Technical Note 1967-48

A. Bertolini

A Trajectory Analysis Program (TRAP)

29 September 1967

Prepared for the Advanced Research Projects Agency under Electronic Systems Division Contract AF 19(628)-5157 by Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Lexington, Massachusetts
The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This research is a part of Project DEFENDER, which is sponsored by the U.S. Advanced Research Projects Agency of the Department of Defense; it is supported by ARPA under Air Force Contract AF 19(628)-5167 (ARPA Order 498).

This report may be reproduced to satisfy needs of U.S. Government agencies.

This document has been approved for public release and sale; its distribution is unlimited.
A TRAJECTORY ANALYSIS PROGRAM (TRAP)

A. BERTOLINI

Group 42

TECHNICAL NOTE 1967-48

29 SEPTEMBER 1967
ABSTRACT

A FORTRAN IV program package has been written for the generation, analysis, and estimation of re-entry trajectories. The following functions may be performed by the program:

(1) Generation of simulated noiseless trajectory data by integrating the differential equations of motion, using a predictor-corrector method.

(2) Error analyses on trajectories generated by the program, in which the effects of hypothetical errors in the initial state and subsequent measurements may be studied.

(3) Estimation of state vector quantities, using a recursive Kalman-filter scheme, from
   (a) Simulated noisy data generated by the program.
   (b) Noisy radar measurements, supplied externally to the program.
# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>iii</td>
</tr>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>II. MAIN PROGRAM (VERSION I)</td>
<td>2</td>
</tr>
<tr>
<td>III. MAIN PROGRAM (VERSION II)</td>
<td>7</td>
</tr>
<tr>
<td>IV. SUBROUTINE TRAJGX</td>
<td>7</td>
</tr>
<tr>
<td>A. Description</td>
<td>7</td>
</tr>
<tr>
<td>B. Mathematical Discussion</td>
<td>11</td>
</tr>
<tr>
<td>V. SUBROUTINE ESTMAT</td>
<td>15</td>
</tr>
<tr>
<td>A. Description</td>
<td>15</td>
</tr>
<tr>
<td>B. Mathematical Discussion</td>
<td>20</td>
</tr>
<tr>
<td>VI. SUBROUTINE DENS</td>
<td>21</td>
</tr>
<tr>
<td>A. Description</td>
<td>21</td>
</tr>
<tr>
<td>B. Mathematical Discussion</td>
<td>26</td>
</tr>
<tr>
<td>VII. SUBROUTINE XYTOR</td>
<td>26</td>
</tr>
<tr>
<td>VIII. SUBROUTINE RTOXY</td>
<td>27</td>
</tr>
<tr>
<td>IX. SUBROUTINES XCOVTR AND RCOVTX</td>
<td>28</td>
</tr>
<tr>
<td>A. Description of XCOVTR</td>
<td>28</td>
</tr>
<tr>
<td>B. Description of RCOVTX</td>
<td>29</td>
</tr>
<tr>
<td>C. Mathematical Discussion</td>
<td>29</td>
</tr>
<tr>
<td>X. SUBROUTINE GAUSSN</td>
<td>32</td>
</tr>
<tr>
<td>A. Description</td>
<td>32</td>
</tr>
<tr>
<td>B. Mathematical Discussion</td>
<td>33</td>
</tr>
<tr>
<td>XI. SUBROUTINE DMTMUL</td>
<td>33</td>
</tr>
<tr>
<td>XII. SUBROUTINE MINV</td>
<td></td>
</tr>
<tr>
<td>APPENDIX A – Some Relations Used in Calculation of Derivatives for $A$ Matrix $\frac{\partial f(x)}{\partial x}$</td>
<td>37</td>
</tr>
<tr>
<td>APPENDIX B – Program Listings</td>
<td>39</td>
</tr>
</tbody>
</table>
A TRAJECTORY ANALYSIS PROGRAM (TRAP)

I. INTRODUCTION

A FORTRAN IV double-precision program package has been written for the generation, analysis, and estimation of re-entry trajectories. The trajectories are characterized by a seven-component state vector specifying position, first time derivative of position, and drag parameter defined as the reciprocal of the weight-to-drag ratio. The program assumes a spherical, rotating earth with an atmosphere and uses either English or metric units. The following functions may be performed by the program package:

(a) Generation of simulated noiseless trajectory data by integrating the differential equations of motion, using a predictor-corrector method.

