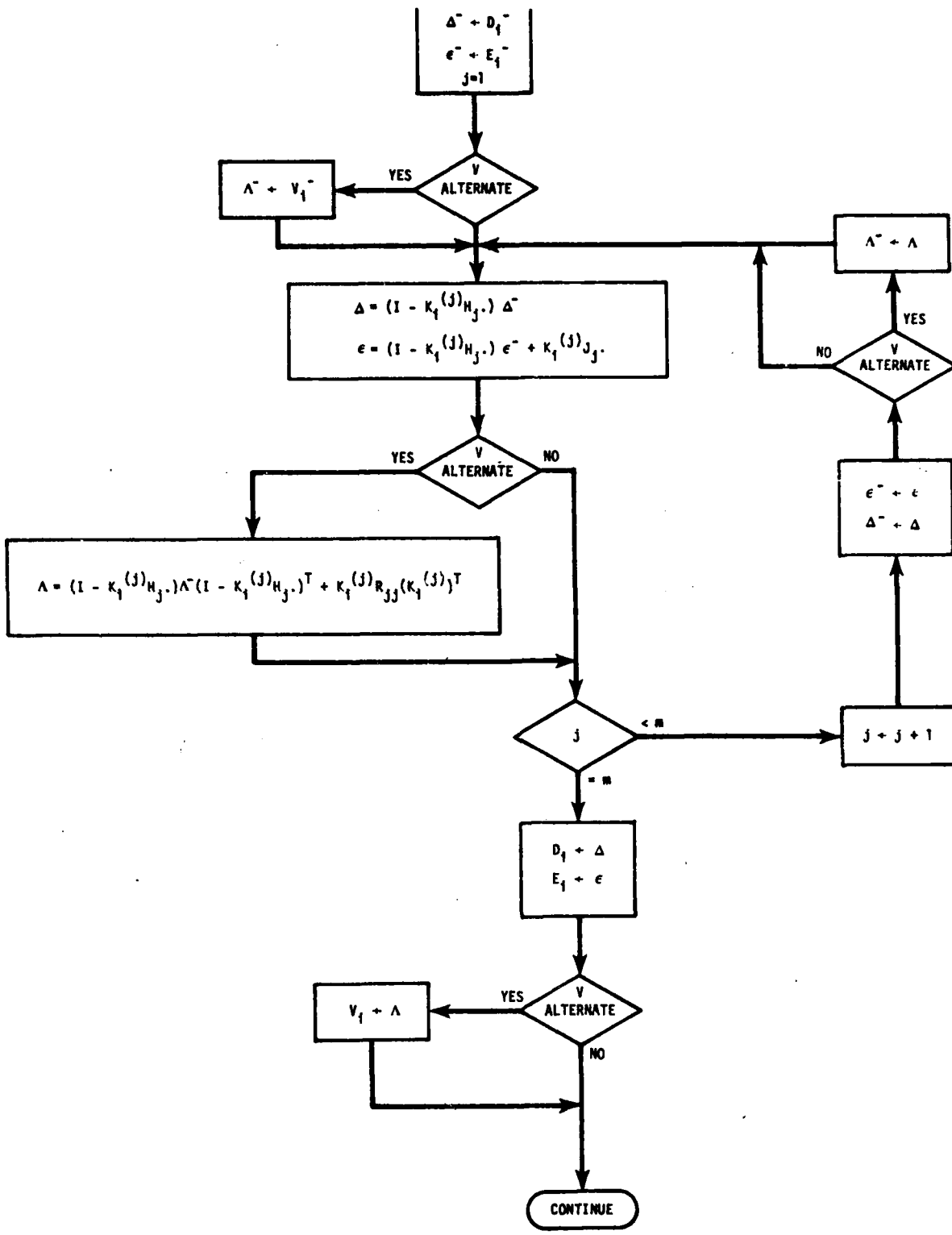


FIGURE 6.2 ERROR PROPAGATION MODIFIED FOR SCALAR MEASUREMENT UPDATE.

The square root filter, which can be realized by any of several algorithms, is mathematically equivalent to the standard filter. The square root filter requires sequential calculation, not of P , but a square root of P , designated here by S .

An $n \times n$ matrix S which satisfies the condition

$$P = SS^T$$

is called a square root of P . If P is nonnegative, then a real valued square root of P exists, but it is not unique. Conversely, for any real valued $n \times n$ matrix S , the $n \times n$ matrix product SS^T is nonnegative, since

$$a^T(SS^T)a = (S^T a)^T(S^T a) \geq 0$$

for all n -vectors a . This property is important in square root filtering, because it guarantees the nonnegativity of the filter error covariance matrix P .

Square root filters have two distinct advantages. First they are much less sensitive to roundoff error than standard filters. In fact empirical computer studies have demonstrated in some instances that square root filters have single precision accuracy that can be achieved with a standard filter only by using double precision arithmetic. Second, and most important, filter instability due to violation of the nonnegativity condition cannot occur, since the error covariance matrix P (which need not be computed but which is implicitly defined by the square root matrix S) is guaranteed to be nonnegative.

Of the available square root filter algorithms, the one selected for application in TRAM is due to Carlson. This algorithm, which is given in detail in Appendix A, is very efficient. The required execution time is comparable to the most efficient standard filter algorithm.

The Carlson algorithm has the distinctive feature that the square root matrix S is maintained in triangular form. S may be either upper triangular or lower triangular at the option of the user. Recall that a matrix is upper (lower)

triangular if all elements below (above) the main diagonal are zero. The importance of triangularity is that it reduces the computational and storage requirements of S by nearly 50%.

The Carlson algorithm parallels the sequential operations of the standard filter, except that it propagates S instead of P.

The matrix S is initialized by extracting a triangular square root of P_0^- using the Cholesky decomposition algorithm given in Appendix D.

The update operation is constrained to scalar measurement processing as discussed in Section 6.1. If the measurement covariance matrix is nondiagonal, a linear transformation must be performed on the measurement vector in order to decorrelate the measurement noise components.

The extrapolation operation can partially or totally destroy the triangularity of S, and as a consequence a retriangularization procedure may be required.

To illustrate the use of Carlson algorithm consider the m-dimensional measurement vector variation at time t_i ,

$$\delta y_i = H_i \delta x_i + v_i$$

with measurement error covariance R_i . It is assumed that $\hat{\delta x}_i^-$ and S_i^- are available where S_i^- is a triangular square root of P_i^- .

If R_i is nondiagonal, the first step is decorrelate the measurement noise components. This is accomplished by applying Gaussian elimination with complete pivoting (Cf. Appendix D) to R_i . This procedure yields a lower triangular matrix L and an upper triangular matrix U such that

$$L \Pi R_i \Pi^T = U,$$

where Π is an invertible matrix constructed by row permutations on the identity matrix.

The matrix L has all ones on its main diagonal, and thus, by virtue of triangularity, L^{-1} exists.

Now consider the transformed measurement equation

$$\delta y_i^i = H_i^i \delta x_i + v_i^i$$

where

$$\delta y_i^i = L \Pi \delta y_i$$

$$H_i^i = L \Pi H_i$$

$$v_i^i = L \Pi v_i$$

and the covariance of v_i^i is simply

$$R_i^i = L \Pi R_i \Pi^T L^T = U L^T.$$

Using the fact that $(L \Pi)^{-1}$ exists, it is easily shown that the standard filter update with δy_i^i , H_i^i , R_i^i is entirely equivalent to update with δy_i , H_i , R_i . But it is readily seen that R_i^i is diagonal, since it is both symmetric and upper triangular. (R_i^i is symmetric since it is a covariance matrix, and it is upper triangular since it is the product of two upper triangular matrices, U and L^T .) In fact, element by element, the main diagonal of R_i^i is equal to the main diagonal of U . That is,

$$R_i^i = \begin{bmatrix} U_{11} & & & \\ & U_{22} & & \\ & & \bigcirc & \\ & & & \ddots \\ & & & & U_{mm} \end{bmatrix}.$$

As a consequence of R_i^i being diagonal, the transformed measurements can be processed by the scalar measurement update procedure of Section 6.1.

The next step is to process the individual scalar components of the transformed measurement vector using the Carlson update algorithm. Suppressing both the subscript i and the prime notation, let the transformed measurement quantities be given by

$$\delta y = \begin{bmatrix} \delta y_1 \\ \delta y_2 \\ \vdots \\ \delta y_m \end{bmatrix}, \quad H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_m \end{bmatrix}, \quad R = \begin{bmatrix} R_{11} & & & \\ & R_{22} & & \\ & & \circ & \\ & & \cdot & \\ \circ & & & R_{mm} \end{bmatrix}$$

The scalar update procedure consists of applying the Carlson update equations sequentially to the elements of δy . The combined decorrelation and Carlson update procedures are illustrated in Figure 6.3.

In the figure dummy variables have been used for intermediate results which need not be saved. The starred box contains the key calculations of the Carlson update procedure, and the detailed algorithm for the functions in this box is provided in Appendix A. Notice that the matrix J_i of measurement partial derivatives with respect to constrained states, required for error propagation (Cf. (43), (56) of Section 5.0), is also subject to the decorrelation transformation L . The quantities which must be saved for smoothing and error propagation functions include H , J , $K_i^{(j)}$, $j = 1, \dots, m$, and R in addition to $\hat{\delta x}_i^-$, $\hat{\delta x}_i^+$, S_i^- , and S_i^+ .

The final step is the extrapolation operation. The nominal state and the state variation are extrapolated in the same way in the square root filter as in the standard filter. To extrapolate S_i^- , a triangular matrix S_{i+1}^- must be found such that

$$P_{i+1}^- = S_{i+1}^- (S_{i+1}^-)^T$$

where (Cf. (28), Section 5.0)

$$P_{i+1}^- = \phi_i P_i \phi_i^T + Q_i,$$

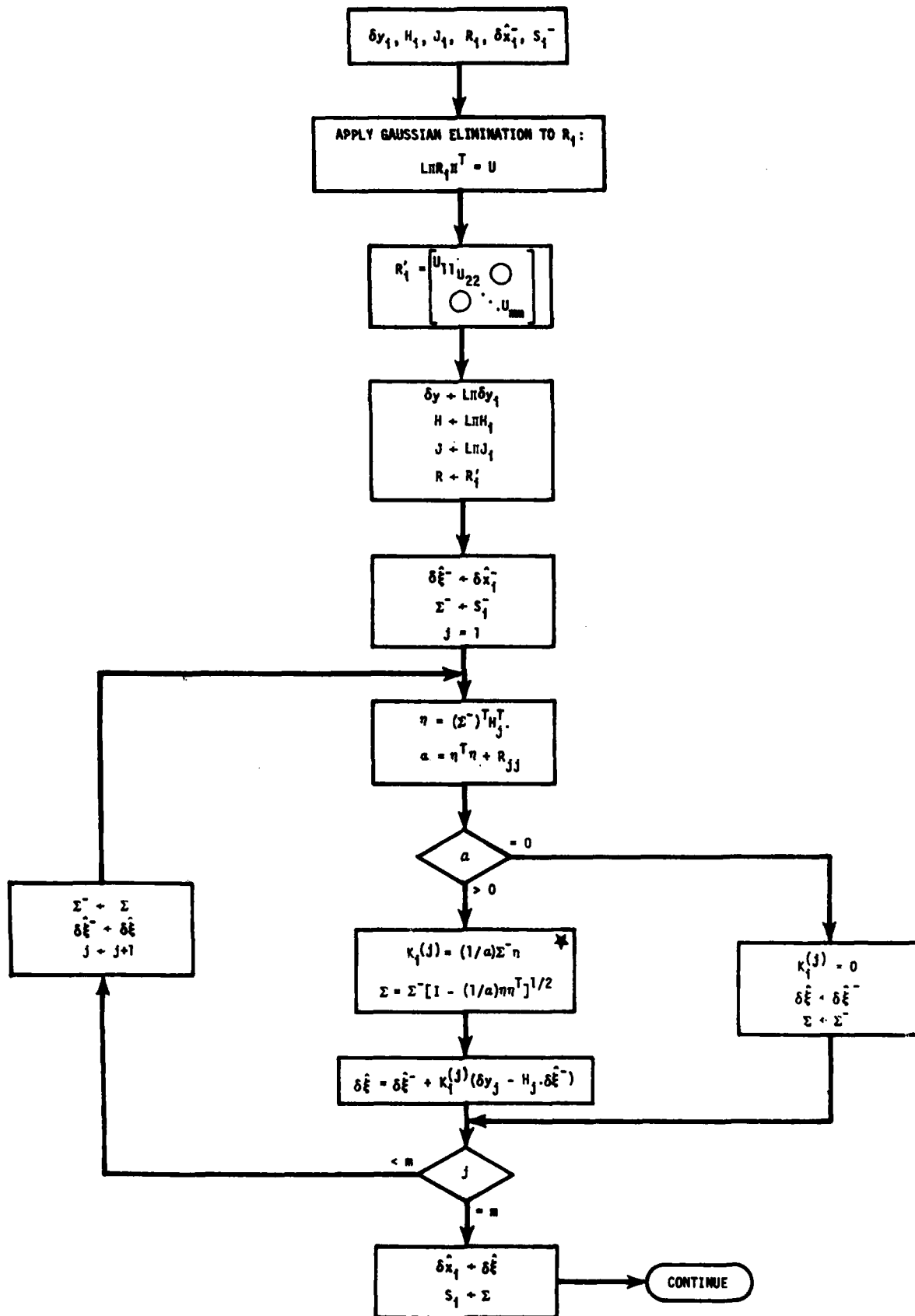


FIGURE 6.3 SQUARE ROOT FILTER UPDATE

and

$$P_i = S_i S_i^T.$$

This could, of course, be accomplished by computing P_{i+1}^- directly and applying the Cholesky decomposition to obtain S_{i+1}^- . But this would defeat the purpose of the square root filter which is to avoid sequential computation of P and thereby gain certain advantages with respect to computational efficiency, numerical precision, and filter stability.

The indirect calculation of S_{i+1}^- is performed in the following manner. Let Γ_i be a square root of Q_i , and observe that

$$[\phi_i S_i \ ; \ \Gamma_i] [\phi_i S_i \ ; \ \Gamma_i]^T = \phi_i P_i \phi_i^T + Q_i = P_{i+1}^-.$$

Thus the augmented matrix $[\phi_i S_i \ ; \ \Gamma_i]$ satisfies one property required of S_{i+1}^- but it is not square.

Now let T be orthogonal matrix, i.e., $T^{-1} = T^T$, of the proper dimension to allow pre-multiplication by $[\phi_i S_i \ ; \ \Gamma_i]$ and observe that

$$\{[\phi_i S_i \ ; \ \Gamma_i]T\} \{[\phi_i S_i \ ; \ \Gamma_i]T\}^T = [\phi_i S_i \ ; \ \Gamma_i] [\phi_i S_i \ ; \ \Gamma_i]^T = P_{i+1}^-.$$

Thus if T is selected such that

$$[\phi_i S_i \ ; \ \Gamma_i]T = [S' \ ; \ 0],$$

then S' is a square root of P_{i+1}^- .

By applying the modified Gram-Schmidt (MGS) process (Cf. Appendix D) to the rows of $[\phi_i S_i \ ; \ \Gamma_i]$, augmented by the rows of $\begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$, an orthogonal matrix T can always be constructed such that

$$[\phi_i S_i \ ; \ \Gamma_i]T = [S' \ ; \ 0],$$

where S' is triangular. The order in which the rows of $[\phi_i S_i ; \Gamma_i]$ are processed to obtain T determines whether S' is upper triangular or lower triangular.

In the particular case where $Q_i = 0$, it is clear that the matrix $\phi_i S_i$ is itself a square root of P_{i+1}^- . However, depending on the structure of ϕ_i , the product $\phi_i S_i$ is not necessarily triangular. Consequently, even when $Q_i = 0$, the MGS process, applied to the rows of $\phi_i S_i$ augmented by the rows of I if necessary, may still be required to obtain a triangular square root of P_{i+1}^- . In this case the procedure is appropriately referred to as retriangularization.

The efficiency of the square root covariance extrapolation is greatly influenced by the structure of ϕ_i and Q_i which in turn is dependent on the ordering of the elements within the estimated state vector.

To illustrate efficient extrapolation, assume that the upper triangular form of S has been selected and that x , ϕ , and Q can be partitioned as follows:

$$x = \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix}, \quad \phi = \begin{bmatrix} \phi_{aa} & \phi_{ab} & \phi_{ac} \\ 0 & \phi_{bb} & \phi_{bc} \\ 0 & 0 & I \end{bmatrix}, \quad Q = \begin{bmatrix} Q_{aa} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

This form, which can always be achieved by proper ordering of the elements of x , has grouped all dynamic states which are driven by noise in x_a , all dynamic states which are not driven by noise in x_b , and all static states in x_c . Denote the respective dimensions of x_a , x_b , x_c by n_a , n_b , n_c .

The corresponding partitioned form of S is given by

$$S = \begin{bmatrix} S_{aa} & S_{ab} & S_{ac} \\ 0 & S_{bb} & S_{bc} \\ 0 & 0 & S_{cc} \end{bmatrix},$$

where S_{aa} , S_{bb} , and S_{cc} are each upper triangular.

Now define

$$\Sigma = \Phi S .$$

Then, in partitioned form

$$\Sigma = \begin{bmatrix} \Sigma_{aa} & \Sigma_{ab} & \Sigma_{ac} \\ 0 & \Sigma_{bb} & \Sigma_{bc} \\ 0 & 0 & \Sigma_{cc} \end{bmatrix}$$

where Σ_{aa} and Σ_{bb} are not necessarily triangular. However, Σ_{cc} is upper triangular; in fact $\Sigma_{cc} = S_{cc}$.

Using the above partitioned forms it follows that

$$[\Phi S ; \Gamma] = \left[\begin{array}{ccc|ccc} \Sigma_{aa} & \Sigma_{ab} & \Sigma_{ac} & \Gamma_{aa} & 0 & 0 \\ 0 & \Sigma_{bb} & \Sigma_{bc} & 0 & 0 & 0 \\ 0 & 0 & \Sigma_{cc} & 0 & 0 & 0 \end{array} \right] ,$$

where Γ_{aa} is a square root of Q_{aa} .

Now an orthogonal matrix T , such that

$$[\Sigma ; \Gamma] T = [\Sigma^- ; 0] ,$$

where Σ^- is upper triangular, can be computed in partitioned form as follows:

1. Apply the MGS process to the rows of

$$\left[\begin{array}{c|c} I & 0 \\ 0 & I \\ \Sigma_{aa} & \Gamma_{aa} \end{array} \right] ,$$

proceeding in the order from bottom to top, skipping linearly dependent rows, and stopping when n_a orthonormal vectors have been obtained. Denote by $[T_{11}^T \ \vdots \ T_{41}^T]$

the matrix whose rows are given by the n_a orthonormal vectors placed, from bottom to top, in the order in which they are computed.

The matrix $[T_{11}^T \mid \mid T_{41}^T]$ is $n_a \times 2n_a$. Denote by $[T_{14}^T \mid \mid T_{44}^T]$ the $n_a \times 2n_a$ matrix whose rows are given, in any order, by the n_a orthonormal vectors which were not computed.

2. Apply the MGS process to the rows of

$$\begin{bmatrix} I \\ \Sigma_{bb} \end{bmatrix},$$

proceeding in the order from bottom to top, skipping linearly dependent rows, until the full set of n_b orthonormal vectors is obtained. (In the special case where $n_a = 0$, the process can be terminated when the rows of Σ_{bb} are exhausted.) Denote by T_{22}^T the matrix whose rows are given by the n_b orthonormal vectors placed, from bottom to top, in the order in which they are computed.

The complete matrix T is given by

$$T = \begin{bmatrix} T_{11} & 0 & 0 & T_{14} & 0 & 0 \\ 0 & T_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 & 0 \\ T_{41} & 0 & 0 & T_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & I \end{bmatrix},$$

but only T_{11} , T_{22} , T_{41} are actually computed. Now the resulting product

$$[\Sigma^- \mid 0] = [\Sigma \mid \Gamma] T,$$

with upper triangular Σ^- , is expressed in partitioned form by

$$\Sigma^- = \begin{bmatrix} \Sigma_{aa}^- & \Sigma_{ab}^- & \Sigma_{ac}^- \\ 0 & \Sigma_{bb}^- & \Sigma_{bc}^- \\ 0 & 0 & \Sigma_{cc}^- \end{bmatrix},$$

where

$$\Sigma_{aa}^- = \Sigma_{aa} T_{11} + \Gamma_{aa} T_{41}$$

$$\Sigma_{ab}^- = \Sigma_{ab} T_{22}$$

$$\Sigma_{bb}^- = \Sigma_{bb} T_{22}$$

$$\Sigma_{ac}^- = \Sigma_{ac}$$

$$\Sigma_{bc}^- = \Sigma_{bc}$$

$$\Sigma_{cc}^- = \Sigma_{cc}$$

The complete square root covariance extrapolation is illustrated in Figure 6.4. The inputs are ϕ_i , S_i , and Q_i , and the output is S_{i+1}^- . The partitioned form of computation is very efficient in those cases where $n_a + n_b \ll n_c$. Note in particular that the partitioned block of S represented by S_{cc} does not change during the extrapolation process.

6.3 Computation of Nominal State Vector and Transition Matrix

The mathematical development of Section 5.0 requires that solutions be obtained to several differential equations. The state vector differential equation is integrated to obtain the nominal state vector. Then the state vector differential equation is linearized about the nominal state and this linear equation is solved to obtain the transition matrix.

Let the state vector be permuted and partitioned into dynamic states x_a , static states x_b which affect the dynamic states, and static states x_c which do not

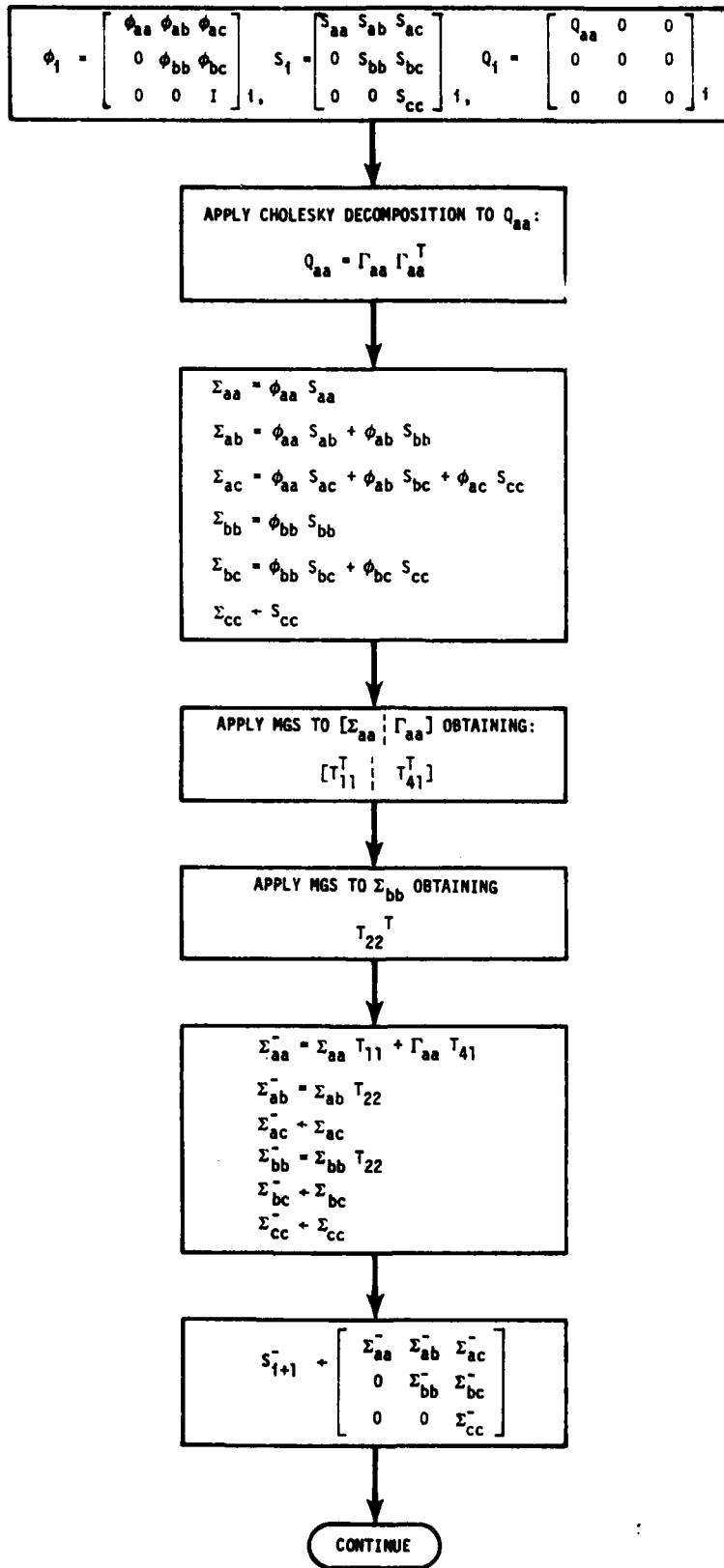


FIGURE 6.4 SQUARE ROOT COVARIANCE EXTRAPOLATION

affect the dynamic states. (Note this is not the same partition as used in Section 6.2.) Then the differential equation for the nominal state vector is given by

$$(1) \quad \begin{bmatrix} \dot{x}_a \\ \dot{x}_b \\ \dot{x}_c \end{bmatrix} = \begin{bmatrix} f_a[x_a(t), x_b, t] \\ 0 \\ 0 \end{bmatrix}, \quad t' \leq t \leq t'',$$

where the notation (\sim) has been suppressed.

Linearization of (1) about the nominal state vector leads to the differential equation for the fundamental matrix

$$(2) \quad \begin{bmatrix} \dot{\Psi}_{aa}(t) & \dot{\Psi}_{ab}(t) & \dot{\Psi}_{ac}(t) \\ \dot{\Psi}_{ba}(t) & \dot{\Psi}_{bb}(t) & \dot{\Psi}_{bc}(t) \\ \dot{\Psi}_{ca}(t) & \dot{\Psi}_{cb}(t) & \dot{\Psi}_{cc}(t) \end{bmatrix} = \begin{bmatrix} F_{aa}(t) & F_{ab}(t) & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Psi_{aa}(t) & \Psi_{ab}(t) & \Psi_{ac}(t) \\ \Psi_{ba}(t) & \Psi_{bb}(t) & \Psi_{bc}(t) \\ \Psi_{ca}(t) & \Psi_{cb}(t) & \Psi_{cc}(t) \end{bmatrix}, \quad t' \leq t \leq t'',$$

with initial condition

$$\begin{bmatrix} \Psi_{aa}(t') & \Psi_{ab}(t') & \Psi_{ac}(t') \\ \Psi_{ba}(t') & \Psi_{bb}(t') & \Psi_{bc}(t') \\ \Psi_{ca}(t') & \Psi_{cb}(t') & \Psi_{cc}(t') \end{bmatrix} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix},$$

where

$$F_{aa} \equiv \frac{\partial f_a}{\partial x_a^T},$$

$$F_{ab} \equiv \frac{\partial f_a}{\partial x_b^T}.$$

It is clear that $\psi_{ac}(t) = 0$, $\psi_{ba}(t) = 0$, $\psi_{bb}(t) = I$, $\psi_{bc}(t) = 0$, $\psi_{ca}(t) = 0$, $\psi_{cb}(t) = 0$, $\psi_{cc}(t) = I$ for all t in $[t', t'']$.

Consequently,

$$(3) \quad \dot{\psi}_{aa}(t) = F_{aa}(t)\psi_{aa}(t), \quad t' \leq t \leq t'' ,$$

and

$$(4) \quad \dot{\psi}_{ab}(t) = F_{aa}(t)\psi_{ab}(t) + F_{ab}(t), \quad t' \leq t \leq t'' .$$

Once the solution of (3) has been obtained, (4) can be solved by direct integration. Thus

$$(5) \quad \psi_{ab}(t) = \int_{t'}^t \psi_{aa}(t)\psi_{aa}^{-1}(r)F_{ab}(r)dr .$$

Now the transition matrix is given by

$$(6) \quad \begin{bmatrix} \phi_{aa}(t,s) & \phi_{ab}(t,s) & \phi_{ac}(t,s) \\ \phi_{ba}(t,s) & \phi_{bb}(t,s) & \phi_{bc}(t,s) \\ \phi_{ca}(t,s) & \phi_{cb}(t,s) & \phi_{cc}(t,s) \end{bmatrix} = \begin{bmatrix} \psi_{aa}(t) & \psi_{ab}(t) & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \psi_{aa}(s) & \psi_{ab}(s) & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}^{-1}$$

for all t, s in $[t', t'']$. From (6) it follows immediately that

$$(7) \quad \phi(t,s) = \begin{bmatrix} \phi_{aa}(t,s) & \phi_{ab}(t,s) & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} ,$$

where

$$(8) \quad \phi_{aa}(t,s) = \psi_{aa}(t)\psi_{aa}^{-1}(s)$$

and

$$(9) \quad \phi_{ab}(t,s) = \psi_{ab}(t) - \phi_{aa}(t,s)\psi_{ab}(s)$$

for all t, s in $[t', t'']$.

To summarize the procedure for constructing the nominal state vector and the transition matrix, denote the dimensions of x_a, x_b, x_c by n_a, n_b, n_c , respectively. The steps are:

1. Given the nominal state initial condition

$$\begin{bmatrix} x_a(t') \\ x_b(t') \\ x_c(t') \end{bmatrix} = \begin{bmatrix} x_a^0 \\ x_b^0 \\ x_c^0 \end{bmatrix},$$

solve the n_a - vector differential equation

$$\dot{x}_a(t) = f_a[x_a(t), x_b^0, t], \quad x_a(t') = x_a^0, \quad t' \leq t \leq t'',$$

and put $x_b(t) = x_b^0, x_c(t) = x_c^0$ for all t in $[t', t'']$.

