AD-A066 512 ARMY MATERIEL SYSTEMS ANALYSIS ACTIVITY ABERDEEN PROV--ETC F/G 17/9

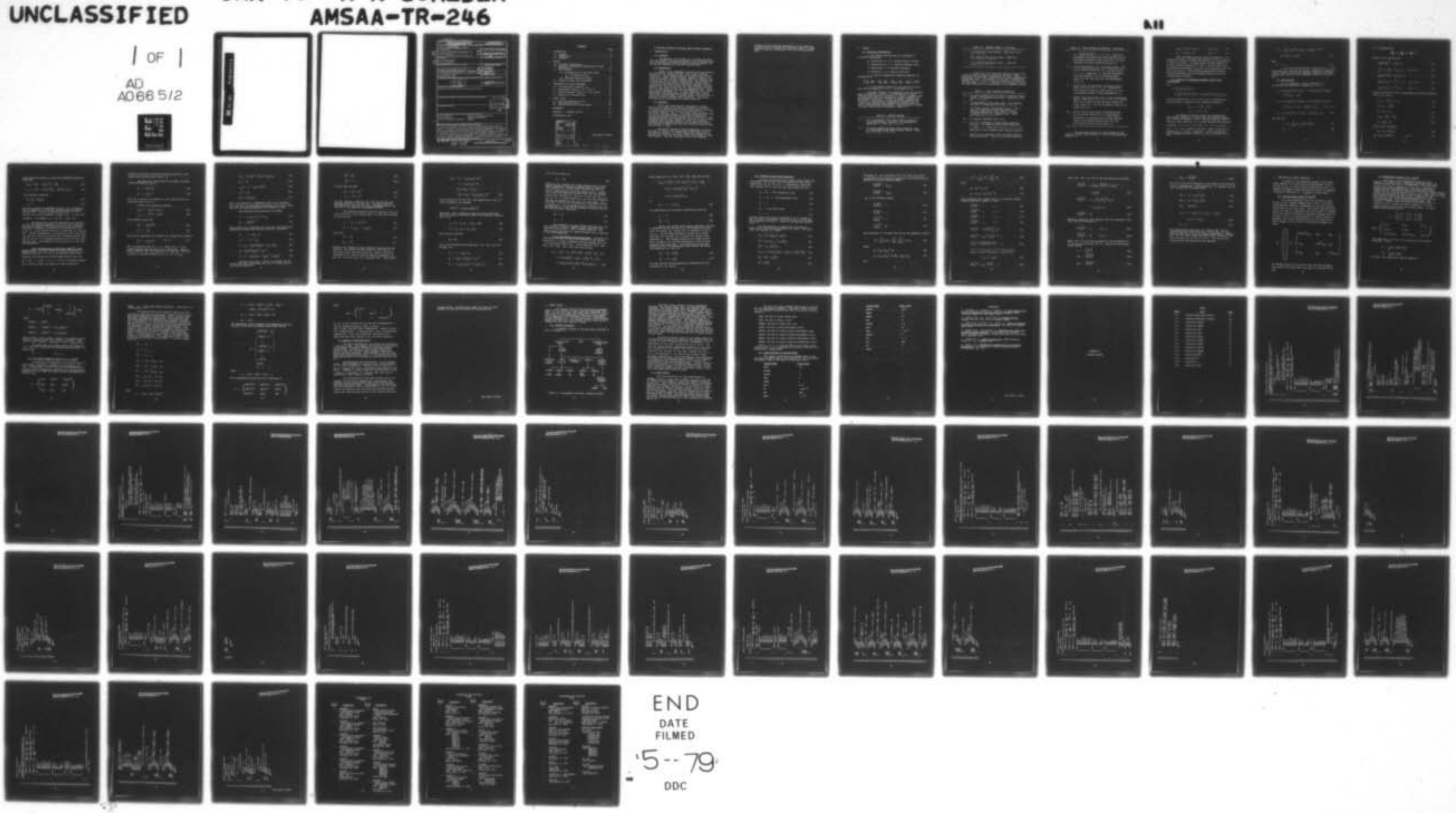
A COMPUTER PROGRAM FOR DOUBLE SWEEP OPTIMAL SMOOTHING.(U)

JAN 79 R A SCHEDER

UNCLASSIFIED AMSAA-TR-246

1 OF 1

AD A066 512

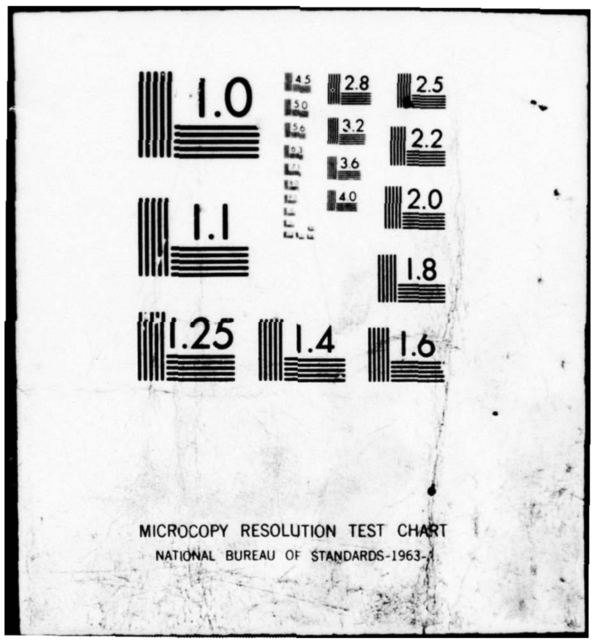


END

DATE FILMED

5--79

DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER (14) AMSAA-TR-
Technical Report No. 246

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
(6) A Computer Program for Double Sweep Optimal Smoothing.

5. TYPE OF REPORT & PERIOD COVERED
(9) Technical Rept.,

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
(10) Robert A. Scheder

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
US Army Materiel Systems Analysis Activity
Aberdeen Proving Ground, MD 21005

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
DA Project No.
(16) 1R765706M541

11. CONTROLLING OFFICE NAME AND ADDRESS
Commander
US Army Materiel Development and Readiness Command
5001 Eisenhower Avenue
Alexandria, VA 22333

12. REPORT DATE
(11) January 1979

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
(12) 74p.

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)

Filter Evaluation	Quadratic Programming Problem
Optimal Estimation	Double Sweep Smoothing
Top Down Structured Programming	FORTRAN

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

This report documents the double sweep smoothing program used at AMSAA to determine an aircraft's position, velocity, and acceleration from radar tracking data. It includes a tutorial guide to the underlying mathematics. A quadratic programming problem with linear constraints is formulated, reduced to a system of difference equations with initial and endpoint boundary conditions, and solved using Bryson's double sweep method. The iterative solution is programmed in simple, machine independent FORTRAN with top down structured programming. A listing is supplied and card decks are available.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

403 910

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

Next page is blank

A COMPUTER PROGRAM FOR DOUBLE SWEEP OPTIMAL SMOOTHING

1. INTRODUCTION

1.1 Purpose.

The purpose of this paper is to explain and document the computer routines developed by the author and used by the Army Materiel Systems Analysis Activity (AMSAA) for the optimal smoothing of aircraft tracking data.

1.2 Background.

AMSAA is the independent evaluator for the Division Air Defense Gun (DIVAD) program. One of the principal features of this program is the use of modern fire control techniques. These include optimal, Kalman filtering, routines to estimate aircraft (target) position, velocity, and acceleration from aircraft tracking data. One method used to study modern fire control consists of running target flight paths obtained during field tests through a digital simulation of the fire control and then comparing the estimates of the target state; that is, target position, velocity, and acceleration, with the corresponding "true" target state values (Reference 1). If this comparison is to be meaningful, the method used to obtain the "true" target state time history needs to provide greater accuracy than is possible to obtain through filtering alone. The method described below guarantees this needed accuracy.

1.3 Approach.

Field test tracking data consisting of target position as a function of time is processed in two steps. It is first filtered using a Kalman filter which is at least as sophisticated as the filter to be evaluated (e.g., nonlinear, adaptive, etc.). The target state estimates from the filter are then optimally corrected by sweeping in reverse chronological order through the tracking data. The corrected estimates are necessarily superior to the filtered estimates since at each instant of time, the corrected estimates were computed using all the tracking data while the filtered estimates were derived using only tracking data from earlier in time.

The general theory of optimal smoothing as considered here is fairly well established (References 2 and 3). The details, however, are often outside the background of analysts who could use (and have used) this program in other Army studies. Section 2 provides a brief but complete tutorial of the underlying mathematics of the smoothing program which will enable the interested reader to understand and modify the program to fill his own particular need.

2. THEORY

2.1 Principal Definitions.

- All vectors and matrices are expressed in cartesian coordinates.
- Superscript -1, $(^{-1})$, denotes matrix inverse.
- Superscript bar, $(\bar{\ })$, denotes expected value.
- Superscript T, $(^{T})$ denotes transpose.
- Subscript i, $(_{i})$, denotes time step.
- ∇, the 3n+1 dimensional gradiant operator, is defined to be

$$\left(\frac{\partial}{\partial X_0}, \frac{\partial}{\partial X_1}, \ldots \frac{\partial}{\partial X_n}, \frac{\partial}{\partial W_0}, \ldots \frac{\partial}{\partial W_{n-1}}, \frac{\partial}{\partial V_1}, \ldots \frac{\partial}{\partial V_n} \right) \text{; where}$$

n is the maximum number of time steps or the maximum value of subscript i (i.e., i goes from 0 to n).

The basic problem is to determine from the tracking data the position, velocity, and acceleration or state vector of the target as a function of time in a manner which is consistent with our knowledge of aircraft dynamics and measurement uncertainty. As will be shown in Subsections 2.2 and 2.3, the process of determining estimates of the target state requires that estimates of several other vectors be simultaneously determined. These vectors are termed "derived" vectors and are defined in Table 2-1.

TABLE 2-1 DERIVED VECTORS

X	:	The 9 dimensional (9-D) state vector consisting of target position, velocity, and acceleration. There are n + 1 x's, X_i, i = 0...n.
W	:	9-D state transition error vector which can also be interpreted as the state control vector. There are n W's, W_i, i = 0...n-1.

TABLE 2-1 DERIVED VECTORS (Continued)

V	:	3-D measurement error vector. There are n V's, V_i, i = 1...n.
λ	:	9-D, Lagrange Multiplier vector. There are n + 1 λ's, λ_i, i = 0...n.
μ	:	3-D, Lagrange Multiplier vector. There are n μ's, μ_i, i = 1...n.

In addition to the "derived" vectors, we will need to define several "input" vectors and matrices. These quantities are mathematic expressions of target dynamics and measurement error. The analysis in the next 3 subsections assumes that the "input" vectors and matrices are known for all time steps, i = 0...n. Section 3 will explain in detail how these quantities are constructed in the computer program.

TABLE 2-2 INPUT VECTORS AND MATRICES

$\overline{W}$	:	9-D state transition error mean (or nominal control) vector. The expected value of W. There are n $\overline{W}$'s, $\overline{W}_i$, i = 0...n-1.
$\overline{V}$	:	3-D measurement error mean vector. The expected value of V. There are n $\overline{V}$'s, $\overline{V}_i$, i = 1...n.
Z	:	3-D target position measurement vector. The Z data as a function of time, obtained from a tracking radar during field tests, is stored in digital format on a computer tape. There are n Z's, Z_i, i = 1...n.
$\overline{X}_o$	:	9-D apriori expected value of X_o.
Φ	:	9 x 9-D^2, nonsingular, target state transition matrices. Φ_i takes the state vector from time i to time i + 1. There are n Φ's, Φ_i, i = 0...n-1. Physically, the Φ_i embody basic aircraft dynamics.
Q	:	9x9-D^2 state transition error (or state control) covariance matrix. The Q_i are positive definite;

TABLE 2-2 INPUT VECTORS AND MATRICES (Continued)

		$Q_i = \overline{(W_i-\overline{W}_i)(W_i-\overline{W}_i)^T}$, i = 0...n-1. Physically, Q_i measures the amount that the expected value of the state transition error vector, $\overline{W}_i$, can vary from the true state transition error, W_i.
R	:	3x3-D[2] measurement error covariance matrix. The R_i are positive definite; $R_i = \overline{(V_i-\overline{V}_i)(V_i-\overline{V}_i)^T}$, i = 1...n. Physically, R_i measures the amount that the expected value of the measurement error, $\overline{V}_i$, can vary from the true measurement error, V_i.
H	:	3x9-D[2] state to measurement transition matrix. There are n H's, H_i, i = 1...n. Physically, the H_i matrix takes the state vector, X_i, into the coordinate frame of the measurements, Z_i.
Γ	:	9x9-D[2] state transition error to state transition matrix, There are n Γ's, Γ_i, i = 0...n-1. Physically, the Γ_i matrix takes the state transition error vector, W_i, into the coordinate frame of the state vector X_i.
$\overline{P}_o$	:	9x9-D[2] initial state error covariance matrix. $\overline{P}_o$ is positive definite; $\overline{P}_o = \overline{(X-\overline{X}_o)(X-\overline{X}_o)^T}$. Physically, $\overline{P}_o$ measures how much the expected value of the initial state vector, $\overline{X}_o$, can vary from the true initial state, X_o.

The above definitions are tied together by the dynamical target state equation (1) and by the measurement equation (2).

$$X_{i+1} = \Phi_i X_i + \Gamma_i W_i \qquad i=0...n-1 \qquad (1)$$

$$Z_i = H_i X_i + V_i \qquad i=1...n \qquad (2)$$

From the definitions of Q and R we expect W_i to be close to $\overline{W}_i$ when Q_i is small and similarly V_i to be close to $\overline{V}_i$ when R_i is small. In other words, we seek estimates of X_i which satisfy equations (1) and (2) and for which the corresponding estimates of W_i and V_i are consistent with the size of Q_i and R_i. The following formulation satisfies these requirements and has the added advantage of being mathematically solvable.