(b) Error analyses on trajectories generated by the program, in which the effects of hypothetical errors in the initial state and subsequent measurements may be studied.

(c) Estimation of state vector quantities, using a recursive Kalman-filter scheme, from

1) Simulated noisy data generated by the program.

2) Noisy radar measurements, supplied externally to the program.

A modified version of the main program is required to perform function (c) (2).

The state vector which describes the trajectory motion has seven components, and may be considered in two coordinate systems. In radar-centered rectangular coordinates, with x pointing east, y north, and z vertically upward, we have the following:

\[
\begin{align*}
  x_1 &= x \\
  x_2 &= y \\
  x_3 &= z \\
  x_4 &= \dot{x} \\
  x_5 &= \dot{y} \\
  x_6 &= \dot{z} \\
  x_7 &= \alpha = \text{drag parameter} = 1/\beta
\end{align*}
\]

In radar-centered polar coordinates, which is the usual measurement coordinate system

\[
\begin{align*}
  r_1 &= \text{range} \\
  r_2 &= \text{azimuth} \\
  r_3 &= \text{elevation} \\
  r_4 &= \text{range rate} \\
  r_5 &= \text{azimuth rate} \\
  r_6 &= \text{elevation rate} \\
  r_7 &= \alpha = \text{drag parameter}
\end{align*}
\]
Input and output quantities may be given in polar or rectangular coordinates but all calculations are done in rectangular coordinates. English (pounds, feet, seconds) or metric (meters, kilograms, seconds) units may be specified.

The program package consists of the main program plus ten subroutines. The subroutines perform various calculations, ranging from integrating the differential equations of motion to multiplying matrices. Some of these subroutines may be of general interest for use outside of the program package. The functions of the main program and all the subroutines are summarized in Table I, with full details given in later sections.

II. MAIN PROGRAM (VERSION I)

The main program reads input cards, calls the appropriate subroutines to perform the desired calculations, and prints output data. Data on the input cards completely specifies the program function (e.g., trajectory generation, error analysis, or estimation), furnishes required initial conditions, and specifies the amount of printed output desired. If required, the user may change the program so that the output data may be written on tape, plotted, or punched on cards. The formats of the seven required input cards are given in Table II. Any number of trajectories may be run in succession by stacking the input cards for each case in succession. A block diagram of the main program is shown in Fig. 1.

The following COMMON statements are used:

```
COMMON/ACOM/COVAR(7), SIGMA(7)/FCOM/COORD, DLAT,
PRNT/ICOM/KLAMP, MDATA, NN
```

An explanation of these variables is given in Table II.

Varying amounts of printed output may be obtained by proper specification of the print-selector parameter PRNT. These are summarized below:

PRNT < 0. Every 50 data points, the following tabulations are printed for the preceding 50 data points.

(a) Time, height, and the seven components of the nominal state vector in radar-centered polar coordinates.

(b) Time, aspect angle, and the seven components of the nominal state vector in radar-centered rectangular coordinates.

(c) Time, plus the measurement vector in radar-centered polar coordinates (printed only in estimation cases).

(d) Time, and the estimated state vector in radar-centered polar and rectangular coordinates (printed only in estimation cases).

(e) Time, and the errors (difference between estimated and nominal state vector) in polar and rectangular coordinates.

(f) Theoretical rms errors, obtained from covariance matrices, in polar and rectangular coordinates.