2. Evaluate the partial derivative matrices

$$F_{aa} = \frac{\partial f_a}{\partial x_a^T}$$

$$F_{ab} = \frac{\partial f_a}{\partial x_b^T}$$

on $[t', t'']$ using the nominal state vector.

3. Solve the $n_a \times n_a$ - matrix differential equation

$$\dot{\Psi}_{aa}(t) = F_{aa}(t)\Psi_{aa}(t), \Psi_{aa}(t') = I, t' \leq t \leq t'' .$$

4. Compute $\Psi_{aa}^{-1}(t)$ on (t', t'') by direct matrix inversion or, alternatively, by solving the differential equation

$$\dot{\Psi}_{aa}^{-1}(t) = -\Psi_{aa}^{-1}(t)F_{aa}(t), \Psi_{aa}^{-1}(t') = I, t' \leq t \leq t'' .$$

5. Compute $\Phi_{aa}(t,s)$ for t, s in $[t', t'']$ by

$$\Phi_{aa}(t,s) = \Psi_{aa}(t)\Psi_{aa}^{-1}(s) .$$

6. Integrate the $n_a \times n_b$ matrix

$$[\Psi_{aa}^{-1}\Psi_{ab}](t) = \int_{t'}^t \Psi_{aa}^{-1}(r)F_{ab}(r)dr, t' \leq t \leq t'' .$$

7. Compute $\Phi_{ab}(t,s)$ for t, s in $[t', t'']$ by

$$\Phi_{ab}(t,s) = \Psi_{aa}(t) \{ [\Psi_{aa}^{-1}\Psi_{ab}](t) - [\Psi_{aa}^{-1}\Psi_{ab}](s) \} .$$

8. Put $\Phi_{ac}(t,s) = 0$, $\Phi_{ba}(t,s) = 0$, $\Phi_{bb}(t,s) = I$, $\Phi_{bc}(t,s) = 0$, $\Phi_{ca}(t,s) = 0$, $\Phi_{cb}(t,s) = 0$, $\Phi_{cc}(t,s) = I$ for t, s in $[t', t'']$.

In the above procedure, standard numerical integration methods can be used to obtain solutions to the differential equations at discrete points in the interval $[t', t'']$. Then, standard interpolation methods can be used to obtain solutions at arbitrary points in $[t', t'']$.

The results of a TRAM related computer study, in which a fourth order Runge-Kutta integration method was used and in which fifth order spline functions were used for trajectory interpolation and third order spline functions were used for interpolation of the fundamental matrix, is reported in [2]. The study, which was limited to an investigation of a freefall case, demonstrated that (with a step size of 5 seconds) the integration algorithms and interpolation methods were efficient and sufficiently accurate for TRAM applications in freefall.

6.4 Numerical Calculation of Partial Derivatives

Partial derivatives expressions are required in conjunction with the linearization of the state and measurement equations. If a function to be differentiated is represented by a closed form analytical expression, its derivatives can be calculated directly using standard formulas and the chain rule. If, on the other hand, the function in question cannot be expressed in closed form, then numerical calculation of some of its derivatives may be required.

In some cases where direct calculation of derivatives by formula is feasible, it may be undesirable for several reasons. First, in order to use direct methods, analytical expressions for each partial derivative must be developed, and then these expressions must be programmed. Often, this is a tedious and error prone process for analyst and programmer alike. Second, in many cases, the complexity of partial derivative expressions greatly exceeds that of the primitive function, and consequently the execution time required for derivative evaluations can greatly exceed that required for prime function evaluation.

Calculation of partial derivatives by numerical methods can be accomplished very simply using only primitive function evaluations. These functions must be programmed regardless of how their derivatives are calculated, and the additional analysis and programming required for numerical differentiation is trivial. However, caution must be exercised to prevent the introduction of excessive amounts of either roundoff or truncation error.

To illustrate the numerical differentiation procedure and bound its errors, let g be a function whose derivative at z_0 is approximated by

$$g'(z_0) \cong \frac{g(z_0 + \alpha) - g(z_0)}{\alpha},$$

for some $\alpha > 0$.

The roundoff error in this approximation is given by

$$r[g'(z_0)] = \frac{1}{\alpha} \{r[g(z_0 + \alpha)] - r[g(z_0)]\}$$

where $r(g)$ denotes the roundoff error in the evaluation of g .

If g' is continuous on $[z_0, z_0 + \alpha]$ and g'' is finite on $(z_0, z_0 + \alpha)$, the truncation error in the approximation of $g'(z_0)$ is, by Taylor's formula with remainder,

$$t[g'(z_0)] = \frac{\alpha}{2} g''(z_1)$$

for some z_1 in $(z_0, z_0 + \alpha)$.

Observe that roundoff error magnitude increases with decreasing α , while truncation error magnitude increases with increasing α . Consequently, judicious selection of α is essential in order that the total error in the approximation to $g'(z_0)$ not be excessive.

When g' is required in conjunction with a linearization procedure, a rationale for the selection of α can be developed.

Linearization of a functional relation

$$w = g(z)$$

about a point \tilde{z} implies that the approximation

$$w = g(\tilde{z}) + g'(\tilde{z})\delta z$$

is invoked, where $\delta z = z - \tilde{z}$. When this representation is used for computation there are, in addition to the error in $g'(\tilde{z})$, errors in w due to both roundoff and truncation. Excluding any error in $g'(\tilde{z})$, the roundoff error in w is given by

$$r[w] = r[g(\tilde{z})],$$

and, assuming g' is continuous and g'' is finite, the truncation error in w is given by

$$t[w] = \frac{1}{2} g''(\zeta)(\delta z)^2$$

for some ζ interior to the interval joining \tilde{z} and z .

Now let $z_0 = \tilde{z}$ and assume g'' is continuous on some closed interval $[a, b]$ containing z_0 , $z_0 + \alpha$, and z . Define

$$||r[g]|| = \max_{a \leq \zeta \leq b} |r[g(\zeta)]|$$

and

$$||g''|| = \max_{a \leq \zeta \leq b} |g''(\zeta)|$$

Then the total roundoff and truncation error in the representation of w is bounded by

$$|e(w)| \leq ||r[g]|| + \frac{1}{2} ||g''|| (\delta z)^2 + \left\{ \frac{2}{\alpha} ||r[g]|| + \frac{\alpha}{2} ||g''|| \right\} (\delta z),$$

where only the terms involving α are due to error in $g'(z_0)$.

Now suppose $\alpha = \delta z$. Then

$$|e(w)| \leq 3||r[g]|| + ||g''|| (\delta z)^2.$$

Thus, with $\alpha = \delta z$, the roundoff error bound does no worse than triple, and the truncation error bound no worse than double, when errors in g' are added to the errors which already exist in w . Consequently, if the algorithm for evaluation of g has negligible roundoff error, and if the linearization of g has negligible truncation error, then g' has negligible roundoff and truncation error when $\alpha = \delta z$.

The above discussion shows that when $g'(z_0)$ is approximated by

$$g'(z_0) \approx \frac{g(z_0 + \alpha) - g(z_0)}{\alpha} ,$$

the roundoff error is bounded by

$$|r[g']| \leq \frac{2}{\alpha} ||r[g]|| ,$$

and the truncation error is bounded by

$$|t[g']| \leq \frac{\alpha}{2} ||g''|| .$$

If $g'(z_0)$ is approximated by the symmetric formula

$$g'(z_0) \approx \frac{g(z_0 + \alpha/2) - g(z_0 - \alpha/2)}{\alpha} ,$$

it can be shown that the roundoff error is likewise bounded by

$$|r[g']| \leq \frac{2}{\alpha} ||r[g]|| ,$$

and the truncation error is bounded by

$$|t[g']| \leq \frac{\alpha^2}{24} ||g'''||$$

provided g''' is continuous on $[a, b]$ (this time selected to contain $z_0 + \alpha/2$ and $z_0 - \alpha/2$).

The roundoff error bound is the same for the two formulas for $g'(z_0)$, but the truncation error is one order higher in the symmetric formula. Consequently in comparison with the truncation error of the linearization process, the truncation error of the symmetric differentiation formula should be negligible.

Since $g(\tilde{z})$ is required in the linearization procedure, calculation of $g'(\tilde{z})$ by the unsymmetric formula requires only one additional evaluation of g . Calculation of $g'(\tilde{z})$ by the symmetric formula requires, two additional evaluations of g , but the additional computation is worth consideration because of the truncation error protection it affords.

The use of $\alpha = \delta z$ in the above discussion was not intended for any purpose other than error analysis. In practice it is desirable to select fixed values of α for each variable with respect to which numerical partial differentiation is to be performed. For each state variable, a value of α roughly equal to the magnitude of the anticipated estimation error in that variable should be selected.

6.5 Transit Time, Refraction, and Doppler Calculations

Let t' denote the time at which a signal leaves a target vehicle and let t denote the arrival time of the signal at a sensor which is tracking the vehicle. The time difference

$$\tau \equiv t - t'$$

is called the transit time from target to tracker.

The apparent range to the target at time t' is defined in terms of transit time by

$$R_A(t') = c\tau,$$

where c is the speed of light in a vacuum. The apparent range $R_A(t')$ and the true range $R(t')$ differ only because of refraction.

If the true range and the range refraction error are known functions, then both transit time τ and time of transmission t' can be computed for any time of reception t by applying Newton's method to the function

$$g(t') = R_A(t') - c(t - t').$$

This results in the algorithm

$$\tau_{i+1} = \tau_i + \frac{R_A(t - \tau_i) - c\tau_i}{c + \dot{R}_A(t - \tau_i)},$$

which is solved iteratively beginning with $\tau_0 = 0$ and ending when the condition

$$|\tau_{i+1} - \tau_i| < \eta$$

is satisfied for an arbitrary $\eta > 0$, or alternatively, when two successive values of τ differ by less than computer roundoff error.

In a two way doppler system in which the vehicle transponds (repeats an exact replica of the signal received) with negligible delay, the doppler frequency is defined as the difference between the transmitted and received frequencies at the tracker. This difference frequency is expressed by

$$\begin{aligned} D(t) &= \frac{1}{2\pi} \frac{d}{dt} \{ \Delta\phi(t) \} \\ &= \frac{1}{2\pi} \frac{d}{dt} \{ \phi_r(t) - \phi_t(t) \} \\ &= \frac{1}{2\pi} \frac{d}{dt} \{ 2\pi f_0(t - 2\tau) - 2\pi f_0 t \} \\ &= -2f_0 \frac{d\tau}{dt} \end{aligned}$$

where f_0 is the transmitted frequency.

Now, since τ satisfies the relation

$$c\tau = R_A(t - \tau),$$

it follows that

$$c \frac{d\tau}{dt} = \dot{R}_A(t - \tau) \left[1 - \frac{d\tau}{dt} \right].$$

Thus

$$\frac{d\tau}{dt} = \frac{\dot{R}_A(t - \tau)}{c + \dot{R}_A(t - \tau)},$$

and consequently

$$D(t) = -2f_0 \left[\frac{\dot{R}_A(t')}{c + \dot{R}_A(t')} \right] \quad t' = t - \tau$$

The transit time and doppler calculations involve apparent range and apparent range rate, which in turn involve refraction errors.

Under the usual assumptions regarding time invariance and spherical symmetry of the refracting medium it follows that

$$R_A = R + \rho[R, E]$$

$$A_A = A$$

$$E_A = E + \varepsilon[R, E]$$

$$\dot{R}_A = \dot{R} + \frac{\partial \rho}{\partial R} \dot{R} + \frac{\partial \rho}{\partial E} \dot{E}$$

where R, A, E , are true range, azimuth, and elevation of the target relative to the tracker, ρ and ε are range and elevation refraction errors, and R_A, A_A, E_A are apparent range, azimuth, and elevation. Observe that ρ and ε are functions of true range and elevation only.

The usual method of computing refraction errors involves "ray-tracing" through the refractive medium with an assumed spherically symmetric refraction profile. Viewed as a black box, such a refraction algorithm performs in the following manner. The inputs are the apparent values R_A and E_A , and the outputs are the true values R and E together with the errors ρ and ε . Within the black box operation begins by tracing a ray from the tracker with elevation angle E_A and continuing until the apparent range along the ray equals R_A . Then R and E are computed from the endpoints of the ray.

Now if the refraction algorithm produces smooth values of ρ and ε , the algorithm can also be used to obtain numerical partial derivatives of ρ and ε with respect to R and E . However, since the inputs to the refraction algorithm are R_A and E_A , rather than R and E , it is necessary to employ the chain rule. Thus

$$\begin{bmatrix} \frac{\partial \rho}{\partial R} & \frac{\partial \rho}{\partial E} \\ \frac{\partial \epsilon}{\partial R} & \frac{\partial \epsilon}{\partial E} \end{bmatrix} = \begin{bmatrix} \frac{\partial \rho}{\partial R_A} & \frac{\partial \rho}{\partial E_A} \\ \frac{\partial \epsilon}{\partial R_A} & \frac{\partial \epsilon}{\partial E_A} \end{bmatrix} \begin{bmatrix} \frac{\partial R_A}{\partial R} & \frac{\partial R_A}{\partial E} \\ \frac{\partial E_A}{\partial R} & \frac{\partial E_A}{\partial E} \end{bmatrix},$$

or

$$\begin{bmatrix} \frac{\partial \rho}{\partial R} & \frac{\partial \rho}{\partial E} \\ \frac{\partial \epsilon}{\partial R} & \frac{\partial \epsilon}{\partial E} \end{bmatrix} = \begin{bmatrix} \frac{\partial \rho}{\partial R_A} & \frac{\partial \rho}{\partial E_A} \\ \frac{\partial \epsilon}{\partial R_A} & \frac{\partial \epsilon}{\partial E_A} \end{bmatrix} \begin{bmatrix} \frac{\partial R}{\partial R_A} & \frac{\partial R}{\partial E_A} \\ \frac{\partial E}{\partial R_A} & \frac{\partial E}{\partial E_A} \end{bmatrix}^{-1}.$$

But since

$$\frac{\partial \rho}{\partial R_A} = 1 - \frac{\partial R}{\partial R_A}, \quad \frac{\partial \rho}{\partial E_A} = -\frac{\partial R}{\partial E_A},$$

$$\frac{\partial \epsilon}{\partial R_A} = -\frac{\partial E}{\partial R_A}, \quad \frac{\partial \epsilon}{\partial E_A} = 1 - \frac{\partial E}{\partial E_A},$$

it follows that only ρ , ϵ , and the partial derivative matrix

$$\begin{bmatrix} \frac{\partial R}{\partial R_A} & \frac{\partial R}{\partial E_A} \\ \frac{\partial E}{\partial R_A} & \frac{\partial E}{\partial E_A} \end{bmatrix}$$

need be computed.

The complete procedure, using the symmetric central difference formula, is outlined below.

1. At the tracker, trace a ray with elevation angle E_A to apparent ranges $R_A - \alpha/2$, R_A and $R_A + \alpha/2$, and compute the corresponding true ranges and elevations:

$$R^-, R, R^+,$$

$$E^-, E, E^+.$$

2. Put $\rho = R_A - R$, $\epsilon = E_A - E$

$$\frac{\partial R}{\partial R_A} = \frac{R^+ - R^-}{\alpha}, \quad \frac{\partial E}{\partial R_A} = \frac{E^+ - E^-}{\alpha}.$$

3. At the tracker trace a ray with elevation angle $E_A - \alpha/2$ to apparent range R_A and compute the corresponding true range and elevation: R^-, E^- .

4. Repeat 3. with elevation angle $E_A + \alpha/2$ and compute the true range and elevation: R^+, E^+ .

5. Using the results of 3. and 4. compute

$$\frac{\partial R}{\partial E_A} = \frac{R^+ - R^-}{\alpha}, \quad \frac{\partial E}{\partial E_A} = \frac{E^+ - E^-}{\alpha}.$$

6. Put $\Delta = \left(\frac{\partial R}{\partial R_A}\right) \left(\frac{\partial E}{\partial E_A}\right) - \left(\frac{\partial E}{\partial R_A}\right) \left(\frac{\partial R}{\partial E_A}\right)$

$$\frac{\partial \rho}{\partial R} = \left(\frac{1}{\Delta}\right) \left(\frac{\partial E}{\partial E_A}\right) - 1$$

$$\frac{\partial \rho}{\partial E} = - \left(\frac{1}{\Delta}\right) \left(\frac{\partial R}{\partial E_A}\right)$$

$$\frac{\partial \epsilon}{\partial R} = - \left(\frac{1}{\Delta}\right) \left(\frac{\partial E}{\partial R_A}\right)$$

$$\frac{\partial \epsilon}{\partial E} = \left(\frac{1}{\Delta}\right) \left(\frac{\partial R}{\partial R_A}\right) - 1.$$

6.6 Measurement Processing

The measurement variational equation for a four channel metric tracking radar is given by (19) in Appendix C.2, and is repeated here for convenience.

$$\begin{aligned}
(1) \quad \delta y(t) = & \left[\tilde{H}_a - \frac{\tilde{H}_a \tilde{x}'_a}{c + \tilde{R}'_A} \left(\frac{\partial \tilde{R}_A}{\partial x'_a} \right) \right] \delta x_a(\tilde{t}') \\
& + \tilde{H}_b \delta x_b + \tilde{H}_c \delta x_c \\
& - \frac{c}{c + \tilde{R}'_A} \tilde{H}_a \tilde{x}'_a m_s^T(\tilde{t}) \delta x_d \\
& + \frac{c}{c + \tilde{R}'_A} \tilde{H}_a \tilde{x}'_a m_v^T(\tilde{t}') \delta x_e \\
& + \left[\tilde{H}_f - \frac{\tilde{H}_a \tilde{x}'_a}{c + \tilde{R}'_A} \left(\frac{\partial \tilde{R}_A}{\partial x'_f} \right) \right] \delta x_f \\
& + \left[\tilde{H}_g - \frac{\tilde{H}_a \tilde{x}'_a}{c + \tilde{R}'_A} \left(\frac{\partial \tilde{R}_A}{\partial x'_g} \right) \right] \delta x_g \\
& + v(t) .
\end{aligned}$$

The variation δy is formed as the vector difference of the actual and nominal measurements.

Thus

$$\delta y \equiv y - \tilde{y} = \begin{bmatrix} r \\ \phi \\ \psi \\ d \end{bmatrix} - \begin{bmatrix} \tilde{r} \\ \tilde{\phi} \\ \tilde{\psi} \\ \tilde{d} \end{bmatrix} .$$

But if only the encoder measurements are available for processing, the dynamic error terms replace the actual measurements, and

$$\delta y \equiv \bar{y} - \tilde{y} = \begin{bmatrix} \bar{r} \\ \bar{\phi} \\ \bar{\psi} \\ \bar{d} \end{bmatrix} - \begin{bmatrix} \tilde{r} \\ \tilde{\phi} \\ \tilde{\psi} \\ \tilde{d} \end{bmatrix} .$$

The expressions for the dynamic error terms are given in Appendix C.3.

The nominal measurement vector \tilde{y} is computed by evaluating the measurement equation using nominal values for all states. The form of the measurement equation which is suitable for this purpose is called the dynamic measurement equation. The algorithm for the dynamic measurement equation is given below. The inputs are the encoder values R_E, A_E, E_E, R'_E , sensor indicated measurement time θ_S , and nominal values of state variables. It is assumed also that the range and elevation refraction corrections and their respective partial derivatives, together with the true range and elevation at which they apply, have been obtained by the procedure outlined in Section 6.5. These quantities are denoted by $\rho, \epsilon, \frac{\partial \rho}{\partial R}, \frac{\partial \rho}{\partial E}, \frac{\partial \rho}{\partial R'}, \frac{\partial \rho}{\partial E'}, R, E$, respectively. The outputs of the algorithm are the nominal measurement variables, $\tilde{r}, \tilde{\phi}, \tilde{\psi}, \tilde{d}$.

6.6.1 Dynamic Measurement Algorithm

1. Correct sensor time by solving

$$\theta_S = \tilde{t} + m_S^T(\tilde{t}) \tilde{x}_d$$

for \tilde{t} , iteratively if necessary by Newton's method.

2. Compute the coordinate transformation from the geocentric to the topocentric system at the sensor site using astronomical longitude $\tilde{\lambda}_A$ and latitude $\tilde{\theta}_A$:

$$C_G^T = \begin{bmatrix} -\sin \tilde{\lambda}_A & \cos \tilde{\lambda}_A & 0 \\ -\sin \tilde{\theta}_A \cos \tilde{\lambda}_A & -\sin \tilde{\theta}_A \sin \tilde{\lambda}_A & \cos \tilde{\theta}_A \\ \cos \tilde{\theta}_A \cos \tilde{\lambda}_A & \cos \tilde{\theta}_A \sin \tilde{\lambda}_A & \sin \tilde{\theta}_A \end{bmatrix} .$$

3. Compute the geocentric coordinates of the sensor site using the geodetic longitude $\tilde{\lambda}_G$, latitude $\tilde{\theta}_G$, and height \tilde{H}_G :

$$X_s = \frac{a \cos \tilde{\theta}_G}{\sqrt{1 - \epsilon^2 \sin^2 \tilde{\theta}_G}} \cos \tilde{\lambda}_G + \tilde{H}_G \cos \tilde{\theta}_G \cos \tilde{\lambda}_G,$$

$$Y_s = \frac{a \cos \tilde{\theta}_G}{\sqrt{1 - \epsilon^2 \sin^2 \tilde{\theta}_G}} \sin \tilde{\lambda}_G + \tilde{H}_G \cos \tilde{\theta}_G \sin \tilde{\lambda}_G,$$

$$Z_s = \frac{a (1 - \epsilon^2) \sin \tilde{\theta}_G}{\sqrt{1 - \epsilon^2 \sin^2 \tilde{\theta}_G}} + \tilde{H}_G \sin \tilde{\theta}_G,$$

where a is the mean equatorial radius of the ellipsoid and ϵ is its eccentricity.

4. Apply the iterative procedure developed in Section 6.5 to solve for transit time, refraction and doppler:

(i) Initialize: $\tilde{\tau} = 0$, $\tilde{t}' = \tilde{t}$.

(ii) In the nominal trajectory time tags have been corrected for nominal vehicle timing errors, put $\tilde{t}'' = \tilde{t}'$; otherwise $\tilde{t}'' = \tilde{t}' + m_V^T(\tilde{t}') \tilde{x}_e$.

(iii) Interpolate the nominal trajectory to \tilde{t}'' obtaining geocentric coordinates \tilde{x} , \tilde{y} , \tilde{z} , $\tilde{\dot{x}}$, $\tilde{\dot{y}}$, $\tilde{\dot{z}}$, and compute:

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = C_G^T \begin{bmatrix} \tilde{x} - X_s \\ \tilde{y} - Y_s \\ \tilde{z} - Z_s \end{bmatrix}$$

$$\begin{bmatrix} \tilde{\dot{u}} \\ \tilde{\dot{v}} \\ \tilde{\dot{w}} \end{bmatrix} = C_G^T \begin{bmatrix} \tilde{\dot{x}} \\ \tilde{\dot{y}} \\ \tilde{\dot{z}} \end{bmatrix}$$

$$\tilde{R} = \sqrt{\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2},$$

$$\tilde{A} = \tan_2^{-1}(\tilde{u}, \tilde{v}) ,$$

$$\tilde{E} = \sin^{-1}(\tilde{w}/\tilde{R}) ,$$

$$\tilde{R} = (\tilde{u}\tilde{u} + \tilde{v}\tilde{v} + \tilde{w}\tilde{w})/\tilde{R} ,$$

$$\dot{\tilde{E}} = \frac{1}{\sqrt{\tilde{u}^2 + \tilde{v}^2}} \left[\dot{\tilde{w}} - \left(\frac{\tilde{w}}{\tilde{R}} \right) \dot{\tilde{R}} \right] .$$

(iv) Compute apparent coordinates in presence of refraction

$$\tilde{\rho} = \rho + \frac{\partial \rho}{\partial R} (\tilde{R} - R) + \frac{\partial \rho}{\partial E} (\tilde{E} - E)$$

$$\tilde{\epsilon} = \epsilon + \frac{\partial \epsilon}{\partial R} (\tilde{R} - R) + \frac{\partial \epsilon}{\partial E} (\tilde{E} - E)$$

$$\dot{\tilde{\rho}} = \frac{\partial \rho}{\partial R} \dot{\tilde{R}} + \frac{\partial \rho}{\partial E} \dot{\tilde{E}}$$

$$\tilde{R}_A = \tilde{R} + \tilde{\rho}$$

$$\tilde{A}_A = \tilde{A}$$

$$\tilde{E}_A = \tilde{E} + \tilde{\epsilon}$$

$$\dot{\tilde{R}}_A = \dot{\tilde{R}} + \dot{\tilde{\rho}} .$$

(v) Compute transit time increment:

$$\delta\tau = \frac{\tilde{R}_A - c\tau}{c + \dot{\tilde{R}}_A} .$$

(vi) Test for convergence:

IF ($\delta\tau = 0$) GO TO (viii).

(vii) Adjust transit time and retarded measurement time and repeat above steps:

$$\begin{aligned}\tilde{\tau} &\leftarrow \tilde{\tau} + \delta\tau \\ \tilde{t}' &= \tilde{t} - \tilde{\tau}\end{aligned}$$

GO TO (ii).

(viii) Compute apparent doppler shift:

$$\tilde{D}_A = \frac{-2f_0 \dot{\tilde{R}}_A}{c + \dot{\tilde{R}}_A}$$

where f_0 is the transmitter frequency.

5. Apply target dependent errors at \tilde{t}' :

$$\begin{aligned}\tilde{R}_R &= \tilde{R}_A + \Delta\tilde{R}_T \\ \tilde{A}_R &= \tilde{A}_A + \Delta\tilde{A}_T \\ \tilde{E}_R &= \tilde{E}_A + \Delta\tilde{E}_T \\ \tilde{D}_R &= \tilde{D}_A + \Delta\tilde{D}_T\end{aligned}$$

6. Recover doppler encoder value:

$$D_E = \frac{-2f_0 \dot{\tilde{R}}_E}{c + \dot{\tilde{R}}_E}$$

7. Correct encoder angles for encoder errors.

$$\begin{aligned}\tilde{A}_S &= A_E - \Delta\tilde{A}_E, \\ \tilde{E}_S &= E_E - \Delta\tilde{E}_E,\end{aligned}$$

iteratively if necessary using Newton's method.

8. Denote the tracker outputs by

$$\begin{aligned}\tilde{R}_O &= \tilde{R}_E \\ \tilde{A}_O &= \tilde{A}_S \\ \tilde{E}_O &= \tilde{E}_S \\ \tilde{D}_O &= \tilde{D}_E ,\end{aligned}$$

apply the feedback errors,

$$\begin{aligned}\tilde{R}_F &= \tilde{R}_O - \Delta\tilde{R}_F \\ \tilde{D}_F &= \tilde{D}_O - \Delta\tilde{D}_F ,\end{aligned}$$

and compute the topocentric (locally level) to electronic boresite coordinate transformation, \tilde{C}_{LL}^{EB} .

9. Compute the discriminator outputs:

$$\begin{aligned}\tilde{r}_O &= \tilde{R}_R - \tilde{R}_F , \\ \tilde{d}_O &= \tilde{D}_R - \tilde{D}_F , \\ \tilde{\phi}_O &= \tan^{-1}(e_x/e_y) , \\ \tilde{\psi}_O &= \tan^{-1}(e_z/e_y) ,\end{aligned}$$

where

$$\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} = \tilde{C}_{LL}^{EB} \begin{bmatrix} \sin\tilde{A}_R \cos\tilde{E}_R \\ \cos\tilde{A}_R \cos\tilde{E}_R \\ \sin\tilde{E}_R \end{bmatrix} .$$

10. Apply the sensor errors to obtain the nominal sensor outputs:

$$\begin{aligned}\tilde{r} &\equiv \tilde{r}_s = \tilde{r}_o + \Delta\tilde{r}_s \\ \tilde{d} &\equiv \tilde{d}_s = \tilde{d}_o + \Delta\tilde{d}_s \\ \begin{bmatrix} \tilde{\phi} \\ \tilde{\psi} \end{bmatrix} &\equiv \begin{bmatrix} \tilde{\phi}_s \\ \tilde{\psi}_s \end{bmatrix} = \begin{bmatrix} \tilde{\phi}_o \\ \tilde{\psi}_o \end{bmatrix} + \begin{bmatrix} SF_{\phi\phi} & SF_{\phi\psi} \\ SF_{\psi\phi} & SF_{\psi\psi} \end{bmatrix} \begin{bmatrix} \tilde{\phi}_o \\ \tilde{\psi}_o \end{bmatrix}.\end{aligned}$$

6.6.2 Measurement Equation Partial Derivatives

The coefficients of the state vector variations in (1) constitute the total or dynamic partial derivatives of the measurement with respect to the respective state variables. Thus

$$\frac{\partial y}{\partial x_a} = H_a - \frac{H_a \dot{x}'_a}{c + R'_A} \left(\frac{\partial R_A}{\partial x'_a} \right)$$

$$\frac{\partial y}{\partial x_b} = H_b$$

$$\frac{\partial y}{\partial x_c} = H_c$$

$$\frac{\partial y}{\partial x_d} = - \frac{\dot{x}'_a}{c + R'_A} H_a x'_a m_s^T$$

$$\frac{\partial y}{\partial x_e} = \frac{\dot{x}'_a}{c + R'_A} H_a x'_a m_v^T$$

$$\frac{\partial y}{\partial x_f} = H_f - \frac{H_a \dot{x}'_a}{c + R'_A} \left(\frac{\partial R_A}{\partial x'_f} \right)$$

and

$$\frac{\partial y}{\partial x_g} = H_g - \frac{H_a \dot{x}'_a}{c + \dot{R}'_A} \left(\frac{\partial R_A}{\partial x_g} \right).$$

The quantities H_a , H_b , H_c , H_f and H_g , appearing in the above expressions, are called static partial derivatives of the measurement with respect to the respective state variables.