2.2 Quadratic Programming Problems (with Linear Constraints).

Let

$$L = 1/2(X_o-\overline{X}_o)^T \overline{P}_o^{-1}(X-X_o) +$$

$$1/2 \sum_{i=0}^{n-1} [(W_i-\overline{W}_i)^T Q_i^{-1}(W_i-\overline{W}_i) + (V_{i+1}-\overline{V}_{i+1})^T R^{-1}_{i+1}(V_{i+1}-\overline{V}_{i+1})]$$

find the estimates $\hat{X}_i$, $\hat{W}_i$, $\hat{V}_i$ of X_i, W_i, V_i respectively which minimize L subject to the constraint equations

$$\hat{X}_{i+1} = \Phi_i \hat{X}_i + \Gamma_i \hat{W}_i$$

$$Z_i = H_i \hat{X}_i + \hat{V}_i \qquad (3)$$

The Theory of Optimal Control and Mathematical Programming (Reference 3) proves that this type of problem has a solution whenever it has a feasible solution, that is, whenever there exists $\hat{X}_i$, $\hat{W}_i$, $\hat{V}_i$ which satisfy the constraint equations. $\hat{X}_i = 0$, $i = 0...n$, $\hat{W}_i = 0$, $i = 0...n$, and $\hat{V}_i = Z_i$, $i = 1...n$ is a feasible solution. Hence a solution to formulation (3) exists. Reference 3 further proves that there necessarily exists vectors λ_i and μ_i such that if we let

$$J = L + \sum_{i=1}^{n} [\lambda_{i-1}^T(\Phi_{i-1}X_{i-1} + \Gamma_{i-1}W_{i-1} - X_i) +$$

$$\mu_i^T (H_i X_i + V_i - Z_i)] \qquad (4)$$

Then

$$\nabla J = 0. \qquad (5)$$

J is called the Hamiltonian function and λ_i, μ_i are the traditional Lagrange multipliers (Reference 2 supplies proof). Since L is a convex function, equation (5) is also a sufficient condition for a feasible solution to be optimal.

2.3 The Solution.

We can highlight a useful property of J by defining the Hamiltonian sequence J_i, i=0...n.

$$J_o = 1/2(X_o-\overline{X}_o)^T\overline{P}_o^{-1}(X_o-\overline{X}_o) + 1/2(W_o-\overline{W}_o)^T Q_o^{-1}(W_o-\overline{W}_o)$$

$$+ \lambda_o^T(\Phi_o X_o + \Gamma_o W_o)$$

$$J_i = 1/2(W_i-\overline{W}_i)^T Q_i^{-1}(W_i-\overline{W}_i) + 1/2(V_i-\overline{V}_i)^T R_i^{-1}(V_i-\overline{V}_i)$$

$$+ \lambda_i^T(\Phi_i X_i + \Gamma_i W_i) + \mu_i^T(H_i X_i + V_i) \qquad (i=1...n-1)$$

$$J_n = 1/2(V_n-\overline{V}_n)^T R_n^{-1}(V_n-\overline{V}_n) + \mu_n^T(H_n X_n + V_n) \qquad (6)$$

We find that

$$J = J_o + \sum_{i=1}^{n} [J_i - \lambda_{i-1}^T X_i - \mu_i^T Z_i] \qquad (7)$$

$$\lambda_n = 0. \qquad (8)$$

$\nabla J = 0$ implies that

$$\frac{\partial J}{\partial X_i} = 0, \; \frac{\partial J}{\partial W_i} = 0, \; \frac{\partial J}{\partial V_i} = 0$$

which in turn implies that

$$(\hat{X}_0 - \overline{X}_0)^T \overline{P}_0^{-1} + \lambda_0^T \Phi_0 = 0 \tag{9}$$

$$\lambda_i^T \Phi_i + \mu_i^T H_i - \lambda_{i-1}^T = 0 \qquad i=1...n-1 \tag{10}$$

$$(\hat{W}_0 - \overline{W}_0)^T Q_0^{-1} + \lambda_0^T \Gamma_0 = 0 \tag{11}$$

$$(\hat{W}_i - \overline{W}_i)^T Q_i^{-1} + \lambda_i^T \Gamma_i = 0 \qquad i=1...n-1 \tag{12}$$

$$(\hat{V}_i - \overline{V}_i)^T R_i^{-1} + \mu_i^T = 0 \qquad i=1...n \tag{13}$$

Hence we arrive at a set of simultaneous difference equations for $\hat{X}_i$, $\hat{W}_i$, and $\hat{V}_i$.

$$\lambda_{i-1} = H_i^T \mu_i + \Phi_i \lambda_i \tag{14}$$

$$\hat{W}_i = \overline{W}_i - Q_i \Gamma_i^T \lambda_i \tag{15}$$

$$\hat{V}_i = \overline{V}_i - R_i \mu_i \tag{16}$$

$$\hat{X}_{i+1} = \Phi_i \hat{X}_i + \Gamma_i \hat{W}_i \tag{17}$$

$$Z_i = H_i \hat{X}_i + \hat{V}_i \tag{18}$$

with initial condition;

$$\hat{X}_0 = \overline{X}_0 - \overline{P}_0 \Phi_0^T \lambda_0 \tag{19}$$

and final condition $\lambda_n = 0$ (20)

These equations combine to yield our fundamental system of equations;

$$\hat{X}_{i+1} = \Phi_i \hat{X}_i - \Gamma_i Q_i \Gamma_i^T \lambda_i + \Gamma_i \overline{W}_i \tag{21}$$

$$\lambda_{i-1} = \Phi_i^T \lambda_i + H_i^T R_i^{-1} H_i \hat{X}_i - H_i^T R_i^{-1} (Z_i - \overline{V}_i) \tag{22}$$

with boundary conditions

$$\hat{X}_0 = \overline{X}_0 - \overline{P}_0 \Phi_0^T \lambda_0 \tag{23}$$

$$\lambda_n = 0 \tag{24}$$

Hence the quadratic programming problem, (3), is reduced to solving equations (21) and (22) with boundary conditions (23) and (24). Note all other derived vectors in (14) through (20) can be expressed in terms of $\hat{X}_i$, λ_i, and input variables. For example, $\mu_i = -R_i^{-1}(Z_i - H_i \hat{X}_i - \overline{V}_i)$.

The difficulty in solving equations (21) and (22) is that the boundary conditions, (23) and (24), are split -- one initial condition and one end point or final condition. Reference 4 suggests a technique for obtaining the final value, $\hat{X}_n$, of the optimal solution to equations (21) through (24). Having determined $\hat{X}_n$, we replace the split conditions (23) and (24) with the two end point conditions and can then find $\hat{X}_i$ for all i by iterating equations (21) and (22) backwards in time. The mathematics are carried out in the next two subsections.

<u>2.3a Determination of the Final State $\hat{X}_n$ - The Kalman Filter</u>. We first establish certain properties of a homogenous solution X_i^h, and a non-homogenous particular solution, X_i^p, whose sum is the optimal solution $\hat{X}_i$ (i.e., $X_i^h + X_i^p = \hat{X}_i$ for i = 0...n). We do not actually compute X_i^h or X_i^p but rather use certain of their analytical

properties to derive recursive relations which will ultimately yield the end point vector, $\hat{X}_n$.

For notational convenience we introduce the additional variables A_i and B_i.

$$A_i = H_i^T R_i^{-1} H_i \quad (25)$$

$$B_i = \Gamma_i Q_i \Gamma_i^T \quad (26)$$

Note the A_i and B_i are composed of input matrices and are therefore known for all i.

Consider the homogenous problem;

$$X_{i+1}^h = \Phi_i X_i^h - B_i \lambda_i^h \quad (27)$$

$$\lambda_{i-1}^h = \Phi^T{}_i \lambda^h{}_i + A_i X_i^h \quad (28)$$

with boundary conditions

$$X_0^h = -\bar{P}_0 \Phi_0^T \lambda_0^h \quad (29)$$

$$\lambda_n^h = -\lambda_n^P. \quad (30)$$

The form of equations (25) through (29) suggests the relation

$$X_i^h = -P_i \Phi_i^T \lambda_i^h \qquad i=0\ldots n \quad (31)$$

This is certainly true for i = 0. The P_i for i > 0 are proportionately matrices still to be determined. Substituting relation (31) into equations (27) and (28), we arrive directly at uncoupled equations for P_i, λ_i^h, and X_i^h.

$$P_{i+1} = [\Phi_i P_i \Phi_i^T + B_i][I - A_{i+1}P_{i+1}] \qquad (32)$$

$$P_0 = \bar{P}_0 \qquad (33)$$

$$\lambda_{i-1}^h = [I - A_i P_i]\, \Phi_i^T \lambda_i^h \qquad (34)$$

$$\lambda_n^h = -\lambda_n^p \qquad (35)$$

$$X_i^h = -P_i \Phi_i^T \lambda_i^h \qquad (36)$$

The P_i matrices play a fundamental role in our analysis. Note that they are completely determined from initial condition (33) by iterating equation (32) forward in time.

The following definitions will be useful:

$$\tilde{P}_i = \Phi_{i-1} P_{i-1} \Phi_{i-1}^T + B_{i-1} \qquad (37)$$

$$K_i = P_i H_i^T R_i^{-1} \qquad (38)$$

These definitions, equations (32) and (33), and some matrix manipulation yields the following important relations;

$$P_i = [I - P_i A_i]\, \tilde{P}_i \qquad (39)$$

$$P_i^{-1} = \tilde{P}_i^{-1} + A_i \qquad (40)$$

$$P_i = \tilde{P}_i - \tilde{P}_i H_i^T [H_i \tilde{P}_i A_i^T + R_i]^{-1} H_i \tilde{P}_i \qquad (41)$$

$$K_i = \tilde{P}_i H_i^T [H_i \tilde{P}_i H_i^T + R_i]^{-1} \qquad (42)$$

$$P_i = (I - K_i H_i)\tilde{P}_i (I - K_i H_i)^T + K_i R_i K_i^T \qquad (43)$$

We shall also find it useful to consider the particular solution to equations (21) and (22) which satisfy the initial condition

$$x_0^p = \bar{x}_0 \tag{44}$$

$$\lambda_0^p = 0 \tag{45}$$

In this case the sums

$$\hat{x}_i = x_i^h + x_i^p \tag{46}$$

$$\lambda_i = \lambda_i^h + \lambda_i^p \tag{47}$$

not only satisfy equations (21) and (22) but also the boundary conditions (23) and (24). In other words the sums defined in (46) and (47) are a representation of the solution to the quadratic programming problem.

The promised recursion equation leading to the end point vector $\hat{x}_n$ is now within reach. To this end we define the new vectors;

$$y_i = x_i^p + P_i \Phi_i^T \lambda_i^p \tag{48}$$

$$\tilde{y}_i = \Phi_{i-1} y_{i-1} + \Gamma_{i-1} \bar{w}_{i-1} \tag{49}$$

Note that

$$y_0 = \bar{x}_0 \tag{50}$$

$$y_n = \hat{x}_n \tag{51}$$

Equation (51) assures us that finding an equation for computing y_i (by forward iteration from y_0) will lead us to the value of the end point vector $\hat{x}_n$. Such an iterative expression for y_i can be found by applying equations (21) and (22) to the x_i^p and λ_i^p in equation (48). This yields

$$Y_{i+1} = [\Phi_i - P_{i+1}A_{i+1}\Phi_i]\, X_i^p +$$

$$[\Gamma_i - P_{i+1}A_{i+1}\Gamma_i]\, \overline{W}_i +$$

$$K_{i+1}\, [Z_{i+1} - \overline{V}_{i+1}] +$$

$$[-B_i + P_{i+1} + P_{i+1}A_{i+1}B_i]\, \lambda_i^p \quad (52)$$

Using relations (37) and (39), the coefficient of the λ_i^p term in equation (52) becomes

$$[\Phi_i P_i \Phi_i^T - P_{i+1}A_{i+1}\Phi_i P_i \Phi_i^T]$$

Therefore, after recombining terms and using definition (49), we arrive at the final form of the iterative equation for the Y_i;

$$Y_i = \tilde{Y}_i + K_i\, [Z_i - H_i\tilde{Y}_i] - K_i\overline{V}_i \quad (53)$$

$$\tilde{Y}_i = \Phi_{i-1}\, Y_{i-1} + \Gamma_{i-1}\, \overline{W}_{i-1} \quad (54)$$

with initial condition

$$Y_0 = \overline{X}_0 \quad (55)$$

the K_i are determined from equations (32), (33), and (37) through (43);

$$P_i = (I - K_iH_i)\, \tilde{P}_i \quad (56)$$

$$K_i = \tilde{P}_iH_i^T\, [H_i\tilde{P}_iH_i^T + R_i]^{-1} \quad (57)$$

$$\tilde{P}_i = \Phi_{i-1}P_{i-1}\Phi_{i-1}^T + \Gamma_{i-1}Q_i\Gamma_{i-1}^T \quad (58)$$

with initial condition

$$P_0 = \bar{P}_0 \tag{59}$$

Equations (53) through (59) taken together form a closed system completely solvable by simple forward iteration. They comprise our "forward sweep" through the data, Z_i, and are traditionally called the Kalman filter. Clearly, the Y_k have the important property that they are optimal target state estimates at time i=k if given only data, Z_i, for i=0...k. In other words the Y_i are the best estimates that can be obtained on line (i.e., in real time). The variables Y_i and P_i will be physically interpreted further in section 2.4. For the moment we are concerned with solving equations (21) and (22) by backward iteration starting with the end point conditions

$$\hat{X}_n = Y_n \tag{60}$$

$$\lambda_n = 0 \tag{61}$$

One approach is to just iterate equations (21) and (22) directly. Another, computationally more efficient approach utilizes the Y_i and P_i already computed. This latter method comprises the "Backward Sweep" and is detailed in the next subsection.