Note: The above may be printed out at data intervals other than every 50 points by properly specifying the value of IMAX in statement 9 of the main program (see Tables B-I and B-II in Appendix B). All one-dimensional arrays of size 54 should be redimensioned to be equal to or greater than the new value of IMAX.
Add noise samples to initial state in rectangular coordinates

Call XYTOR to convert to polar coordinates

Note: Noise - 0 if MDATA = 0

Add noise samples to initial state in radar polar coordinates

Call RTOXY to convert to rectangular coordinates

Set initial state estimate = initial noisy state

Compute errors = estimate - nominal state

Fig. 1. Flow chart of TRAP (version I) main program.
Fig. 1. Continued.
### TABLE I
LIST OF PROGRAMS IN TRAP PACKAGE

<table>
<thead>
<tr>
<th>Program or Subroutine</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main Program</strong> (Versions I and II)</td>
<td>Reads input cards specifying desired calculations and initial conditions, calls appropriate subroutines, and prints out results.</td>
</tr>
<tr>
<td>TRAJGX</td>
<td>Integrates differential equations of motion in radar-centered rectangular coordinates. A spherical rotating earth with a realistic atmosphere is used.</td>
</tr>
<tr>
<td>ESTMAT</td>
<td>Estimates the true value of a state vector by combining a predicted value of the state vector with a noisy measurement. Also calculates covariance matrices for error analyses.</td>
</tr>
<tr>
<td>DENS</td>
<td>Calculates atmospheric density at any given altitude.</td>
</tr>
<tr>
<td>XYTOR</td>
<td>Converts from radar-centered rectangular coordinates to polar coordinates.</td>
</tr>
<tr>
<td>RTOXY</td>
<td>Converts from radar-centered polar coordinates to rectangular coordinates.</td>
</tr>
<tr>
<td>XCOVTR</td>
<td>Converts a mean-square error covariance matrix in rectangular coordinates to one in polar coordinates. It also calculates a partial derivative matrix $H = \partial \hat{r}/\partial \hat{x}$ where $\hat{r}$ and $\hat{x}$ are the state vector in radar-centered polar and rectangular coordinates respectively.</td>
</tr>
<tr>
<td>RCOVTX</td>
<td>Converts a mean-square error covariance matrix in polar coordinates to one in rectangular coordinates.</td>
</tr>
<tr>
<td>GAUSSN</td>
<td>Generates random numbers with a Gaussian distribution.</td>
</tr>
<tr>
<td>DMATMUL</td>
<td>Multiplies two matrices in double precision.</td>
</tr>
<tr>
<td>MINV</td>
<td>Matrix inversion subroutine from IBM 360 Scientific Subroutine Package (double-precision version used).</td>
</tr>
</tbody>
</table>

**PRNT = 0.** All of the above are printed, as well as the following which are printed for every data point.

(a) Time, height, aspect angle, and the nominal state vector in both polar and rectangular coordinates.

(b) The transition matrix, $\partial x(t_k)/\partial x(t_{k-1})$ (see discussion of subroutine TRAJGX, Sec. IV), along the nominal trajectory in rectangular coordinates.

(c) The noisy measurement vector, in polar coordinates (printed only in estimation cases).

(d) The predicted state vector, in polar and rectangular coordinates (printed only in estimation cases).

(e) The covariance matrices of the estimated state vector (see discussion of subroutine ESTMAT, Sec. V).

(f) The estimated state vector in polar and rectangular coordinates.

**PRNT > 0.** All of the above are printed, as well as the following which are printed by subroutine ESTMAT for every data point.

(a) Transition matrix used in the estimation.

(b) Covariance matrices for predicted and estimated data points.