The total partial derivatives of the measurement can be calculated by numerical differentiation using the dynamic measurement algorithm specified in the preceding subsection. Alternatively, the total partial derivatives can be constructed from the static measurement partial derivatives, where the static partial derivatives are computed by numerical differentiation using the static measurement algorithm to be given below. The latter method of obtaining measurement partial derivatives is useful in conjunction with the measurement variation averaging processing method to be considered in Section 6.6.5.

The static measurement algorithm is very similar to the dynamic algorithm, except that, in addition to the other inputs, \tilde{t} and \tilde{t}' are specified, and thus it is not necessary to solve for transit time by iteration. Steps in the static algorithm which are identical with corresponding steps in the dynamic algorithm will be listed without elaboration.

6.6.3 Static Measurement Algorithm

1. Compute the coordinate transformation from the geocentric to the topocentric system.
2. Compute the geocentric coordinates of the sensor site.
3. If the nominal trajectory time tags have been corrected for nominal vehicle timing errors, put $\tilde{t}'' = \tilde{t}'$; otherwise $\tilde{t}'' = \tilde{t}' + m_V^T(\tilde{t}') \tilde{x}_e$.
4. Interpolate the nominal trajectory to \tilde{t}'' obtaining geocentric coordinates, and compute \tilde{R} , \tilde{A} , \tilde{E} , \tilde{R} , \tilde{E} .

5. Compute apparent coordinates $\tilde{R}_A, \tilde{A}_A, \tilde{E}_A, \tilde{R}_A$.
6. Compute apparent doppler shift \tilde{D}_A .
7. Apply target dependent errors at t' .
8. Recover doppler encoder value.
9. Correct encoder angles for encoder error.
10. Apply range and doppler feedback errors, and compute the coordinate transformation \tilde{C}_{LL}^{EB} .
11. Compute the discriminator outputs $\tilde{r}_O, \tilde{d}_O, \tilde{\phi}_O, \tilde{\psi}_O$.
12. Apply sensor errors to obtain sensor outputs $\tilde{r}_S, \tilde{d}_S, \tilde{\phi}_S, \tilde{\psi}_S$.

In using either measurement algorithm for the purpose of obtaining partial derivatives by numerical differentiation, computational efficiency can be enhanced by judicious application of the chain rule. For example, if $\beta \equiv x_{d1}$ denotes sensor timing bias, it follows that

$$\frac{\partial y}{\partial x_d^T} = \frac{\partial y}{\partial \beta} m_s^T(t),$$

$$\frac{\partial y}{\partial x_e^T} = - \frac{\partial y}{\partial \beta} m_v^T(t').$$

Thus differentiation with respect to the scalar β suffices to compute the derivatives with respect to the vectors x_d and x_e .

6.6.4 Measurement Processing with Adjustable Estimation Times

The measurement variation equation, which has been the subject of this section, can be concisely written as follows

$$\begin{aligned}
 (2) \quad \delta y(t) = & \frac{\partial y}{\partial x_a} \delta x_a(\tilde{t}') + \frac{\partial y}{\partial x_b} \delta x_b(\tilde{t}') + \frac{\partial y}{\partial x_c} \delta x_c(\tilde{t}) \\
 & + \frac{\partial y}{\partial x_d} \delta x_d(\tilde{t}) + \frac{\partial y}{\partial x_e} \delta x_e(\tilde{t}') + \frac{\partial y}{\partial x_f} \delta x_f \\
 & + \frac{\partial y}{\partial x_g} \delta x_g + v(t) .
 \end{aligned}$$

For purposes of illustration all states except those pertaining to survey and refraction have been represented as time varying. Observe that there are three categories of states in (2); those which apply at \tilde{t}' , those which apply at \tilde{t} , and those which are constant.

Now let x_α denote all states which relate to the vehicle or its trajectory (including navigation and geopotential error states which do not appear in (2)), let x_β denote all states which relate to the metric sensor system, and let x_γ denote geodetic and refraction error states. Then the measurement variation can be expressed by

$$(3) \quad \delta y(t) = \frac{\partial y}{\partial x_\alpha} \delta x_\alpha(\tilde{t}') + \frac{\partial y}{\partial x_\beta} \delta x_\beta(\tilde{t}) + \frac{\partial y}{\partial x_\gamma} \delta x_\gamma(\tilde{t}) + v(t) .$$

Assume there is no state noise, let t^* be arbitrary, and observe that

$$(4) \quad \begin{bmatrix} \delta x_\alpha(\tilde{t}') \\ \delta x_\beta(\tilde{t}) \\ \delta x_\gamma(\tilde{t}) \end{bmatrix} = \begin{bmatrix} \Phi_{\alpha\alpha}(\tilde{t}', t^*) & 0 & 0 \\ 0 & \Phi_{\beta\beta}(\tilde{t}, t^*) & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \delta x_\alpha(t^*) \\ \delta x_\beta(t^*) \\ \delta x_\gamma(t^*) \end{bmatrix} .$$

Substitution of (4) in (3) yields

$$(5) \quad \delta y(t) = \begin{bmatrix} \frac{\partial y}{\partial x_\alpha} \Phi_{\alpha\alpha}(\tilde{t}', t^*) \\ \frac{\partial y}{\partial x_\beta} \Phi_{\beta\beta}(\tilde{t}, t^*) \\ \frac{\partial y}{\partial x_\gamma} \end{bmatrix} \begin{bmatrix} \delta x_\alpha(t^*) \\ \delta x_\beta(t^*) \\ \delta x_\gamma(t^*) \end{bmatrix} + v(t)$$

in which the measurement variation at time t is related linearly to the state vector variation at an arbitrary time t^* .

Equation (5) forms the basis for several processing options in which a collection of measurements is used to update the estimate of the state vector at an arbitrarily specified time. For example the state estimation times can be made periodic, even though the measurement times are aperiodic. It must be remembered, however, that (5) is valid only in the absence of state noise.

6.6.5 Processing with Measurement Variation Average

A suboptimal method of processing will now be considered in which the measurement variations, in each channel of each sensor, are averaged over a specified time interval and collectively used to update the state estimate at a specified time within the interval. An important distinction here is that it is the measurement variations, not the measurement themselves, which are averaged.

The averaging technique to be developed here is based on (5) of the preceding subsection and consequently is valid only in the absence of state noise.

Averaging of the measurement variations prior to update of the state vector estimate reduces the filter processing rate and thus enhances computational efficiency. Also, averaging tends to reduce the magnitude of the serial correlation coefficients of the filter input noise without deletion of any measurements. Thus when serial correlation of measurement noise is a potential problem, such as when encoder measurements are used exclusively, averaging can be used to more nearly satisfy the requirement that the measurement noise be serially uncorrelated (Cf. Appendix A).

The reason that the technique of measurement variation averaging is suboptimal is that higher order terms in a Taylor series for the averaged variation are ignored. The error introduced by the neglected terms increases with the duration of the time interval over which the averaging

takes place. To maintain the truncation error within acceptable limits, processing intervals are partitioned into subintervals and the measurement variations on each subinterval are separately averaged.

To develop the technique of measurement variation averaging, consider a single channel of an arbitrary metric sensor. Let an interval be selected over which the variations are to be averaged. Let $\{t_i, i \in I\}$ denote the set of measurement times in the interval, and similarly let $\{\tilde{t}_i; i \in I\}$ denote the set of corresponding retarded measurement times. Let t_I denote the state estimation time specified for the interval. Then from (5) it follows for each $i \in I$ that

$$(6) \quad \delta y(t_i) = \begin{bmatrix} \frac{\partial y}{\partial x_\alpha} \phi_{\alpha\alpha}(\tilde{t}_i, t_I) & \frac{\partial y}{\partial x_\beta} \phi_{\beta\beta}(\tilde{t}_i, t_I) & \frac{\partial y}{\partial x_\gamma} \end{bmatrix} \begin{bmatrix} \delta x_\alpha(t_I) \\ \delta x_\beta(t_I) \\ \delta x_\gamma(t_I) \end{bmatrix} + v(t_i)$$

$$\equiv B_i \delta x(t_I) + v(t_i),$$

where

$$B_i = \begin{bmatrix} \frac{\partial y}{\partial x_\alpha} \phi_{\alpha\alpha}(\tilde{t}_i, t_I) & \frac{\partial y}{\partial x_\beta} \phi_{\beta\beta}(\tilde{t}_i, t_I) & \frac{\partial y}{\partial x_\gamma} \end{bmatrix}$$

and

$$\delta x(t_I) = \begin{bmatrix} \delta x_\alpha(t_I) \\ \delta x_\beta(t_I) \\ \delta x_\gamma(t_I) \end{bmatrix}.$$

Since (6) is written for a single channel, B_i is a row vector. In fact, B_i is the total partial derivative of $y(t_i)$ with respect to $x(t_I)$, i.e.,

$$B_i = \frac{\partial y(t_i)}{\partial x^T(t_I)}.$$

If B_j is viewed in terms of its constituent static partial derivatives, as evidenced in (1), and transition matrix components, it is found that, for fixed t_I , B_j is a function of $\tilde{x}_\alpha(\tilde{t}'_i)$, $\dot{\tilde{x}}_\alpha(\tilde{t}'_i)$, $\tilde{x}_\beta(\tilde{t}_i)$, $\dot{\tilde{x}}_\beta(\tilde{t}_i)$, \tilde{x}_γ , \tilde{t}'_i , \tilde{t}_i and the encoder measurement vector $z(t_i)$.

Let \bar{x}_α , $\dot{\bar{x}}_\alpha$, \bar{x}_β , $\dot{\bar{x}}_\beta$, \bar{t} , \bar{t}' , \bar{z} be arbitrary and expand B to first order in (6). Thus

$$\begin{aligned}
 (7) \quad \delta y(t_i) &= \bar{B} \delta x(t_I) \\
 &+ \delta x^T(t_I) \left[\frac{\partial \bar{B}^T}{\partial x_\alpha} (\tilde{x}_\alpha(\tilde{t}'_i) - \bar{x}_\alpha) + \frac{\partial \bar{B}^T}{\partial \dot{x}_\alpha} (\dot{\tilde{x}}_\alpha(\tilde{t}'_i) - \dot{\bar{x}}_\alpha) \right. \\
 &\quad + \frac{\partial \bar{B}^T}{\partial x_\beta} (\tilde{x}_\beta(\tilde{t}_i) - \bar{x}_\beta) + \frac{\partial \bar{B}^T}{\partial \dot{x}_\beta} (\dot{\tilde{x}}_\beta(\tilde{t}_i) - \dot{\bar{x}}_\beta) \\
 &\quad + \frac{\partial \bar{B}^T}{\partial t} (\tilde{t}_i - \bar{t}) + \frac{\partial \bar{B}^T}{\partial t'} (\tilde{t}'_i - \bar{t}') \\
 &\quad \left. + \frac{\partial \bar{B}^T}{\partial z} (z(t_i) - \bar{z}) \right] + v(t_i),
 \end{aligned}$$

where B and its derivatives are evaluated at \bar{x}_α , $\dot{\bar{x}}_\alpha$, \bar{x}_β , $\dot{\bar{x}}_\beta$, \bar{x}_γ , \bar{t} , \bar{t}' , \bar{z} with $\bar{x}_\gamma \equiv \tilde{x}_\gamma(t_I)$.

Now let $\langle \cdot \rangle_I$ denote the operator which performs simple averaging over I. Thus if $N(I)$ is the number of elements of I, then

$$\langle \epsilon_i \rangle_I = \frac{1}{N(I)} \sum_{i \in I} \epsilon_i.$$

Next put

$$\bar{x}_\alpha = \langle \tilde{x}_\alpha(\tilde{t}') \rangle_I$$

$$\dot{\bar{x}}_\alpha = \langle \dot{\tilde{x}}_\alpha(\tilde{t}') \rangle_I$$

$$\bar{x}_\beta = \langle \tilde{x}_\beta(\tilde{t}.) \rangle_I$$

$$\dot{\bar{x}}_\beta = \langle \dot{\tilde{x}}_\beta(\tilde{t}.) \rangle_I$$

$$\bar{t} = \langle \tilde{t}. \rangle_I$$

$$\bar{t}' = \langle \tilde{t}' \rangle_I$$

$$\bar{z} = \langle z(t.) \rangle_I .$$

Then to first order

$$(8) \quad \langle \delta y(t.) \rangle_I = \bar{B} \delta x(t_I) + \langle v(t.) \rangle_I .$$

Extending the above analysis to all channels of a given metric sensor and reverting to earlier notation it follows that

$$(9) \quad \langle \delta y(t.) \rangle_I = \left[\frac{\partial \bar{y}}{\partial x_\alpha} \phi_{\alpha\alpha}(\bar{t}', t_I) \quad \left| \quad \frac{\partial \bar{y}}{\partial x_\beta} \phi_{\beta\beta}(\bar{t}, t_I) \quad \left| \quad \frac{\partial \bar{y}}{\partial x_\gamma} \right. \right] \begin{bmatrix} \delta x_\alpha(t_I) \\ \delta x_\beta(t_I) \\ \delta x_\gamma(t_I) \end{bmatrix} + \langle v(t.) \rangle_I$$

where the total partial derivatives $\frac{\partial \bar{y}}{\partial x_\alpha}$, $\frac{\partial \bar{y}}{\partial x_\beta}$, $\frac{\partial \bar{y}}{\partial x_\gamma}$ are evaluated at \bar{x}_α , $\dot{\bar{x}}_\alpha$, \bar{x}_β , $\dot{\bar{x}}_\beta$, \bar{x}_γ , \bar{t} , \bar{t}' , \bar{z} .

Because of the decoupling of \bar{x}_α and $\dot{\bar{x}}_\alpha$, \bar{x}_β and $\dot{\bar{x}}_\beta$, and \bar{t} and \bar{t}' which occurred in the above development, it is necessary that the total partial derivatives $\frac{\partial \bar{y}}{\partial x_\alpha}$, $\frac{\partial \bar{y}}{\partial x_\beta}$, $\frac{\partial \bar{y}}{\partial x_\gamma}$ be computed from static partial derivatives

using the expressions given at the outset of Section 6.6.2.

Finally, the measurement error covariance matrix for use in conjunction with measurement variation averaging must be computed. Extending (8) to include the measurements from all channels of a given sensor which are to be processed over the interval I, let

$$R_{ij} = E[v(t_i)v(t_j)^T] .$$

It then follows that

$$\text{cov} (\langle v(t.) \rangle_I) = \frac{1}{N(I)^2} \sum_{i \in I} \sum_{j \in I} R_{ij} .$$

In the particular case in which the measurement noise is serially uncorrelated,

$$\text{cov} (\langle v(t.) \rangle_I) = \frac{1}{N(I)^2} \sum_{i \in I} R_{ii} = \frac{1}{N(I)} \langle R_{..} \rangle_I .$$

6.7 Augmentation and Permutation of the State Vector

At any stage of the filtering operation it is possible to augment the state vector with additional states. For example, when an RV is deployed it is appropriate to augment the state vector with the RV position and velocity states.

Moreover, when the filter operation is switched from one RV to another, the trajectory states of the new RV become active while those of the old RV become inactive. Consequently, in order to use the partitions defined either in 6.2 or 6.3, it is necessary to permute the trajectory states of the two RVs in question.

To illustrate the state augmentation procedure at RV deployment, assume the state vector prior to augmentation is partitioned in the form

$$x = \begin{bmatrix} x_a \\ x_c \end{bmatrix},$$

where x_a includes all dynamic states of the bus vehicle (BV) and x_c includes all static states. Let x_b denote the dynamic states of the RV.

For a specified deployment configuration, a separation model based on energy and momentum relations can be developed. In general the model has the form

$$\begin{bmatrix} \delta x_a^+ \\ \delta x_b^+ \end{bmatrix} = \begin{bmatrix} \phi_{aa} \\ \phi_{ba} \end{bmatrix} \delta x_a^- + p + w$$

where ϕ_{aa} and ϕ_{ba} are known matrices, p is a known vector, and w is a zero mean random vector with covariance

$$Q = \Gamma \Gamma^T .$$

δx_a^- is the variation of x_a prior to deployment, and δx_a^+ , δx_b^+ are variations after deployment.

The state vector is augmented to include x_b . Thus

$$x = \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} .$$

After deployment, the variation estimates are given by

$$\begin{bmatrix} \hat{\delta x}_a^+ \\ \hat{\delta x}_b^+ \end{bmatrix} = \begin{bmatrix} \phi_{aa} \\ \phi_{ba} \end{bmatrix} \hat{\delta x}_a^- + p$$

and

$$\hat{\delta x}_c^+ = \hat{\delta x}_c^- .$$

The square root covariance matrix prior to deployment is given in upper triangular form by

$$\begin{bmatrix} S_{aa}^- & S_{ac}^- \\ 0 & S_{cc}^- \end{bmatrix}.$$

The augmented square root covariance matrix after deployment can be obtained by applying the extrapolation technique developed in Section 6.2 to the matrix

$$\left[\begin{array}{cc|cc} 0 & \phi_{aa} S_{aa}^- & \phi_{aa} S_{ac}^- & \\ 0 & \phi_{ba} S_{aa}^- & \phi_{ba} S_{ac}^- & \Gamma \\ \hline 0 & 0 & S_{cc}^- & 0 \end{array} \right].$$

The procedure is actually applied only to the submatrix

$$\left[\begin{array}{c|c} 0 & \phi_{aa} S_{aa}^- \\ \hline 0 & \phi_{ba} S_{aa}^- \end{array} \right] \Gamma \quad \text{to obtain} \quad \left[\begin{array}{c|c|c} S_{aa}^+ & S_{ab}^+ & 0 \\ \hline 0 & S_{bb}^+ & 0 \end{array} \right].$$

The resulting square root covariance matrix is given in upper triangular form by

$$\begin{bmatrix} S_{aa}^+ & S_{ab}^+ & S_{ac}^+ \\ 0 & S_{bb}^+ & S_{bc}^+ \\ 0 & 0 & S_{cc}^+ \end{bmatrix}$$

where $S_{ac}^+ = \phi_{aa} S_{ac}^-$, $S_{bc}^+ = \phi_{ba} S_{ac}^-$, and $S_{cc}^+ = S_{cc}^-$.

To illustrate the state vector permutation procedure, assume the state vector prior to permutation has the form

$$x = \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix},$$

and the square root covariance is given in upper triangular form by

$$\begin{bmatrix} S_{aa}^- & S_{ab}^- & S_{ac}^- \\ 0 & S_{bb}^- & S_{bc}^- \\ 0 & 0 & S_{cc}^- \end{bmatrix}.$$

Assume further that the state vector permutation consists of an interchange of x_a and x_b .

Permutation of the state vector variation estimates is accomplished by simply interchanging $\hat{\delta x}_a$ and $\hat{\delta x}_b$. Similarly the square root covariance matrix is permuted by a simple row interchange to obtain

$$\begin{bmatrix} 0 & S_{bb}^- & S_{bc}^- \\ S_{aa}^- & S_{ab}^- & S_{ac}^- \\ 0 & 0 & S_{cc}^- \end{bmatrix}.$$

Since the row interchange destroys the triangularity of S , the retriangularization method of 6.2 is applied to the submatrix

$$\begin{bmatrix} 0 & S_{bb}^- \\ S_{aa}^- & S_{ab}^- \end{bmatrix}$$

to obtain

$$\begin{bmatrix} S_{bb}^+ & S_{ba}^+ \\ 0 & S_{aa}^+ \end{bmatrix}.$$

Thus the permuted and retriangularized S matrix is given by

$$\begin{bmatrix} S_{bb}^+ & S_{ba}^+ & S_{bc}^+ \\ 0 & S_{aa}^+ & S_{ac}^+ \\ 0 & 0 & S_{cc}^+ \end{bmatrix},$$

where $S_{bc}^+ = S_{bc}^-$, $S_{ac}^+ = S_{ac}^-$, and $S_{cc}^+ = S_{cc}^-$.

6.8 General Partitioned Structure of the State Vector, Transition Matrix, and Measurement Sensitivity Matrix

Extensive use of vector element permutation and matrix partitioning has been employed throughout this report for the purposes of illustrating theoretical features and demonstrating computationally efficient algorithms. To this point, the partitions which have been used were selected on a case by case basis and no general structure has been evident. The purpose of this section is to consolidate the piecemeal use of permutation and partitioning into a general TRAM structure.

The state vector may be augmented with additional elements when an RV is launched, and the state vector elements may undergo a permutation when the estimation process passes from one trajectory segment to another. For a given processing interval, the general structure of the state vector and transition matrix depends on whether TMIG data is used to construct the trajectory over the interval in question. The structure of the measurement sensitivity matrix depends further on the measurement processing option which is selected.

When TMIG data is used to construct the trajectory, the general structure of the state vector is as follows.

$$x = \begin{bmatrix} x_a \\ x_b \\ x_c \\ x_d \\ x_e \\ x_f \\ x_g \end{bmatrix},$$

where

- x_a - trajectory states of vehicle on segment in process including dynamic IMU states
- x_b - trajectory states of other vehicles which have been augmented including dynamic IMU states of those vehicles so equipped
- x_c - metric sensor states including timing, survey, and refraction states*
- x_d - static IMU states, including timing, of other vehicles which have been augmented
- x_e - IMU timing of vehicle on segment in process
- x_f - static IMU states of vehicle on segment in process
- x_g - geopotential states.

The corresponding partitioned structure for the transition matrix is given by

$$\Phi = \begin{bmatrix} \phi_{aa} & 0 & 0 & 0 & \phi_{ae} & \phi_{af} & \phi_{ag} \\ 0 & I & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I \end{bmatrix}.$$

*Throughout this subsection, it is assumed that all metric sensor states are static, since SAMTEC sensors can be modeled in this fashion.

If the measurements are processed asynchronously, the general structure of the measurement sensitivity matrix is given by

$$H = \left[\begin{array}{c|c|c|c|c|c|c} \frac{\partial y}{\partial x_a} & 0 & \frac{\partial y}{\partial x_c} & 0 & \frac{\partial y}{\partial x_e} & 0 & 0 \\ \hline \hline \hline \hline \hline \hline \hline \end{array} \right].$$

However, if either adjustable estimation time processing or measurement variation averaging is employed, the structure is

$$H = \left[\begin{array}{c|c|c|c|c|c|c} \frac{\partial y}{\partial x_a} \phi_{aa} & 0 & \frac{\partial y}{\partial x_c} & 0 & \frac{\partial y}{\partial x_e} + \frac{\partial y}{\partial x_a} \phi_{ae} & \frac{\partial y}{\partial x_a} \phi_{af} & \frac{\partial y}{\partial x_a} \phi_{ag} \\ \hline \hline \hline \hline \hline \hline \hline \end{array} \right].$$

When a trajectory segment is constructed without the use of TMIG data, the general structure of the state vector is given by

$$x = \begin{bmatrix} x_a \\ x_b \\ x_c \\ x_d \\ x_e \end{bmatrix},$$

where

- x_a - trajectory states of vehicle on segment in process
- x_b - trajectory states of other vehicles which have been augmented including dynamic IMU states of those vehicles so equipped.
- x_c - metric sensor states including timing, survey and refraction states
- x_d - static IMU states including timing, of all augmented vehicles so equipped
- x_e - geopotential states.

The transition matrix for this case has the structure

$$\Phi = \begin{bmatrix} \Phi_{aa} & 0 & 0 & 0 & \Phi_{ae} \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & I \end{bmatrix}.$$

If the measurements are processed asynchronously,

$$H = \begin{bmatrix} \frac{\partial y}{\partial x_a} & 0 & \frac{\partial y}{\partial x_c} & 0 & 0 \end{bmatrix}.$$

But if adjustable estimation time processing or measurement variation averaging is employed, then

$$H = \begin{bmatrix} \frac{\partial y}{\partial x_a} \Phi_{aa} & 0 & \frac{\partial y}{\partial x_c} & 0 & \frac{\partial y}{\partial x_a} \Phi_{ae} \end{bmatrix}.$$

6.9 Suboptimal Processing in the Presence of IMU Noise

In the development of the navigation variational equations in Appendix B, it was necessary to introduce additional states to account for the noise in the IMU velocity output. The function of these additional states is to model the propagation of velocity noise into position and velocity variations.

In an optimal processor, the IMU noise states are included in the state vector and are estimated. If, however, the position error induced by velocity noise* is negligible in comparison with metric sensor position errors, a suboptimal processing scheme in which the IMU noise states are deleted can be employed with only slight degradation in estimation accuracy. The principal advantage of the suboptimal processor is that computational efficiency is greatly enhanced.

*For MMIII, this error is estimated to be less than one foot at boost termination.

Although the IMU noise states are deleted and the effect of IMU noise on position error is neglected in the suboptimal processor, IMU velocity noise is not ignored, but is instead treated as an equivalent measurement error in each of the metric sensor doppler channels. As a consequence the measurement error covariance matrix of the collective doppler channels is modified by the addition of another matrix to represent IMU velocity noise. Since this results in a nondiagonal measurement error covariance, the decorrelation process developed in 6.2 must be applied if scalar measurement update is to be used.

In order to achieve a substantial increase in computational efficiency, it is necessary to employ measurement variation averaging in conjunction with the suboptimal processor. (Note that this is made possible by the deletion of the IMU noise states, since this eliminates state noise in the navigation variation equations.)

An algorithm for calculation of the measurement error covariance induced by IMU velocity noise is developed below.

Let D_1, \dots, D_M denote the doppler channels to be processed on an interval over which the measurement variations are averaged. Let t_0, \dots, t_N denote the discrete readout times of IMU velocity in the averaging interval (t_0, t_N), and assume the IMU velocity noise is sequentially uncorrelated.

1. For each $i = 1, \dots, M$, compute the retarded measurement times $t_{i1}^i, \dots, t_{iL(i)}^i$ of the i -th doppler channel which are in the interval (t_0, t_N).
2. For each $i = 1, \dots, M$ and each $j = 1, \dots, L(i)$, determine k such that $t_{k-1} \leq t_{ij}^i < t_k$ and set C_{ij} equal to the $3 \times 3(N+1)$ matrix

$$0_3 \quad \dots \quad 0_3 \quad \underbrace{\begin{pmatrix} t_k - t_{ij}^i \\ t_k - t_{k-1} \end{pmatrix} I_3}_{\text{kth block}} \quad \underbrace{\begin{pmatrix} t_{ij}^i - t_{k-1} \\ t_k - t_{k-1} \end{pmatrix} I_3}_{\text{(k+1)th block}} \quad 0_3 \quad \dots \quad 0_3 \quad 3 \times 3(N+1)$$

3. For each $i = 1, \dots, M$, put

$$\bar{c}_i = \frac{1}{N(i)} \sum_{j=1}^{L(i)} c_{ij} .$$

where $N(i)$ is the number of measurements in the averaging interval from the i -th doppler channel.

4. For each $i = 1, \dots, M$, evaluate the partial derivative of the i -th doppler channel measurement with respect to velocity at the mid point of the averaging interval. Denote this by

$$\frac{\partial \bar{d}_i}{\partial v^T} .$$

5. Construct the $M \times 3(N+1)$ matrix \bar{C} given by

$$\bar{C} = \begin{bmatrix} \left(\frac{\partial \bar{d}_1}{\partial v^T} \right) \bar{c}_1 \\ \vdots \\ \left(\frac{\partial \bar{d}_M}{\partial v^T} \right) \bar{c}_M \end{bmatrix} .$$

6. Partition \bar{C} into $M \times 3$ blocks, i.e.,

$$\bar{C} = [\bar{C}_0 \mid \dots \mid \bar{C}_N]$$

where each \bar{C}_k is $M \times 3$.

Now denote the IMU velocity noise sequence by r_0, \dots, r_N , and observe that equivalent measurement error in the doppler channels is given by

$$- [\bar{c}_0 | \dots | \bar{c}_N] \begin{bmatrix} r_0 \\ \vdots \\ r_N \end{bmatrix} .$$

Then if the IMU velocity noise covariance is given by

$$C_V = E[r_k r_k^T], \quad k = 0, \dots, N,$$

it follows that the covariance of the equivalent doppler measurement error is given by

$$\Delta R = \bar{c}_0 \bar{c}_V \bar{c}_0^T + \dots + \bar{c}_N \bar{c}_V \bar{c}_N^T .$$

The total doppler measurement error covariance is then formed by adding ΔR to the doppler measurement error covariance due to receiver noise.

7.0 PROGRAM REQUIREMENTS

The purpose of this section is to establish guidelines for TRAM program development. These guidelines are intended to insure that on the one hand the functional requirements of estimation and error analysis are satisfied, and on the other, the program structure is designed to be sufficiently flexible and modular to provide for analysis of future as well as current types of test vehicles and range support instrumentation.

The end-to-end execution of the TRAM program requires essentially three phases of operation, consisting of respectively, data preparation, estimation, and error analysis.

In the data preparation or pre-processing phase, metric sensor and telemetry data are organized into files suitable for use in the estimation phase. Included in this phase are computation of refraction errors and their partial derivatives, gross editing, and statistical analysis of raw metric data. Also included is the extraction of trajectory information from the telemetry data.