<u>2.3b The Backward Correcting Sweep</u>. Utilizing the preceeding analysis we can arrive directly at the optimal solution $\hat{X}_i$, λ_i using a backward iteration scheme where the λ_i are uncoupled from the $\hat{X}_i$. Using equations (22), (39), (40), (41), (42), (43), (48), and (53) we find

$$\lambda_{i-1} = \Phi_i^T \lambda_i + A_i \, [X_i^h + X_i^p] - H_i^T R_i^{-1} \, [Z_i - \bar{V}_i]$$

$$= [I - A_i P_i][\Phi_i^T \lambda_i + A_i Y_i - H_i^T R_i^{-1} (Z - \bar{V}_i)]$$

$$= [I - A_i P_i][\Phi_i^T \lambda_i - H_i^T R_i^{-1} (Z_i - \bar{V}_i - H_i \tilde{Y}_i)]. \tag{62}$$

Using equations (21), (34), (37), (39), and (48) we find

$$\hat{X}_{i+1} = \Phi_i(X_i^h + X_i^p) - B_i(\lambda_i^h + \lambda_i^p) + \Gamma_i\overline{W}_i$$

$$= Y_{i+1} - P_{i+1}\Phi_{i+1}^T\lambda_{i+1}^p - \tilde{P}_{i+1}\lambda_i^h$$

$$= Y_{i+1} - P_{i+1}\Phi_{i+1}^T\lambda_{i+1}$$

or

$$\hat{X}_i = Y_i - P_i\Phi_i^T\lambda_i \tag{63}$$

For completeness the end point conditions are written

$$\lambda_n = 0 \tag{64}$$

$$\hat{X}_n = Y_n \tag{65}$$

Hence, the "Double Sweep Optimal Smoother" consists of sweeping through the data Z_i in the forward direction using the system (53) through (59) to determine P_i and Y_i $i=0...n$. We then sweep backwards through the Z_i using equations (62) through (65) to obtain the optimal state estimates $\hat{X}_i$. This backward sweep can be viewed as optimally correcting the filtered estimates Y_i with the data received after time step i. The estimates of the state transition error and measurement error vectors are found from equations (15) and (16)

$$\hat{W}_i = \overline{W}_i - Q_i\Gamma_i^T\lambda_i \tag{66}$$

$$\hat{V}_i = Z_i - H_i\hat{X}_i\ . \tag{67}$$

The next section derives measures of confidence for the estimates $\hat{X}_i$, $\hat{W}_i$, and $\hat{V}_i$.

2.4 Complete Double Sweep Equations.

If the true values of the initial state vector, the measurement error vector, and the transition error vector equalled $\overline{X}_0$, $\overline{V}_i$, $\overline{W}_i$ (for all i), respectively, equations (14) through (20) or (21) and (22) would imply for all i,

$$\hat{W}_i = \overline{W}_i = \text{true transition error} \tag{68}$$

$$\hat{V}_i = \overline{V}_i = \text{true measurement error} \tag{69}$$

$$\lambda_i = 0 \tag{70}$$

$$\hat{X}_i = \text{true state vector} \tag{71}$$

$$L = 0 \tag{72}$$

In other words, the optimal estimates $\hat{X}$, $\hat{W}$, $\hat{V}$, necessarily take on their corresponding true values since in this case the performance criteria, L, takes on its smallest possible value, 0.

Let the operator, δ, denote the variation of a vector value from the corresponding true value. Then equations (62) through (72) imply,

$$\delta Y_i = \delta \tilde{Y}_i - K_i H_i \delta \tilde{Y}_i - K_i \delta \overline{V}_i \tag{73}$$

$$\delta \tilde{Y}_i = \Phi_{i-1} \delta Y_{i-1} + \Gamma_{i-1} \delta \overline{W}_{i-1} \tag{74}$$

$$\delta \hat{X}_i = \delta Y_i - P_i \Phi_i^T \delta \lambda_i \tag{75}$$

$$\delta \lambda_{i-1} = [I - A_i P_i][\Phi_i \delta \lambda_i + A_i \delta Y_i + H_i^T R_i^{-1} \delta \overline{V}_i] \tag{76}$$

$$\delta \hat{W}_i = \delta \overline{W}_i - Q_i \Gamma_i^T \delta \lambda_i \tag{77}$$

$$\delta \hat{V}_i = H_i \delta \hat{X}_i \tag{78}$$

We assume that the variations of $\overline{V}_i$, $\overline{W}_i$ and $\overline{X}_0$ from their corresponding true values are given as in the definitions of Section 2.1 and are entirely random;

$$\overline{\delta\overline{V}_i \delta\overline{V}_i^T} = R_i \delta_{ij} \tag{79}$$

$$\overline{\delta\overline{W}_i \delta\overline{W}_j^T} = Q_i \delta_{ij} \tag{80}$$

(δ_{ij} is the Krenecker delta)

$$\overline{\delta\overline{V}_i \delta\overline{W}_j^T} = 0 \tag{81}$$

$$\overline{\delta\overline{V}_i \delta\overline{X}_0^T} = 0 \tag{82}$$

$$\overline{\delta\overline{W}_i \delta\overline{X}_0^T} = 0 \tag{83}$$

$$\overline{\delta\overline{X}_0 \delta\overline{X}_0^T} = \overline{P}_0 \, . \tag{84}$$

From equations (73) through (84) we get the important results,

$$\delta Y_i = \prod_{k=0}^{i-1} C_k \delta Y_0 + \sum_{k=0}^{i-1} \left(\prod_{\ell=k+1}^{i-1} C_\ell \right) c_k \tag{85}$$

where

$$C_k = P_{k+1} \tilde{P}_{k+1}^{-1} \Phi_k \tag{86}$$

$$c_k = P_{k+1} \tilde{P}_{k+1}^{-1} \Gamma_k \overline{W}_k - K_{k+1} \overline{V}_{k+1} \tag{87}$$

and

$$\delta\lambda_i^T = \sum_{\ell=i+1}^{n} d_\ell^T \left(\prod_{k=i+1}^{\ell-1} D_k \right) \tag{88}$$

where

$$D_k = \tilde{P}_k^{-1} P_k \Phi_k^T \tag{89}$$

$$d_k = A_k Y_k + H_k^T R_k^{-1} \overline{V}_k \tag{90}$$

Using equations (85) through (90) it is straight forward, although sometimes lengthy, to show;

$$\overline{\delta Y_i \delta \overline{V}_\ell^T} = 0 \qquad \ell > i \tag{91}$$

$$\overline{\delta \tilde{Y}_i \delta \overline{V}_\ell^T} = 0 \qquad \ell \geq i \tag{92}$$

$$\overline{\delta Y_i \delta \overline{W}_\ell^T} = 0 \qquad \ell \geq i \tag{93}$$

$$\overline{\delta \tilde{Y}_i \delta \overline{W}_\ell^T} = 0 \qquad \ell \geq i \tag{94}$$

$$\overline{\delta Y_i \delta Y_\ell^T} = \overline{\delta Y_i \delta Y_i^T} \prod_{k=1}^{\ell-1} C_k^T \tag{95}$$

$$\overline{\delta Y_i \delta \lambda_i^T} - P_i \Phi_i^T \overline{\delta \lambda_i \delta \lambda_i^T} = 0 \tag{96}$$

$$\overline{(A_i \delta\lambda_i + H_i^T R_i^{-1} \delta \overline{V}_i) \delta\lambda_i^T} = 0 \tag{97}$$

$$\overline{[A_i \delta\lambda_i + H_i^T R_i^{-1} \delta \overline{V}_i][A_i \delta Y_i + H_i^T R_i^{-1} \delta \overline{V}_i]^T}$$

$$= A_i (I - P_i A_i) \tag{98}$$

$$\overline{\delta \overline{W}_\ell \delta\lambda_i^T} - Q_i \Gamma_i^T \overline{\delta\lambda\delta\lambda^T} = 0 \tag{99}$$

Using (91), (92), and (95) we get the recursive relations;

$$\overline{\delta Y_i \delta Y_i^T} = (I-K_i H_i)\ \overline{\delta \tilde{Y}_i \delta \tilde{Y}_i^T}\ (I-K_i H_i)^T$$

$$+ K_i R_i K_i^T \tag{100}$$

$$\overline{\delta \tilde{Y}_i \delta \tilde{Y}_i^T} = \Phi_{i-1}\ \overline{\delta Y_{i-1} \delta Y_{i-1}^T}\ \Phi_{i-1}^T$$

$$+ \Gamma_{i-1} Q_{i-1} \Gamma_{i-1} \tag{101}$$

$$\overline{\delta Y_0 \delta Y_0^T} = \overline{P}_0 \tag{102}$$

Comparing equations (100) through (102) with equations (33), (37), and (43) we arrive at

$$\overline{\delta Y_i \delta Y_i^T} = P_i \qquad i=0\ldots n \tag{103}$$

$$\overline{\delta \tilde{Y}_i \delta \tilde{Y}_i^T} = \tilde{P}_i \qquad i=1\ldots n \tag{104}$$

Hence, the P ($\tilde{P}$) are the covariance of the variations in Y ($\tilde{Y}$) due to the variations $\delta\overline{V}$, $\delta\overline{W}$. Continuing along these lines, define;

$$\Lambda_i = \overline{\delta\lambda_i \delta\lambda_i^T} \tag{105}$$

$$SP_i = \overline{\delta\hat{X}_i \delta\hat{X}_i^T} \tag{106}$$

$$SQ_i = \overline{\delta\hat{W}_i \delta\hat{W}_i^T} \tag{107}$$

$$SR_i = \overline{\delta\hat{V}_i \delta\hat{V}_i^T} \quad (108)$$

SP, SQ, SR are the covariances of the errors in the estimates $\hat{X}$, $\hat{W}$, $\hat{V}$, respectively. Using equations (73) through (78) and (91) through (99) we easily find;

$$SP_i = P_i - P_i \Phi_i^T \Lambda_i \Phi_i P_i \quad (109)$$

$$SQ_i = Q_i - Q_i \Gamma_i \Lambda_i \Gamma_i^T Q_i \quad (110)$$

$$SR_i = H_i SP_i H_i^T \quad (111)$$

$$\Lambda_{i-1} = (I - P_i A_i)^T \Phi_i^T \Lambda_i \Phi_i (I - P_i A_i) + A_i (I - P_i A_i) \quad (112)$$

$$\Lambda_n = 0 \quad (113)$$

The forward sweep equations, (53) through (59), and the complete backward sweep equations, (62) through (67) and (109) through (113), form the "Double Sweep Optimal Smoother." Note that the double sweep equations require values for the input variables $\overline{X}_0$, $\overline{P}_0$, $\overline{W}$, $\overline{V}$, R and Q. The next section describes how these quantities are computed.

3. COMPUTATION OF INPUT VARIABLES

In order for the iterative process of the double sweep smoother to be initialized and maintained, the $\overline{X}_0$, $\overline{P}_0$, $\overline{W}_i$, $\overline{V}_i$, R_i, Q_i, and Φ_i must be supplied for all i. A complete analysis of these quantities is given in "Adaptive Estimation" (Reference 5). A summary is given below.

3.1 Initialization Inputs $\overline{P}_0$ and $\overline{X}_0$.

All nine components of $\overline{X}_0$ are set equal to zero. $\overline{P}_0$ has all off diagonal elements set to zero. Its first three diagonal elements, corresponding to initial position uncertainty are set very large. The initial velocity uncertainty is large but recognizes that the aircraft is subsonic while the initial acceleration uncertainty assumes that the aircraft is not in a severe maneuver at initial detection. Smoother (and filter) estimates are not very sensitive to small chnages in $\overline{P}_0$ as long as the diagonal elements are large.

$$\overline{X}_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \overline{P}_0 = \begin{pmatrix} (3000)^2 I_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & (240)^2 I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & (25)^2 I_{3x3} \end{pmatrix}$$

The assumed values of the initial state and the corresponding blocks of $\overline{P}_0$ given above can easily be changed by the user.

3.2 Measurement Inputs R, H, Z, and $\bar{V}$.

Since most of the tracking data that we use is supplied in a fixed inertial cartesian coordinate frame, the smoother assumes that the Z_i are expressed in the same frame as the state vector, although it can be easily modified to account for any measurement frame. The model of the measurement errors in the program accounts for constant variance angular errors and also constant variance linear, glint type, errors in both azimuth and elevation and for a fixed variance linear error in range. The covariance matrix of these errors is computed in the radar line of sight frame and then expressed, via a similarity transformation, in the fixed reference frame. The reference to sensor frame coordinate transformation TRS_i is computed at each i using the latest estimate, Y_i, of the target position vector during the forward sweep. The measurement related matrices are;

$$H_i = \begin{pmatrix} 1\,0\,0 & 0\,0\,0 & 0\,0\,0 \\ 0\,1\,0 & 0\,0\,0 & 0\,0\,0 \\ 0\,0\,1 & 0\,0\,0 & 0\,0\,0 \end{pmatrix}$$

$$TRS_i = \begin{pmatrix} {}^1Y \div |Y| & {}^2Y \div |Y| & {}^3Y \div |Y| \\ -{}^2Y \div |Yg| & {}^1Y \div |Yg| & 0 \\ -{}^1Y\,{}^3Y \div |Y|\,|Yg| & -{}^2Y\,{}^3Y \div |Y|\,|Yg| & |Yg| \div |Y| \end{pmatrix}$$

(the subscript i's on the Y's are omitted for notational convience). Where

$$|Y| = \sqrt{{}^1Y^2 + {}^2Y^2 + {}^3Y^2}$$

$$|Yg| = \sqrt{{}^1Y^2 + {}^2Y^2}$$

and upper left superscript denotes component.

$$R_i = TRS_i^T \begin{pmatrix} sigrxs_i & 0 & 0 \\ 0 & sigrys_i & 0 \\ 0 & 0 & sigrzs_i \end{pmatrix} TRS_i$$

where

$$sigrxs_i = sigmr^2$$

$$sigrys_i = (stdaz1)^2 + (|Y_i|stdaz2)^2$$

$$sigrzs_i = (stdel1)^2 + (|Y_i|stdel2)^2.$$

sigmr, stdaz1, stdaz2, stdel1, stdel2, are constants set by the user. Nominal values are 5(meters), 1(meter), .0005 (mils ÷ 1000), 1(meter), .0005(mils ÷ 1000), respectively.