(c) Various vectors and matrices used in the calculation of the final covariance matrices and the estimate (see description of subroutine ESTMAT, Sec. V).
<table>
<thead>
<tr>
<th>Card</th>
<th>Format</th>
<th>Arguments and/or Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18A4</td>
<td>Title card with 72 alpha-numerical characters available.</td>
</tr>
<tr>
<td>2</td>
<td>7F10.3</td>
<td>(STATEI(J), J = 1, 7) where STATEI is initial value of nominal state vector, in either polar or rectangular radar-centered coordinates.</td>
</tr>
</tbody>
</table>
| 3    | 5F10.3, F5.3, 3F5 | TZERO = initial time (in seconds)  
DELT = time of first measurement with respect to time of initial state  
TINT = interval, in seconds, between data points  
TINCR = trajectory integration step size, in seconds  
DLAT = latitude of radar (reference) site in degrees  
PRNT = print-selector parameter (see text, p. 2)  
NDATA = number of data points to be processed  
KLAMP = "memory" of estimation algorithm, given in number of data points (see description of ESTMAT, Sec. V); KLAMP = 0 is used to indicate infinite memory  
MDATA = mode parameter  
MDATA < 0 specifies error analysis  
MDATA = 0 specifies noiseless trajectory generation  
MDATA > 0 specifies estimation  
The magnitude of MDATA < 7 and specifies the size of the measurement vector used in calculations (see description of ESTMAT, Sec. V). |
| 4    | 7F10.3 | (SIGMA(J), J = 1, 6) = rms noise levels associated with measurement vector. No value is given to SIGMA(7) since the drag parameter, which is the seventh component of the state vector, is not measurable directly.  
COORD = Coordinate – specification parameter  
COORD = -2. Initial state vector given in radar-centered polar coordinates. Metric units used for all calculations.  
COORD = -1. Initial state vector given in radar-centered rectangular coordinates. Metric units used for all calculations.  
COORD = 1. Initial state vector given in radar-centered rectangular coordinates. English units used for all calculations.  
COORD = 2. Initial state vector given in radar-centered polar coordinates. English units used for all calculations. |
| 5    | 7F10.3 | (COVAR(J), J = 1, 7) = rms noise levels associated with initial state vector in units specified by the parameter COORD. |
| 6    | 7F10.3 | (FNOISE(J), J = 1, 7) = noise samples associated with initial state vector, in units specified by the parameter COORD. |
| 7    | 7F10.3 | (ERRMAX(J), J = 1, 7) = magnitude of maximum allowable convergence errors. The convergence errors are an indication of the accuracy of the integration of the trajectory equations – the smaller the errors, the better the accuracy. Values of 0.1 for all seven elements in the ERRMAX array have been found suitable for most cases. See the mathematical discussion of subroutine TRAJGX, Sec. IV, for a description of the convergence errors. |
III. MAIN PROGRAM (VERSION II)

A second main program is available for use in estimating trajectory parameters from real radar measurement data. The measurement data are read in on cards but it is a simple matter to change the input medium and/or format. No error analysis or simulated trajectory generation is done. Four input cards, followed by the measurement data cards, are required (see Table III).

<table>
<thead>
<tr>
<th>Card</th>
<th>Format</th>
<th>Argument and/or Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18A4</td>
<td>Title card, with 72 alpha-numerical characters available.</td>
</tr>
<tr>
<td>2</td>
<td>4F10.3, F5.3, 315</td>
<td>TZERO, TINT, TINCR, DLAT, PRNT, NDATA, KLAMP, MDATA (see text below).</td>
</tr>
<tr>
<td>3</td>
<td>7F10.3</td>
<td>(SIGMA(J), J = 1, 6), COORD</td>
</tr>
<tr>
<td>4</td>
<td>7F10.3</td>
<td>(ERRMAX(J), J = 1, 7) convergence errors</td>
</tr>
<tr>
<td>5</td>
<td>D15.2, 4D15.6</td>
<td>Data cards for each measurement, giving time, range, azimuth, elevation, range rate</td>
</tr>
</tbody>
</table>

The title card and the arguments TINCR, DLAT, PRNT, NDATA, KLAMP, SIGMA, and ERRMAX are exactly as described previously for Version I. The arguments MDATA and COORD are also as described previously except that MDATA is limited to positive values while COORD must equal ±2.