In the estimation phase, the recursive filtering and smoothing operations are performed iteratively until the estimates of the state vector elements converge.

In the error analysis phase, error propagation parameters and error budget values are combined to obtain estimates of the mean and covariance of the state vector estimation error.

The error propagation parameters, required in the error analysis phase, can be obtained during the estimation phase. This is accomplished by augmenting the filtering and smoothing equations to include the sensitivity of the estimation error to initialization errors (in all estimated states together with a selected subset of the constrained states) and, in certain cases, the estimation error covariance due to random measurement error.

However, the option should exist to exercise the filtering and smoothing equations in an error propagation mode in which no measurements are processed, but in which all error propagation parameters are obtained.

In the sequel, the three phases of TRAM operation will be considered separately and in more detail. Functional requirements, program modules and structure, user options, and alternative modes of operation will be evident from the discussion.

To maintain perspective, only those program operations which are essential for estimation and error analysis will be considered. Auxiliary operations required to provide analyst aids, in the form of various kinds of program output, are not discussed. The reason for this is that the types of auxiliary operations required are somewhat dependent on the user, and the capability to perform these operations should reside in separate program modules which are readily augmented or modified.

7.1 Pre-Processing Phase

The purpose of the pre-processing phase is to prepare files of data for use in the estimation phase. The particular operations which must be performed are highly dependent on the data source. However, since the estimator performs operations of filtering, smoothing, and systematic error compensation, these operations should not be performed by the pre-processor, on data to be supplied to the estimator, regardless of the data source.

7.1.1 Metric Data Pre-Processing

The required pre-processing functions for metric data are:

- (i) gross editing
- (ii) refraction calculations
- (iii) statistical analysis
- (iv) file organization

The gross editing function serves first to identify intervals over which each metric station is tracking properly. Second within each track interval, subintervals are identified over which each sensor channel is tracking. Finally, isolated points are identified by means of an edit flag, where loss of data or transmission error occurs.

In order to avoid aliasing of data, the level at which isolated points are edited must be well above the level of systematic and random measurement error. Thus, if M is a measurement, R is the reference used for editing, σ is the rms value of total measurement and reference error, and the measurement is edited whenever

$$|M - R| > L ,$$

then L should be at least as great as 10σ .

Refraction calculations are required in the pre-processor in order to avoid repeating the time consuming ray trace operations on each iteration during the estimation phase.

The refraction parameters are computed as functions of geometric, rather than refracted, range and elevation angle. These parameters are required at each point where a range, elevation or doppler measurement is to be processed. Since the independent variables, i.e., geometric range and elevation angle, are known only approximately for each measurement, the partial derivatives of the refraction parameters with respect to the independent variables are also required.

For the purpose of calculating the refraction parameters only, the raw measurements should be corrected to obtain the best a priori value of geometric range and elevation angle, and the refraction parameters should be evaluated at this point.

Furthermore, when ionospheric refraction is significant, both group and phase refraction parameters must be computed for each affected measurement.

The statistical analysis of the metric data is required to determine the variances due to random measurement error. Unlike the measurement error proper, the variances are smooth functions of time. Accurate estimation of measurement error variances requires smoothing and calculation of sample statistics over intervals of sufficient duration to reduce statistical noise to a negligible level.

When both sensor element outputs and encoder outputs are included in the measurement set for a given channel, the variance of the sensor element measurement noise is required at each sensor element measurement point. If only encoder outputs are available, the variance of the encoded random error is required at each encoder measurement point.

The raw data, refraction parameters, and measurement variances are organized into separate files for each tracking station. A header must be applied to each file which defines the tracking intervals over which the individual channels of the station maintain track.

The data in each file is stored sequentially, in the order of increasing time. The data to be stored includes:

encoder data: T, R, A, E, D

encoder variances: σ_R^2 , σ_A^2 , σ_E^2 , σ_D^2

sensor signals: t, r, ϕ , ψ , d

sensor variances: σ_r^2 , σ_ϕ^2 , σ_ψ^2 , σ_d^2

refraction parameters: ρ , ϵ , $\frac{\partial \rho}{\partial R}$, $\frac{\partial \rho}{\partial E}$, $\frac{\partial \epsilon}{\partial R}$, $\frac{\partial \epsilon}{\partial E}$; R, E

track indicator flags

edit flags

signal strength

7.1.2 Telemetry Data Pre-Processing

The pre-processing of telemetry data involves nothing more than extraction of trajectory data and file organization. A file header must be generated which defines intervals of data loss, and the times of guidance system initialization, staging events, vehicle deployment, and system shutdown.

The data in the file is stored sequentially, in the order of increasing time. The stored data includes:

major cycle data: time, position, velocity

minor cycle data: time, accelerometer, rate gyro and platform gimbal angle outputs

time insertion data: time of receipt at TM station

7.2 Estimation Phase

The operations which are performed in connection with the estimation phase of the TRAM program can be categorized as follows:

1. Input
2. Initialization
3. Filter
4. Smoother
5. Convergence Test
6. Reset

Input operations include those which must be performed by the user in order to initiate the estimation phase.

Initialization operations consist of those which must be performed by the program before any filter or smoother operations can occur.

Filter operations consist of measurement processing and/or error propagation calculations. Each time the filter is activated, it performs operations over the entire trajectory. The filter operations are recursive, and proceed in the direction of increasing time on each trajectory segment. Each segment is partitioned into intervals over which the filter configuration is fixed. Thus the measurement coverage and state dynamics do not change on an interval, and all state vector augmentation and permutation operations occur at interval junctures. Finally, the intervals are partitioned into subintervals, and all bulk storage I/O operations are performed at sub-interval junctures.

Smoother operations consist of filter output processing and/or error propagation calculations. The smoother is activated at the conclusion of each set of filter operations over the entire trajectory. The smoother operations, once activated, are also performed recursively over the entire trajectory, but in precisely the reverse order to those of the filter and in the direction of decreasing time*.

*There is one exception to this which will be explained later.

A convergence test is performed at the conclusion of each set of smoother operations over the entire trajectory. This test determines if the state vector estimates have converged or if the maximum number of iterations has occurred. If neither part of the test is satisfied, the program performs another iteration of the filter and smoother operations.

The reset operation is performed whenever an iteration of the filter and smoother operations is required. The reset operation is required to reinitialize the filter state vector and square root covariance to the original input values. Also the nominal state vector is equated to the estimated state vector obtained by the preceding smoothing operation.

The requirements for each of the above operational categories will now be considered.

7.2.1 Input Requirements

A. Schedules

The user must define the order in which the trajectory segments are to be processed. A measurement schedule for each tracking station, a schedule indicating portions of the trajectory which are inertially instrumented, and a schedule indicating thrust termination and reentry points on the trajectory. (Some of these schedule items are derivable from the file headers generated in the pre-processing phase.)

The schedule inputs are required in order to develop processing intervals over which the filter configuration is invariant.

B. Parameters

The user must specify the complete set of parameters to be employed in the program. These include a priori estimates of all state vector elements and the state covariance matrix. Parameters which are not included in the state vector but which enter either the state or measurement equations must be specified. Also, the state noise covariance must be specified, and if a measurement covariance is to be used which differs from that implied by the measurement variances included in the data files, it must also be specified.

C. State Vector Categorization

The user must specify, for each state vector element, whether it is to be estimated or constrained. From the subset of constrained elements, the user must specify those which are to be propagated for error analysis.

D. Measurement Processing

For each filter processing interval, the user must specify the type of measurement processing to be used. The options are:

1. asynchronous
2. adjusted estimation time
3. measurement variation averaging

The first may be used in any case. The latter two are recommended only when state noise can safely be ignored.

When the adjusted estimation time option is to be employed, the user must specify the estimation times.

When the measurement variation averaging option is to be employed, the user must specify the intervals (within the filter processing subintervals) over which the averaging is performed and the estimation time within each averaging interval.

In addition to the above, the user must also indicate whether the processing is to be restricted to the use of encoder outputs. If this is to be the case, the user must specify the dynamic error coefficients to be used for each sensor.

E. Nominal State Vector

For the initial set of filter operations, the user must specify the nominal state vector. (On subsequent iterations the nominal state vector is

automatically set by the program.) The user must always specify the initial condition for the nominal state vector, but he also has the option of specifying the nominal state vector at arbitrary times over the entire trajectory. When the user specifies only the initial condition of the nominal state vector, he has the option of allowing the program to reset the nominal state vector at arbitrary times by equating it to the filter estimate of the state vector.

It should be noted, that although the user can in effect specify nominal state vector resets arbitrarily, the program will actually restrict the occurrence of resets to certain points within the filter cycle.

F. Convergence Criteria

The user must specify the convergence criteria of the estimator. The user must also specify the maximum number of iterations of the filter/smoothen which are to be allowed in attempting to satisfy the convergence test.

The convergence criteria will be based on the differences between elements of the estimated states obtained on successive iterations. For each estimated state element, a test of the form "Is $|x_k^{(i+1)} - x_k^{(i)}| < \epsilon_k$?" will be applied, where i denotes the iteration count, k denotes the state vector element, and $\epsilon_k \geq 0$ is specified by the user.

7.2.2 Initialization Requirements

Initialization operations consist of those which are performed prior to the first set of filter/smoothen operations and which are not repeated on subsequent iterations.

A. Control Logic

The operations to be performed by the filter and smoother are dependent on user inputs and program constraints. The control of these operations can be accomplished by the use of various logical variables (flags) and the specification of times at which discrete transitions in processing are to occur.

Each state vector element has a flag to indicate in which of three distinct categories it falls:

1. Estimated
2. Propagated
3. Constant

The constant states are invariant throughout all program operations. The propagated states are constant for all program operations other than those involving numerical partial derivative calculations with respect to these states as required for error propagation.

The estimator state vector configuration is fixed on each processing interval, and includes all estimated states which have been augmented up to and including the epoch of the interval in process. A permutation array, or its logical equivalent, must be generated for each processing interval to indicate (i) which elements are in the estimated state vector and (ii) the order of these elements within the estimated state vector.

The beginning and ending times for each processing interval must be set, and flags must be generated to indicate the measurements to be processed. The specification of measurements includes the designation of tracking stations and the sensor channels to be processed.

Flags to indicate the type of measurement processing and the type of smoother to be used on each interval must also be set.

Finally within each processing interval, the beginning and ending times for the I/O subintervals must be set.

At the juncture of processing intervals, augmentation and/or permutation of the elements of the estimated state vector can occur. This requires discrete processing of the estimator state vector and square root covariance matrix. The logic to control these discrete operations must be set during initialization.

On the first filtering pass only, the nominal state vector is reset at arbitrary times which have been designated during initialization. Filter logic will inhibit nominal state vector reset except at designated points within its cycle, and a reset will actually occur at the first of these points to be reached after the designated reset time.

As an alternative to the specification of arbitrary reset times, flags must be available which provide for either of two extreme cases:

- (i) no reset
- (ii) reset at the highest possible rate, i.e., at every allowable point of reset in the filter cycle.

A flag must also be set to indicate whether the nominal state vector is to be reset externally or internally. In the former instance, the reset is accomplished by interpolation of the external reference to the actual reset time. In the latter case, the reset is accomplished by equating the nominal state vector to the value obtained from the filter estimate at the actual reset time. (In either case, the state variation estimate must be reset such that the whole state vector estimate at reset is invariant.)

Finally, flags must be set to indicate whether error propagation calculations are to be performed during the estimation phase or postponed until the error analysis phase, and to control the optional calculations associated with error propagation.

B. Calculation of Constants

Depending on the assigned category for each state, a number of program variables may actually be constant during either the estimation phase or the error analysis phase, or both. For example, station coordinates, coordinate transformations, geopotential parameters, and so on, may be constant during an entire phase of program operation.

It is important to compute, to the extent possible, all program constants during initialization. Furthermore, the structure of the routines which are used in the TRAM program should be designed so that the flags which

define the category of each state element can be used to bypass program blocks whenever these blocks serve only to recompute constants.

C. Filter Initialization

Prior to filter operation, the filter nominal state vector, state vector variation estimate, and square root covariance must all be initialized.

The filter nominal state vector is equated to the initial condition which the user has specified for the nominal state vector. However, the initial condition of the nominal state vector must be brought into time commensuration with the a priori state vector estimate, if this condition is not already satisfied. This is accomplished by interpolation if possible; otherwise it is accomplished by integration of the state differential equation using the nominal state vector initial condition.

The state vector variation estimate is equated to the difference (estimate minus nominal) between the a priori state vector estimate and the initial value of the nominal state vector.

The square root covariance is initialized by applying the Cholesky decomposition to the a priori covariance of the state vector estimation error. The object matrix to which the Cholesky decomposition is applied is obtained by row/column permutations of the a priori covariance to conform with the initial state vector configuration.

7.2.3 Filter Requirements

At interval junctures, elements may be augmented to the filter state vector, and the filter state vector may then undergo a permutation. Correspondingly, augmentation and permutation operations must be performed on the square root covariance matrix of the filter, which must also be retriangularized.

All filter I/O operations are performed at subinterval junctures. The inputs include measurement data and the nominal trajectory. The required outputs are dependent on the particular processing options in effect, but basically include all filter outputs and error propagation parameters required by the smoother.

Within each subinterval the filter operates in a cyclical manner performing functions of extrapolation, update, error propagation, and nominal resets when required. The specific filter requirements will be considered separately for each of the measurement processing options.

A. Asynchronous Processing

Asynchronous measurement processing is recommended on each interval over which state noise is not negligible, but it may in fact be used on any interval.

Consider an interval on which asynchronous measurement processing is to be employed. On this interval, the measurements from a fixed set of tracking stations are scheduled for processing.

The processing is performed recursively on blocks of measurements, each consisting of one measurement from each scheduled sensor channel of each scheduled tracking station on the interval under consideration.

The first step in the processing of a block is to check the edit flags and delete those channels which have been edited during the pre-processing phase.

Next, for each measurement that remains in the block, the nominal measurement is computed along with the nominal receive time and retarded time of the measurements at each station, and measurement variations are formed.

The measurement variations of each station are then processed in turn, where the order of station processing is the same as the order of increasing retarded measurement times.

For each station, the processing consists of extrapolation operations followed by update operations. The extrapolation operations are performed to bring the filter and error propagation parameters up to the station retarded measurement time. Then, filter and error propagation update operations are performed for each sensor channel of the station.

If nominal state vector resets have been scheduled on the first filter pass, they only can occur immediately following the processing of one measurement block, prior to computing the nominal retarded measurement times of the next block. Thus at the conclusion of the processing of each measurement block, on the first filter pass only, a test is performed to determine if a scheduled reset time has been passed during the processing of that block. If so, the nominal state vector and state variation estimate are reset at the time corresponding to the end of the block.

On subsequent filter passes, nominal state vector resets do not occur, since the nominal state vector is always equal to the whole state estimate obtained on the preceding smoother pass.

However, when fixed-interval smoothing is employed an operation which resembles reset must be performed as a part of each extrapolation. This operation is required because the variation estimates are extrapolated with state noise equal to zero.

Thus when fixed-interval smoothing is employed, the nominal state vector and the state variation estimate are each extrapolated (to the retarded measurement time of the station in process) with state noise equal to zero. Then the difference between the extrapolated state vector and the nominal state vector, interpolated to the retarded time, is computed (extrapolated minus nominal), and the difference is added to the extrapolated state variation estimate to obtain an adjusted state variation estimate at the retarded measurement time. The adjusted state variation estimate is used in subsequent processing, and the extrapolated state vector and extrapolated state variation estimate have no further use.

Intermediate filter outputs are required on processing intervals in which fixed-interval smoothing is to be employed. In this case filter outputs must be saved before and after the update operations for each station in every processing block.

Thus, within each processing block, when the extrapolation operations for a given station are complete, the nominal state vector and state variation estimate along with the nominal received and retarded measurement times,

the filter square root covariance matrix, the filter transition matrix, and the error propagation parameters must be saved. Then when the update operations for all sensor channels of the station are complete, the state variation estimate, the filter square root covariance matrix, and the error propagation parameters must again be saved.

B. Processing with Adjustable Estimation Time

Adjustable estimation time processing is recommended only when state noise is negligible*.

Adjustable estimation time processing is identical with asynchronous processing to the point where the measurement variation, receive time, and retarded time have been computed for each measurement in a block.

At this point, extrapolation operations are performed to bring the filter and error propagation parameters up to the estimation time which has been designated for the measurement block as a whole.

Then, filter and error propagation update operations are performed on the entire block of measurement variations in any convenient order.

If nominal state vector resets have been scheduled on the first filter pass, they can only occur immediately following the completion of the set of update operations for an entire measurement block. Thus, whenever, on the first filter pass only, a scheduled reset time has been passed during the processing of a measurement block, the nominal state vector and the state variation estimate are reset at the estimation time associated with the block.

On subsequent filter passes, nominal state vector resets do not occur. However, when fixed-interval smoothing is employed, the same reset-like operation described for asynchronous processing must be performed at the close of each extrapolation.

*Although the actual state noise process may be negligible, this does not preclude the use of a nonzero state noise covariance matrix in the filter mechanization and subsequent use of the fixed-interval smoother. This type of mechanization can be useful in reducing unmodeled error growth in the estimates.

On processing intervals in which fixed-interval smoothing is to be employed, filter outputs must be saved before and after the set of update operations for each measurement block.

Thus when the extrapolation operation for a measurement block is complete, the nominal state vector, the state variation estimate, the designated estimation time, the filter square root covariance matrix, the filter transition matrix, and the error propagation parameters must be saved. Then, when the set of update operations for the block as a whole is complete, the state variation estimate, the filter square root covariance matrix, and the error propagation parameters must be saved.

C. Processing with Measurement Variation Averaging

Processing with measurement variation averaging is an extension of adjustable estimation time processing, and is likewise recommended only when state noise is negligible.

When measurement variation averaging is employed, each processing subinterval is partitioned into averaging intervals. Then, for each averaging interval, a block of measurements is formed, and an estimation time within the interval is designated.

The measurements which constitute a block consist of all measurements, within the averaging interval of the block, from all scheduled tracking stations for the interval in process.

Once a block is formed, edit flags are tested, and individual measurements which have been edited during the pre-processing phase are deleted from the block. Then, for each measurement remaining in the block, calculation of the nominal measurement and nominal receive and retarded measurement times is performed.

Next the measurement variations for the block as whole are computed and, for each sensor channel, the average measurement variation over the averaging interval is computed.

Extrapolation operations are now performed to bring the filter and error propagation parameters up to the estimation time which has been designated for the averaging interval.

After extrapolation, the filter and error propagation update operations are performed on the entire block of averaged measurement variations (consisting of at most one per sensor channel) in any convenient order.

If nominal state vector resets have been scheduled on the first filter pass, they can only occur following the completion of the set of update operations for an entire averaging interval. Thus, on the first filter pass only, at the conclusion of the set of update operations, a test is performed to determine whether a nominal reset has been scheduled at any time within the averaging interval.

If no nominal reset has been scheduled within an averaging interval, the filter processing described above is repeated for the next block of measurements and the next averaging interval.

But if a nominal reset has been scheduled within the averaging interval, it is now performed. Moreover, regardless of the designated reset time, the actual reset takes place at the estimation time of the interval. If the nominal is to be reset from an external source, the source value is first interpolated or integrated to the interval estimation time. Then the nominal state vector and the state variation estimate are reset, and the filter processing described above is repeated for the next block measurements on the next averaging interval.

On processing intervals in which fixed-interval smoothing is to be employed, filter outputs must be saved before and after the set of update operations for each averaging interval.

Thus when the extrapolation operation for an averaging interval is complete, the nominal state vector, the state variation estimate, the designated estimation time, the filter square root covariance matrix, the filter transition matrix (from the preceding estimation time to the current

estimation time), and the error propagation parameters must be saved. Then, when the set of update operations for the averaging interval is complete, the state variation estimate, the filter square root covariance matrix, and the error propagation parameters must be saved.

As in all other cases, between the points at which filter outputs before and after update are saved for the smoother, no nominal reset is allowed.

7.2.4 Smoother Requirements

The two smoother types which may be employed in the TRAM program differ substantially in mechanization and operation, and, for this reason, the two will first be considered separately.

A. Retrograde-Integration Smoother

This smoother is by far the simpler of two types, but it can be effectively employed only when state noise is negligible.

On intervals in which retrograde-integration smoothing is to be employed, it is only necessary to integrate the dynamic states and the error propagation parameters. The integration process is initialized at the end of the interval and proceeds in the direction of decreasing time. Static states are held fixed during each retrograde-integration.

This type of smoother requires no intermediate filter output, but its own output can be stored at the same subinterval junctures as used for filter inputs.

If the retrograde-integration smoother is used on an interval at the end of a trajectory segment, then it is initialized by the final filter estimate of the whole state on that interval. When this case occurs, the final time on the interval, i.e., initial time for the retrograde-integrator, must be adjusted by applying to the nominal vehicle time tag at the end of the interval, the difference between the filter estimate of and the nominal value of the vehicle timing correction at the end of the interval.

B. Fixed-Interval Smoother

The fixed-interval smoother is employed on processing intervals in which state noise is not negligible. It may also be used at interval junctures, coinciding with vehicle deployment, where random separation errors occur.

The fixed-interval smoother operates recursively on filter outputs and error propagation parameters, but in reverse order to the filter recursion. The smoother requires the filter estimates of the whole state vector before and after each filter update. These are obtained, respectively, by combining the nominal state vector and the filter estimates of the state variation before and after each update. The subinterval structure used for filter I/O operations is also used for smoother I/O.

At each stage in fixed-interval smoothing, the time tag of the filter estimate must be adjusted by applying the difference between the smoother estimate and the nominal value of the vehicle timing correction at the estimation time.

At the conclusion of the entire smoothing process, the smoothed estimates can be interpolated to any convenient set of times, designated by the user, and stored for future use. For this it is only necessary that the stored smoothed estimates be sufficiently closely spaced such that smoothed estimates at arbitrary times can be accurately obtained using low order interpolating splines.

C. Comments on the Smoothing Process for the Trajectory as a Whole

The smoothing operation is performed over the trajectory as a whole in exactly the reverse order to the filtering operations. At interval junctures the smoother may change from one type to another, but the process is essentially continuous. Furthermore, when discontinuities occur at interval juncture such as with separation errors at deployment, the fixed-interval smoother is used to make the transition, regardless of the smoother types used in either of the intervals forming the juncture.

It was mentioned earlier that there is one exception to the rule that the smoother operates in reverse order to the filter. The single exception occurs, as a processing option, when state noise is negligible in the inertial guidance system of the boost vehicle.

In this case, at the conclusion of the filter pass on any iteration, the initial conditions for the smoother estimate of the boost vehicle trajectory are completely known, and since there is no state noise, the smoothed boost vehicle trajectory can be reconstructed by forward integration of the navigation equations. (It is tacitly assumed that the launch point survey error has been estimated, or it is negligible, in which case the uncertainty in the navigation initial position and velocity is zero.)

When this option is exercised on the boost segment, the smoother processing on all other segments is unaffected, except that it terminates at points just after vehicle deployment, and it is no longer necessary to smooth the discontinuity at deployment events.

7.2.5 Convergence Test

The number of filter/smoother iterations which are performed during the estimation phase is determined by the convergence test which is performed at the end of each iteration.

The first part of the test is performed by comparing differences, in each of the components of the estimated state vector, obtained on successive iterations. For convergence, each element of the estimated state vector must satisfy the convergence criterion which has been specified by the user. This part of the test is performed on the second and all subsequent estimation cycles.

The other part of the test simply counts the number of estimation cycles which have been performed and compares this with the maximum number the user has allowed

If the maximum number of cycles permitted by the user is attained without satisfaction of the convergence criteria, program operation is halted and an error message to this effect is generated.

7.2.6 Reset

When the convergence test results in another iteration of the estimator, it is necessary to perform a reset operation.

The reset is required to set the nominal state vector and reinitialize the filter.

The nominal state vector is equated to the estimated state vector obtained by the smoother. For this purpose, the smoother trajectory estimates can be interpolated to a designated set of time points. Then, for intermediate points required on the next filter cycle, spline interpolation can be used between the designated time points.

Reinitialization is required for the filter state variation estimate and the filter square root covariance matrix. The state variation estimate is equated to the difference between the a priori whole state vector estimate and the value of the new nominal state vector at initialization time.

7.3 Error Analysis Phase

In this phase of processing an error analysis of the estimation process is performed. This is accomplished by processing an error budget, specified by the user, with the error propagation parameters of the estimation process. The result of the error analysis is the bias and covariance of estimation error based on the specified error budget.

In its complete form the error budget must include a schedule of state and measurement noise covariances as well as the mean and covariance of the initial value of the vector whose elements are the estimated and propagated states.

The error propagation parameters may be obtained during the estimation phase. However, it is usually more efficient to defer calculation of these parameters until the estimation process converges. Also, it may be required for planning purposes to perform an error analysis in advance of the estimator phase. For these reasons, an error propagation mode is required in which the error propagation parameters are obtained separately from the estimation phase.

In the error propagation mode, the nominal state vector is supplied externally, and it is necessary to perform all estimator functions, except measurement processing, as well as the error propagation functions. In the particular case in which the error propagation mode follows the estimation phase, the externally supplied nominal state vector is equal to the whole state vector estimate obtained during the estimation phase.

In order that the noise induced errors be properly accounted for, the filter and smoother gains actually used for estimation must, with one exception, be based on the noise covariance schedules in the error budget. The one exception occurs on intervals in which state noise is negligible, in which case the covariance of estimation error due to measurement noise can be computed by the alternate method given in Section 5.0, regardless of how the filter gains are obtained.

REFERENCES

- [1] Polard, Joseph J., "Orbital Parameter Determination by Weighted Least Square Error and Kalman Filtering Methods," Master Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, June 1971.
- [2] Thompson, G. T., Computation of State Vector and Transition Matrix for TRAM, Technical Report Number PA100-77-18, Space and Missile Test Center, Vandenberg AFB, California, June 1977.
- [3] Kalman, R. E., "A New Approach to Linear Filtering and Prediction Problems," Journal of Basic Engineering, Transaction ASME. Volume 82D, 1960, pages 35-50.
- [4] Meditch, J. S., Stochastic Optimal Linear Estimation and Control, McGraw-Hill, New York, 1969.
- [5] Potter, J. E., "New Statistical Formulas," Space Guidance Analysis Memo 40, Instrumentation Laboratory, MIT, Cambridge, Massachusetts, April 1963.
- [6] Andrews, A., "A Square Root Formulation of the Kalman Covariance Equations," AIAA Journal, Volume 5, Number 7, July 1967, pages 1309-1314.
- [7] Carlson, N. A., "Fast Triangular Formulation of the Square Root Filter," AIAA Journal, Volume 11, Number 9, September 1973, pages 1259-1265.
- [8] Brooks, R. A., Technical Note on Steady State Doppler Tracking Error, Federal Electric Corporation, Vandenberg AFB, California, February 1976.
- [9] Halmos, P. R., Finite Dimensional Vector Spaces, 2nd Edition, Van Nostrand, Princeton, 1958.
- [10] Bellman, R. E., Introduction to Matrix Analysis, 2nd Edition, McGraw-Hill, New York, 1970.

- [11] Ralston, A., A First Course in Numerical Analysis, McGraw-Hill, New York, 1965.
- [12] Hoffman, K. M. and Kunze, R. A., Linear Algebra, 2nd Edition, Prentice-Hall, Englewood Cliffs, 1971, page 280.
- [13] Hewitt, E. and Stromberg, K., Real and Abstract Analysis, Springer-Verlag, New York, 1965, pages 240-242.

APPENDICES

A. LINEAR ESTIMATION

Consider a linear stochastic system described by

$$(1) \quad x_{i+1} = \Phi_i x_i + u_i, \quad i = 0, 1, \dots, N-1,$$

$$(2) \quad y_i = H_i x_i + v_i, \quad i = 0, 1, \dots, N.$$

In these equations x is the state vector, and y is the measurement vector. $\{u_i, i = 0, 1, \dots, N-1\}$ is a sequence of random vectors called the state noise process, and $\{v_i, i = 0, 1, \dots, N\}$ is a similar sequence called the measurement noise process. These processes are assumed to be zero mean, sequentially uncorrelated, and mutually uncorrelated with each other and x_0 . Mathematically these assumptions are expressed as follows:

$$E(u_i) = 0, \quad E(u_i u_j^T) = Q_i \delta_{ij}; \quad i, j = 0, 1, \dots, N-1,$$

$$E(v_i) = 0, \quad E(v_i v_j^T) = R_i \delta_{ij}; \quad i, j = 0, 1, \dots, N$$

and

$$E(u_i v_j^T) = 0, \quad E(x_0 u_i^T) = 0, \quad E(x_0 v_j^T) = 0;$$

where

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}.$$

Q and R are the state noise and measurement noise covariance matrices, respectively. To complete the system description, it is assumed that the a priori mean and covariance of x_0 , denoted by \hat{x}_0^- and P_0^- , respectively, are also specified.

The estimation procedures to be considered in the remainder of this appendix will apply to the linear system given by (1) and (2).

A.1 Optimal Linear Estimation

For the system (1), (2), let $\tilde{x}(i|j)$ denote a function which is in the form of a constant plus a linear function of the measurement set y_0, \dots, y_j and which has the property $E[\tilde{x}(i|j)] = E(x_i)$. Such a function is said to be a linear unbiased estimate of x_i given y_0, \dots, y_j .

Let $\hat{x}(i|j)$ be a linear unbiased estimate of x_i given y_0, \dots, y_j , and suppose $\hat{x}(i|j)$ has the property*

$$E[|\hat{x}(i|j) - x_i|^2] \leq E[|\tilde{x}(i|j) - x_i|^2]$$

for all linear unbiased estimates $\tilde{x}(i|j)$ of x_i given y_0, \dots, y_j . Then $\hat{x}(i|j)$ is said to be an optimal linear estimate of x_i given y_0, \dots, y_j .