We assume that the tracking radar which generated the Z_i was well calibrated and therefore had zero bias. In other words,

$$\overline{V}_i = 0 \qquad i=1...n.$$

3.3 The Target Dynamical Inputs Φ, Q, Γ, and $\overline{W}$.

Φ_i represents the dynamics of an aircraft which is controlled at time step i through its acceleration rate, $\Gamma_i W_i$. Alternatively, we can look on Φ_i as representing an aircraft whose acceleration is constant over any particular transition, $i \rightarrow i+1$, where the error in this representation is given by $\Gamma_i W_i$.

$$\Phi_i = \begin{pmatrix} I_{3x3} & I_{3x3}\Delta t & .5I_{3x3}\Delta t^2 \\ O_{3x3} & I_{3x3} & I_{3x3}\Delta t \\ O_{3x3} & O_{3x3} & I_{3x3} \end{pmatrix}$$

where Δt = time step between iterations. Note that Φ is constant over all i.

The computations of the inputs $\overline{W}_i$ and Q_i are complicated in that they are physically related to the aircraft body frame variables of thrust rate, g rate, and roll rate and their corresponding covariances. The transition from body frame dynamics to reference frame dynamics is worked out in Reference 5. At any particular instant, a circle is matched to the target's flight path with the properties that the component of target acceleration perpendicular to the velocity vector points towards the circle's center. The target's body frame has its first component axis along the velocity vector and its second component axis along the normal acceleration vector. It is also called the Frenet frame. Γ is the body, or Frenet frame, to reference frame coordinate transformation. It is computed at each i based on the latest available state estimate. Γ's matrix components are;

$$^{11}\Gamma = {}^{4}Y \div V$$

$$^{21}\Gamma = {}^{5}Y \div V$$

$$^{31}\Gamma = {}^{6}Y \div V$$

$$^{12}\Gamma = ({}^{7}Y - {}^{4}Y\ at) \div gn$$

$$^{22}\Gamma = ({}^{8}Y - {}^{5}Y\ at) \div gn$$

$$^{32}\Gamma = ({}^{9}Y - {}^{6}Y\ at) \div gn$$

$$^{13}\Gamma = {}^{21}\Gamma\ {}^{32}\Gamma - {}^{31}\Gamma\ {}^{22}\Gamma$$

$$^{23}\Gamma = {}^{12}\Gamma\ {}^{31}\Gamma - {}^{11}\Gamma\ {}^{32}\Gamma$$

$$^{33}\Gamma = {}^{11}\Gamma\ {}^{22}\Gamma - {}^{12}\Gamma\ {}^{21}\Gamma$$

where

$$V = ({}^{4}Y^2 + {}^{5}Y^2 + {}^{6}Y^2)^{1/2}$$

$$k = [({}^5Y^9Y - {}^6Y^8Y)^2 + ({}^4Y^9Y - {}^7Y^6Y)^2$$

$$+ ({}^4Y^8Y - {}^5Y^7Y)^2]^{1/2} \div V^3$$

$$at = ({}^4Y^7Y + {}^5Y^8Y = {}^6Y^9Y) \div V^2$$

$$gn = kV^2.$$

The computation of $\overline{W}$ is based on the assumption that the aircraft flies along the Frenet circle over time Δt.

$$\overline{W} = \begin{pmatrix} -k^2V^3\Delta t^3 \div 6 \\ k\dot{V}V\Delta t^3 \div 6 \\ 0 \\ -k^2V^3\Delta t^2 \div 2 \\ k\dot{V}V\Delta t^2 \div 2 \\ 0 \\ -k^2V^3\Delta t \\ k\dot{V}V\Delta t \\ 0 \end{pmatrix}$$

where

$$\dot{V} = ({}^4Y^7Y + {}^4Y^8Y + {}^6Y^9Y) \div V.$$

The corresponding transition error covariance is

$$Q = \begin{pmatrix} QAP\Delta t^4 \div 36 & QAP\Delta t^3 \div 12 & QAP\Delta t^2 \div 2 \\ QAP\Delta t^3 \div 12 & QAP\Delta t^2 \div 2 & QAP\Delta t \\ QAP\Delta t^2 \div 2 & QAP\Delta t & QAP \end{pmatrix}$$

where

$$QAP = \begin{pmatrix} (C_1\Delta t)^2 & 0 & 0 \\ 0 & (C_2\Delta t)^2 & 0 \\ 0 & 0 & (gn*C_3\Delta t)^2 \end{pmatrix} .$$

C_1, C_2, C_3 are constants set by the user corresponding to the one standard deviation (STD) of thrust rate, g rate, and roll rate, respectively. Nominal values are 1.0(meters/sec^3), 7.0(meters/sec^3), and 1.0(radians/sec respectively. Note, for notational convenience the i subscripts have been omitted in all the above equations.

3.4 Remarks on Nonlinearities.

The input variables can in principle be defined in a variety of ways. For example, they could each be referred to any number of possibly different coordinate systems. Unfortunately, the nature of the tracking problem will result in any, but the most simple minded, formulation having nonlinear portions. The investigator can influence where these nonlinearities occur. In our formulation they have been shifted to the coordinate transition matrices TRS and Γ.

Nonlinearities have two effects. On the forward sweep, Γ and TRS depend on the most recent state estimates, Y_i, for their computation. Therefore the P_i cannot be a priori computed but must be determined simultaneously with the Y_i. Secondly, we must change our interpretation of the δ operator in equations (73) through (108) from that of variation to that of small variation.

The nonlinearities raise some question as to the adequacy of the $\overline{W}$ and Q computations, particularly for severely jinking targets where lags in the forward sweep estimates may become large. Conceivably we can reduce the lags by increasing Q, that is increasing C_1, C_2, C_3, and/or running the smoother back and forth over the data several times, each time starting with $\overline{X}_0$ and $\overline{P}_0$ from the previous backward sweep. Doubtless the reader can think of other schemes for reducing the effects of nonlinearities.

Next page is blank

4. USERS' GUIDE

The program uses Top Down Structured Programming (TDSP) and is written in simple, machine independent FORTRAN. In line with TDSP, the coding uses only five logical structures, is self-documented (requiring no flow charts) and is completely straightforward (avoiding "programming tricks")(Reference 6). A brief description of the program's overall structure and a few remarks on input data are all a perspective user will need for a guide.

4.1 Overall Structure.

A schematic diagram of the functional hierarchy is given in Figure 4.1.

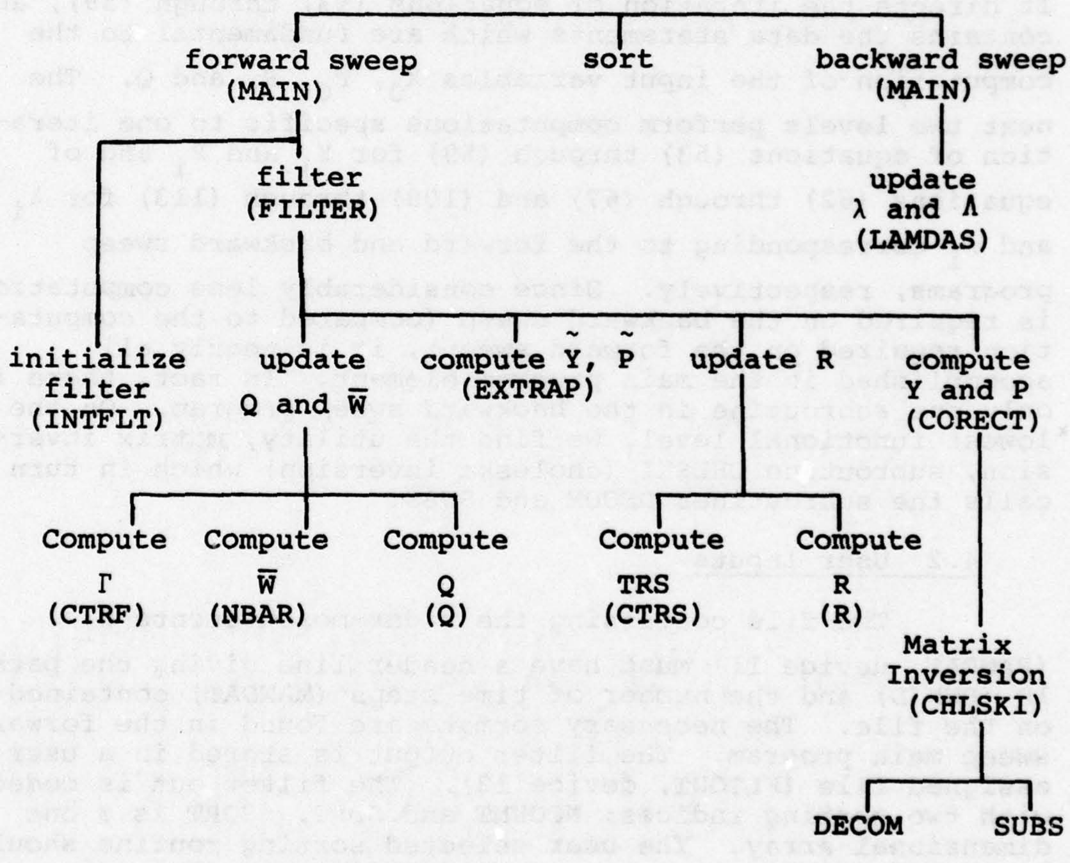


Figure 4.1 Programming Structure (Subroutine Name)

The first level consists of three independent programs, the forward sweep, sort, and backward sweep programs. The forward sweep reads the tracking data from a user selected storage device (e.g., FASTRAND, tape, disc), filters it, and writes the results on user selected devices. The sort program reverses the chronological order of the filtered output in preparation for the backward sweep. The user is urged to consult his own computer facility for a machine optimized sort routine. For example, AMSAA's Air Defense Evaluation Branch uses a sort program optimized for the Univac 1108 consisting of one executive command, @MISD* LIB$.SORTSDF 12., 12.,#records, #chars/per, KEY/127/4.D, KEY/131/2.A. The backward sweep reads the filtered output in reverse chronological order and writes the final smoothed results on user selected devices. The user may wish to use SORT again since the smoothed results are in reverse chronological order.

The next functional level is the FILTER subroutine. It directs the iteration of equations (53) through (59), and contains the data statements which are fundamental to the computation of the input variables $\overline{X}_0$, $\overline{P}_0$, R, and Q. The next two levels perform computations specific to one iteration of equations (53) through (59) for Y_i and P_i and of equations (62) through (67) and (109) through (113) for λ_i and Λ_i corresponding to the forward and backward sweep programs, respectively. Since considerably less computation is required on the backward sweep (compared to the computation required on the forward sweep), it is nearly all accomplished in the main program element. In fact, there is only one subroutine in the backward sweep program. On the lowest functional level, we find the utility, matrix inversion, subroutine CHLSKI (choleski inversion) which in turn calls the subroutines DECOM and SUBS.

4.2 User Inputs.

The file containing the radar measurements Z_i (RAWDAT, device 11) must have a header line giving the path ID (RUNID) and the number of time steps (MAXDAT) contained on the file. The necessary formats are found in the forward sweep main program. The filter output is stored in a user assigned file (FLTOUT, device 12). The filter out is coded with two sorting indices; NCOUNT and SORT. SORT is a one dimensional array. The user selected sorting routine should sort in decreasing numerical order on NCOUNT and in normal ascending order on SORT. The backward sweep program reads the sorted (reverse chronological order) data from FLTOUT and writes the smoothed data in a user assigned file (SMTOUT, device 13).

The data time step, DELTAT, must be set in each of the two main program elements. Subroutine INTFLT contains all the remaining variables that are to be set by the user. They are;

CONE, the STD of target thrust rate,

CTWO, the STD of target g rate,

CTHREE, the STD of target roll rate,

SIGMR, the STD of range measurement error,

STDAZ1, the STD of linear azimuth measurement error,

STDAZ2, the STD of angular azimuth measurement error,

STDEL1, the STD of linear elevation measurement error,

STDEL2, the STD of angular elevation measurement error,

PVAL(i), i = 1...9, are the STDs of the initial state vector in the X, Y, Z directions of position, velocity, and acceleration, respectively.

4.3 Identification of Program Names.

This section identifies the FORTRAN names of the principal variables used in the program with the corresponding symbols used in the theory of sections 2 and 3.

Program Name	Theory Name
BARN	$\Gamma\overline{W}$
CAPLAM	Λ
DELTAT	Δt
HKAL	H
LAMDA	λ
PKAL	P, $\tilde{P}$
QA	Γ QAP Γ^T
QAP	QAP
QKAL	Γ Q Γ^T

Program Name	Theory Name
RESDUL	Z-HY
RMEAS	R
RMEASI	R^{-1}
SAB	A
SMTBRN	$\Gamma \hat{W}$
SMTP	SP
SMTQ	Γ SQ Γ^T
SMTSPS	$\hat{X}$
SPS	Y, $\tilde{Y}$
TRS	TRS
TRV	Γ^T
ZMEAS	Z

REFERENCES

1. Meredith, J., Schëder, R., Lufkin, B., Evaluation of the Gun Low Altitude Fire Control Test Bed (GLAADS), AMSAA Technical Report No. 149, April 1977.

2. Bryson, A.E., Jr., and Yu-Chi Ho, Applied Optimal Control, Waltham, MA, Ginn and Company, 1969.

3. Canon, M.D., Clifton, D.C., Polak, E., Theory of Optimal Control and Mathematical Programming, New York, NY, McGraw-Hill, 1970.

4. Bryson, A.E., and Frazier, M., Smoothing for Linear and Nonlinear Dynamic Systems, Proceedings of the Optimum System Synthesis Conference, Wright-Patterson Air Force Base, Ohio, September 1963.

5. Scheder, R.A., Adaptive Esitmation, AMSAA Technical Report No. 166, December 1976.

6. Mills, H.D., Mathematical Foundations for Structured Programming, IBM Corporation, Federal Systems Division, Gaithersburg, MD, 1972.