The arguments TZERO and TINT specify the starting time of the desired data and the minimum desired data interval, respectively. All data prior to time TZERO are ignored, as well as succeeding data points which are less than TINT seconds later than the previously processed point.

It should be noted that an initial state vector to start the program is obtained from the measurements at time TZERO or later. Any required time derivatives not contained in the measurement are calculated by taking differences of measurements TINT seconds, or more, apart.

A block diagram of this program is given in Fig. 2.

IV. SUBROUTINE TRAJGX

A. Description

Subroutine TRAJGX is a double-precision FORTRAN subroutine which integrates the differential equations of motion in radar-centered rectangular coordinates over any specified time interval, using a predictor-corrector method. It assumes a spherical rotating earth with a rigid atmosphere. A choice of English units (feet, pounds, seconds), or metric units (meters, kilograms, seconds), is available by specifying the parameter COORD, in COMMON storage. Atmospheric data are obtained by calling subroutine DENS via the entry point ATM, described in Sec. VI. Subroutine TRAJGX may be used to generate a ballistic trajectory or to predict the position and velocity of a re-entry vehicle at any given time, given the position, velocity, and ballistic coefficient at some other time.
Fig. 2. Flow chart of TRAP (version II) main program.
Last data point

Set $K_{\text{MAX}} = \text{No. of remaining data points to be printed}$

Here data from a sufficient number ($I_{\text{MAX}}$) of points been processed since last printout

Yes

$K_{\text{MAX}} = I_{\text{MAX}}$

Print tabulated output data from last $K_{\text{MAX}}$ points

No

Last data point

$N_{\text{N}} = N_{\text{N}} + 1$

Fig. 2. Continued.
The subroutine is called by the following statement:

\[
\text{CALL TRAJGX (XI, Y1, Z1, XDOT1, YDOT1, ZDOT1, BETA1, TAU, TSTEP, ERRMAX, X2, Y2, Z2, XDOT2, YDOT2, ZDOT2, BETA2, PHIMAT)}
\]

The input arguments are:

- \( X1 \) initial position in radar-centered rectangular coordinates
- \( Y1 \)
- \( Z1 \)
- \( XDOT1 = \dot{X1} \) initial velocity components
- \( YDOT1 = \dot{Y1} \)
- \( ZDOT1 = \dot{Z1} \)
- \( BETA1 = \) initial ballistic coefficient
- \( TAU = \) integration time interval (sec)
- \( TSTEP = \) integration step size (sec)
- \( ERRMAX = \) convergence errors (see description below)

The output arguments are:

- \( X2 \) position at time \( TAU \) seconds after initial time
- \( Y2 \)
- \( Z2 \)
- \( XDOT2 = \dot{X2} \) velocity components at time \( TAU \) seconds after initial time
- \( YDOT2 = \dot{Y2} \)
- \( ZDOT2 = \dot{Z2} \)
- \( BETA2 = \) ballistic coefficient at time \( TAU \) seconds after initial time.

(At present, TRAJGX assumes that the ballistic coefficient is constant and sets \( BETA2 = BETA1 \). Provision can be made, however, for a varying ballistic coefficient).

\( PHIMAT = \) transition matrix, whose elements are defined by

\[
\varphi_{ij} = \frac{\partial x_i(t_o + TAU)}{\partial x_j(t_o)}
\]

where \( x_i \) is the \( i \)th component of a state vector \( x \), whose components are

\[
\begin{align*}
x_1(t_o) &= X1 \\
x_2(t_o) &= Y1 \\
x_3(t_o) &= Z1 \\
x_4(t_o) &= XDOT1 \\
x_5(t_o) &= YDOT1 \\
x_6(t_o) &= ZDOT1 \\
x_7(t_o) &= 1/BETA1 \\
\end{align*}
\]

Similarly
\[ x_1(t_0 + TAU) = X2 \]
\[ \vdots \]
\[ x_7(t_0 + TAU) = 1.0/BETA2 \]