It can be shown that an optimal linear estimate of x_i given y_0, \dots, y_j always exists and is unique. The notation $\hat{x}(i|j)$ will be used exclusively to denote the optimal estimate defined above, and the notation $P(i|j)$ will be used to denote the error covariance of $\hat{x}(i|j)$ defined by

$$P(i|j) \equiv E[(\hat{x}(i|j) - x_i)(\hat{x}(i|j) - x_i)^T].$$

$P(i|j)$ is also called the state covariance of x_i given y_0, \dots, y_j .

A.2 Kalman Estimation

A recursive procedure for realization of the optimal linear estimator for the system (1), (2) has been developed by Kalman [3], [4]. The procedure consists of two stages. The first stage employs a filter algorithm, while the second uses a smoother algorithm**.

* The notation $||z||^2 \equiv z^T z$ denotes the ordinary Euclidean norm of z .

**The terminology employed here is due to N. Wiener and has been adopted by R. Kalman. An estimator which estimates x_i given measurements with indices up to and including j is called a filter if $i = j$; it is called a predictor if $i > j$, and it is called a smoother if $i < j$.

Application of the Kalman filter to the system (1), (2) yields $\hat{x}(i|i) \equiv \hat{x}_i$ and $P(i|i) \equiv P_i$ by recursion. The algorithm is

$$(3) \quad K_i = P_i^- H_i^T [H_i P_i^- H_i^T + R_i]^{-1},$$

$$(4) \quad \hat{x}_i = \hat{x}_i^- + K_i (y_i - H_i \hat{x}_i^-),$$

$$(5) \quad P_i = P_i^- - K_i H_i P_i^-, \quad i = 0, 1, \dots, N,$$

$$(6) \quad \hat{x}_{i+1}^- = \Phi_i \hat{x}_i,$$

$$(7) \quad P_{i+1}^- = \Phi_i P_i \Phi_i^T + Q_i, \quad i = 0, 1, \dots, N-1.$$

The smoother algorithm is initiated when the filter stage is complete. The smoother uses the filter outputs in a recursive process, which runs in reverse order to the filter recursion, to compute $\hat{x}(i|N)$ and $P(i|N)$, $i = 0, 1, \dots, N$. The algorithm for smoothing is given by

$$(8) \quad A_i = P_i \Phi_i^T (P_{i+1}^-)^{-1},$$

$$(9) \quad \hat{x}(i|N) = \hat{x}_i + A_i [\hat{x}(i+1|N) - \hat{x}_{i+1}^-],$$

$$(10) \quad P(i|N) = P_i + A_i [P(i+1|N) - P_{i+1}^-] A_i^T, \quad i = N-1, \dots, 0.$$

A.3 Carlson Square Root Filter Formulation

There are several algorithms which are mathematically equivalent to the Kalman filter algorithm, but which recursively compute a square root S of the state covariance matrix P , rather than the covariance matrix itself [5], [6], [7]. In each of these algorithms, the condition

$$P = SS^T,$$

which is the defining relation for the square root of P , holds. These algorithms differ markedly because of the nonuniqueness of S . The distinctive feature of the Carlson algorithm is that S is maintained in triangular form.

The Carlson algorithm is initialized by applying the Cholesky decomposition and, if necessary, the Gram Schmidt process (Cf Appendix D) to P_0^- to obtain the appropriate (i.e., upper or lower) triangular form of S_0^- .

The Carlson algorithm will be given for the case where the measurement is a scalar. When the measurement is a vector, the Carlson algorithm can be applied sequentially to each scalar component of the measurement vector by the procedure developed in Section 6.2.

The measurement update relations for the Kalman filter are given by (3), (4), and (5) and the time update, or extrapolation, relations are given by (6) and (7). The corresponding update and extrapolate relations will now be developed for the Carlson square root filter.

Assume that \hat{x}_i^- , S_i^- , and y_i are given, where S_i^- is triangular and satisfies

$$P_i^- = S_i^- (S_i^-)^T .$$

Assume further that y_i is a scalar measurement of the form

$$y_i = h_i x_i + v_i ,$$

where v_i has mean zero and variance r_i . Then the Carlson update algorithm can be used to compute \hat{x}_i and S_i , where S_i is triangular and satisfies

$$S_i S_i^T = P_i \equiv P_i^- - K_i h_i P_i^- .$$

The Carlson update algorithm will be derived for the case in which S is maintained in upper triangular form. However, by a simple reversal of index order, this algorithm can also be used to maintain S in lower triangular form.

For notational brevity, the index i will be dropped. Thus

$$y = hx + v, \quad E(v) = 0, \quad C(v) = r,$$

and

$$P^- = S^-(S^-)^T.$$

Put

$$g = (S^-)^T h^T$$

and

$$\alpha = hP^-h^T + r = g^Tg + r.$$

If $\alpha = 0$, then $\hat{x} = \hat{x}^-$ and $S = S^-$, and the update process is complete. If $\alpha \neq 0$, it follows from (3) that

$$(11) \quad K = \left(\frac{1}{\alpha}\right)S^-g.$$

Then from (4)

$$(12) \quad \hat{x} = \hat{x}^- + K(y - h\hat{x}^-),$$

and from (5)

$$P = S^-(S^-)^T - \left(\frac{1}{\alpha}\right)S^-gg^T(S^-)^T,$$

or

$$(13) \quad P = S^{-1}[I - \alpha^{-1}gg^T](S^{-1})^T.$$

Now let U be an upper triangular square root of $[I - \alpha^{-1}gg^T]$, i.e.,

$$[I - \alpha^{-1}gg^T] = UU^T.$$

Then with

$$(14) \quad S = S^{-1}U,$$

it follows that

$$P = SS^T.$$

Since the product of two upper triangular matrices is itself upper triangular, it is clear that S is the desired form of the updated square root covariance.

All that remains to complete the development of the Carlson update procedure is to compute U . To this end let

$$g^T = (0, \dots, 0, g_{m+1}, \dots, g_n)$$

where $0 \leq m \leq n-1$. (If $g_i = 0$ for all i , then $U = I$, $K = 0$, $\hat{x} = \hat{x}^-$, and $S = S^-$.)

Then, by the Cholesky decomposition,

$$U = \begin{bmatrix} I_m & 0 \\ 0 & W \end{bmatrix},$$

where W is upper triangular and

$$w_{ii} = \sqrt{\frac{r + \sum_{j < i} g_j^2}{r + \sum_{j \leq i} g_j^2}}, \quad i = m+1, \dots, n,$$

$$w_{ki} = \frac{-g_k g_i}{\sqrt{(r + \sum_{j < i} g_j^2) (r + \sum_{j \leq i} g_j^2)}}, \quad k = m+1, \dots, i-1, \quad i = m+2, \dots, n.$$

Now let

$$\alpha_i = \alpha_{i-1} + g_i^2, \quad i = m+1, \dots, n, \quad \alpha_m = r,$$

$$\beta_i = \sqrt{\alpha_i}, \quad i = m, \dots, n,$$

and observe that

$$w_{ii} = \beta_{i-1} / \beta_i, \quad i = m+1, \dots, n,$$

$$w_{ki} = -g_k g_i / (\beta_{i-1} \beta_i), \quad k = m+1, \dots, i-1, \quad i = m+2, \dots, n,$$

and

$$\alpha = \alpha_n.$$

Define

$$c_i = \begin{cases} 1 & , i \leq m, \\ \beta_{i-1} / \beta_i & , i = m+1, \dots, n, \end{cases}$$

$$d_i = \begin{cases} 0 & , i \leq m+1, \\ g_i / (\beta_{i-1} \beta_i), & i = m+2, \dots, n, \end{cases}$$

$$C = \begin{bmatrix} c_1 & 0 & \dots & \dots & \dots & 0 \\ 0 & c_2 & \dots & \dots & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \dots & \dots & c_n \end{bmatrix},$$

$$D = \begin{bmatrix} d_1 & 0 & \dots & \dots & \dots & 0 \\ 0 & d_2 & \dots & \dots & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \dots & \dots & d_n \end{bmatrix},$$

and

$$G = \begin{bmatrix} 0 & g_1 & g_1 & \dots & \dots & g_1 \\ 0 & 0 & g_2 & \dots & \dots & g_2 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \dots & \dots & g_{n-1} \\ 0 & 0 & \dots & \dots & \dots & 0 \end{bmatrix}.$$

Then

$$U = C - GD.$$

Now with the column partition notation

$$S^- = [s_1^- | \dots | s_n^-],$$

$$S = [s_1 | \dots | s_n],$$

it follows from

$$S = S^-U = S^-C - S^-GD$$

that

$$s_i = c_i s_i^- - d_i \left(\sum_{j < i} g_j s_j^- \right).$$

Define

$$p_{i+1} = p_i + g_i s_i^-, \quad i = m+1, \dots, n, \quad p_i = 0, \quad i \leq m+1.$$

Then $s_i = c_i s_i^- - d_i p_i$, $i = 1, \dots, n$.

Moreover, since

$$p_{n+1} = \sum_{j=1}^n g_j s_j^- = S^- g,$$

and

$$\alpha_n = \alpha,$$

it follows from (11) that

$$K = \alpha_n^{-1} p_{n+1}.$$

From the above development, it is seen that the Carlson update procedure can be carried out with recursive operations on the columns of S . An auxiliary sequence of vectors $\{p_i, i = 1, \dots, n+1\}$ is computed, and the Kalman filter gain is given by $K = \alpha_n^{-1} p_{n+1}$.

The complete algorithm, valid for both upper and lower triangular forms is summarized below. For the upper triangular form $i_1=1$, $i_n=n$, and $i_d=+1$, while for the lower triangular form, $i_1=n$, $i_n=1$, and $i_d=-1$. Notice that the algorithm includes tests to prevent divisions, as well as certain multiplications, by zero.

Carlson Update Algorithm

1. $g^T \equiv (g_1, \dots, g_n) = hS^-$
2. $K = 0, i_p = 0$
3. $\alpha = r, \gamma = \sqrt{\alpha}$
4. $i = i_1$
5. IF ($r > 0$) GO TO 11
6. IF ($g_i^2 > 0$) GO TO 12
7. $s_i = s_i^-$
8. IF ($i = i_n$) GO TO 35
9. $i \leftarrow i + i_d$
10. GO TO 6
11. IF ($g_i^2 = 0$) PUT $s_i = s_i^-$ AND GO TO 21
12. $\alpha = r + g_i^2$
13. $\beta = \gamma$
14. $\gamma = \sqrt{\alpha}$
15. IF ($\beta > 0$) GO TO 18
16. $s_i = 0$
17. GO TO 20
18. $c = \beta/\alpha$
19. $s_i = c s_i^-$
20. $K = g_i s_i^-, i_p = 1$
21. IF ($i = i_n$) GO TO 34
22. $i \leftarrow i + i_d$
23. IF ($g_i^2 = 0$) PUT $s_i = s_i^-$ AND GO TO 33
24. $\alpha \leftarrow \alpha + g_i^2$
25. $\beta = \gamma$
26. $\gamma = \sqrt{\alpha}$

27. $c = \beta/\gamma$
28. $s_i = c s_i^-$
29. IF ($i_p = 0$) GO TO 32
30. $d = g_i/(B\gamma)$
31. $s_i \leftarrow s_i - dK$
32. $K \leftarrow K + g_i s_i^-$, $i_p = 1$
33. IF ($i \neq i_n$) GO TO 22
34. IF ($i_p = 1$) $K = \alpha^{-1} K$
35. $\hat{x} = \hat{x}^-$
36. IF ($i_p = 1$) $\hat{x} \leftarrow \hat{x}^- + K (y - h\hat{x}^-)$

To develop the extrapolation relations for the Carlson square root filter, assume that \hat{x}_i , S_i are given, where S_i is triangular and satisfies

$$P_i = S_i S_i^T.$$

Assume also that

$$x_{i+1} = \phi_i x_i + u_i,$$

where u_i has mean zero and covariance Q_i . Then $\hat{x}_{i+1}^- = \phi_i \hat{x}_i$, and all that is required of the Carlson extrapolate algorithm is to compute S_{i+1}^- , where S_{i+1}^- is triangular and satisfies

$$S_{i+1}^- (S_{i+1}^-)^T = P_{i+1}^- = \phi_i P_i \phi_i^T + Q_i.$$

The Carlson extrapolate procedure is based on the Gram-Schmidt orthogonalization procedure which is discussed in Appendix D. To illustrate the procedure, drop the subscripts for brevity and let the dimension of S and Q each be $n \times n$.

The first step in the procedure is to apply the Cholesky decomposition to obtain a square root of Q , i.e.,

$$Q = \Gamma \Gamma^T.$$

Then the $n \times 2n$ matrix

$$[\phi S \mid \Gamma]$$

is formed. Finally, an orthogonal $2n \times 2n$ matrix T is determined such that

$$(15) \quad [S^- \mid 0] = [\phi S \mid \Gamma] T$$

where S^- is an $n \times n$ triangular matrix. Since

$$\begin{aligned}
S^-(S^-)^T &= [\phi S \mid \Gamma] T T^T \begin{bmatrix} S^T \phi^T \\ \Gamma^T \end{bmatrix} \\
&= [\phi S \mid \Gamma] \begin{bmatrix} S^T \phi^T \\ \Gamma^T \end{bmatrix} \\
&= \phi S S^T \phi^T + \Gamma \Gamma^T = \phi P \phi^T + Q,
\end{aligned}$$

it is clear that S^- is the desired form of the extrapolated square root covariance.

Thus all that remains is to construct an orthogonal matrix T which satisfies (15). This is accomplished by application of the Gram-Schmidt process to the rows of $[\phi S \mid \Gamma]$, augmented as necessary by the rows of the $2n \times 2n$ identity matrix I_{2n} , to obtain a set of $2n$ orthonormal vectors. These orthonormal vectors then form the columns of T .

If S^- is to be upper triangular, the Gram-Schmidt process is applied to the rows of

$$\begin{bmatrix} I_n & 0 \\ 0 & I_n \\ \phi S & \Gamma \end{bmatrix}$$

in the order from bottom to top, until a complete orthonormal set is obtained*, to form the columns of T in the order $n, \dots, 1, n+1, \dots, 2n$.

If S^- is to be lower triangular, the orthonormalization procedure is applied to the rows of

$$\begin{bmatrix} \phi S & \Gamma \\ I_n & 0 \\ 0 & I_n \end{bmatrix}$$

* Linearly dependent rows are simply skipped over.

in the order from top to bottom, until a complete set is obtained, to form the columns of T in the order from left to right.

That T satisfies (15) follows by construction. Also, it should be noted that at most, n columns of T need be computed since the products involving the remaining columns of T in (15) are all zero.

B. VARIATIONAL FORM OF THE NAVIGATION EQUATIONS

In this appendix, the variational form of the navigation equations with three degrees of freedom (DOF) will be developed. The reference coordinate system is taken as earth fixed (EF), and the 3 DOF are the EF position coordinates of the navigation system. Extension of the results in this appendix to 6 DOF can be readily accomplished by augmenting the three position coordinates with the three Euler angles which express the spatial attitude of the navigation system with respect to the EF coordinate frame. Extension to 6 DOF is required when the navigation system is of the "strap-down" variety commonly used to provide on-board instrumentation data during reentry.

Let P denote the position vector of the navigation system relative to the earth center of mass. Denote by $\left[\frac{d}{dt}\right]_I$ and $\left[\frac{d}{dt}\right]_E$, the time derivatives with respect to inertial and earth fixed frames, respectively. Then

$$(1) \quad \left[\frac{d^2}{dt^2}\right]_I P = A_G(P) + A_F$$

where $A_G(P)$ is the acceleration due to gravity at P , and A_F is the acceleration of the navigation system due to all forces except gravity.

The velocity vector of the navigation system relative to the earth is defined by

$$(2) \quad V \equiv \left[\frac{d}{dt}\right]_E P \equiv \dot{P}.$$

But

$$\begin{aligned} \left[\frac{d}{dt}\right]_I P &= \left[\frac{d}{dt}\right]_E P + \omega \times P \\ &= V + \omega \times P, \end{aligned}$$

where ω is the angular velocity vector of the earth with respect to an inertial frame.

Consequently

$$\begin{aligned} (3) \quad \left[\frac{d^2}{dt^2} \right]_I P &= \left[\frac{d}{dt} \right]_I (V + \omega \times P) \\ &= \left[\frac{d}{dt} \right]_E (V + \omega \times P) + \omega \times (V + \omega \times P) \\ &= \dot{V} + 2\omega \times V + \omega \times (\omega \times P) \\ &= A + A_R(P, V) , \end{aligned}$$

where

$$A \equiv \left[\frac{d}{dt} \right]_E V \equiv \dot{V}$$

and

$$A_R(P, V) \equiv 2\omega \times V + \omega \times (\omega \times P) .$$

The quantity $A_R(P, V)$, due to earth rotation, is the sum of Coriolis and centripetal accelerations.

Now from (1), (2), and (3), the navigation system equations of motion (EOM) are expressed by

$$\begin{aligned} \dot{P} &= V \\ \dot{V} &= A_G(P) - A_R(P, V) + A_F , \end{aligned}$$

or

$$\begin{aligned} (4) \quad \dot{P} &= V \\ \dot{V} &= A_O(P, V) + A_F , \end{aligned}$$

where

$$A_O(P, V) = A_G(P) - A_R(P, V) .$$

B.1 Boost Phase

Let t_0 denote the time at which the navigation system is initialized. Then the solution of (4) can be expressed by

$$(5) \quad \begin{bmatrix} P(t) \\ V(t) \end{bmatrix} = \int_{t_0}^t \begin{bmatrix} V(t') \\ A_0[P(t'), V(t')] \end{bmatrix} dt' + \begin{bmatrix} 0 \\ V_F(t) \end{bmatrix} + \begin{bmatrix} P(t_0) \\ V(t_0) \end{bmatrix}$$

where

$$V_F(t) \equiv \int_{t_0}^t A_F(t') dt' .$$

The navigation system constructs a solution of the form given in (5) using outputs from an inertial measuring unit (IMU) which senses A_F and a clock which measures time.

The acceleration sensed by the IMU, denoted by A_{FN} , is given by

$$A_{FN}(t) = A_F(t) + M[A_F(\cdot), t]b_I$$

where b_I is the vector of IMU error coefficients* and M is a matrix which is a function of t and a functional of $\{A_F(\tau), t_0 \leq \tau \leq t\}$. But if the IMU accelerometers are of the integrating variety, the output of the IMU is sensed velocity which is given by

$$V_{FN}(t) = \int_{t_0}^t A_{FN}(t') dt' + b_0 + r(t) ,$$

where b_0 and $r(t)$ are the respective bias and zero mean random components of instrument output error. Thus for integrating accelerometers, the IMU output is given by

$$(6) \quad V_{FN}(t) = V_F(t) + b_0 + \int_{t_0}^t M[A_F(\cdot), t'] dt' b_I + r(t) .$$

*The components of b_I include coefficients of initial platform misalignment error, uncompensated gyro precession, and uncompensated accelerometer error.

The time measured by the navigation clock is given by

$$(7) \quad \Theta_N(t) = t + \Delta t_N(t) = (1 + \Delta t_N) \circ t$$

where t denotes true time (i.e., time indicated by an earth fixed master clock) and $\Delta t_N(t)$ is the navigation clock error. This error can be expressed in the form

$$(8) \quad \Delta t_N(t) = m^T(t) b_c,$$

where b_c is the vector of clock error coefficients and m is a vector function of t .

If the IMU and navigation clock outputs are given at discrete times t_i , $i = 0, 1, 2, \dots$, a navigation solution of the EOM can be obtained by numerical integration. Similarly a nominal solution (about which the variational equations are obtained) can be obtained by applying nominal corrections to the IMU and clock outputs prior to integration. Assuming the integrator truncation error is negligible, the nominal solution is expressed by

$$(9) \quad \begin{bmatrix} \tilde{P}(t) \\ \tilde{V}(t) \end{bmatrix} = \int_{t_0}^t \begin{bmatrix} \tilde{V}(t') \\ \tilde{A}_0[\tilde{P}(t'), \tilde{V}(t')] \end{bmatrix} \tilde{t}(t') dt' + \begin{bmatrix} 0 \\ \tilde{V}_F(t) \end{bmatrix} + \begin{bmatrix} \tilde{P}(t_0) \\ \tilde{V}(t_0) \end{bmatrix},$$

where \tilde{V}_F and \tilde{t} denote the nominal corrected values of V_{FN} and Θ_N , respectively, and \tilde{A}_0 denotes A_0 evaluated using the nominal geopotential model. Observe that

$$\tilde{t}(t) = (1 + \Delta t_N)^{-1} \circ \Theta_N(t),$$

or, approximately,

$$\tilde{t}(t) \approx t + m^T(t) \delta b_c,$$

and thus

$$\tilde{t}(t) \cong 1 + \dot{m}^T(t) \delta b_c .$$

Also, note that

$$\tilde{V}_F(t) \cong V_F(t) + \delta b_0 + \int_{t_0}^t M[\tilde{A}_F(\cdot), t'] dt' \delta b_I + \delta r(t) .$$

The geopotential function can be expressed by

$$(10) \quad A_G(P, c) = A_G(P, \tilde{c}) + \left(\frac{\partial A_G}{\partial c^T} \right)_c (c - \tilde{c}) ,$$

where c is the vector of true geopotential coefficients and \tilde{c} is the value of these coefficients used in the nominal solution. Thus

$$\tilde{A}_0(P, V) \cong A_G(P, \tilde{c}) - A_R(P, V) ,$$

and

$$A_0(P, V) \cong A_G(P, c) - A_R(P, V) = \tilde{A}_0(P, V) + \left(\frac{\partial A_G}{\partial c^T} \right)_c \delta c .$$

The solution expressed by (9) is called the on-board nominal solution. In order to relate this solution to off-board measurements, it is necessary to use the navigation clock (corrected for nominal errors) and construct an off-board nominal solution. The off-board solution is given by

$$(11) \quad \begin{bmatrix} \tilde{P}'(t) \\ \tilde{V}'(t) \end{bmatrix} = \begin{bmatrix} \tilde{P}[\tilde{t}^{-1}(t)] \\ \tilde{V}[\tilde{t}^{-1}(t)] \end{bmatrix} , \quad t \geq \tilde{t}(t_0) .$$

The variational equations will be developed exclusively for the on-board solution, and (11) will be used to relate on-board trajectory variations to off-board measurement variations.

In order to obtain the variational equations, (9) is subtracted from (5) and the right hand side of the resulting difference is expanded to first order about the nominal. The variational equations thus obtained are given by

$$\begin{aligned}
 (12) \quad \begin{bmatrix} \delta P(t) \\ \delta V(t) \end{bmatrix} &= \int_{t_0}^t \left\{ \begin{bmatrix} 0 & I \\ \frac{\partial \tilde{A}_0}{\partial P^T} & \frac{\partial \tilde{A}_0}{\partial V^T} \end{bmatrix} \begin{bmatrix} \delta P(t') \\ \delta V(t') \end{bmatrix} \right. \\
 &+ \begin{bmatrix} 0 \\ \frac{\partial A_G}{\partial c^T} \end{bmatrix} \tilde{P}(t'), \tilde{c} \delta c - \begin{bmatrix} \tilde{V}(t') \\ \tilde{A}_0[\tilde{P}(t'), \tilde{V}(t')] \end{bmatrix} \dot{m}^T(t') \delta b_c \quad dt' \\
 &- \begin{bmatrix} 0 \\ I \end{bmatrix} \delta b_0 - \begin{bmatrix} 0 \\ \int_{t_0}^t M[\tilde{A}_F(\cdot), t'] dt' \end{bmatrix} \delta b_I - \begin{bmatrix} 0 \\ \delta r(t) \end{bmatrix} \\
 &+ \begin{bmatrix} \delta P(t_0) \\ \delta V(t_0) \end{bmatrix} .
 \end{aligned}$$

The above equation expresses the trajectory variations in integral form. Differentiation of both sides and augmentation with the error coefficient variations yields the complete set of navigation variational equations in differential form. Thus

$$(13) \quad \begin{bmatrix} \dot{\delta P} \\ \dot{\delta V} \\ \dot{\delta b}_0 \\ \dot{\delta b}_c \\ \dot{\delta b}_I \\ \dot{\delta c} \end{bmatrix} = \begin{bmatrix} 0 & I & 0 & -\tilde{V}m^T & 0 & 0 \\ \frac{\partial \tilde{A}_0}{\partial P^T} & \frac{\partial \tilde{A}_0}{\partial V^T} & 0 & -\tilde{A}_0 \dot{m}^T & -\tilde{M} & \frac{\partial \tilde{A}_G}{\partial c^T} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta P \\ \delta V \\ \delta b_0 \\ \delta b_c \\ \delta b_I \\ \delta c \end{bmatrix} - \begin{bmatrix} 0 \\ \dot{\delta r}(t) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Equation (13) is exact only for the case where V_{FN} is provided continuously with respect to time. When V_{FN} is given only at discrete times, the form of \tilde{M} and \dot{r} will depend on the method of interpolation of V_{FN} used in integrating the EOM. For example, suppose V_{FN} is given at t_0, t_1, t_2, \dots , and linear interpolation is used in the intervals $[t_i, t_{i+1}]$, $i = 0, 1, 2, \dots$. That is, for $t_i \leq t < t_{i+1}$, $i = 0, 1, 2, \dots$,

$$(14) \quad \tilde{V}_F(t) = \tilde{V}_F(t_i) + \left[\frac{\tilde{t}(t) - \tilde{t}(t_i)}{\tilde{t}(t_{i+1}) - \tilde{t}(t_i)} \right] [\tilde{V}_F(t_{i+1}) - \tilde{V}_F(t_i)] .$$

Then ignoring velocity interpolation error,

$$\begin{aligned} \delta \tilde{V}_F &= \left[\frac{\dot{\tilde{t}}(t)}{\tilde{t}(t_{i+1}) - \tilde{t}(t_i)} \right] \left[\int_{t_i}^{t_{i+1}} M[\tilde{A}_F(\cdot), t'] dt' \delta b_I + \delta r(t_{i+1}) - \delta r(t_i) \right] \\ &= \left(\frac{1}{t_{i+1} - t_i} \right) \left[\int_{t_i}^{t_{i+1}} M[\tilde{A}_F(\cdot), t'] dt' \delta b_I + \delta r(t_{i+1}) - \delta r(t_i) \right] , \end{aligned}$$

and therefore, in (13) it follows that the substitutions

$$(15) \quad \tilde{M} \leftarrow \left(\frac{1}{\Delta t_i} \right) \int_{t_i}^{t_{i+1}} M[\tilde{A}_F(\cdot), t'] dt'$$

and

$$(16) \quad \dot{\delta r} \leftarrow \left(\frac{1}{\Delta t_i} \right) [\delta r(t_{i+1}) - \delta r(t_i)] ,$$

where $\Delta t_i \equiv t_{i+1} - t_i$, must be made for all t in the interval $[t_i, t_{i+1}]$.

Suppose also that the navigation system is initialized while the vehicle is at rest on the launch pad. In this case

$$V_{FN}(t_0) = V_F(t_0) = 0 ,$$

and thus

$$r(t_0) = -b_0 .$$

Furthermore, if $\tilde{V}_F(t_0) = 0$,

$$\delta r(t_0) = -\delta b_0.$$

Now define the state noise process

$$u_i = r(t_{i+1}), \quad i = 0, 1, 2, \dots,$$

and introduce the correlated noise states ξ and η defined by:

$$\dot{\xi}(t) = 0,$$

$$\dot{\eta}(t) = 0,$$

for

$$t_i < t < t_{i+1}, \quad \text{and}$$

$$\begin{bmatrix} \xi(t_i^+) \\ \eta(t_i^+) \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \xi(t_i^-) \\ \eta(t_i^-) \end{bmatrix} + \begin{bmatrix} 0 \\ u_i \end{bmatrix},$$

$i = 0, 1, 2, \dots$, with initial condition

$$\begin{bmatrix} \xi(t_0^-) \\ \eta(t_0^-) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Then from (16)

$$\delta r(t) = \begin{cases} \left(\frac{1}{\Delta t_i}\right) [\delta \eta(t) - \delta \xi(t)], & t_i < t < t_{i+1}, \quad i = 1, 2, \dots, \\ \left(\frac{1}{\Delta t_0}\right) [\delta \eta(t) + \delta b_0], & t_0 < t < t_1. \end{cases}$$

Thus, augmenting the variational equations with the correlated noise states, it follows that (13) can be expressed by:

$$(17) \quad \begin{bmatrix} \dot{\delta P} \\ \dot{\delta V} \\ \dot{\delta \xi} \\ \dot{\delta \eta} \\ \dot{\delta b}_o \\ \dot{\delta b}_c \\ \dot{\delta b}_I \\ \dot{\delta c} \end{bmatrix} = \begin{bmatrix} 0 & I & 0 & 0 & 0 & -\tilde{V}_m^T & 0 & 0 \\ \frac{\partial A_o}{\partial P^T} & \frac{\partial A_o}{\partial V^T} & \left(\frac{1-\delta_{io}}{\Delta t_i}\right)I & -\left(\frac{1}{\Delta t_i}\right)I & -\left(\frac{\delta_{io}}{\Delta t_o}\right)I & -\tilde{A}_o^T & -\tilde{M} & \frac{\partial A_G}{\partial c^T} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta P \\ \delta V \\ \delta \xi \\ \delta \eta \\ \delta b_o \\ \delta b_c \\ \delta b_I \\ \delta c \end{bmatrix}$$

for $t_i < t < t_{i+1}$, where

$$\delta_{io} = \begin{cases} 0, & i = 1, 2, \dots, \\ 1, & i = 0, \end{cases}$$

together with

$$(18) \quad \begin{bmatrix} P(t_i^+) \\ V(t_i^+) \\ \xi(t_i^+) \\ \eta(t_i^+) \\ b_o(t_i^+) \\ b_c(t_i^+) \\ b_I(t_i^+) \\ c(t_i^+) \end{bmatrix} = \begin{bmatrix} I & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I \end{bmatrix} \begin{bmatrix} P(t_i^-) \\ V(t_i^-) \\ \xi(t_i^-) \\ \eta(t_i^-) \\ b_o(t_i^-) \\ b_c(t_i^-) \\ b_I(t_i^-) \\ c(t_i^-) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} u_i,$$

for $i = 0, 1, \dots$, with initial condition

$$(19) \quad \begin{bmatrix} P(t_0^-) \\ V(t_0^-) \\ \xi(t_0^-) \\ \eta(t_0^-) \\ b_0(t_0^-) \\ b_c(t_0^-) \\ b_I(t_0^-) \\ c(t_0^-) \end{bmatrix} = \begin{bmatrix} P(t_0) \\ V(t_0) \\ 0 \\ 0 \\ b_0 \\ b_c \\ b_I \\ c \end{bmatrix} .$$

If u_i , $i = 0, 1, 2, \dots$, is a sequence of uncorrelated random vectors, then equations (17), (18), and (19) provide the desired structure for application of the linear estimation results discussed in Appendix A and Section 5.0. If, on the other hand, the sequence of random vectors is correlated, additional correlated noise states must be augmented to the system given by (17), (18), and (19) in order to obtain the structure required for the estimator to be optimal.