Next page is blank

APPENDIX A

PROGRAM LISTING

INDEX

Table A-1 Forward Sweep Main Program

```
C THIS PROGRAM ESTIMATES TARGET POSITION, VELOCITY, AND ACCELERATION
C BY SWEEPING THRU RAW TRACKING (POSITION) DATA USING A SELF
C ADAPTING, NON-LINEAR, KALMAN FILTER.
C
C             REFERENCES
C
C              1.  SCHEDER R.A., ADAPTIVE ESTIMATION,
C                  AMSAA TECHNICAL REPORT 166
C                  APG, MD 1976.
C
C              2.  BRYSON, A.E. JR., AND HO YU-CHI,
C                  APPLIED OPTINAL CONTROL,  GINN AND COMPANY,
C                  WALTHAM,  MA 1969,  CHAPTER 12
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /KALDAT/ PVAL  ,DELTAT,CONE  ,CTWO  ,CTHREE,SIGMR
     A               ,STDAZ1,STDAZ2,STDEL1,STDEL2
C
      REAL  RESDUL(3)
     A     ,BARN(9)
     B     ,SPS(9)
     C     ,PKAL(9,9)
     D     ,QA(3,3)
     E     ,RMEASI(3,3)
     F     ,QKAL(9,9)
     G     ,ZMEAS(3)
     H     ,HKAL(3,9)
     I     ,PHI(9,9)
     J     ,RMEAS(3,3)
C
      REAL  PVAL(9)
     A     ,DELTAT
     B     ,CONE
     C     ,CTWO
     D     ,CTHREE
     E     ,SIGMR
     F     ,STDAZ1
     G     ,STDAZ2
     H     ,STDEL1
     I     ,STDEL2
C
      REAL  ADOT(3)
C
      INTEGER  RAWDAT
     A        ,FLTOUT
     B        ,RUNID
     C        ,MAXDAT
     D        ,SORT(20)
C
100   FORMAT  (4F10.2)
200   FORMAT  (I10,F10.2,6F14.7,22X,I4,I2,/,(9F14.7,I4,I2))
300   FORMAT  (1X,I4,F10.2,9F9.1,2X,3F9.2)
400   FORMAT  (1X,'STATE ERROR COVARIANCE',9(/,1X,9F10.2),
     A         /,1X,'ACCELERATION  TRANSITION ERROR COVARIANCE'
```

```
     B        ,3(/,1X,3F8.2)
     C        ,/,1X,'MEASUREMENT ERROR COVARIANCE',3(/,1X,3F8.2))
  500 FORMAT  (I10,F10.2,6F12.3,/,9F12.3,/,9(9F12.3,/),9F12.3)
  600 FORMAT  (I4,1X,I4)
  700 FORMAT  (I10,116X,I4,I2)
C
C                  INITIALIZATION
C
        DELTAT = 0.05
        RAWDAT = 11
        FLTOUT = 12
C
        REWIND RAWDAT
        REWIND FLTOUT
C
        ID = 0
        NCOUNT = 0
C
        DO 1000 I = 1,20
          SORT(I) = I
 1000   CONTINUE
C
        READ  (RAWDAT,600)  RUNID ,MAXDAT
        WRITE (FLTOUT,700)  RUNID ,MAXDAT, ID
        WRITE (6,600)  RUNID ,MAXDAT
C
        CALL INTFLT
C
C                    FORWARD SWEEP THRU RAW DATA
C
        DO 3000 II = 1, MAXDAT
          READ  (RAWDAT,100)  TIME   ,(ZMEAS(I),I=1,3)
          NCOUNT = NCOUNT + 1
          CALL FILTER
C
C                 OUTPUT
C
          WRITE(FLTOUT,200) NCOUNT,TIME,(RESDUL(I),I=1,3)
     A                      ,(BARN(I),I=7,9),NCOUNT,SORT(1)
     B                      ,(SPS(I),I=1,9),NCOUNI,SORT(2)
     C                      ,((PKAL(I,J),J=1,9),NCOUNT,SORT(I+2),I=1,9)
     D                      ,((QA(I,J),J=1,3),I=1,3),NCOUNT,SORT(12)
     E                      ,((RMEASI(I,J),J=1,3),I=1,3),NCOUNT,SORT(13)
C
          ADOT(1) = BARN(7) / DELTAT
          ADOT(2) = BARN(8) / DELTAT
          ADOT(3) = BARN(9) / DELTAT
C
          WRITE (6,300)  NCOUNT,TIME,(SPS(I),I=1,9),(ADOT(I),I=1,3)
C
          IF  ( MOD((NCOUNT - 1),20) .NE. 0 )  GO TO 2500
            WRITE (6,400)   ((PKAL(I,J),J=1,9),I =1,9)
     A                      ,((QKAL(I,J),J=7,9),I =7,9)
     B                      ,((RMEAS(I,J),J=1,3),I =1,3)
 2500     CONTINUE
 3000   CONTINUE
C
```

```
      ENDFILE FLTOUT
C
      STOP
      END
```

Table A-2 Backward Sweep Main Program

```
C THIS PROGRAM SMOOTHS TRACKING DATA BY SWEEPING BACKWARDS
C THROUGH PREVIOUSLY FILTERD DATA.
C
C              REFERENCES
C
C              *.   MEREDITH J., SCHEDER R., LUFKIN B.,  EVALUATION OF
C                   THE GUN LOW ALTITUDE AIR DEFENSE FIRE CONTROL TEST
C                   BED (GLAAD), AMSAA TECHNICAL REPORT 149,
C                   APG, MD 1977.  CHAPTER 3
C
C              2.   BRYSON, A.E, JR.,  AND  HO,  YU-CHI,
C                   APPLIED OPTIMAL CONTROL,  GINN AND COMPANY,
C                   WALTHAM,  MA  1969.  CHAPTER 13
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /ABIO  / LAMDA ,CAPLAM
C
      DOUBLE PRECISION  LAMDA(9)
     A     ,CAPLAM(9,9)
C
      DOUBLE PRECISION  RESDUL(3)
     A     ,BARN(9)
     B     ,SPS(9)
     C     ,PKAL(9,9)
     D     ,QA(3,3)
     E     ,RMEASI(3,3)
     F     ,QKAL(9,9)
     G     ,ZMEAS(3)
     H     ,HKAL(3,9)
     I     ,PHI(9,9)
     J     ,RMEAS(3,3)
C
      DOUBLE PRECISION  TEMP3(9,9)
     A     ,SMTSPS(9)
     B     ,SMTBRN(9)
     D     ,TEMP4(9,9)
     E     ,SMTP(9,9)
     F     ,SMTQ(9,9)
C
      REAL  ADOT(3)
C
      INTEGER  FLTOUT
     A         ,SORT(20)
     B         ,SMTOUT
     C         ,RUNID
     D         ,MAXDAT
C
  200 FORMAT  (10X,F10.2,6F14.7,/,9F14.7,/,9(9F14.7,/),9F14.7,/,9F14.7)
  300 FORMAT  (1X,I4,F10.2,9F9.1,2X,3F9.2)
  400 FORMAT  (1X,'STATE ERROR COVARIANCE',9(/,1X,9F10.2)
     A        ,/,1X,'ACCELERATION  TRANSITION ERROR COVARIANCE'
     B        ,3(/,1X,3F8.2))
  500 FORMAT  (I10,F10.2,106X,I4,I2,/,(9F14.7,I4,I2))
  600 FORMAT  (I10)
```

```
700    FORMAT  (I10,116X,I4,I2)
C
C                INITIALIZE SMOOTHER
C
         DELTAT = 0.05
         FLTOUT = 12
         SMTOUT = 13
C
         REWIND SMTOUT
         REWIND FLTOUT
C
         READ  (FLTOUT,700)  RUNID ,MAXDAT, IC
         WRITE  (SMTOUT,700)  RUNID ,MAXDAT, IC
         WRITE  (6,600)  RUNID
C
         DO 0950 I = 1,20
           SORT(I) = I
0950     CONTINUE
C
C               STATE TO MEASURMENT  TRANSITION MATRIX HKAL
C
         DO 1500  I = 1,3
           DO 1000  J = 1,9
             HKAL(I,J) = 0.0
1000       CONTINUE
1500     CONTINUE
         HKAL(1,1) = 1.0
         HKAL(2,2) = 1.0
         HKAL(3,3) = 1.0
C
C               STATE TRANSITION MATRIX PHI
C
         DO 3000  I = 1,9
           DO 2000 J = 1,9
             PHI(I,J) = 0.0
2000       CONTINUE
3000     CONTINUE
         DO 4000 I = 1,3
           PHI(I,I) = 1.0
           PHI(I+3,I+3) = 1.0
           PHI(I+6,I+6) = 1.0
4000     CONTINUE
C
         PHI(1,4) = DELTAT
         PHI(2,5) = DELTAT
         PHI(3,6) = DELTAT
         PHI(1,7) = DELTAT**2 / 2.0
         PHI(2,8) = DELTAT**2 / 2.0
         PHI(3,9) = DELTAT**2 / 2.0
         PHI(4,7) = DELTAT
         PHI(5,8) = DELTAT
         PHI(6,9) = DELTAT
C
C             LAMDA AND CAPLAM
C
         DO 6000 I = 1,9
           LAMDA(I) = 0.0
```

```
            DO 5000 J = 1,9
              CAPLAM(I,J) = 0.0
      5000  CONTINUE
      6000 CONTINUE
C
C                BACKWARD SWEEP
C
          DO 15000 NCOUNT = 1, MAXDAT
C
C                 INPUT
C          ( FILTERD DATA MUST BE IN REVERSE CHRONOLOGICAL ORDER ).
C
            READ(FLTOUT,200) TIME,(RESDUL(I),I=1,3)
         A                   ,(BARN(I),I=7,9)
         B                   ,(SPS(I),I=1,9)
         C                   ,((PKAL(I,J),J=1,9),I=1,9)
         D                   ,((QA(I,J),J=1,3),I=1,3)
         E                   ,((RMEASI(I,J),J=1,3),I=1,3)
C
C           EXPAND  BARN AND QA TO FULL DIMENSION
C
            DO 7100 I = 1,3
              BARN(I) = 0.1666667*BARN(I+6)*DELTAT**2
              BARN(I+3) = 0.5*BARN(I+6)*DELTAT
      7100  CONTINUE
C
            DO 7400 I = 1,3
              DO 7300 J = 1,3
                QKAL(I,J) = QA(I,J)*DELTAT**4/36.
                QKAL(I,J+3)=QA(I,J)*DELTAT**3/12.
                QKAL(I,J+6) = QA(I,J) * DELTAT**2 / 6.0
                QKAL(I+3,J)=QA(I,J)*DELTAT**3/12.
                QKAL(I+3,J+3) = QA(I,J) * DELTAT**2 / 4.0
                QKAL(I+3,J+6) = QA(I,J) * DELTAT / 2.0
                QKAL(I+6,J) = QA(I,J) * DELTAT**2 / 6.0
                QKAL(I+6,J+3) = QA(I,J) * DELTAT / 2.0
                QKAL(I+6,J+6) = QA(I,J)
      7300    CONTINUE
      7400  CONTINUE
C
C                   COMPUTE SMOOTHED STATE
C
C               PKAL * PHI(TRANSPOSE)      STORE IN TEMP3
C
            DO 9000 I = 1,9
              DO 8000 J = 1,9
                TEMP3(I,J) = 0.0
                DO 7500 K = 1,9
                  TEMP3(I,J) = TEMP3(I,J) + PKAL(I,K) * PHI(J,K)
      7500      CONTINUE
      8000    CONTINUE
      9000  CONTINUE
C
C               SMTSPS = SPS - TEMP3 * LAMDA --- SMOOTHED STATE
C               SMTBRN = BARN - QFLT * LAMDA --SMOOTHED TRANSITION ERROR
C
            DO 10000 I = 1,9
```

```
             SMTSPS(I) = SPS(I)
             SMTBRN(I) = BARN(I)
             DO 9500 J = 1,9
               SMTSPS(I) = SMTSPS(I) - TEMP3(I,J)*LAMDA(J)
               SMTBRN(I) = SMTBRN(I) - QKAL(I,J) * LAMDA(J)
    9500     CONTINUE
   10000   CONTINUE
C
C                   COMPUTE SMOOTHED COVARIANCES
C
C              TEMP3 * CAPLAM * TEMP3(TRANSPOSE)    STORE IN TEMP4
C
           DO 1'000  I = 1,9
             DO 10500 J = 1,9
               TEMP4(I,J) = 0.0
               DO 10200 M = 1,9
                 DO 10100 K = 1,9
                   TEMP4(I,J) = TEMP4(I,J) + TEMP3(I,M) * CAPLAM(M,K) *
        A          TEMP3(J,K)
   10100         CONTINUE
   10200       CONTINUE
   10500     CONTINUE
   11000   CONTINUE
C
C              QKAL * CAPLAM  * QKAL     STORE IN  TEMP3
C
           DO 12000 I = 1,9
             DO 11500 J = 1,9
               TEMP3(I,J) = 0.0
               DO 11200 M = 1,9
                 DO 11100 K = 1,9
                   TEMP3(I,J)= TEMP3(I,J)+ QKAL(I,M) * CAPLAM(M,K) *
        A          QKAL(K,J)
   11100         CONTINUE
   11200       CONTINUE
   11500     CONTINUE
   12000   CONTINUE
C
C              SMTP = PKAL - TEMP4  --- SMOOTHED STATE ERROR COVARIANCE
C              SMTQ = QKAL - TEMP3  --- SMOOTHED PLANT ERROR COVARIANCE
C
           DO 13000 I = 1,9
             DO 12500 J = 1,9
               SMTP(I,J) = PKAL(I,J) - TEMP4(I,J)
               SMTQ(I,J) = QKAL(I,J) - TEMP3(I,J)
   12500     CONTINUE
   13000   CONTINUE
C
C                   OUTPUT
C
           WRITE (SMTOUT,500) NCOUNT,TIME,NCOUNT,SORT(1)
        A                     ,(SMTSPS(I),I=1,9),NCOUNT,SORT(2)
        B                     ,((SMTP(I,J),J=1,9),NCOUNT,SORT(I+2),I=1,9)
        C                     ,((SMTQ(I,J),J=/,9),I=7,9),NCOUNT,SORT(12)
        D                     ,(SMTBRN(I),I=1,9),NCOUNT,SORT(13)
C
           DO 1350 I = 1,3
```

```
      C
           ADOT(I) = SMTBRN(I+6) / DELTAT
 1350     CONTINUE
      C
          WRITE (6,300)  NCOUNT,TIME,(SMTSPS(I),I=1,9),(ADOT(I),I=1,3)
      C
          IF  ( MOD((NCOUNT - 1),20) .NE. 0 )  GO TO 14000
           WRITE (6,400)  ((SMTP(I,J),J=1,9),I =1,9)
     A                    ,((SMTQ(I,J),J=7,9),I =7,9)
14000     CONTINUE
      C
      C              UPDATE  LAMDA  AND CAPLAM
      C
          CALL  LAMDAS
      C
15000    CONTINUE
      C
         ENDFILE SMTOUT
      C
         STOP
        END
```