The following statements are also required in the calling program:

```plaintext
COMMON/FCOM/COORD, DLAT, PRNT/ICOM/KLAMP, MDATA, NN
DIMENSION ERRMAX(7)
```

where

- **DLAT** = latitude, in degrees, of radar (reference) site
- **COORD** = 0. indicates English system units of feet, pounds, seconds
- **COORD** < 0. indicates metric system units of meters, kilograms, seconds
- **NN** indicates the data point under current consideration (NN = 1 for the first data point and indicates that TRAJGX is being called for the first time. This is the only case of interest to TRAJGX)
- **ERRMAX** is an array of convergence errors associated with the seven components of the state vector \( \mathbf{x} \). In general, the smaller the specified convergence errors, the more accurate the integration, with a resulting increase in computation time. The significance of the convergence errors is discussed in Sec. IV. B. Values of 0.1 seem to be suitable for all seven convergence errors.

The following values have been used for some fundamental parameters.

- \( R_E = 2.0925738 \times 10^7 \text{ ft} \) (radius of the earth)
- \( G_M = 1.40763488 \times 10^{16} \text{ ft}^3/\text{sec}^2 \) (Newton's gravitational constant x mass of earth)
- \( \omega = 7.27 \times 10^{-5} \text{ radians/sec} \) (earth's angular rotation rate)
- 1 meter = 3.2808333 ft
- \( \pi = 3.14159265 \)

### B. Mathematical Discussion

The program assumes the following basic equations of motion in radar-centered rectangular coordinates:

\[
\begin{align*}
\dot{x} &= P_{xx} \dot{x} + Bx + Cy + Ez \\
\dot{y} &= -Cx + Gy + P_{yy} \dot{y} + Jz + K \\
\dot{z} &= -Ex + Jy + Nz + P_{zz} \dot{z} + Q
\end{align*}
\]

where

\[ \alpha = 1/\beta = \text{reciprocal of ballistic coefficient = drag parameter} \]

\[ P = -\rho V_D/2 \]

\[ V_D = \text{drag velocity} = (\dot{x}^2 + \dot{y}^2 + \dot{z}^2)^{1/2} \]
TABLE IV

\[ A \text{ MATRIX } = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \]

\[
\frac{\partial \mathbf{F}}{\partial \mathbf{x}} = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
B + x \frac{\partial B}{\partial x} & x \frac{\partial B}{\partial y} & x \frac{\partial B}{\partial z} & (\alpha P + \alpha x \frac{\partial P}{\partial x}) & (\dot{x} \alpha \frac{\partial P}{\partial x} + C) & (\dot{x} \alpha \frac{\partial P}{\partial x} + E) & (P\dot{x}) \\
\dot{x}(\alpha \frac{\partial P}{\partial y} + P \frac{\partial \alpha}{\partial x}) & \dot{x}(\alpha \frac{\partial P}{\partial y} + P \frac{\partial \alpha}{\partial x}) & \dot{x}(\alpha \frac{\partial P}{\partial y} + P \frac{\partial \alpha}{\partial x}) & (\dot{y} \alpha \frac{\partial P}{\partial y} - C) & [\alpha (P + \dot{y} \frac{\partial P}{\partial y})] & (\dot{y} \alpha \frac{\partial P}{\partial y}) & (P\dot{y}) \\
y(P \frac{\partial \alpha}{\partial x} + \alpha \frac{\partial \alpha}{\partial x}) & y(P \frac{\partial \alpha}{\partial y} + \alpha \frac{\partial \alpha}{\partial y}) & y(P \frac{\partial \alpha}{\partial z} + \alpha \frac{\partial \alpha}{\partial z}) & (\dot{z} \alpha \frac{\partial P}{\partial x} - E) & (\dot{z} \alpha \frac{\partial P}{\partial y}) & [\alpha (P + \dot{z} \frac{\partial P}{\partial z})] & (P\dot{z}) \\
\frac{\partial Q}{\partial x} + z \frac{\partial \alpha}{\partial x} & \frac{\partial Q}{\partial y} + z \frac{\partial \alpha}{\partial y} & \frac{\partial Q}{\partial z} + z \frac{\partial \alpha}{\partial z} & (\dot{x} \alpha \frac{\partial P}{\partial x}) & (\dot{y} \alpha \frac{\partial P}{\partial y}) & (\dot{z} \alpha \frac{\partial P}{\partial z} + \alpha \frac{\partial P}{\partial z}) & (\dot{z} \alpha \frac{\partial P}{\partial z}) \\
\frac{\partial \alpha}{\partial x} & \frac{\partial \alpha}{\partial y} & \frac{\partial \alpha}{\partial z} & (\dot{\alpha}) & (\dot{\alpha}) & (\dot{\alpha}) & (\dot{\alpha}) \\
\end{bmatrix}
\]