B.2 Free-Fall Phase

The position and velocity of an RV at deployment can be computed from (9) using vehicle Euler angles and their rates together with a separation model based on energy and momentum relations. Following release, the position and velocity of the RV can be computed using an equation analogous to (9) with $\tilde{V}_F = 0$. Thus if t_d is the time of deployment,

$$(20) \quad \begin{bmatrix} \tilde{P}_{RV}(t) \\ \tilde{V}_{RV}(t) \end{bmatrix} = \int_{t_d}^t \begin{bmatrix} \tilde{V}_{RV}(t') \\ \tilde{A}_O[\tilde{P}_{RV}(t'), \tilde{V}_{RV}(t')] \end{bmatrix} dt' + \begin{bmatrix} \tilde{P}_{RV}(t_d) \\ \tilde{V}_{RV}(t_d) \end{bmatrix}, \quad t \geq t_d .$$

Now (20) is the nominal on-board solution for the RV. The only timing error associated with this solution is that which exists at t_d . Thus the nominal off-board solution for the RV is given by

$$(21) \quad \begin{bmatrix} \tilde{P}'_{RV}(t) \\ \tilde{V}'_{RV}(t) \end{bmatrix} = \begin{bmatrix} \tilde{P}_{RV}(t + t_d - \tilde{t}(t_d)) \\ \tilde{V}_{RV}(t + t_d - \tilde{t}(t_d)) \end{bmatrix}, \quad t \geq t(t_d) .$$

The variational equations for the RV on-board solution are given in differential form by

$$(22) \quad \begin{bmatrix} \dot{\delta P}_{RV} \\ \dot{\delta V}_{RV} \\ \dot{\delta c} \end{bmatrix} = \begin{bmatrix} 0 & I & 0 \\ \frac{\partial \tilde{A}_O}{\partial P_{RV}^T} & \frac{\partial \tilde{A}_O}{\partial V_{RV}^T} & \frac{\partial \tilde{A}_G}{\partial c^T} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta P_{RV} \\ \delta V_{RV} \\ \delta c \end{bmatrix}, \quad t \geq t_d ,$$

with

$$(23) \quad \begin{bmatrix} \delta P_{RV}(t_d) \\ \delta V_{RV}(t_d) \end{bmatrix} = \begin{bmatrix} \delta P(t_d) \\ \delta V(t_d) \end{bmatrix} + \begin{bmatrix} \epsilon_p \\ \epsilon_v \end{bmatrix},$$

where ϵ_p and ϵ_v are due to attitude, attitude rate, and separation errors at deployment.

B.3 Reentry Phase with On-Board Instrumentation

If the RV includes on-board instrumentation for use during reentry, the procedure used for the boost navigation system can be used to develop nominal and variational equations for the reentry navigation system. These equations will be initialized by the free fall solution at or near pierce point, and the initialization procedure will be subject to both the free fall timing error and the reentry navigation clock error.

The time measured by the reentry clock is given by

$$\Theta_R(t) = t + \Delta t_R(t) = (1 + \Delta t_R) \cdot t ,$$

where t is true time. Correction of the reentry clock for nominal errors yields

$$\tilde{t}_R(t) = (1 + \Delta \tilde{t}_R)^{-1} \cdot \Theta_R(t) ,$$

and if

$$\Delta t_R(t) = m_R^T(t) b_{RC} ,$$

then

$$\tilde{t}_R(t) \cong t + m_R^T(t) \delta b_{RC} .$$

The nominal on-board reentry solution is initialized at time t_p using the nominal off-board freefall solution as follows.

$$(24) \quad \begin{bmatrix} \tilde{P}_R(t_p) \\ \tilde{V}_R(t_p) \end{bmatrix} = \begin{bmatrix} \tilde{P}'_{RV}(\tilde{t}_R(t_p)) \\ \tilde{V}'_{RV}(\tilde{t}_R(t_p)) \end{bmatrix} .$$

But from (21)

$$\begin{bmatrix} \tilde{P}'_{RV}(\tilde{t}_R(t_p)) \\ \tilde{V}'_{RV}(\tilde{t}_R(t_p)) \end{bmatrix} = \begin{bmatrix} \tilde{P}_{RV}(\tilde{t}_R(t_p) + t_d - \tilde{t}(t_d)) \\ \tilde{V}_{RV}(\tilde{t}_R(t_p) + t_d - \tilde{t}(t_d)) \end{bmatrix} .$$

Therefore

$$\begin{aligned} \begin{bmatrix} \tilde{P}_R(t_p) \\ \tilde{V}_R(t_p) \end{bmatrix} &= \begin{bmatrix} \tilde{P}_{RV}(t_p) \\ \tilde{V}_{RV}(t_p) \end{bmatrix} + \begin{bmatrix} \dot{\tilde{P}}_{RV}(t_p) \\ \dot{\tilde{V}}_{RV}(t_p) \end{bmatrix} [\tilde{t}_R(t_p) - t_p + t_d - \tilde{t}(t_d)] \\ &= \begin{bmatrix} \tilde{P}_{RV}(t_p) \\ \tilde{V}_{RV}(t_p) \end{bmatrix} + \begin{bmatrix} \dot{\tilde{P}}_{RV}(t_p) \\ \dot{\tilde{V}}_{RV}(t_p) \end{bmatrix} (m_R^T(t_p)\delta b_{RC} - m^T(t_d)\delta b_C) . \end{aligned}$$

Consequently

$$(25) \quad \begin{bmatrix} \delta P_R(t_p) \\ \delta V_R(t_p) \end{bmatrix} = \begin{bmatrix} \delta P_{RV}(t_p) \\ \delta V_{RV}(t_p) \end{bmatrix} - \begin{bmatrix} \dot{\tilde{P}}_{RV}(t_p) \\ \dot{\tilde{V}}_{RV}(t_p) \end{bmatrix} (m_R^T(t_p)\delta b_{RC} - m^T(t_d)\delta b_C) .$$

Observe that (24) is used to initialize the nominal on-board reentry solution, while (25) is used to reset filter estimates and covariances (or square root covariances) at the outset of reentry.

Finally, in order to relate the reentry navigation solution to off-board sensor measurements, the nominal off-board reentry solution is required. This solution is given by

$$(26) \quad \begin{bmatrix} \tilde{P}'_R(t) \\ \tilde{V}'_R(t) \end{bmatrix} = \begin{bmatrix} \tilde{P}_R(\tilde{t}_R^{-1}(t)) \\ \tilde{V}_R(\tilde{t}_R^{-1}(t)) \end{bmatrix} , \quad t \geq \tilde{t}_R(t_p) .$$

C. METRIC SENSOR SYSTEMS

The diagram shown in Figure C.1 depicts the general structure of a metric sensor system, the signal flow through the system, and the sources and injection points of various errors. The sensor elements generally have rather narrow operational limits, and for this reason, it is necessary to control the sensor with a servo mechanism. The loop which includes the sensor elements, servo electronics, feedback elements (and in some cases the encoder) performs a tracking function which maintains the target within the sensor operating limits.

In an angle tracking system, the sensor elements consist typically of a monopulse receiver. The receiver senses the apparent displacement of the target relative to the receiver boresite (or tracking axis) in the form of two angles. The servo electronics and control elements contain amplifiers, filters, and antenna drive motors. The inputs to this block are the sensed angles out of the receiver, and the outputs are motor shaft angles. The feedback element is an antenna which is positioned by the motors, and the sensor input elements are rigidly mounted to the antenna. The motor shaft angles are encoded outside the loop, i.e., the encoders are not within the control loop.

In a range tracking system, the sensor element is typically a device which measures the time difference between the leading edge, centroid, or some other point on the received pulse, and a timing pulse output from the feedback element. The sensor element output is an analog signal which is input to the servo electronics and control elements. This block consists of amplifiers, filters, and integrator circuits. The output of this block is roughly proportional to target range and is digitized in the encoder which is typically inside the control loop. The feedback element produces a timing pulse delayed in time (relative to pulse transmission time) by an amount controlled by the encoder.

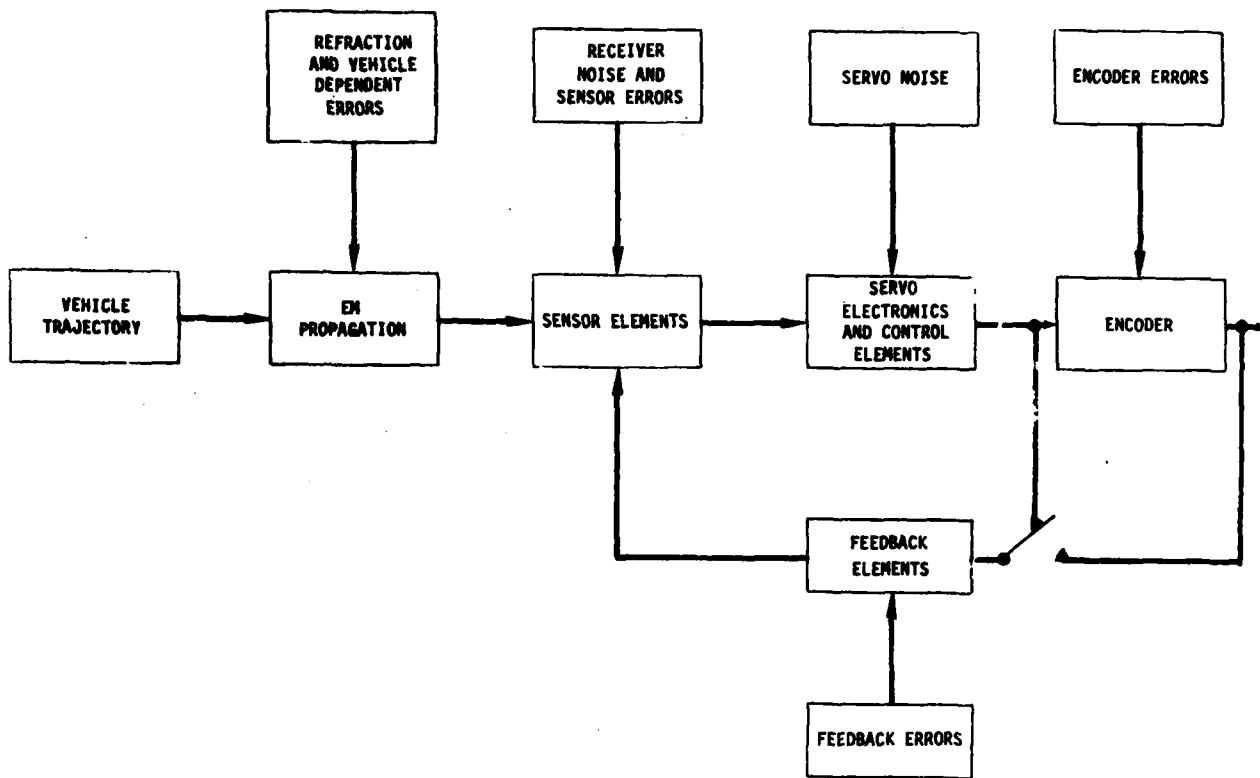


FIGURE C.1 METRIC SENSOR SYSTEM FLOW DIAGRAM

In a range rate or doppler system, the sensor element produces a signal with phase equal to the difference of the phases of the received signal and a frequency controlled oscillator in the feedback element. This phase modulated signal is input to the servo electronics and control block which consists of a frequency discriminator, amplifiers, filters, and integrators. The output of this block (which is roughly proportional to the doppler shift of the received signal relative to the transmitted signal) is digitized in the encoder. Then depending on whether the oscillator in the feedback element is controlled by an analog or digital signal, either the servo output or the encoder output is input to the feedback element.

In a pulse radar system, the optimal sensor element is a matched filter, but in practice the implementation is usually a suboptimal approximation to a matched filter. The output of the sensor element (regardless of whether it includes a matched filter) consists of a sequence of pulses which is synchronous with the received pulse sequence. This sequence of pulses contains target information corrupted by various systematic errors and sequentially uncorrelated noise. Since the information in this pulse sequence is relative to the sensor track point which is monitored by the encoder, it is necessary to use the encoder measurements to relate the sensor outputs to a reference coordinate system.

C.1 Metric Sensor Measurement Equations

From the foregoing discussion, it is clear that the sensor element (i.e., receiver) outputs in the range, doppler, and angle channels constitute the fundamental measurements of a metric sensor, while the encoders merely provide the means whereby these measurements are related to a reference coordinate system.

Consider a four channel tracking radar, and let y denote the measurement vector at time t . The components of y are simply the receiver outputs in range, doppler, and angles at t . Let τ denote the transit time of the signal which is received at t , and let t' denote the time when the signal left the target. Then

$$(1) \quad t' = t - \tau ,$$

and it is the target position and velocity at t' which ultimately affect the measurements made at time t .

Let the range, azimuth, and elevation of the target be defined with respect to the topocentric coordinate system located at the sensor as shown in Figure C.2. Now define apparent values of these spherical coordinates to be equal to true values modified by refraction.

Thus

$$(2) \quad R_A(t') = R(t') + \rho[R(t'), E(t')] ,$$

$$(3) \quad A_A(t') = A(t') ,$$

$$(4) \quad E_A(t') = E(t') + \epsilon[R(t'), E(t')] ,$$

where the apparent values are subscripted, true values are unsubscripted, and ρ and ϵ denote range and elevation refraction, respectively. Observe that for a specified refractivity between target and sensor, the apparent values of target position relative to the sensor are functions only of true position relative to the sensor. Note also that

$$(5) \quad c\tau = R_A(t') ,$$

and thus for a given t , (1), (2), and (5) together with the true target trajectory relative to the sensor suffice to determine transit time.

The apparent range rate is given by

$$(6) \quad \dot{R}_A(t') = \dot{R}(t') + \left(\frac{\partial \rho}{\partial (R, E)} \right) t' \begin{bmatrix} \dot{R}(t') \\ \dot{E}(t') \end{bmatrix} ,$$

and this quantity determines apparent doppler by

$$(7) \quad D_A(t') = \frac{-2f_0 \dot{R}_A(t')}{c + \dot{R}_A(t')} ,$$

where f_0 is the transmitter frequency.

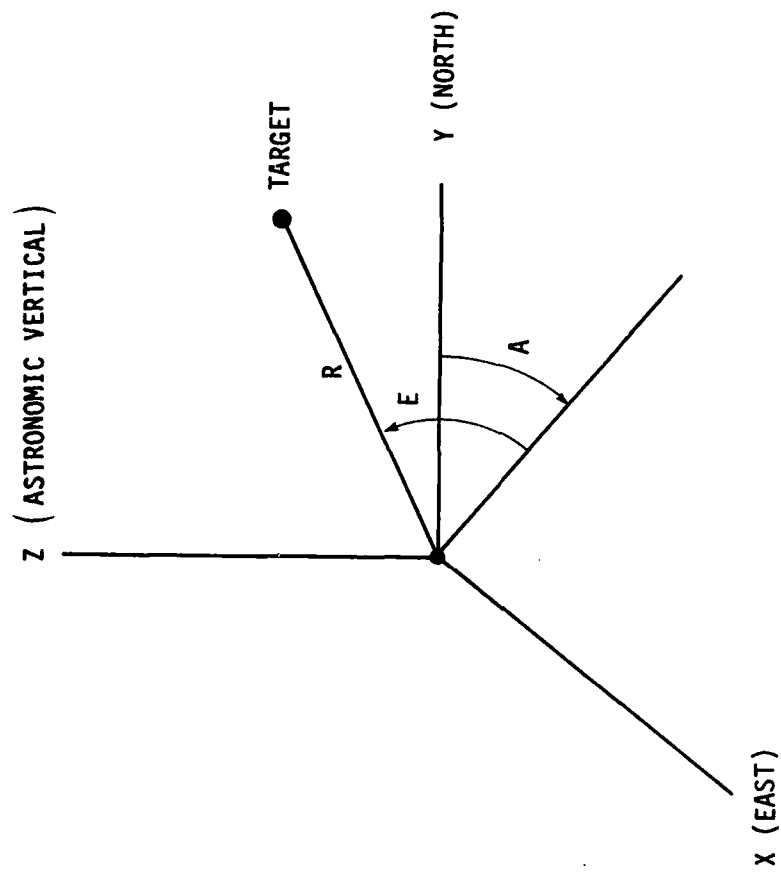


FIGURE C.2 SPHERICAL COORDINATES WITH RESPECT TO TOPOCENTRIC FRAME AT SENSOR

The apparent quantities given by equations (2), (3), (4), and (7), possibly corrupted by target dependent error, provide the inputs which generate y . Thus the input quantities of interest are the received values

$$(8) \quad R_R(t) = R_A(t') + \Delta R_T(t') \quad ,$$

$$(9) \quad A_R(t) = A_A(t') + \Delta A_T(t') \quad ,$$

$$(10) \quad E_R(t) = E_A(t') + \Delta E_T(t') \quad ,$$

$$(11) \quad D_R(t) = D_A(t') + \Delta D_T(t') \quad ,$$

where ΔR_T is due to beacon delay, ΔA_T and ΔE_T are due to phase front distortion, and ΔD_T is due to beacon oscillator drift.

In the range channel, the sensor element forms the difference

$$(12) \quad r_o(t) = R_R(t) - R_F(t) \quad ,$$

where $R_F(t)$ is the output of the feedback element. Assuming the range encoder is inside the tracking loop,

$$(13) \quad R_F(t) = R_E(t) - \Delta R_F(t) \quad ,$$

where $R_E(t)$ is the encoder output and $\Delta R_F(t)$ is the error introduced in the feedback path. If the feedback error is due only to a bias and a scale factor, then

$$(14) \quad \Delta R_F(t) = B_R + SF_R \cdot R_E(t) \quad .$$

Now the sensor element output consists of the range difference r_o corrupted by sensor errors and receiver noise. Thus

$$(15) \quad r_s(t) = r_o(t) + \Delta r_s(t) + v_r(t) \quad ,$$

where $\Delta r_s(t)$ is the error introduced by the sensor and $v_r(t)$ is the receiver noise error. If the sensor error is due only to a scale factor, then

$$(16) \quad \Delta r_s(t) = SF_r \cdot r_o(t) .$$

In the doppler channel the sensor element produces a signal with phase equal to the difference in the phase of the received signal and the phase of a frequency element. In an optimum mechanization, this phase difference would be measured directly and would constitute one component of the measurement vector. Typically, however, this phase modulated signal is filtered and input to a frequency discriminator which forms the difference:

$$(17) \quad d_o(t) = D_R(t) - D_F(t) ,$$

where $D_F(t)$ is the doppler modulation in the feedback element. Because of the filtering operation, the frequencies in (17) are not instantaneous quantities. Instead they are approximately equal to the average frequencies over a time interval determined by filter bandwidth. (Actually, if the filter and discriminator parameters are known, a state variable representation can be used to relate the frequency output given by (17) to the phase difference of the received signal and the feedback oscillator. This approach will not be taken here, however.)

Now assuming the doppler encoder is inside the tracking loop,

$$(18) \quad D_F(t) = D_E(t) - \Delta D_F(t) ,$$

where $D_E(t)$ is the encoder output and $\Delta D_F(t)$ is the error introduced in the feedback path. In general the feedback error may include bias and scale factor terms. However, because $D_F(t)$ in (17) is an average value due to filtering in the sensor element, $\Delta D_F(t)$ will also include a term which accounts for the average lag error of the feedback oscillator in response to inputs from the encoder. In a pulsed doppler radar, with period T_r , in which the feedback oscillator is reset only once each period, the average lag error* is approximately given by $T_r \dot{D}_R(t)/2$ [8]. Thus accounting for bias, scale factor, and lag errors,

*The existence of this error has been verified both experimentally and analytically for the SAMTEC C-Band systems.

$$(19) \quad \Delta D_F(t) = B_D + SF_D \cdot D_E(t) + \frac{T_r}{2} \dot{D}_R(t) .$$

The frequency discriminator output consists of d_0 corrupted by sensor errors and filtered receiver noise. Thus

$$(20) \quad d_s(t) = d_0(t) + \Delta d_s(t) + v_d(t) ,$$

where $\Delta d_s(t)$ is the error introduced by the sensor and $v_d(t)$ is receiver noise error. Assuming the sensor error is due only to a scale factor,

$$(21) \quad \Delta d_s(t) = SF_d \cdot d_0(t) .$$

In the angle channels, angles of arrival of the received signal with respect to the tracking axis (i.e., electronic boresite) are sensed. In order to develop expressions for the sensed angles, it is convenient to first define an electronic boresite coordinate system. This system is defined by a sequence of rotations applied to the locally level (i.e., topocentric) coordinate system located at the sensor. The transformation from the locally level system to the electronic boresite system is denoted by $C_{LL}^{EB}(t)$.

Assuming the angle encoders are outside the loop, $C_{LL}^{EB}(t)$ is a function of the encoder outputs, encoder errors, and feedback errors. This functional representation of $C_{LL}^{EB}(t)$ will now be developed.

The shaft angles of the tracking system are related to the encoder angles by

$$(22) \quad \begin{bmatrix} A_S(t) \\ E_S(t) \end{bmatrix} = \begin{bmatrix} A_E(t) \\ E_E(t) \end{bmatrix} - \begin{bmatrix} \Delta A_E(t) \\ \Delta E_E(t) \end{bmatrix} ,$$

where $\Delta A_E(t)$ and $\Delta E_E(t)$ are the encoder errors, which include bias and nonlinearity terms.

The azimuth shaft angle measures the antenna pedestal rotation in a plane which is determined by the North and East mislevel coefficients, μ_N and μ_E . These coefficients define the total mislevel, μ , and the azimuth angle of the mislevel axis, A_m , by means of

$$(23) \quad \begin{bmatrix} \mu_N \\ \mu_E \end{bmatrix} = \begin{bmatrix} \mu \cos A_m \\ \mu \sin A_m \end{bmatrix} .$$

The transformation from the locally level system to the mislevel system is then given by

$$(24) \quad C_{LL}^{ML} = \begin{bmatrix} \cos A_m & \sin A_m & 0 \\ -\sin A_m & \cos A_m & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \mu & 0 & -\sin \mu \\ 0 & 1 & 0 \\ \sin \mu & 0 & \cos \mu \end{bmatrix} \begin{bmatrix} \cos A_m & -\sin A_m & 0 \\ \sin A_m & \cos A_m & 0 \\ 0 & 0 & 1 \end{bmatrix} .$$

Now an azimuth shaft coordinate system is defined by a rotation of the mislevel system through the angle $A_S(\cdot)$.

This rotation is given by

$$(25) \quad C_{ML}^{AS}(t) = \begin{bmatrix} \cos A_S(t) & -\sin A_S(t) & 0 \\ \sin A_S(t) & \cos A_S(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} .$$

Next a nonorthogonality coordinate system is defined by a rotation of the azimuth shaft system through an angle, η , equal to the nonorthogonality of the elevation trunnion. This rotation is given by

$$(26) \quad C_{AS}^{NO}(t) = \begin{bmatrix} \cos \eta & 0 & -\sin \eta \\ 0 & 1 & 0 \\ \sin \eta & 0 & \cos \eta \end{bmatrix} .$$

Now an elevation shaft coordinate system is defined by a rotation of the nonorthogonality system through the elevation shaft angle $E_S(t)$. Thus

$$(27) \quad C_{NO}^{ES}(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos E_S(t) & \sin E_S(t) \\ 0 & -\sin E_S(t) & \cos E_S(t) \end{bmatrix} .$$

A mechanical boresight coordinate system is now defined in terms of misalignment angles in traverse and elevation, denoted by m_T and m_E respectively.

Thus

$$(28) \quad C_{ES}^{MB} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos m_E & \sin m_E \\ 0 & -\sin m_E & \cos m_E \end{bmatrix} \begin{bmatrix} \cos m_T & -\sin m_T & 0 \\ \sin m_T & \cos m_T & 0 \\ 0 & 0 & 1 \end{bmatrix} .$$

Next a gravity droop coordinate system is defined. The droop angle, δ is a function of the true elevation angle of the mechanical boresight, and it is a rotation about an axis which lies in the horizontal plane and is orthogonal to the mechanical boresight. The transformation from the mechanical boresight to the gravity droop system is given by

$$(29) \quad C_{MB}^{GB} = \begin{bmatrix} \cos \gamma & 0 & \sin \gamma \\ 0 & 1 & 0 \\ -\sin \gamma & 0 & \cos \gamma \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \delta & \sin \delta \\ 0 & -\sin \delta & \cos \delta \end{bmatrix} \begin{bmatrix} \cos \gamma & 0 & -\sin \gamma \\ 0 & 1 & 0 \\ \sin \gamma & 0 & \cos \gamma \end{bmatrix} ,$$

where γ and δ are determined as follows.

Define unit vectors e_x , e_y , e_v by

$$e_x = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad e_y = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad e_v = C_{LL}^{MB} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} .$$

If $|e_v \times e_y| = 0$, then $\delta = 0$, and $C_{MB}^{GD} = I$.

Otherwise, define

$$e_h = \frac{e_v \times e_y}{|e_v \times e_y|} .$$

Then

$$\begin{aligned} \cos \gamma &= e_x \cdot e_h , \\ \sin \gamma &= (e_x \times e_h) \cdot e_y . \end{aligned}$$

and the elevation angle of the mechanical boresite is given by

$$E_{MB} = \tan^{-1} \left(\frac{e_v \cdot e_y}{|e_v \times e_y|} \right),$$

with $-\pi/2 < e_{MB} < \pi/2$. Assuming a simple cantilever model for droop, it follows that

$$\delta = \delta_0 \cos E_{MB},$$

where δ_0 is the coefficient of droop.

Finally the electronic boresite coordinate system is defined in terms of electrical misalignment angles in traverse, elevation, and skew, denoted by ϵ_T , ϵ_E , and ϵ_S , respectively. Thus

$$(30) \quad C_{GD}^{EB} = \begin{bmatrix} \cos \epsilon_S & 0 & -\sin \epsilon_S \\ 0 & 1 & 0 \\ \sin \epsilon_S & 0 & \cos \epsilon_S \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \epsilon_E & \sin \epsilon_E \\ 0 & -\sin \epsilon_E & \cos \epsilon_E \end{bmatrix} \begin{bmatrix} \cos \epsilon_T & -\sin \epsilon_T & 0 \\ \sin \epsilon_T & \cos \epsilon_T & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The transformation from the locally level system to the electronic boresite system is given by the product of successive rotations. Thus

$$(31) \quad C_{LL}^{EB} = C_{GD}^{EB} C_{MB}^{GD} C_{ES}^{MB} C_{NO}^{ES} C_{AS}^{NO} C_{ML}^{AS} C_{LL}^{ML}.$$

Let $e(t)$ denote the unit vector defined by the angles of arrival of the received signal. Expressed in the topocentric, or locally level coordinate system, this unit vector is simply

$$(32) \quad e_{LL}(t) = \begin{bmatrix} \sin A_R(t) & \cos E_R(t) \\ \cos A_R(t) & \cos E_R(t) \\ & \sin E_R(t) \end{bmatrix}.$$

Therefore, in the electronic boresite system this unit vector is given by

$$(33) \quad \mathbf{e}_{EB}(t) = \begin{bmatrix} e_{EB}^x(t) \\ e_{EB}^y(t) \\ e_{EB}^z(t) \end{bmatrix} = C_{LL}^{EB}(t) \begin{bmatrix} \sin A_R(t) & \cos E_R(t) \\ \cos A_R(t) & \cos E_R(t) \\ & \sin E_R(t) \end{bmatrix} .$$

Now, define the discriminator traverse and elevation angles in the electronic boresite system by

$$(34) \quad \phi_0(t) = \tan^{-1}[e_{EB}^x(t)/e_{EB}^y(t)]$$

and

$$(35) \quad \psi_0(t) = \tan^{-1}[e_{EB}^z(t)/e_{EB}^y(t)] ,$$

respectively. Observe, however, that these angles may be approximated by

$$(36) \quad \phi_0(t) = e_{EB}^x(t)$$

and

$$(37) \quad \psi_0(t) = e_{EB}^z(t) ,$$

where the truncation error is third order. Consequently, if ϕ_0 and ψ_0 are less than ten milliradians, the errors in these approximations are roughly one microradian or less.