Table A-3 Subroutine CHLSKI

```
      SUBROUTINE CHLSKI
C
C CHOLESKI DECOMPOSITION  ( MATRIX INVERSION)
C
      COMMON /KALIN2/  T1    ,T2    ,T3
C
      COMMON /MATINV/ Z      ,SZ
C
      REAL  T1(9,9)
     F     ,T2(3,3)
     G     ,T3(3,9)
C
      DOUBLE PRECISION  Z(3,3)
     A                 ,SZ(3)
C
        DO 2000 I = 1,3
          DO 1000 J = 1,3
            Z(I,J) = T2(I,J)
 1000     CONTINUE
 2000   CONTINUE
        CALL DECOM
        DO 5000  I = 1,9
          DO 3000  J = 1,3
            SZ(J)  = T1(J,I)
 3000     CONTINUE
          CALL SUBS
          DO 4000  J = 1,3
            T3(J,I) = SZ(J)
 4000     CONTINUE
 5000   CONTINUE
C
        RETURN
       END
```

Table A-4 Subroutine CORECT

```
      SUBROUTINE CORECT
C
C UPDATE  EXTRAPOLATION WITH  MEASUREMENTS
C
      COMMON /KALIO / RESDUL,BARN   ,SPS    ,PKAL   ,QA     ,RMEASI,QKAL
     A                ,ZMEAS ,HKAL  ,PHI    ,RMEAS
C
      COMMON /KALIN2/  T1     ,T2     ,T3
C
      REAL  RESDUL(3)
     A      ,BARN(9)
     B      ,SPS(9)
     C      ,PKAL(9,9)
     D      ,QA(3,3)
     E      ,RMEASI(3,3)
     F      ,QKAL(9,9)
     G      ,ZMEAS(3)
     H      ,HKAL(3,9)
     I      ,PHI(9,9)
     J      ,RMEAS(3,3)
C
      REAL  T1(9,9)
     F      ,T2(3,3)
     G      ,T3(3,9)
C
C           COMPUTE  HKAL * PKAL       STORE IN T1
C
        DO 3000 K = 1,3
          DO 2000 I = 1,9
            T1(K,I) = 0.0
            DO 1000  J = 1,9
              T1(K,I) = T1(K,I) + HKAL(K,J) * PKAL(J,I)
 1000       CONTINUE
 2000     CONTINUE
 3000   CONTINUE
C
C
C
C           RMEAS + HKAL * PKAL * HKAL(TRANSPOSE)      STORE IN T2
C
        DO 6000  K = 1,3
          DO 5000  I = 1,3
            T2(K,I) = RMEAS(K,I)
            DO 4000  J = 1,9
              T2(K,I) = T2(K,I) + T1(K,J) * HKAL(I,J)
 4000       CONTINUE
 5000     CONTINUE
 6000   CONTINUE
C
C           CHLSKI COMPUTES T2 INVERSE * T1       STORE IN T3
C
        CALL CHLSKI
C
C           CORRECT PKAL   ( NOTE PKAL SYMETRIC )
C
        DO 9000  K = 1,9
```

```
               DO 8000  I = 1,9
                 DO 7000  J = 1,3
                   PKAL(K,I) = PKAL(K,I) - T1(J,K) * T3(J,I)
       7000      CONTINUE
       8000    CONTINUE
       9000  CONTINUE
      C
      C            STABILIZE PKAL
      C
             DO 9700 I = 1,9
               PKAL(I,I) = AMAX1(PKAL(I,I),0.000001 )
               DO 9500 J = I,9
                 PKAL(J,I) = 0.5*(PKAL(I,J) + PKAL(J,I))
       9500    CONTINUE
       9700  CONTINUE
      C
      C            INNOVATION  (  ZMEAS - HKAL * SPS )    STORE IN RESDUL
      C
             DO 11000 I = 1,3
               RESDUL(I) = ZMEAS(I)
               DO 10000 J = 1,9
                 RESDUL(I) = RESDUL(I) - HKAL(I,J) * SPS(J)
      10000    CONTINUE
      11000  CONTINUE
      C
      C            CORRECT SPS  ( T3 TRANSPOSE IS THE KALMAN GAIN )
      C
             DO 13000 I = 1,9
               DO 12000 J = 1,3
                 SPS(I) = SPS(I) + T3(J,I) * RESDUL(J)
      12000    CONTINUE
      13000  CONTINUE
      C
             RETURN
           END
```

Table A-5 Subroutine CTRF

```
      SUBROUTINE CTRF
C
C COMPUTES COORDINATE TRANSITION MATRIX, TRV, FROM  REFERENCE TO FRENET FRAME
C AND FRENET FRAME ACCELERATION ERROR COVARIANCE - QAP
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /KALUT / QAP   ,VMAG  ,TRV   ,ADOTVU,AP    ,GN    ,RSPS
     A               ,TRS   ,THRUST
C
      COMMON /KALDAT/ PVAL  ,DELTAT,CONE  ,CTWO  ,CTHREE,SIGMR
     A               ,STDAZ1,STDAZ2,STDEL1,STDEL2
C
      REAL  RESDUL(3)
     A     ,BARN(9)
     B     ,SPS(9)
     C     ,PKAL(9,9)
     D     ,QA(3,3)
     E     ,RMEASI(3,3)
     F     ,QKAL(9,9)
     G     ,ZMEAS(3)
     H     ,HKAL(3,9)
     I     ,PHI(9,9)
     J     ,RMEAS(3,3)
C
      REAL  QAP(3,3)
     A     ,VMAG
     B     ,TRV(3,3)
     C     ,ADOTVU
     D     ,AP
     E     ,GN
     F     ,RSPS
     G     ,TRS(3,3)
     H     ,THRUST
C
      REAL  PVAL(9)
     A     ,DELTAT
     B     ,CONE
     C     ,CTWO
     D     ,CTHREE
     E     ,SIGMR
     F     ,STDAZ1
     G     ,STDAZ2
     H     ,STDEL1
     I     ,STDEL2
C
        VMAG = SQRT ( SPS(4)*SPS(4) + SPS(5)*SPS(5) + SPS(6)*SPS(6) )
        VMAGP = SQRT ( SPS(4)*SPS(4) + SPS(5)*SPS(5) )
C
C              CHECK FOR NEAR ZERO VELOCITY
C
        IF  ( VMAGP .LT. 50.0 )  GO TO 3000
C
C              FAST TARGET
C
```

```
                VMAGIN = 1.0 / VMAG
                VMAGPI = 1.0 / VMAGP
      C
                TRV(1,1) = SPS(4) * VMAGIN
                TRV(1,2) = SPS(5) * VMAGIN
                TRV(1,3) = SPS(6) * VMAGIN
      C
      C              COMPUTE NUMBER OF G'S
      C
                ADOTVU =   TRV(1,1)*SPS(7) + TRV(1,2)*SPS(8) + TRV(1,3)*SPS(9)
                AN1 = SPS(7) - TRV(1,1) * ADOTVU
                AN2 = SPS(8) - TRV(1,2) * ADOTVU
                AN3 = SPS(9) - TRV(1,3) * ADOTVU
                GN2 = AN1*AN1 + AN2*AN2 + AN3*AN3
      C
      C              CHECK FOR NEAR ZERO ACCELERATION
      C
                IF  ( GN2 .LT. 5.0 )  GO TO 1000
      C
      C              MANEUVERING TARGET
      C
                  GN  = SQRT ( GN2 )
                  TRV(2,1) = AN1 / GN
                  TRV(2,2) = AN2 / GN
                  TRV(2,3) = AN3 / GN
      C
                  TRV(3,1) = TRV(1,2) * TRV(2,3) - TRV(2,2) * TRV(1,3)
                  TRV(3,2) = TRV(2,1) * TRV(1,3) - TRV(1,1) * TRV(2,3)
                  TRV(3,3) = TRV(1,1) * TRV(2,2) - TRV(1,2) * TRV(2,1)
      C
      C
      C              COMPUTE TRUST, AIR PRESSURE FORCE ON WINGS AP, AND QAP
      C
                  THRUST = ADOTVU + 9.8 * TRV(1,3)
                  APN = 9.8 * TRV(2,3) + GN
                  AP = SQRT( 96.0*TRV(3,3)**2 + APN**2 )
      C
                  QAP(1,1) = ( CONE*DELTAT )**2
                  QAP(2,2) = ( CTWO*DELTAT )**2
                  QAP(3,3) = ( AMIN1(GN*CTHREE,30.0)*DELTAT )**2
      C
                  GO TO 2000
      1000      CONTINUE
      C
      C              NON MANEUVERING TARGET (ACCELERATION SMALL)
      C
                  TRV(2,1) = - SPS(5) * VMAGPI
                  TRV(2,2) =   SPS(4) * VMAGPI
                  TRV(2,3) =   0.0
                  TRV(3,1) =   - SPS(4)* SPS(6) * VMAGIN * VMAGPI
                  TRV(3,2) =   - SPS(5)* SPS(6) * VMAGIN * VMAGPI
                  TRV(3,3) = VMAGP * VMAGIN
      C
                  QAP(1,1) =   ( CONE * DELTAT )**2
                  QAP(2,2) =   ( CTWO * DELTAT )**2
                  QAP(3,3) =   ( CTWO * DELTAT )**2
      C
```

```
        GN = 0.0
C
2000    CONTINUE
        GO TO 4000
3000  CONTINUE
C
C           SLOW TARGET (VELOCITY SMALL)
C
        DO 3500 I = 1,3
          DO 3250 J = 1,3
            TRV(I,J) = 0.0
            QAP(I,J) = 0.0
3250      CONTINUE
          TRV(I,I) = 1.0
          QAP(I,I) = ( CONE*DELTAT)**2
3500    CONTINUE
4000  CONTINUE
C
      RETURN
      END
```

Table A-6 Subroutine CTRS

```
      SUBROUTINE CTRS
C
C COMPUTES TRANSITION MATRIX, TRS, FROM REFERENCE TO ESTIMATED  L.O.S FRAME
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /KALUT / QAP   ,VMAG  ,TRV   ,ADOTVU,AP    ,GN    ,RSPS
     A               ,TRS   ,THRUST
C
      REAL  RESDUL(3)
     A     ,BARN(9)
     B     ,SPS(9)
     C     ,PKAL(9,9)
     D     ,QA(3,3)
     E     ,RMEASI(3,3)
     F     ,QKAL(9,9)
     G     ,ZMEAS(3)
     H     ,HKAL(3,9)
     I     ,PHI(9,9)
     J     ,RMEAS(3,3)
C
      REAL  QAP(3,3)
     A     ,VMAG
     B     ,TRV(3,3)
     C     ,ADOTVU
     D     ,AP
     E     ,GN
     F     ,RSPS
     G     ,TRS(3,3)
     H     ,THRUST
C
        RSPS = SQRT ( SPS(1)**2 + SPS(2)**2 + SPS(3)**2 )
        RSPSP = SQRT ( SPS(1)**2 + SPS(2)**2 )
C
C               CHECK FOR NEAR ZERO RANGE
C
        IF  ( RSPSP .LT. 5.0 )  GO TO 1000
C
C               RANGE OK
C
          TRS(*,1) = SPS(1) / RSPS
          TRS(*,2) = SPS(2) / RSPS
          TRS(*,3) = SPS(3) / RSPS
          TRS(2,1) = - SPS(2) / RSPSP
          TRS(2,2) = SPS(1) / RSPSP
          TRS(2,3) = 0.0
          TRS(3,1) = - SPS(1)*SPS(3) / ( RSPS * RSPSP )
          TRS(3,2) = - SPS(2)*SPS(3) / ( RSPS * RSPSP )
          TRS(3,3) = RSPSP / RSPS
          GO TO 4000
1000    CONTINUE
C
C               RANGE SMALL
C
          DO 3000 I = 1,3
```

```
            DO 2000 J = 1,3
               TRS(I,J) = 0.0
 2000       CONTINUE
            TRS(I,I) = 1.0
 3000    CONTINUE
 4000   CONTINUE
C
        RETURN
      END
```

Table A-7 Subroutine DECOM

```
      SUBROUTINE DECOM
C
C DECOMPOSE  (MATRIX INVERSION)
C
      COMMON /MATINV/ Z      ,SZ
C
      DOUBLE PRECISION  Z(3,3)
     A                 ,SZ(3)
C
      DO 3000  I = 1,3
        K = I - 1
        DO 2000  J = 1,3
          IF  ( K .EQ. 0 )  GO TO 1500
            DO 1000  M = 1,K
              Z(I,J) = Z(I,J) - Z(M,J) * Z(M,I)
 1000       CONTINUE
 1500     CONTINUE
          IF  ( (J-I) .EQ. 0 )  GO TO 1750
            Z(I,J) = Z(I,J) / Z(I,I)
            GO TO 1875
 1750     CONTINUE
            Z(I,J) = DSQRT( DABS(Z(I,J)) )
 1875     CONTINUE
 2000   CONTINUE
 3000 CONTINUE
C
      RETURN
      END
```