1. Since this subroutine assumes a constant drag parameter \( \alpha \), all derivatives of \( \alpha \) are set = 0. If it were desired to provide for a variable \( \alpha \), the seventh row of the matrix would have to be filled in accordingly.

2. See Appendix A for relations used in calculation of derivatives of \( A \) matrix \( \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \).
The position and motion of the vehicle are described in terms of the state vector

\[ \mathbf{x} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \alpha \end{bmatrix} \]

The equations of motion are numerically integrated using a predictor-corrector method, which makes use of the state vector \( \mathbf{x} \), the vector \( \mathbf{b} = \mathbf{f}(\mathbf{x}) = \partial \mathbf{x} / \partial t \), and the matrix \( \mathbf{A} = \partial \mathbf{f}(\mathbf{x}) / \partial \mathbf{x} \). A block diagram of the integration scheme, proposed by M. Gruber, is shown in Fig. 3. The vector \( \mathbf{b} \) is written more explicitly below, while Table IV gives the matrix \( \mathbf{A} \).

\[ \mathbf{b} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \end{bmatrix} \]

Figure 3 shows that the vector \( \mathbf{x}(t_k) \) at time \( t_k \) is integrated over one step size to time \( t_{k+1} \), giving a predicted state vector \( \hat{\mathbf{x}}(t_{k+1}) \) which is based on knowledge of the vectors \( \mathbf{x}(t_k) \) and \( \mathbf{x}(t_{k-1}) \). A corrected vector \( \mathbf{x}(t_{k+1}) \) is then calculated, using the newly obtained prediction \( \hat{\mathbf{x}}(t_{k+1}) \) and \( \mathbf{x}(t_k) \). If the convergence errors, defined as \( |\hat{\mathbf{x}}(t_{k+1}) - \mathbf{x}(t_{k+1})| \), are less than the values specified in the ERRMAX array for all seven components of \( \mathbf{x} \), \( \mathbf{x}(t_{k+1}) \) is accepted as
Fig. 3. Flow chart of TRAJGX calculations.
the value of $x$ at time $t_{k+1}$. If not, $\hat{x}(t_{k+1})$ is set equal to $x(t_{k+1})$ and another value of $x(t_{k+1})$ is calculated based on $x(t_k)$ and the corrected vector $\hat{x}(t_{k+1})$. New convergence errors are calculated and the comparison test and subsequent procedure are repeated until sufficiently small convergence errors are obtained. The above procedure is repeated for each integration step until the desired interval has been covered. Figure 4 is a flow chart of the subroutine.

V. SUBROUTINE ESTMAT

A. Description

Subroutine ESTMAT estimates the true value of a state vector by combining a predicted value, given in radar-centered rectangular coordinates, with a noisy measurement given in radar-centered polar coordinates. It is essentially a recursive Kalman-filter scheme. Double-precision calculations are used throughout. In making the estimate, the prediction and the measurement are weighted in proportion to the inverse of their respective mean-square error covariance matrices. ESTMAT may also be used to perform trajectory error analyses, in which the effects of hypothetical errors in the initial state and subsequent measurements are calculated. Suitable variances are inserted in the appropriate covariance matrices which are evaluated and propagated along the nominal trajectory.