The outputs of the sensor element are given by ϕ_0 and ψ_0 corrupted by sensor errors and receiver noise. Assuming the sensor errors consist of scale factor and crosstalk, the sensed angles are given by

$$(38) \quad \begin{bmatrix} \phi_s(t) \\ \psi_s(t) \end{bmatrix} = \begin{bmatrix} \phi_0(t) \\ \psi_0(t) \end{bmatrix} + \begin{bmatrix} SF_{\phi\phi} & SF_{\phi\psi} \\ SF_{\psi\phi} & SF_{\psi\psi} \end{bmatrix} \begin{bmatrix} \phi_0(t) \\ \psi_0(t) \end{bmatrix} + \begin{bmatrix} v_\phi(t) \\ v_\psi(t) \end{bmatrix} ,$$

where $SF_{\phi\phi}$ and $SF_{\psi\psi}$ are scale factor errors, $SF_{\phi\psi}$ and $SF_{\psi\phi}$ are cross talk errors, and $v_\phi(t)$ and $v_\psi(t)$ are receiver noise errors.

C.2 Nominal and Variational Measurement Equations

The measurement equations for a four channel tracking radar were developed in the preceding subsection. If the sensor element outputs at time t are denoted by $r_s(t)$, $\phi_s(t)$, $\psi_s(t)$, and $d_s(t)$ in range, traverse, elevation, and doppler, respectively, then the measurement vector is defined by

$$(1) \quad y(t) = \begin{bmatrix} r_s(t) \\ \phi_s(t) \\ \psi_s(t) \\ d_s(t) \end{bmatrix} .$$

By introducing appropriate state variables for vehicle dynamics and the various vehicle and measurement related error sources, the measurement vector at time t can be expressed in the following algorithmic form:

$$(2) \quad t = \theta_s(t) - m_s^T(t)x_d$$

$$(3) \quad t' = t - \tau$$

$$(4) \quad c\tau = R_A^i(t')$$

$$(5) \quad y(t) = h[x_a^i(t'), x_b, x_c, \dots, x_g, z(t)] + v(t) .$$

The state variables which appear in the algorithm are defined below.

- x_a^i - vehicle trajectory states (off-board)
- x_b - vehicle dependent measurement error states
- x_c - sensor measurement error states
- x_d - sensor timing error states
- x_e - vehicle timing error states
- x_f - refraction profile states
- x_g - sensor survey states (coordinates of geodetic position and astronomic vertical)

In equation (2), $\theta_s(t)$ denotes the sensor clock indicated time at t , and $m_s(t)$ is a known vector function of t .

Equation (3) expresses the relation between the measurement time t , transit time τ , and the retarded measurement time t' .

The transit time constraint is expressed in equation (4), where the apparent range $R'_A(t')$ is derived from $x'_a(t')$, x_f , and x_g .

Finally, the measurement $y(t)$ is expressed by (5), in which $z(t)$ is the encoder measurement vector at t , and $v(t)$ is the measurement noise vector at t .

The complete set of equations, (2) through (5), constitutes the dynamic measurement algorithm, while equation (5), in and of itself, is called the static measurement algorithm.

The sequence of operations in the dynamic algorithm is as follows. First the sensor clock measurement is corrected to obtain t from equation (2). Next equations (3) and (4) are solved iteratively, using the off-board trajectory x'_a to determine τ , t' , and $x'_a(t')$. Finally, the measurement vector $y(t)$ is obtained from equation (5).

If nominal values of the state variables (including a nominal off-board trajectory x'_a) are specified, the dynamic measurement algorithm can be used to obtain the nominal measurement $y(t)$, computed with $v(t) \equiv 0$, and the partial derivatives which appear in the measurement variation equation:

$$(6) \quad \delta y(t) = \frac{\partial y}{\partial x_a} \delta x_a(t') + \frac{\partial y}{\partial x_b} \delta x_b + \dots + \frac{\partial y}{\partial x_g} \delta x_g + v(t) ,$$

where $\delta y(t) \equiv y(t) - \tilde{y}(t)$, and the partial derivatives are evaluated at the nominal state.

The relationship which exists between the nominal off-board and on-board trajectories of Appendix B, denoted respectively by \tilde{x}'_a and \tilde{x}_a , can be expressed in terms of the vehicle timing variation. Let $\theta_v(t')$ be the

vehicle clock indicated time at t' , and suppose that

$$(7) \quad \theta_V(t') = t' + m_V^T(t')x_e,$$

where $m_V(t')$ is a known vector function of t' . Then if \tilde{t}' is the nominal value of t' obtained from the measurement algorithm, it follows that

$$(8) \quad \tilde{x}'_a(\tilde{t}') = \tilde{x}_a(\tilde{t}' - m_V^T(\tilde{t}')\delta x_e).$$

Observe that equation (8) indicates the mechanism by which vehicle timing variation enters the measurement variation equation.

Now, the partial derivatives in equation (6) can be obtained readily by numerical differentiation using the dynamic measurement algorithm, and in many instances this is the preferred method. However, it is possible, and sometimes more convenient, to express the measurement partial derivatives in terms of partial derivatives of the function h in equation (5). The partial derivatives of h with respect to the state variable elements are called static measurement partial derivatives, and, when clarification is desired, the partial derivatives in equation (6) are called dynamic measurement partial derivatives.

The remainder of this subsection is devoted to expressing the dynamic measurement partial derivatives in terms of the static measurement partial derivatives. The latter can of course be obtained by numerical differentiation using the static measurement algorithm.

In the calculation of measurement partial derivatives it is most useful to adopt the viewpoint that these derivatives are of $\tilde{y}(t)$ with respect to $\tilde{x}_a(t')$, \tilde{x}_b , ..., \tilde{x}_g , rather than derivatives of $y(t)$ with respect to $x_a(t')$, x_b , ..., x_g , evaluated at the nominal state. (These alternative viewpoints are entirely equivalent.) Also, for the sake of notational brevity, the ($\tilde{\quad}$) notation will be dropped in the sequel with the understanding that it applies to all quantities other than variations, i.e., all quantities, except variations, are understood to be nominal values.

Now, except for the evaluation of the partial derivative with respect to x_e , it may be assumed, because of (8), that

$$x'_a(t') = x_a(t') .$$

Let x_i be any component of the nominal state vector, except a component of x_e . That is x_i is any component of $[x_a(t'), x_b, x_c, x_d, x_f, x_g]$. Then from (5),

$$(9) \quad \frac{\partial y}{\partial x_i} = \frac{\partial h}{\partial x_i} + \frac{\partial h}{\partial x_a^T} \dot{x}_a \frac{\partial t'}{\partial x_i} .$$

Now

$$\frac{\partial t'}{\partial x_i} = \frac{\partial t}{\partial x_i} - \frac{\partial \tau}{\partial x_i}$$

and

$$\frac{\partial t}{\partial x_i} = -m_s^T \frac{\partial x_d}{\partial x_i} .$$

Thus

$$\frac{\partial t'}{\partial x_i} = -m_s^T \frac{\partial x_d}{\partial x_i} - \frac{\partial \tau}{\partial x_i} .$$

But

$$c \frac{\partial \tau}{\partial x_i} = \frac{\partial R_A}{\partial x_i} + \dot{R}_A \frac{\partial t'}{\partial x_i} .$$

Therefore

$$\frac{\partial \tau}{\partial x_i} = \frac{1}{c + \dot{R}_A} \left[\frac{\partial R_A}{\partial x_i} - \dot{R}_A m_s^T \frac{\partial x_d}{\partial x_i} \right] ,$$

and thus

$$(10) \quad \frac{\partial t'}{\partial x_i} = - \frac{1}{c + R_A} \frac{\partial R_A}{\partial x_i} - \frac{c}{c + R_A} m_s^T \frac{\partial x_d}{\partial x_i} .$$

From (9) and (10), using the notation

$$H_\alpha \equiv \frac{\partial h}{\partial x_\alpha^T} ,$$

it follows that

$$(11) \quad \frac{\partial y}{\partial x_a^T} = H_a - \frac{H_a \dot{x}_a}{c + R_A} \frac{\partial R_A}{\partial x_a^T}$$

$$(12) \quad \frac{\partial y}{\partial x_b^T} = H_b$$

$$(13) \quad \frac{\partial y}{\partial x_c^T} = H_c$$

$$(14) \quad \frac{\partial y}{\partial x_d^T} = - \frac{c H_a \dot{x}_a}{c + R_A} m_s^T$$

$$(15) \quad \frac{\partial y}{\partial x_f^T} = H_f - \frac{H_a \dot{x}_a}{c + R_A} \frac{\partial R_A}{\partial x_f^T}$$

$$(16) \quad \frac{\partial y}{\partial x_g^T} = H_g - \frac{H_a \dot{x}_a}{c + R_A} \frac{\partial R_A}{\partial x_g^T} .$$

The evaluation of the partial derivative with respect to x_e can be accomplished most readily by indirect means.

First, observe that if β denotes a bias in the vehicle clock, it follows by the chain rule of partial differentiation that

$$(17) \quad \frac{\partial y}{\partial x_e^T} = \frac{\partial y}{\partial \beta} m_v^T .$$

Then from the measurement algorithm and equation (8), observe that if a bias α is introduced into the sensor clock and an equal bias β is introduced into the vehicle clock, there is no change in the calculated value of y . (Recall from Appendix B that a vehicle clock bias has no affect whatsoever on the on-board nominal trajectory \tilde{x}_a .) Consequently

$$\frac{\partial y}{\partial \alpha} + \frac{\partial y}{\partial \beta} = 0 ,$$

and it follows by equations (14) and (17) that

$$(18) \quad \frac{\partial y}{\partial x_e^T} = \frac{c \tilde{H}_a \dot{x}'_a}{c + \dot{R}'_A} m_V^T .$$

To conclude this subsection, the complete measurement variation equation, in terms of static partial derivatives and with complete notation, is given for reference.

$$(19) \quad \delta y(t) = \begin{bmatrix} \tilde{H}_a & -\frac{\tilde{H}_a \dot{x}'_a}{c + \dot{R}'_A} & \frac{\partial \tilde{R}_A}{\partial x_a^T} \end{bmatrix} \delta x_a(\tilde{t}') \\ + \tilde{H}_b \delta x_b + \tilde{H}_c \delta x_c \\ - \frac{c \tilde{H}_a \dot{x}'_a}{c + \dot{R}'_A} m_S^T(\tilde{t}) \delta x_d + \frac{c \tilde{H}_a \dot{x}'_a}{c + \dot{R}'_A} m_V^T(\tilde{t}') \delta x_e \\ + \begin{bmatrix} \tilde{H}_f & -\frac{\tilde{H}_a \dot{x}'_a}{c + \dot{R}'_A} & \frac{\partial \tilde{R}_A}{\partial x_f^T} \end{bmatrix} \delta x_f \\ + \begin{bmatrix} \tilde{H}_g & -\frac{\tilde{H}_a \dot{x}'_a}{c + \dot{R}'_A} & \frac{\partial \tilde{R}_A}{\partial x_g^T} \end{bmatrix} \delta x_g + v(t) .$$

C.3 Measurement Processing Using Only Encoder Measurement

The mathematical development to this point has been based on the assumption that measurements at both the sensor element output and the encoder are made

in each metric information channel. In many cases however only encoder measurements are made, and an alternative measurement processing scheme must be employed.

It should be noted at the outset that there is no method of processing encoder measurements alone which can completely compensate for the lost information carried in the sensor element output. If encoder measurements alone are to be processed in an optimal estimator, the best that can be done is to augment the estimator state vector with additional dynamic states which characterize the servo plant (i.e., all system components between the sensor element and the encoder input), and treat the encoder output as the fundamental metric sensor measurement. But this approach requires explicit knowledge of the differential equations for the servo plant, and such knowledge is not always readily available. Moreover, if this approach were used for every sensor channel, the number of additional states would greatly increase the computational burden of the estimator. Consequently, the servo state augmentation method of processing encoder measurements will not be considered further.

A commonly used method of processing encoder measurements (and the only method which will be given further consideration) is based on the use of a set of so-called servo dynamic error coefficients to approximate the quasi steady state following (or lag) error of the servo. This approximation is then used in lieu of the actual sensor element output in the measurement processing operation.

In the sequel, a linear model for a metric tracking system will be used to develop the theoretical basis for the use of dynamic error coefficients. Then comparisons will be made between three alternative schemes in which the measurements processed consist of, respectively:

- (i) both sensor element and encoder outputs,
- (ii) encoder output compensated by the dynamic error coefficient approximation to sensor element output,
- (iii) encoder output alone.

It will be seen that both (ii) and (iii) suffer from certain deficiencies in comparison to (i).

A linear model for a single channel metric sensor is depicted in Figure C.3. The variables in the diagram are identified as follows:

- x_A - apparent value of channel coordinate
- Δx_T - target dependent error
- x_R - received (or input) value of coordinate
- x_F - feedback value of coordinate
- y - sensor element output
- Δy_S - systematic sensor error
- v - measurement noise
- Δx_F - feedback error
- x_O - output value of coordinate
- Δx_O - output error including servo noise
- z - encoder output
- Δx_E - encoder error
- $G(s)$ - servo plant transfer function, i.e., Laplace transform of servo plant impulse response.

Although the diagram is drawn for the case where the encoder is outside the tracking loop, it can be applied to cases where the encoder is inside the loop by simply equating x_O and z and combining Δx_O and Δx_E .

From the diagram it follows immediately that

$$(1) \quad y + z = x_A + \Delta x_T + \Delta x_F + \Delta x_E + \Delta y_S + v .$$

Now the quantity $(y + z)$ may be regarded as the sensor measurement when both y and z are measured, since the measurement variations obtained from either y or $(y + z)$ are identical. That is

$$\delta(y + z) \equiv (y + z) - (\tilde{y} + \tilde{z}) \equiv \delta y .$$

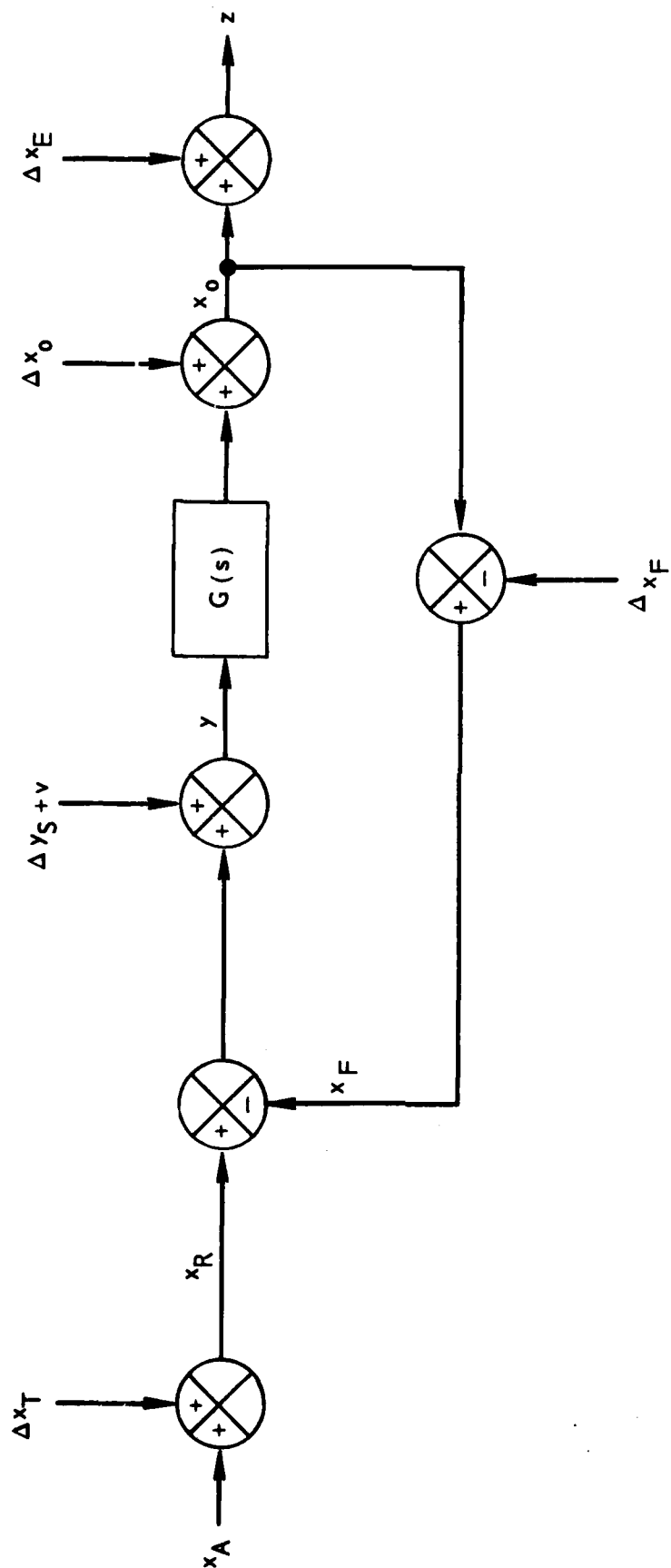


FIGURE C.3 LINEAR MODEL FOR SINGLE CHANNEL METRIC SENSOR

The expression for the encoder measurement is also readily obtained from the diagram. Thus

$$z = p*[x_A + \Delta x_T + \Delta x_F + \Delta y_S + v] + q*\Delta x_O + \Delta x_E ,$$

or

$$(2) \quad z = p*x_A + p * [\Delta x_T + \Delta x_F + \Delta y_S] + q*\Delta x_O + \Delta x_E + p*v ,$$

where

$$P(s) = \frac{G(s)}{1 + G(s)} ,$$

$$Q(s) = \frac{1}{1 + G(s)} ,$$

$$p(t) = \mathcal{L}^{-1} \{P(s)\} ,$$

$$q(t) = \mathcal{L}^{-1} \{Q(s)\} ,$$

\mathcal{L}^{-1} denotes the inverse Laplace transform, and $*$ denotes the convolution operation defined by

$$(p*x)(t) = \int_{-\infty}^{\infty} p(u)x(t-u)du = \int_{-\infty}^{\infty} p(t-u)x(u)du .$$

The constraints imposed on servo design are typically such that in (2) p acts as a low pass filter, and q acts as a high pass filter. The term $p*x_A$ is a delayed and distorted version of x_A . The term $p*v$ is a smoothed version of v which exhibits serial correlation, where the correlation time is roughly equal to the reciprocal of the bandwidth of the filter p . If, as is the usual case, Δx_T , Δx_F , and Δy_S are very low frequency in character, the term $p*[\Delta x_T + \Delta x_F + \Delta y_S]$ is approximately just $[\Delta x_T + \Delta x_F + \Delta y_S]$. Since q is a high pass filter the term $q*\Delta x_O$ will contain only the high frequency servo noise content in Δx_O , and the low frequency servo noise will be rejected.

Now the approximation to the sensor element output obtained with dynamic error coefficients will be derived. From the diagram of Figure C.3, it follows that

$$(3) \quad y = q*[x_A + \Delta x_T + \Delta x_F - \Delta x_O + \Delta y_S + v] .$$

Since q is a high pass filter, while Δx_T , Δx_F and Δy_S are typically low frequency terms, it follows further that

$$(4) \quad y = q*[x_A - \Delta x_O + v] .$$

Deletion of the noise and error terms in (4) leads to the quantity

$$(5) \quad \bar{y} = q*x_A$$

which contains all the information in y regarding x_A . The procedure is now aimed toward approximation of \bar{y} .

The expression for $Q(s)$ is first expanded in a Taylor series about the origin. Thus

$$(6) \quad Q(s) = \frac{1}{1 + G(s)} = \frac{1}{1 + K_0} + \frac{s}{K_1} + \frac{s^2}{K_2} + \frac{s^3}{K_3} + \dots ,$$

where K_i is the dynamic error coefficient of order i for $i = 0, 1, 2, \dots$. If the function $G(s)$ has a pole of order n at the origin, then

$$K_i = \begin{cases} \infty, & i = 0, 1, 2, \dots, n-1 \quad , \\ \lim_{s \rightarrow 0} s^n G(s), & i = n \quad . \end{cases}$$

In this case the system in Figure C.3 is said to be of type n .

Type 0 systems are of little interest. In fact, most metric tracking systems are either type 1 or type 2. If type 0 systems are excluded, then $K_0 = \infty$ and thus

$$(7) \quad Q(s) = \frac{s}{K_1} + \frac{s^2}{K_2} + \frac{s^3}{K_3} + \dots$$

Using (7) in (5) it follows from elementary properties of Laplace transforms that

$$(8) \quad \bar{y}(t) = \frac{\dot{x}_A(t)}{K_1} + \frac{\ddot{x}_A(t)}{K_2} + \frac{\dddot{x}_A(t)}{K_3} + \dots$$

At this point the assumption is made that x_A is a sufficiently smooth function that the series in (8) can be truncated after the second or third term with negligible error. Thus assuming a nominal trajectory is specified for x_A , the resulting approximation for \bar{y} is

$$(9) \quad \bar{y}(t) = \frac{\tilde{\dot{x}}_A(t)}{K_1} + \frac{\tilde{\ddot{x}}_A(t)}{K_2} + \frac{\tilde{\dddot{x}}_A(t)}{K_3} .$$

Combining (9) with (2) there results

$$(10) \quad \bar{y} + z = p^*x_A + \frac{\tilde{\dot{x}}_A}{K_1} + \frac{\tilde{\ddot{x}}_A}{K_2} + \frac{\tilde{\dddot{x}}_A}{K_3} .$$

$$+ p^*[\Delta x_T + \Delta x_F + \Delta y_S] + q^*\Delta x_O + \Delta x_E + p^*v .$$

At this point the three measurement processing schemes can be compared to determine their relative merits. The first alternative, which is feasible only when both encoder and sensor element outputs are available, provides a measurement expressed by (1). In this case the measurement has a simple linear dependence on the measured coordinate, systematic error terms, and measurement noise. The systematic error terms can be expressed as linear functions of state variables, and the measurement noise in the sensor element is sequentially uncorrelated. Thus the measurement given by (1) satisfies all assumptions required in the development of the optimal estimation equations in Appendix A.

The next alternative to be considered is the case in which the uncompensated encoder output is used, and the measurement is expressed by (2). In this case, as has been mentioned, the measured coordinate is delayed and distorted by filtering, the measurement contains high frequency servo noise, and the noise which originated in the sensor element or receiver now appears as smoothed and sequentially correlated measurement noise. It is clear that the measurement in this case does not satisfy the assumptions required by the estimator for optimality, and some degradation in estimator performance will result.

The last alternative is the case in which the encoder output is compensated using dynamic error coefficients, and the measurement is given by (10). Comparison of (2) with (10) shows that all the objectionable features present in (2) are also present in (10), except that, to some extent in the latter, the delay in x_A has been compensated by the introduction of the dynamic error terms.

The application of dynamic error coefficients in the single channel range and doppler trackers is straight forward. Each of these trackers is typically of type 2. Thus the dynamic error terms are given by

$$\bar{r} = \frac{\ddot{R}'_A}{K_2} + \frac{\ddot{R}'_A}{K_3}$$

and

$$\bar{d} = \frac{\ddot{D}'_A}{K_2} + \frac{\ddot{D}'_A}{K_3}$$

respectively, where \ddot{R}'_A and \ddot{D}'_A are derived from the off-board nominal solution.

The application of dynamic error coefficients in angle trackers must account for the fact that the sensing element output which drives the azimuth servo senses angle in the traverse plane. Consequently the gain of the azimuth servo is multiplied by a factor $G(E_s)$ to ensure uniform servo response in azimuth for all elevation angles except those in a small region near zenith. E_s is the elevation shaft angle, and

$$G(E_s) = \begin{cases} \sec E_s, & E_s \leq E_{\max} \\ \sec E_{\max}, & E_{\max} < E_s \leq \pi/2 \end{cases} .$$

For operation with $E_s > \pi/2$, $G(E_s) = -G(\pi - E_s)$.

The dynamic error approximations in traverse and elevation are given by

$$\bar{\phi} = \frac{1}{G(E_s)} \left[\frac{\ddot{A}'_A}{K_1} + \frac{\ddot{A}'_A}{K_2} + \frac{\ddot{A}'_A}{K_3} \right]$$

and

$$\bar{\psi} = \frac{\ddot{E}'_A}{K_1} + \frac{\ddot{E}'_A}{K_2} + \frac{\ddot{E}'_A}{K_3} ,$$

respectively.

D. MISCELLANEOUS TOPICS IN LINEAR ALGEBRA

The results in this appendix can be found in many standard mathematics and numerical analysis texts [9], [10], [11]. They are included here to provide a concise and readily accessible supplement to the algorithms developed in the text.

D.1 Gauss Elimination

Let A be an $n \times n$ matrix. Assume for the moment that A has full rank. The Gauss elimination procedure applied to A consists of a sequence of operations on the rows of A which yields an upper triangular matrix U . To be specific, let

$$A^{(0)} = A = \begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & \dots & a_{1n}^{(0)} \\ a_{21}^{(0)} & a_{22}^{(0)} & \dots & a_{2n}^{(0)} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1}^{(0)} & a_{n2}^{(0)} & \dots & a_{nn}^{(0)} \end{bmatrix} .$$

Using the elements of the first column of $A^{(0)}$ define

$$l_{ki} = \begin{cases} 0, & k = 1, \\ \frac{a_{k1}^{(0)}}{a_{11}^{(0)}}, & k = 2, \dots, n, \end{cases}$$

and put

$$L_1^{-1} = \left[\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline -l_{21} & & & \\ \vdots & & & \\ -l_{n1} & & I_{n-1} & \end{array} \right] .$$

Then let

$$A^{(1)} = L_1^{-1}A^{(0)},$$

and observe that $A^{(1)}$ has the structure

$$A^{(1)} = \begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & \dots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \dots & a_{2n}^{(1)} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} \end{bmatrix},$$

where

$$a_{kj}^{(1)} = a_{kj}^{(0)} - \frac{a_{k1}^{(0)}a_{1j}^{(0)}}{a_{11}^{(0)}}, \quad j, k = 2, \dots, n.$$

Similarly, using the elements of the second column of $A^{(1)}$ define

$$l_{k2} = \begin{cases} 0, & k = 1, 2 \\ \frac{a_{k2}^{(1)}}{a_{22}^{(1)}}, & k = 3, \dots, n \end{cases},$$

and put

$$L_2^{-1} = \begin{bmatrix} 1 & 0 & | & 0 & \dots & 0 \\ 0 & 1 & | & 0 & \dots & 0 \\ \hline 0 & -l_{32} & | & & & \\ \vdots & \vdots & | & & & \\ \vdots & \vdots & | & & & \\ \vdots & \vdots & | & & & \\ 0 & -l_{n2} & | & & & \end{bmatrix}.$$

Then with

$$A^{(2)} = L_2^{-1}A^{(1)} = L_2^{-1}L_1^{-1}A^{(0)},$$

it follows that $A^{(2)}$ has the structure

$$A^{(2)} = \begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & a_{13}^{(0)} & \dots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \dots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & a_{n3}^{(2)} & \dots & a_{nn}^{(2)} \end{bmatrix},$$

where $a_{kj}^{(2)} = a_{kj}^{(1)} - \frac{a_{k2}^{(1)}a_{2j}^{(1)}}{a_{22}^{(1)}}$, $j, k = 3, \dots, n$.