Table A-8 Subroutine EXTRAP

```
      SUBROUTINE EXTRAP
C
C EXTRAPOLATES STATE VECTOR AND STATE COVARIANCE MATRIX
C
      COMMON /KALIO / RESDUL,BARN   ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A                ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      REAL  RESDUL(3)
     A      ,BARN(9)
     B      ,SPS(9)
     C      ,PKAL(9,9)
     D      ,QA(3,3)
     E      ,RMEASI(3,3)
     F      ,QKAL(9,9)
     G      ,ZMEAS(3)
     H      ,HKAL(3,9)
     I      ,PHI(9,9)
     J      ,RMEAS(3,3)
C
      REAL  ST(9)
     E      ,TEMP(9,9)
C
C             PREDICT STATE VECTOR
C
      DO 2000  I = 1,9
        ST(I) = 0.0
        DO 1000 J = 1,9
          ST(I) = ST(I) + PHI(I,J) * SPS(J)
 1000   CONTINUE
 2000 CONTINUE
      DO 3000  I = 1,9
        SPS(I) = ST(I)
 3000 CONTINUE
      DO 3500 I = 1,9
        SPS(I) = SPS(I) + BARN(I)
 3500 CONTINUE
C
C             PKAL * PHI (TRANSPOSE)  STORE IN TEMP
C
      DO 6000 K = 1,9
        DO 5000 I = 1,9
          TEMP(K,I) = 0.0
          DO 4000  J = 1,9
            TEMP(K,I) = TEMP(K,I) + PKAL(K,J) * PHI(I,J)
 4000     CONTINUE
 5000   CONTINUE
 6000 CONTINUE
C
C             PREDICT PKAL AS  QKAL + PHI * PKAL * PHI(TRANPOSE)
C
      DO 9000  K = 1,9
        DO 8000  I = 1,9
          PKAL(K,I) = QKAL(K,I)
          DO 7000  J = 1,9
            PKAL(K,I) = PKAL(K,I) + PHI(K,J) * TEMP(J,I)
 7000     CONTINUE
```

```
 8000      CONTINUE
 9000    CONTINUE
C
C
         RETURN
        END
```

Table A-9 Subroutine FILTER

```
      SUBROUTINE FILTER
C
C KALMAN ESTIMATOR OF POSITION, VELOCITY, AND ACCELERATION
C
C              COMPUTE TRANSITION ERROR COVARIANCE MATRIX QKAL
C              AND TRANSITION ERROR MEAN  BARN
C
        CALL CTRF
        CALL NBAR
        CALL Q
C
C              EXTRAPOLATE STATE
C
        CALL EXTRAP
C
C              COMPUTE MEASUREMENT ERROR COVARIANCE MATRIX
C
        CALL CTRS
        CALL R
C
C              UPDATE STATE AND STATE ERROR COVARIANCE
C
        CALL CORECT
C
        RETURN
      END
```

Table A-10 Subroutine INTFLT

```
      SUBROUTINE INTFLT
C
C INITIALIZATION OF KALMAN ALGORITHMS
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /KALUT / QAP   ,VMAG  ,TRV   ,ADOTVU,AP    ,GN    ,RSPS
     A               ,TRS   ,THRUST
C
      COMMON /KALDAT/ PVAL  ,DELTAT,CONE  ,CTWO  ,CTHREE,SIGMR
     A               ,STDAZ1,STDAZ2,STDEL1,STDEL2
C
      REAL  RESDUL(3)
     A     ,BARN(9)
     B     ,SPS(9)
     C     ,PKAL(9,9)
     D     ,QA(3,3)
     E     ,RMEASI(3,3)
     F     ,QKAL(9,9)
     G     ,ZMEAS(3)
     H     ,HKAL(3,9)
     I     ,PHI(9,9)
     J     ,RMEAS(3,3)
C
      REAL  QAP(3,3)
     A     ,VMAG
     B     ,TRV(3,3)
     C     ,ADOTVU
     D     ,AP
     E     ,GN
     F     ,RSPS
     G     ,TRS(3,3)
     H     ,THRUST
C
      REAL  PVAL(9)
     A     ,DELTAT
     B     ,CONE
     C     ,CTWO
     D     ,CTHREE
     E     ,SIGMR
     F     ,STDAZ1
     G     ,STDAZ2
     H     ,STDEL1
     I     ,STDEL2
C
C        NOMINAL VALUES
C
        PVAL(1) = 9000000.0
        PVAL(2) = 9000000.0
        PVAL(3) = 4000000.0
        PVAL(4) = 10000.0
        PVAL(5) = 10000.0
        PVAL(6) = 10000.0
        PVAL(7) = 400.0
        PVAL(8) = 400.0
```

```
      PVAL(9) = 400.0
C
      CONE = 1.0
      CTWO = 4.0
      CTHREE = 1.0
      SIGMR = 2.0
      STDAZ1 = 1.0
      STDAZ2 = 0.0005
      STDEL1 = 1.0
      STDEL2 = 0.0005
C
C           INITIAL STATE VECTOR
C
      DO 1000 I = 1,9
        SPS(I) = 0.0
 1000 CONTINUE
C
C           INITIAL COVARIANCE MATRIX PKAL
C
      DO 3000  I = 1,9
        DO 2000  J = 1,9
          PKAL(I,J) = 0.0
 2000   CONTINUE
 3000 CONTINUE
      DO 4000 I = 1,9
        PKAL(I,I) = PVAL(I)
 4000 CONTINUE
C
C           INITAL AND CONSTANT MEASURMENT TRANSITION MATRIX HKAL
C
      DO 6000  I = 1,3
        DO 5000  J = 1,9
          HKAL(I,J) = 0.0
 5000   CONTINUE
 6000 CONTINUE
      HKAL(1,1) = 1.0
      HKAL(2,2) = 1.0
      HKAL(3,3) = 1.0
C
C
C
C           INITAL AND CONSTANT STATE TRANSITION MATRIX PHI
C
      DO 8000  I = 1,9
        DO 7000 J = 1,9
          PHI(I,J) = 0.0
 7000   CONTINUE
 8000 CONTINUE
      DO 9000 I = 1,3
        PHI(I,I) = 1.0
        PHI(I+3,I+3) = 1.0
        PHI(I+6,I+6) = 1.0
 9000 CONTINUE
      PHI(1,4) = DELTAT
      PHI(2,5) = DELTAT
      PHI(3,6) = DELTAT
      PHI(1,7) = DELTAT**2 / 2.0
```

```
      PHI(2,8) = DELTAT**2 / 2.0
      PHI(3,9) = DELTAT**2 / 2.0
      PHI(4,7) = DELTAT
      PHI(5,8) = DELTAT
      PHI(6,9) = DELTAT
C
C           ACCELERATION NOISE COVARIANCE  MATRIX IN FRENET  FRAME
C
      DO 11000  I = 1,3
        DO 10000 J = 1,3
          QAP(I,J) = 0.0
10000   CONTINUE
11000 CONTINUE
      QAP(1,') = (CONE*DELTAT)**2
      QAP(2,2) = (CTWO*DELTAT)**2
      QAP(3,3) = (CTHREE*DELTAT)**2
C
C           INITIAL TRANSITION ERROR COVARIANCE MATRIX   QKAL
C
      DO 12000 I = 1,9
        DO 1'500 J = 1,9
          QKAL(I,J) = 0.0
11500   CONTINUE
12000 CONTINUE
C
      DO 13000 I = 1,3
        QKAL(I+6,I+6) = QAP(2,2)
13000 CONTINUE
C
C           INITAL TRANSITION ERROR VECTOR  BARN
C
      DO 14000 I = 1,9
        BARN(I) = 0.0
14000 CONTINUE
C
      RETURN
     END
```

Table A-11 Subroutine LAMDAS

```
      SUBROUTINE LAMDAS
C
C UPDATE LAMDA AND CAPLAM
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /ABIO  / LAMDA ,CAPLAM
C
      DOUBLE PRECISION   RESDUL(3)
     A       ,BARN(9)
     B       ,SPS(9)
     C       ,PKAL(9,9)
     D       ,QA(3,3)
     E       ,RMEASI(3,3)
     F       ,QKAL(9,9)
     G       ,ZMEAS(3)
     H       ,HKAL(3,9)
     I       ,PHI(9,9)
     J       ,RMEAS(3,3)
C
      DOUBLE PRECISION   LAMDA(9)
     A       ,CAPLAM(9,9)
C
      DOUBLE PRECISION   SAB(9,9)
     A       ,TEMAB1(9,9)
     B       ,TEMAB2(9)
     C       ,TEMAB3(9,9)
     D       ,TAB3(9)
     E       ,TAB4(9,9)
     F       ,TAB5(9,9)
C
C                 UPDATE LAMDA
C
C             SAB = HKAL(TRANSPOSE) * RMEASI * HKAL
C
        DO 4000 I = 1,9
          DO 3000 J = 1,9
            SAB(I,J) = 0.0
            DO 2000 M = 1,3
              DO 1000 N = 1,3
                SAB(I,J) = SAB(I,J) + HKAL(M,I) * RMEASI(M,N) *
     A          HKAL(N,J)
 1000         CONTINUE
 2000       CONTINUE
 3000     CONTINUE
 4000   CONTINUE
C
C             COMPUTE I - PKAL * SAB  STORE IN TEMAB1
C
        DO 7000 I = 1,9
          DO 6000 J = 1,9
            TEMAB1(I,J) = 0.0
            TEMAB3(I,J) = 0.0
            DO 5000 K = 1,9
```

```
                TEMAB1(I,J) = TEMAB1(I,J) - PKAL(I,K) * SAB(K,J)
 5000         CONTINUE
 6000       CONTINUE
 7000     CONTINUE
          DO 7500 I = 1,9
            TEMAB1(I,I) = TEMAB1(I,I) + 1.0
            TEMAB3(I,I) = TEMAB3(I,I) + 1.0
 7500     CONTINUE
C
C          PHI(TRANSPOSE) * LAMDA     STORE IN  TEMAB2
C
          DO 9000 I = 1,9
            TEMAB2(I) = 0.0
            DO 8000 J = 1,9
              TEMAB2(I) = TEMAB2(I) +  PHI(J,I) * LAMDA(J)
 8000       CONTINUE
 9000     CONTINUE
C
C               TEMAB2 - HKAL(TRANSPOSE) * RMEASI * RESDUL  STORE IN TAB3
C
          DO 14000 I = 1,9
            TAB3(I) = TEMAB2(I)
            DO 13000 J = 1,3
              DO 12000 K = 1,3
                TAB3(I) = TAB3(I) - HKAL(J,I) * RMEASI(J,K) * RESDUL(K)
12000         CONTINUE
13000       CONTINUE
14000     CONTINUE
C
C               LAMDA = TEMAB1(TRANSPOSE) * TAB3
C
          DO 16000 I = 1,9
            LAMDA(I) = 0.0
            DO 15000 J = 1,9
              LAMDA(I) = LAMDA(I) + TEMAB1(J,I) * TAB3(J)
15000       CONTINUE
16000     CONTINUE
C
C                      UPDATE CAPLAM
C
C               PHI * TEMAB1    STORE IN TAB4
C
          DO 18000 I = 1,9
            DO 17000 J = 1,9
              TAB4(I,J) = 0.0
              DO 16500 K = 1,9
                TAB4(I,J) = TAB4(I,J) + PHI(I,K) * TEMAB1(K,J)
16500         CONTINUE
17000       CONTINUE
18000     CONTINUE
C
C               TAB4(TRANSPOSE) * CAPLAM * TAB4   STORE  IN  TAB5
C
          DO 22000 I = 1,9
            DO 2'000 J = 1,9
              TAB5(I,J) = 0.0
              DO 20000 M = 1,9
```

```
                DO 19000 N = 1,9
                  TAB5(I,J) = TAB5(I,J) + TAB4(M,I) * CAPLAM(M,N) *
     A                        TAB4(N,J)
 19000          CONTINUE
 20000        CONTINUE
 21000      CONTINUE
 22000    CONTINUE
C
C               CAPLAM =   TAB5 + SAB * TEMAB1
C
          DO 25000 I = 1,9
            DO 24000 J = 1,9
              CAPLAM(I,J) = TAB5(I,J)
              DO 23000 K = 1,9
                CAPLAM(I,J) = CAPLAM(I,J) + SAB(I,K) * TEMAB1(K,J)
 23000        CONTINUE
 24000      CONTINUE
 25000    CONTINUE
C
          RETURN
        END
```

Table A-12 Subroutine NBAR

```
      SUBROUTINE NBAR
C
C COMPUTES STATE TRANSITION ERROR VECTOR  BARN
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /KALUT / QAP   ,VMAG  ,TRV   ,ADOTVU,AP    ,GN    ,RSPS
     A               ,TRS   ,THRUST
C
      COMMON /KALDAT/ PVAL  ,DELTAT,CONE  ,CTWO  ,CTHREE,SIGMR
     A               ,STDAZ1,STDAZ2,STDEL1,STDEL2
C
      REAL  RESDUL(3)
     A      ,BARN(9)
     B      ,SPS(9)
     C      ,PKAL(9,9)
     D      ,QA(3,3)
     E      ,RMEASI(3,3)
     F      ,QKAL(9,9)
     G      ,ZMEAS(3)
     H      ,HKAL(3,9)
     I      ,PHI(9,9)
     J      ,RMEAS(3,3)
C
      REAL  QAP(3,3)
     A      ,VMAG
     B      ,TRV(3,3)
     C      ,ADOTVU
     D      ,AP
     E      ,GN
     F      ,RSPS
     G      ,TRS(3,3)
     H      ,THRUST
C
      REAL  PVAL(9)
     A      ,DELTAT
     B      ,CONE
     C      ,CTWO
     D      ,CTHREE
     E      ,SIGMR
     F      ,STDAZ1
     G      ,STDAZ2
     H      ,STDEL1
     I      ,STDEL2
C
        IF ( VMAG .LT. 50.0 )  GO TO 1000
          TBARN1 = -GN**2 / VMAG *DELTAT
          TBARN2 = ADOTVU *GN /VMAG *DELTAT
          TBARN3 =   0.0
          GO TO 2000
 1000   CONTINUE
          TBARN1 = 0.0
          TBARN2 = 0.0
          TBARN3 = 0.0
 2000   CONTINUE
```

```
C
C      TRANSFORM TO REFERENCE FRAME
C
      BARN(7) = TRV(1,1)*TBARN1 + TRV(2,1)*TBARN2 + TRV(3,1)*TBARN3
      BARN(8) = TRV(1,2)*TBARN1 + TRV(2,2)*TBARN2 + TRV(3,2)*TBARN3
      BARN(9) = TRV(1,3)*TBARN1 + TRV(2,3)*TBARN2 + TRV(3,3)*TBARN3
C
      BARN(4) = BARN(7) * 0.5 * DELTAT
      BARN(5) = BARN(8) * 0.5 * DELTAT
      BARN(6) = BARN(9) * 0.5 * DELTAT
C
      BARN(1) = BARN(7) * 0.1666667 * DELTAT**2
      BARN(2) = BARN(8) * 0.1666667 * DELTAT**2
      BARN(3) = BARN(9) * 0.1666667 * DELTAT**2
C
      RETURN
      END
```