In general, for $i = 1, \dots, n-1$, define

$$\ell_{ki} = \begin{cases} 0, & k = 1, \dots, i, \\ \frac{a_{ki}^{(i-1)}}{a_{ii}^{(i-1)}}, & k = i+1, \dots, n, \end{cases}$$

put

$$L_i^{-1} = \begin{bmatrix} I_{(i-1)} & \begin{matrix} 0 \\ \vdots \\ 0 \end{matrix} & \bigcirc \\ \begin{matrix} 0 & \dots & 0 \end{matrix} & 1 & \begin{matrix} 0 & \dots & 0 \end{matrix} \\ \bigcirc & \begin{matrix} -\ell_{i+1,i} \\ \vdots \\ -\ell_{n,i} \end{matrix} & I_{(n-i)} \end{bmatrix},$$

and let

$$A^{(i)} = L_i^{-1} \dots L_1^{-1} A^{(0)}$$

Then $A^{(i)}$ has the structure

$$A^{(i)} = \begin{bmatrix} a_{11}^{(0)} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{2n}^{(1)} \\ \cdot & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ \cdot & \cdot & \cdot & a_{ii}^{(i-1)} & \cdot & \cdot & \cdot & \cdot & \cdot & a_{in}^{(i-1)} \\ \cdot & \cdot & \cdot & 0 & a_{i+1,i+1}^{(i)} & \cdot & \cdot & \cdot & \cdot & a_{i+1,n}^{(i)} \\ \cdot & \cdot & \cdot & \vdots & \vdots & \cdot & \cdot & \cdot & \cdot & \vdots \\ 0 & 0 & \dots & 0 & a_{n,i+1}^{(i)} & \dots & \cdot & \cdot & \cdot & a_{nn}^{(i)} \end{bmatrix},$$

where $a_{kj}^{(i)} = a_{kj}^{(i-1)} - \frac{a_{ki}^{(i-1)} a_{ij}^{(i-1)}}{a_{ii}^{(i-1)}}$, $j, k = i+1, \dots, n$,

and in particular

$$A^{(n-1)} \equiv U = \begin{bmatrix} a_{11}^{(0)} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{2n}^{(1)} \\ \cdot & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ 0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 & a_{nn}^{(n-1)} & \cdot & \cdot \end{bmatrix}.$$

Using the definitions introduced above, the Gaussian elimination procedure can be represented in matrix form by

$$(1) \quad L_{n-1}^{-1} \dots L_1^{-1} A = U .$$

Now let e_i denote the i -th column of the $n \times n$ identity matrix and for each $i = 1, \dots, n-1$, let λ_i denote the n -vector defined by

$$(2) \quad \lambda_i = \begin{bmatrix} l_{1i} \\ l_{2i} \\ \vdots \\ l_{ni} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ l_{i+1,i} \\ \vdots \\ l_{ni} \end{bmatrix} .$$

Then for each $i = 1, \dots, n-1$,

$$(3) \quad L_i^{-1} = I - \lambda_i e_i^T ,$$

and it is obvious that

$$(4) \quad L_i = I + \lambda_i e_i^T .$$

Finally let

$$(5) \quad L \equiv L_1 \dots L_{n-1} ,$$

and observe that

$$(6) \quad L = I + \sum_{i=1}^{n-1} \lambda_i e_i^T .$$

Then from (1) it follows that

$$(7) \quad A = LU ,$$

where

$$(8) \quad L = \begin{bmatrix} 1 & 0 & \dots & \dots & \dots & 0 \\ l_{21} & 1 & 0 & \dots & \dots & 0 \\ \vdots & l_{32} & \cdot & \cdot & \cdot & \vdots \\ \vdots & \vdots & & & & 0 \\ l_{n1} & l_{n2} & \dots & l_{n,n-1} & \cdot & 1 \end{bmatrix}$$

$$(9) \quad U \equiv \begin{bmatrix} u_{11} & u_{12} & \dots & \dots & u_{1n} \\ 0 & u_{22} & \dots & \dots & u_{2n} \\ \vdots & 0 & \cdot & \cdot & \vdots \\ \vdots & \vdots & \cdot & \cdot & \vdots \\ \vdots & \vdots & \cdot & \cdot & \vdots \\ \vdots & 0 & \dots & 0 & u_{nm} \end{bmatrix} = \begin{bmatrix} a_{11}^{(0)} & \dots & \dots & \dots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \dots & \dots & a_{2n}^{(1)} \\ \vdots & 0 & \cdot & \cdot & \vdots \\ \vdots & \vdots & \cdot & \cdot & \vdots \\ \vdots & \vdots & \cdot & \cdot & \vdots \\ 0 & 0 & \dots & \dots & 0 \cdot a_{nn}^{(n-1)} \end{bmatrix}$$

Since an explicit expression for L^{-1} does not appear in the sequel, it is perhaps worth while to note that

$$L^{-1} \neq I - \sum_{i=1}^{n-1} \lambda_i e_i^T,$$

and hence L^{-1} cannot be obtained by simply changing the sign of the elements below the main diagonal in (8).

The Gauss elimination procedure has led to the factorization of A into a product of triangular matrices in (7). This form is particularly useful in solving equations of the form

$$(10) \quad Ax = b.$$

A two step process is used to solve (10) for x . From (7) it follows that (10) is equivalent to the pair of equations

$$(11) \quad Ly = b ,$$

$$(12) \quad Ux = y .$$

Now it is a simple matter, because of triangularity, to first solve (11) for y by forward substitution and then solve (12) for x by backward substitution.

In the course of the Gauss elimination process, the columns of L are formed sequentially using the results of previous calculations. Thus the i -th column of L requires division by the diagonal element $a_{ii}^{(i-1)}$. If $a_{ii}^{(i-1)}$ is zero the process fails outright. And if $a_{ii}^{(i-1)}$ is small in magnitude, it can contain a relatively large roundoff error. In such cases the relative error in the i -th column of L , and hence in successive computations, can be quite large.

Since A is nonsingular, there must exist a nonzero element $a_{jk}^{(i-1)}$ for some $j \geq i$ and $k \geq i$. This fact provides the basis for a means not only of avoiding division by zero, but also of reducing the amount of roundoff error buildup in Gauss elimination. The procedure is called pivoting, and the idea is to interchange rows and columns in the object matrix to bring an element of large magnitude into the i -th diagonal position prior to computing the i -th column of L . The larger this element, relative to other elements of the matrix, the smaller will be the roundoff error in the i -th column of L and in subsequent calculations.

The theoretical justification for the pivoting process will now be given. To this end, the notion of a permutation matrix is formulated.

An $n \times n$ matrix P is called a permutation matrix, if for any n -vector x , the vector y defined by

$$y = Px$$

is obtained by interchanging at most two elements of x . Thus either, $y = x$, or there exist two distinct integers j and k such that

$$y_i = \begin{cases} x_k, & i=j \\ x_j, & i=k \\ x_i, & \text{otherwise} \end{cases} .$$

It is easily established that if P is a permutation matrix, then it is symmetric and orthogonal, i.e.,

$$P = P^T = P^{-1} .$$

Observe also that, if P is an $n \times n$ permutation matrix and B is an arbitrary $n \times n$ matrix, then

PB

is a matrix which differs from B by at most an interchange of two rows, while

BP

differs from B by at most an interchange of two columns.

To illustrate the use of pivoting in Gauss elimination, the row and column interchange operations will be represented by permutation matrices. For each $i = 1, \dots, n-1$, just prior to calculation of the elements in L_i^{-1} , a row permutation R_i and a column permutation C_i are performed on the object matrix to place an element of large magnitude in the i -th position on the diagonal. To preserve triangularity it is necessary and sufficient that R_i permute rows i and j and C_i permute columns i and k , where $j \geq i$ and $k \geq i$. Thus the row and column interchanges bring the element in position (j, k) to position (i, i) in the object matrix.

In matrix form, the sequence of elimination operations can be represented as follows

$$\tilde{L}_1^{-1} R_1 A C_1 = \tilde{A}^{(1)}$$

$$\tilde{L}_2^{-1} R_2 \tilde{L}_1^{-1} R_1 A C_1 C_2 = \tilde{A}^{(2)}$$

⋮

$$\tilde{L}_{n-1}^{-1} R_{n-1} \dots \tilde{L}_1^{-1} R_1 A C_1 \dots C_{n-1} = \tilde{A}^{(n-1)}$$

Now with the definitions

$$\bar{A} \equiv R_{n-1} \dots R_1 A C_1 \dots C_{n-1} ,$$

$$\bar{U} \equiv \tilde{A}^{(n-1)} ,$$

and

$$\bar{L} = R_{n-1} \dots R_2 \tilde{L}_1 R_2 \tilde{L}_2 \dots R_{n-1} \tilde{L}_{n-1} ,$$

it follows that

$$\bar{A} = \bar{L} \bar{U} .$$

But by induction

$$\begin{aligned} \bar{L} &= R_{n-1} \dots R_2 \tilde{L}_1 R_2 \tilde{L}_2 \dots R_{n-1} \tilde{L}_{n-1} \\ &= R_{n-1} \dots R_2 \tilde{L}_1 R_2 (R_3 \dots R_{n-1}) (R_{n-1} \dots R_3) \tilde{L}_2 \dots R_{n-1} \tilde{L}_{n-1} \\ &= [R_{n-1} \dots R_2 \tilde{L}_1 R_2 \dots R_{n-1}] [R_{n-1} \dots R_3 \tilde{L}_2 \dots R_{n-1} \tilde{L}_{n-1}] \\ &= [I + R_{n-1} \dots R_2 \tilde{\lambda}_1 e_1^T] [R_{n-1} \dots R_3 \tilde{L}_2 R_3 \tilde{L}_3 \dots R_{n-1} \tilde{L}_{n-1}] \\ &= [I + R_{n-1} \dots R_2 \tilde{\lambda}_1 e_1^T] \dots [I + R_{n-1} \tilde{\lambda}_{n-2} e_{n-2}^T] [I + \tilde{\lambda}_{n-1} e_{n-1}^T] , \end{aligned}$$

or

$$\begin{aligned} \bar{L} &= [I + \bar{\lambda}_1 e_1^T][I + \bar{\lambda}_2 e_2^T] \dots [I + \bar{\lambda}_{n-1} e_{n-1}^T] \\ &= I + \sum_{i=1}^{n-1} \bar{\lambda}_i e_i^T, \end{aligned}$$

where

$$\bar{\lambda}_i = \begin{cases} R_{n-1} \dots R_{i+1} \tilde{\lambda}_i, & i = 1, \dots, n-2 \\ \tilde{\lambda}_{n-1}, & i = n-1. \end{cases}$$

A simple bookkeeping procedure for factorization of \bar{A} into $\bar{L}\bar{U}$ is now obvious. On the i -th elimination the entire $n \times n$ object matrix is subjected to the appropriate row and column permutation. Then the elements of $\tilde{\lambda}_i$ are computed and the elimination operation is performed. Finally the nonzero elements of $\tilde{\lambda}_i$ are placed in the corresponding eliminated positions in the i -th column below the main diagonal. That is $\tilde{\lambda}_{ki}$ is placed in position (k, i) for each $k = i+1, \dots, n$. After the $(n-1)$ -th elimination, the object matrix is of the form

$$\begin{bmatrix} \bar{u}_{11} & \bar{u}_{12} & \cdot & \cdot & \cdot & \bar{u}_{1n} \\ \bar{\ell}_{21} & \bar{u}_{22} & \cdot & \cdot & \cdot & \bar{u}_{2n} \\ \bar{\ell}_{31} & \bar{\ell}_{32} & \cdot & & & \cdot \\ \cdot & \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \bar{\ell}_{n1} & \bar{\ell}_{n2} & \dots & \bar{\ell}_{n,n-1} & \cdot & \bar{u}_{nn} \end{bmatrix},$$

where

$$\bar{L} = \begin{bmatrix} 1 & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ \bar{l}_{21} & 1 & 0 & \cdot & \cdot & \cdot & 0 \\ \bar{l}_{31} & \bar{l}_{32} & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \bar{l}_{n1} & \bar{l}_{n2} & \cdot & \cdot & \cdot & \cdot & \bar{l}_{n,n-1} & 1 \end{bmatrix}$$

and

$$\bar{U} = \begin{bmatrix} \bar{u}_{11} & \bar{u}_{12} & \cdot & \cdot & \cdot & \bar{u}_{1n} \\ 0 & \bar{u}_{22} & \cdot & \cdot & \cdot & \bar{u}_{2n} \\ \cdot & 0 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & \cdot & \bar{u}_{nn} \end{bmatrix}$$

The solution of equation (10) using Gauss eliminations with pivoting is quite direct. First observe that the system (10) is equivalent to

$$(10') \quad \bar{A}\bar{x} = \bar{b} \quad ,$$

where

$$\bar{A} = R_{n-1} \dots R_1 A C_1 \dots C_{n-1} \quad ,$$

$$\bar{b} = R_{n-1} \dots R_1 b \quad ,$$

and

$$\bar{x} = C_{n-1} \dots C_1 x \quad .$$

But (10') can be solved using the factorization

$$\bar{A} = \bar{L}\bar{U}$$

by solving the equivalent system

$$(11') \quad \bar{L}\bar{y} = \bar{b} ,$$

$$(12') \quad \bar{U}\bar{x} = \bar{y} ,$$

from which \bar{x} is obtained simply by

$$\bar{x} = C_1 \dots C_{n-1} \bar{x} .$$

The pivot procedure described above is sometimes called complete pivoting.

In the special cases in which either $R_1 = \dots = R_{n-1} = I$ or $C_1 = \dots = C_{n-1} = I$, the procedure is called partial pivoting.

If the matrix A is of rank $r < n$, then of course A is singular. In this case the Gauss elimination process with pivoting will terminate immediately after the r -th elimination and the object matrix will be of the form

$$\begin{bmatrix} \bar{u}_{11} & \bar{u}_{12} & \dots & \dots & \bar{u}_{1r} & \dots & \bar{u}_{1n} \\ \bar{l}_{21} & \bar{u}_{22} & \dots & \dots & \bar{u}_{2r} & \dots & \bar{u}_{2n} \\ \bar{l}_{31} & \bar{l}_{32} & \dots & \dots & \dots & \dots & \dots \\ \vdots & \vdots & \dots & \dots & \dots & \dots & \dots \\ \vdots & \vdots & \dots & \dots & \dots & \dots & \dots \\ \bar{l}_{r1} & \bar{l}_{r2} & \dots & \bar{l}_{r,r-1} & \bar{u}_{rr} & \dots & \bar{u}_{rn} \\ \vdots & \vdots & \dots & \vdots & \bar{l}_{r+1,r} & 0 \dots 0 & \vdots \\ \vdots & \vdots & \dots & \vdots & \vdots & \vdots & \vdots \\ \bar{l}_{n1} & \bar{l}_{n2} & \dots & \bar{l}_{n,r-1} & \bar{l}_{n,r} & 0 \dots 0 & \vdots \end{bmatrix}$$

D.2 Cholesky Decomposition

A real valued $n \times n$ matrix A is said to be nonnegative if A is symmetric and the condition

$$x^T A x \geq 0$$

holds for every n -vector x .

A real valued $n \times n$ matrix S which satisfies the condition

$$A = S S^T$$

is called a square root of A .

It is obvious that the matrix product $S S^T$ is nonnegative for any real valued $n \times n$ matrix S . Conversely, as will be shown, if A is a nonnegative matrix then a real valued square root of A exists, but is not unique. The proof of the existence of a square root of a nonnegative matrix will be by construction using Gauss elimination. This constructive process is also called Cholesky decomposition.

Let A be a nonnegative $n \times n$ matrix. It is readily shown that

(i) $a_{kk} \geq 0$ for all $k = 1, \dots, n$

and

(ii) $\alpha = \max \{a_{kk}; k = 1, \dots, n\}$ implies $\alpha = \max \{|a_{jk}|; j, k = 1, \dots, n\}$,

that is, all diagonal elements of A are nonnegative, and no element of A has a magnitude greater than the maximal diagonal value α .

Gauss elimination with pivoting will now be applied to A , and the first permutation matrices R_1 and C_1 will be selected such that

$$R_1 = C_1 = P_1$$

and

$$\tilde{a}_{11}^{(0)} = \alpha (= \text{maximal diagonal value of } A) ,$$

where

$$\tilde{A}^{(0)} = P_1 A P_1 .$$

The first elimination matrix \tilde{L}_1^{-1} is formed such that

$$\tilde{L}_1^{-1} = \begin{bmatrix} 1 & 0^T \\ -\tilde{\lambda}_1 & I \end{bmatrix}$$

and

$$\tilde{L}_1^{-1} \tilde{A}^{(0)} = \left[\begin{array}{c|ccc} \tilde{a}_{11}^{(0)} & \tilde{a}_{12}^{(0)} & \dots & \tilde{a}_{1n}^{(0)} \\ \hline 0 & & \tilde{B}^{(1)} & \end{array} \right] \equiv \tilde{A}^{(1)} ,$$

where

$$\tilde{B}^{(1)} = \begin{bmatrix} \tilde{a}_{22}^{(1)} & \dots & \tilde{a}_{2n}^{(1)} \\ \vdots & & \vdots \\ \tilde{a}_{n2}^{(1)} & \dots & \tilde{a}_{nn}^{(1)} \end{bmatrix} .$$

(In case $\alpha = 0$, take $\tilde{\lambda}_1 = 0$ and $P_1 = I$.)

It follows immediately, by symmetry, that

$$\tilde{L}_1^{-1} \tilde{A}^{(0)} (\tilde{L}_1^{-1})^T = \begin{bmatrix} \tilde{a}_{11}^{(0)} & 0^T \\ 0 & \tilde{B}^{(1)} \end{bmatrix} ,$$

and since A is nonnegative, $\tilde{B}^{(1)}$ must also be nonnegative. But since $\tilde{B}^{(1)}$ is nonnegative it follows by induction that the Gauss elimination process can be continued with $R_i = C_i = P_i$ chosen at each stage to place a maximal value element, from the remaining diagonal elements, in the (i, i) position. If for some $i_0 < n$, the remaining diagonal elements are all zero, take $\tilde{\lambda}_i = 0$ and $R_i = C_i = P_i = I$ for all $i = i_0, \dots, n-1$.

Thus when A is nonnegative, the sequence of elimination operations can be performed with symmetric pivoting and represented as follows.

$$\begin{aligned} \tilde{L}_1^{-1} P_1 A P_1 &= \tilde{A}^{(1)} \\ \tilde{L}_2^{-1} P_2 \tilde{L}_1^{-1} P_1 A P_1 P_2 &= \tilde{A}^{(2)} \\ &\vdots \\ \tilde{L}_{n-1}^{-1} P_{n-1} \dots \tilde{L}_1^{-1} P_1 A P_1 \dots P_{n-1} &= \tilde{A}^{(n-1)} . \end{aligned}$$

Then with the definition

$$\begin{aligned} \bar{A} &= P_{n-1} \dots P_1 A P_1 \dots P_{n-1} , \\ \bar{U} &= \tilde{A}^{(n-1)} , \end{aligned}$$

and

$$\bar{L} \equiv P_{n-1} \dots P_2 \tilde{L}_1 P_2 \tilde{L}_2 \dots P_{n-1} \tilde{L}_{n-1} ,$$

it follows that

$$\bar{A} = \bar{L} \bar{U} .$$

But it is also true that

$$\bar{L}^{-1} \bar{A} (\bar{L}^{-1})^T = \bar{U} (\bar{L}^{-1})^T \equiv \bar{D} ,$$

which implies \bar{D} is nonnegative (hence symmetric) and upper triangular. Thus \bar{D} is a nonnegative diagonal matrix.

Now let $\bar{D}^{1/2}$ denote the (unique) nonnegative diagonal matrix which is a square root of \bar{D} , and put

$$\bar{S} = \bar{L}\bar{D}^{1/2} .$$

Then

$$\bar{A} = \bar{S}\bar{S}^T ,$$

and thus \bar{S} is a lower triangular square root of \bar{A} . It follows immediately that

$$A = SS^T ,$$

where

$$S \equiv P_1 \dots P_{n-1} \bar{S} .$$

Thus S is a square root of A , and the construction is complete. Note that S , unlike \bar{S} , is not necessarily triangular. A method of transforming S into a triangular square root of A using the Gram-Schmidt process is described in D.3.

When the Cholesky decomposition is applied to a nonnegative matrix A , a simplified bookkeeping procedure can be used which takes advantage of symmetry. First, recall that

$$\bar{D} = \bar{U}(\bar{L}^{-1})^T .$$

Then since \bar{L} has unity diagonal values so does \bar{L}^{-1} . Consequently, \bar{D} is simply the diagonal part of \bar{U} , i.e.,

$$\bar{D} = \begin{bmatrix} \bar{u}_{11} & 0 & \dots & 0 \\ 0 & \cdot & \cdot & \cdot \\ \vdots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 0 \\ 0 & \dots & 0 & \bar{u}_{nn} \end{bmatrix}.$$

Thus since

$$U = \bar{D}\bar{L}^T$$

it follows that

$$\bar{u}_{jk} = \bar{u}_{jj}\bar{\ell}_{kj} \text{ for } k > j, j = 1, \dots, n-1.$$

Now if A is of rank r , then $\bar{u}_{jj} > 0$ for $j \leq r$ and $\bar{u}_{jj} = 0$ for $j > r$, $j = 1, \dots, n$.

But if A is of rank r , it also follows that $\bar{\ell}_{kj} \equiv 0$ for $k > j > r$. Consequently, the entries in \bar{L} below the main diagonal are determined from \bar{U} by

$$\bar{\ell}_{kj} = \begin{cases} \frac{\bar{u}_{jk}}{\bar{u}_{jj}}, & j \leq r, \\ 0, & j > r, \end{cases}$$

for $j < k \leq n$, $j = 1, \dots, n-1$.

Second, recall that

$$\bar{S} = \bar{L}\bar{D}^{1/2}.$$

Then if A is of rank r , the elements of \bar{S} on and below the main diagonal are given by

$$\bar{s}_{kj} = \begin{cases} \sqrt{\frac{\bar{u}_{jk}}{\bar{u}_{jj}}}, & j \leq r, \\ 0, & j > r, \end{cases}$$

for $j \leq k \leq n$, $j = 1, \dots, n$.

Since \bar{L} and \bar{S} are readily computed from \bar{U} , the Cholesky decomposition can be carried out without bothering to compute and store \bar{L} . Furthermore, because of symmetry the entire process can be performed using only the upper triangle of the object matrix. Also, the square root normalizations used to obtain \bar{S} from \bar{U} can be performed in conjunction with the eliminations, so that the object matrix contains \bar{S}^T (rather than \bar{U}) when the process is complete. Conceptually, the bookkeeping method used in Cholesky decomposition differs from that given for Gauss elimination as follows:

1. Only symmetric pivoting is used, and P_i is chosen to bring a maximal valued element, from the remaining diagonal elements, into position (i, i) .
2. Instead of computing \tilde{L}_{ki} , $k = i+1, \dots, n$, for the i -th elimination, the elements in the i -th row of the object matrix are normalized by the square root of the diagonal element in that row. That is

$$\tilde{a}_{ik}^{(i-1)} + \frac{\tilde{a}_{ik}^{(i-1)}}{\tilde{a}_{ii}^{(i-1)}}, \quad k = i, \dots, n.$$

3. The i -th elimination is performed as follows:

$$\tilde{a}_{ji}^{(i)} = 0, \quad j = i+1, \dots, n$$

$$\tilde{a}_{jk}^{(i)} = \tilde{a}_{jk}^{(i-1)} - \tilde{a}_{ij}^{(i-1)}\tilde{a}_{ik}^{(i-1)}, \quad j \leq k \leq n, \quad j = i+1, \dots, n,$$

and

$$\tilde{a}_{kj}^{(i)} = \tilde{a}_{jk}^{(i)}, \quad j < k \leq n, \quad j = i+1, \dots, n-1.$$

When all elimination steps are complete and the object matrix contains \bar{S}^T . Thus if A is of rank r , the object matrix contains

$$\begin{bmatrix} \bar{s}_{11} & \dots & \bar{s}_{r1} & \dots & \bar{s}_{n1} \\ 0 & & \vdots & & \vdots \\ \vdots & & \vdots & & \vdots \\ 0 & \dots & 0 & \bar{s}_{rr} & \dots & \bar{s}_{nr} \\ \vdots & & \vdots & & & \vdots \\ \vdots & & \vdots & & 0 & \dots & 0 \\ \vdots & & \vdots & & & & \vdots \\ \vdots & & \vdots & & & & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix}$$

If the nonnegative matrix A satisfies the stronger condition,

$$x^T Ax > 0$$

for every nonzero n-vector, then A is positive. In this case, Cholesky decomposition can be applied without pivoting, theoretically, although it may be inadvisable to do so because of roundoff error buildup. Nonetheless, if pivoting is not used, a simple formula for S, due originally to Cholesky, follows by induction from the Gauss elimination method, modified as above to take advantage of symmetry. The formula is

$$s_{ii} = \left(a_{ii} - \sum_{1 \leq k < i} s_{ik}^2 \right)^{1/2}, \quad i = 1, \dots, n,$$

$$s_{ji} = \frac{1}{s_{ii}} \left(a_{ji} - \sum_{1 \leq k < i} s_{jk} s_{ik} \right), \quad j = i+1, \dots, n, \quad i = 1, \dots, n-1.$$

As a final observation, it is noted that the Cholesky decomposition can be applied to obtain

$$A = \bar{S} \bar{S}^T,$$

where \bar{S} is upper triangular. All that is required is that the row elimination operations be applied to the object matrix from the right (instead of left) and proceed from bottom to top (rather than top to bottom).

D.3 Gram-Schmidt Orthonormalization

The Gram-Schmidt orthonormalization process is a method of constructing an orthonormal set of vectors from an arbitrary set of linearly independent vectors. In this appendix, the treatment is limited to real valued n -vectors. For a complete treatment, the reader is referred to [12], [13].

The inner product of two real n -vectors x and y is given by

$$(x, y) \equiv x^T y = y^T x = (y, x) ,$$

and x and y are said to be orthogonal if

$$(x, y) = 0 .$$

The norm of an arbitrary vector x is defined by

$$\|x\| = (x, x)^{1/2} .$$

A set of vectors x_1, \dots, x_m is said to be linearly independent if the relation

$$c_1 x_1 + \dots + c_m x_m = 0$$

holds only when $c_1 = \dots = c_m = 0$.

A set of vectors u_1, \dots, u_m is said to be orthonormal if

$$(u_i, u_j) = \begin{cases} 1 & i = j , \\ 0 & i \neq j , \end{cases}$$

$i, j = 1, \dots, m$.

A set of vectors which is orthonormal is necessarily also linearly independent, but the converse does not hold. The Gram-Schmidt algorithm for constructing an orthonormal set of vectors from an arbitrary linearly independent set of vectors x_1, \dots, x_m is developed below.

Define a set of vectors y_1, \dots, y_m as follows

$$\begin{aligned}
 y_1 &= x_1 \\
 y_2 &= x_2 - (x_2, y_1) \frac{y_1}{\|y_1\|^2} \\
 &\vdots \\
 y_i &= x_i - (x_i, y_1) \frac{y_1}{\|y_1\|^2} - \dots - (x_i, y_{i-1}) \frac{y_{i-1}}{\|y_{i-1}\|^2} \\
 &\vdots \\
 y_m &= x_m - (x_m, y_1) \frac{y_1}{\|y_1\|^2} - \dots - (x_m, y_{m-1}) \frac{y_{m-1}}{\|y_{m-1}\|^2} .
 \end{aligned}$$

It follows, by linear independence, that $\|y_i\| > 0$, $i = 1, \dots, m$.

It is easily shown, by induction, that y_1, \dots, y_m is an orthogonal set of vectors. Furthermore, for each $i = 1, \dots, m$, the set y_1, \dots, y_i and the set x_1, \dots, x_i span the same subspace, that is every vector which can be expressed as a linear combination of members of one set can also be expressed as a linear combination of members of the other set.

Now define the set u_1, \dots, u_m by normalization of the set y_1, \dots, y_m , that is,

$$u_i = \frac{y_i}{\|y_i\|}, \quad i = 1, \dots, m .$$

Then u_1, \dots, u_m is an orthonormal set of vectors, and for each $i = 1, \dots, m$, the sets x_1, \dots, x_i and u_1, \dots, u_i span the same subspace.

The above algorithm for obtaining the set u_1, \dots, u_m is the ordinary Gram-Schmidt process. For computational reasons, a slightly modified form of the algorithm is preferred. This algorithm, which is called the modified Gram-Schmidt process, is given below. The modified algorithm is entirely equivalent to the ordinary algorithm, theoretically, but is numerically more accurate when executed on a computer.

$$x_i^{(1)} = x_i, \quad i = 1, \dots, m$$

$$y_1 = x_1^{(1)}$$

$$u_1 = y_1 / \|y_1\|$$

$$x_i^{(2)} = x_i^{(1)} - (x_i^{(1)}, u_1)u_1, \quad i = 2, \dots, m$$

$$y_2 = x_2^{(2)}$$

$$u_2 = y_2 / \|y_2\|$$

⋮

$$x_i^{(k)} = x_i^{(k-1)} - (x_i^{(k-1)}, u_{k-1})u_{k-1}, \quad i = k, \dots, m$$

$$y_k = x_k^{(k)}$$

$$u_k = y_k / \|y_k\|$$

⋮

$$x_m^{(m)} = x_m^{(m-1)} - (x_m^{(m-1)}, u_{m-1})u_{m-1}$$

$$y_m = x_m^{(m)}$$

$$u_m = y_m / \|y_m\|$$

To illustrate an application of the Gram-Schmidt process which is of particular interest, let S be an arbitrary square root of an $n \times n$ non-negative matrix A , that is

$$A = SS^T .$$

Now denote the rows of S by the set of n -vectors x_1^T, \dots, x_n^T . Thus

$$S = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} .$$

Apply the Gram-Schmidt process to the set of vectors x_1, \dots, x_n , augmented if necessary by the columns of the $n \times n$ identity to obtain the orthonormal set u_1, \dots, u_n .

Augmentation with columns of I_n is necessary only if x_1, \dots, x_n is not linearly independent. In this case the Gram-Schmidt process is carried out as before, skipping any vector x_i which is linearly dependent on the set x_1, \dots, x_{i-1} . Only when the original set is exhausted does augmentation occur. Thus, in general, the orthonormalization process is applied to the set $x_1, \dots, x_n, e_1, \dots, e_n$, and terminates when a complete orthonormal set u_1, \dots, u_n is obtained.

Now let T be the orthogonal matrix whose columns are given by the set u_1, \dots, u_n . Thus

$$T = [u_1 | \dots | u_n] .$$

By the Gram-Schmidt construction, it follows that

$$ST = \begin{bmatrix} (x_1, u_1) & 0 & \dots & 0 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 0 \\ (x_n, u_1) & \dots & \dots & (x_n, u_n) \end{bmatrix} \equiv S' ,$$

that is S' is lower triangular. But since T is orthogonal, it also follows that

$$S'(S')^T = A ,$$

and thus S' is a lower triangular square root of A .

By simply applying the Gram-Schmidt process to the rows of A in the reverse order and similarly forming the columns of T in the reverse order, an upper triangular square root of A can be obtained. Thus with

$$S = \begin{bmatrix} x_n^T \\ \vdots \\ x_1^T \end{bmatrix}$$

and

$$T = [u_n | \dots | u_1] ,$$

it follows that

$$S' \equiv ST = \begin{bmatrix} (x_n, u_n) & \dots & \dots & (x_n, u_1) \\ 0 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ 0 & \dots & \dots & 0 & (x_1, u_1) \end{bmatrix}$$

is an upper triangular square root of A .