Table A-13 Subroutine Q

```
      SUBROUTINE Q
C
C COMPUTES STATE TRANSITION ERROR COVARIANCE MATRIX  QKAL
C
      COMMON /KALIO / RESDUL,BARN  ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A               ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /KALUT / QAP   ,VMAG  ,TRV   ,ADOTVU,AP    ,GN    ,RSPS
     A               ,TRS   ,THRUST
C
      COMMON /KALDAT/ PVAL  ,DELTAT,CONE  ,CTWO  ,CTHREE,SIGMR
     A               ,STDAZ1,STDAZ2,STDEL1,STDEL2
C
      REAL   RESDUL(3)
     A      ,BARN(9)
     B      ,SPS(9)
     C      ,PKAL(9,9)
     D      ,QA(3,3)
     E      ,RMEASI(3,3)
     F      ,QKAL(9,9)
     G      ,ZMEAS(3)
     H      ,HKAL(3,9)
     I      ,PHI(9,9)
     J      ,RMEAS(3,3)
C
      REAL   QAP(3,3)
     A      ,VMAG
     B      ,TRV(3,3)
     C      ,ADOTVU
     D      ,AP
     E      ,GN
     F      ,RSPS
     G      ,TRS(3,3)
     H      ,THRUST
C
      REAL   TEMP(3,3)
C
      REAL   PVAL(9)
     A      ,DELTAT
     B      ,CONE
     C      ,CTWO
     D      ,CTHREE
     E      ,SIGMR
     F      ,STDAZ1
     G      ,STDAZ2
     H      ,STDEL1
     I      ,STDEL2
C
C            TRANSFORM ACCELERATION COVARIANCE MATRIX (QAP) TO
C            REFERENCE FRAME (QA)
C
C            QAP * TRV       STORE IN TEMP
C
        DO 2000  I = 1,3
          DO 1000  J = 1,3
            TEMP(I,J) = 0.0
```

```
          QA(I,J) = 0.0
 1000   CONTINUE
 2000 CONTINUE
      DO 5000  I = 1,3
        DO 4000  K = 1,3
          DO 3000  J = 1,3
             TEMP(I,K) = TEMP(I,K) + QAP(I,J) * TRV(J,K)
 3000     CONTINUE
 4000   CONTINUE
 5000 CONTINUE
C
C           TRV(TRANSPOSE) * TEMP     STORE IN QA
C
      DO 8000  I = 1,3
        DO 7000  K = 1,3
          DO 6000  J = 1,3
            QA(I,K) = QA(I,K) + TRV(J,I) * TEMP(J,K)
 6000     CONTINUE
 7000   CONTINUE
 8000 CONTINUE
C
C           TRANSITION ERROR COVARIANCE MATRIX QKAL
C
      DO 10000 I = 1,3
        DO 9000  J = 1,3
          QKAL(I,J) = QA(I,J)*DELTAT**4/36.
          QKAL(I,J+3)=QA(I,J)*DELTAT**3/12.
          QKAL(I,J+6) = QA(I,J)*DELTAT**2 /6.0
          QKAL(I+3,J)=QA(I,J)*DELTAT**3/12.
          QKAL(I+3,J+3) = QA(I,J)*DELTAT**2 /4.0
          QKAL(I+3,J+6) = QA(I,J)*DELTAT /2.0
          QKAL(I+6,J) = QA(I,J)*DELTAT**2 /6.0
          QKAL(I+6,J+3) = QA(I,J)*DELTAT /2.0
          QKAL(I+6,J+6) = QA(I,J)
 9000   CONTINUE
10000 CONTINUE
C
      RETURN
      END
```

Table A-14 Subroutine R

```
      SUBROUTINE R
C
C COMPUTES THE MEASURMENT NOISE COVARIANCE MATRIX R
C R INVERSE IS COMPUTED FOR THE BACKWARD SWEEP SMOOTHER
C
      COMMON /KALIO / RESDUL,BARN   ,SPS   ,PKAL  ,QA    ,RMEASI,QKAL
     A                ,ZMEAS ,HKAL  ,PHI   ,RMEAS
C
      COMMON /KALDAT/ PVAL  ,DELTAT,CONE  ,CTWO  ,CTHREE,SIGMR
     A               .,STDAZ1,STDAZ2,STDEL1,STDEL2
C
      COMMON /KALUT / QAP   ,VMAG  ,TRV   ,ADOTVU,AP    ,GN    ,RSPS
     A                ,TRS   ,THRUST
C
      REAL  RESDUL(3)
     A      ,BARN(9)
     B      ,SPS(9)
     C      ,PKAL(9,9)
     D      ,QA(3.3)
     E      ,RMEASI(3,3)
     F      ,QKAL(9,9)
     G      ,ZMEAS(3)
     H      ,HKAL(3,9)
     I      ,PHI(9,9)
     J      ,RMEAS(3,3)
C
      REAL  QAP(3.3)
     A      ,VMAG
     B      ,TRV(3,3)
     C      ,ADOTVU
     D      ,AP
     E      ,GN
     F      ,RSPS
     G      ,TRS(3,3)
     H      ,THRUST
C
      REAL  TEMP1(3,3)
     C      ,RLOS(3,3)
     J      ,RLOSI(3,3)
     K      ,TEMP2(3,3)
C
      REAL  PVAL(9)
     A      ,DELTAT
     B      ,CONE
     C      ,CTWO
     D      ,CTHREE
     E      ,SIGMR
     F      ,STDAZ1
     G      ,STDAZ2
     H      ,STDEL1
     I      ,STDEL2
C
C
C             COMPUTE COVARIANCE MATRIX IN ESTIMATED  L.O.S FRAME
C
        SIGRXS = SIGMR**2
```

```
      SIGRYS =(STDAZ1)**2+ ( RSPS*STDAZ2)**2
      SIGRZS =(STDEL1)**2  + (RSPS*STDEL2)**2
      DO 2000  I = 1,3
        DO 1000  J = 1,3
          RLOS(I,J) = 0.0
          RLOSI(I,J) = 0.0
          RMEAS(I,J) = 0.0
          RMEASI(I,J) = 0.0
          TEMP1(I,J) = 0.0
          TEMP2(I,J) = 0.0
 1000   CONTINUE
 2000 CONTINUE
      RLOS(1,1) = SIGRXS
      RLOS(2,2) = SIGRYS
      RLOS(3,3) = SIGRZS
      RLOSI(',1) = 1.0 / SIGRXS
      RLOSI(2,2) = 1.0 / SIGRYS
      RLOSI(3,3) = 1.0 / SIGRZS
C
C           TRANSFORM COVARIANCE MATRIX, RLOS, TO REFERENCE FRAME
C
C
C           RLOS * TRS, STORE IN TEMP1
C
      DO 5000  I = 1,3
        DO 4000  K = 1,3
          DO 3000  J = 1,3
            TEMP1(I,K) = TEMP1(I,K) + RLOS(I,J) * TRS(J,K)
            TEMP2(I,K) = TEMP2(I,K) + RLOSI(I,J) * TRS(J,K)
 3000       CONTINUE
 4000     CONTINUE
 5000 CONTINUE
C
C           TRS(TRANSPOSE) * TEMP1
C
      DO 8000  I = 1,3
        DO 7000  K = 1,3
          DO 6000  J = 1,3
            RMEAS(I,K) = RMEAS(I,K) + TRS(J,I) * TEMP1(J,K)
            RMEASI(I,K) = RMEASI(I,K) + TRS(J,I) * TEMP2(J,K)
 6000     CONTINUE
 7000   CONTINUE
 8000 CONTINUE
C
      RETURN
    END
```

Table A-15 Subroutine SUBS

```
      SUBROUTINE SUBS
C
C RECURSIVE INVERSION
C
      COMMON /MATINV/ Z      ,SZ
C
      DOUBLE PRECISION  Z(3,3)
     A                  ,SZ(3)
C
        SZ(1) = SZ(1) / Z(1,1)
C
C              FORWARD SUBSTITUTION
C
        DO 2000  I = 2,3
          K = I -1
          DO 1000  J = 1,K
            SZ(I) = SZ(I) - Z(J,I) * SZ(J)
 1000     CONTINUE
          SZ(I) = SZ(I) / Z(I,I)
 2000   CONTINUE
        SZ(3) = SZ(3) / Z(3,3)
        DO 4000  I = 2,3
          K = I - 1
          J1 = 4 - I
C
C              BACK SUBSTITUTION
C
            DO 3000  J = 1,K
            JT = 4 - J
            SZ(J1) = SZ(J1) - SZ(JT) * Z(J1,JT)
 3000     CONTINUE
          SZ(J1) = SZ(J1) / Z(J1,J1)
 4000   CONTINUE
C
        RETURN
      END
```

Next page is blank

DISTRIBUTION LIST
TR-246

No. of Copies	Organization
1	Commander US Army Materiel Development and Readiness Command ATTN: DRCPA-S 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Materiel Development and Readiness Command ATTN: DRCDE-R 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Materiel Development and Readiness Command ATTN: DRCBSI-AD 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Materiel Development and Readiness Command ATTN: DRCBSI-OR 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Materiel Development and Readiness Command ATTN: DRCBSI-SE 5001 Eisenhower Avenue Alexandria, VA 22333
2	Commander Defense Documentation Center ATTN: TCA Cameron Station Alexandria, VA 22314
2	Chief Defense Logistics Studies Information Exchange US Army Logistics Management Center ATTN: DRXMC-D FT Lee, VA 23801
1	HQDA (SAUS-OR) WASH DC 20310
2	HQDA (DAMO-RQA) COL Campbell/LTC Solomon WASH DC 20310
2	Commander US Army Concepts Analysis Agency ATTN: MOCA-MR MOCA-ADL 8120 Woodmont Avenue Bethesda, MD 20014
1	Commander US Army Operational Test and Evaluation Agency ATTN: CSTE-STS-L 5600 Columbia Pike Falls Church, VA 22041
4	Commander US Army Armament Research and Development Command ATTN: DRDAR-SE DRDAR-LCS DRDAR-SCF DRDAR-SCS DRCPM-ADG Dover, NJ 07801
2	Commander US Army Aviation Research and Development Command ATTN: DRDAV-BCA DRDAV-BCE PO Box 209 St. Louis, MO 63166

DISTRIBUTION LIST (Continued)
TR-246

No. of Copies	Organization
1	Director Research and Technology Laboratories ATTN: DAVDL Moffet Field, CA
1	Commander US Army Electronics Research and Development Command ATTN: DRDEL-AP-OA (B. Zarwyn) 2800 Powder Mill Road Adelphi, MD 20783
8	Commander US Army Missile Research and Development Command ATTN: DRDMI-X DRDMI-DS DRDMI-CD DRDMI-CW DRDMI-RSI DRDMI-Y DRCPM-HEL DRCPM-ROC Redstone Arsenal, AL 35809
1	Commander US Army Tank Automotive Research and Development Command ATTN: DRDTA-V Warren, MI 48090
1	Commander US Army Armament Materiel Readiness Command ATTN: DRSAR-LEP-L (Tech Lib) Rock Island, IL 62199
3	Commander US Army Missile Materiel Readiness Command ATTN: DRSMI-D DRCPM-CF DRCPM-HA Redstone Arsenal, AL 35809
1	Commander US Army Troop Support and Aviation Readiness Command ATTN: DRSTS-BA 4300 Goodfellow Boulevard St. Louis, MO 63120
2	Commander US Army Combat Developments Experimentation Command ATTN: ATEC-PL-M ATEC-PL-TL FT Ord, CA 93941
2	Director US Army TRADOC Systems Analysis Activity ATTN: ATAA-PL ATAA-TG White Sands Missile Range, NM 88002
1	Commandant US Army Air Defense School ATTN: ATSA-CD-A FT Bliss, TX 79916
1	Commander US Army Aviation Center and Fort Rucker ATTN: ATZQ-CD FT Rucker, AL 36362
1	President US Army Air Defense Board ATTN: ATZC FT Bliss, TX 79916
2	Commander US Army White Sands Missile Range ATTN: STEWS-TE-ML STEWS-NR-PA White Sands Missile Range, NM 88002

DISTRIBUTION LIST (Continued)
TR-246

No. of Copies	Organization
1	Commander US Army Combined Arms Combat Developments Activity ATTN: ATCA-CA Fort Leavenworth, KS 66027
1	Commandant US Military Academy ATTN: COL John L. Palmer Dept. of Engineering West Point, NY 10996
1	Commander Office of Naval Research ATTN: Mr. S.M. Selig 800 North Quincy Street Arlington, VA 22217
1	Director Special Projects Office Department of the Navy Washington, DC 20360
1	Commander Naval Weapons Center ATTN: Code 753 China Lake, CA 93557
1	AF/XOOFA Washington, DC 20330
1	AF/SAV Washington, DC 20330
1	AFSC (DOVS) Andrews AFB Washington, DC 20331
1	TAWC/OA (Mr. J. Durrenberger) Eglin AFB, FL 32542
1	AFWL (SA) Kirtland AFB, NM 87117
1	Director Institute for Defense Analysis ATTN: Dr. J. Ross 400 Army Navy Drive Arlington, VA 22202
1	United States Air Force Academy Department of Astronautics and Computer Science ATTN: DFACS/LTC E. J. Bauman USAF Academy, CO 80840
	Aberdeen Proving Ground
	Dir, AMSAA ATTN: D.O'Neill, AWD H.Peaker, AWD J.Meredith, AWD R.Scheder, AWD H.Burke, CSD T.Perkins, CSD S.Johnson, MSD
	Cdr, TECOM ATTN: DRSTE-CT-A DRSTE-CM-A DRSTE-AD-A DRSTE-AD-S DRSTE-AD-M
	Dir, HEL ATTN: DRXHE-SP
	Dir, BRL ATTN: DRDAR-BLB (H.Reed) DRDAR-BLB (P.Yip)
	Dir, MTD ATTN: STEAP-MT-M