The subroutine calculates the mean-square error covariance matrix of the estimate, and prints it out if desired. This matrix is saved by the subroutine to be used, along with the transition matrix, in computing the covariance matrix of the next prediction. The transition matrix, defined by the equation

$$\Phi_{k, k-1} = \frac{\partial x(t_k)}{\partial x(t_{k-1})}$$

is required as an input argument to ESTMAT. It is computed by subroutine TRAJGX in the course of calculating the predicted value of $x(t_k)$. If TRAJGX is not used, a suitable transition matrix must be calculated elsewhere.

Subroutines DMTMUL, MINV, RCOVTX, XCOVTR, and XYTOR are also required by ESTMAT.

One problem that arises in linear recursive tracking is that at a sufficiently high data rate, the computed covariance matrix of the estimate will tend towards zero. This has the effect of giving progressively less weight to measurements. As a result of errors in our model of the state equation, or in our covariance matrix calculations, or in the numerical integration routine (especially during re-entry), it is often advisable to weight the measurements higher than the algorithm normally would.

An arbitrarily chosen device (others are possible) that has been used to remedy the situation is to prevent the determinant of the covariance matrix from decreasing below a given reference value. This reference is generally set equal to the determinant of the covariance matrix after a specified number of data points have been processed. For all subsequent estimates, the determinant of the covariance matrix is calculated and the entire covariance matrix is scaled up so that its determinant is equal to the reference. The determinant, and also the covariance matrix, are said to be "clamped," and prevented from decreasing to zero. A parameter KLAMP, referred to as the clamping time, or memory, of the algorithm specifies the number of data points to be processed before calculating the reference determinant.
Fig. 4. Flow chart of subroutine TRAJGX.
Fig. 4. Continued.
Subroutine ESTMAT is called by the following statement:

CALL ESTMAT (STATEM, STATEP, PHIMTX, ESTATE, DEVX, DEVR)

The input arguments are:

- **STATEM** Measurement vector in radar-centered polar coordinates
  - STATEM(1) = range measurement
  - STATEM(2) = azimuth measurement (in radians)
  - STATEM(3) = elevation measurement (in radians)
  - STATEM(4) = range rate (Doppler) measurement (if used)
  - STATEM(5) = azimuth rate measurement (if used)
  - STATEM(6) = elevation rate measurement (if used)
  - STATEM(7) : generally not used

- **STATEP** Predicted state vector in radar-centered rectangular coordinates

- **PHIMTX** Transition matrix (Note: this matrix is destroyed by ESTMAT)

The output arguments are:

- **ESTATE** Estimated state vector in radar-centered rectangular coordinates

- **DEVX** Seven-component vector containing expected rms errors of estimated state vector in rectangular coordinates

- **DEVR** Seven-component vector containing expected rms errors of estimated state vector in radar polar coordinates

The following statements are also required in the calling program:

```plaintext
COMMON/ACOM/COVAR(7), SIGMA(7)/FCOM/COORD, DLAT, PRNT/
ICOM/KLAMP, MDATA, NN'
DIMENSION STATEM(7), STATEP(7), ESTATE(7), PHIMTX(7,7),
DEVX(7), DEVR(7)
```

The quantity MDATA is an integer specifying the mode of operation of ESTMAT, and may have values from −7 to +7, but not zero. (If MDATA = 0, noiseless trajectory data will be generated and subroutine ESTMAT will not be called.) The magnitude of MDATA indicates the number of measurement vector components used, while the sign indicates the type of job to be done – positive for estimation, negative for error analysis. Some common values are given below:

- **MDATA = 4** state-vector estimation, using range, azimuth, elevation, and range rate (Doppler) measurements
- **MDATA = 3** state-vector estimation, using range, azimuth, and elevation measurements
- **MDATA = 1** state-vector estimation, using range measurements only
- **MDATA = −4** error analysis, assuming range, azimuth, elevation, and range rate (Doppler) measurements
- **MDATA = −3** error analysis, assuming range, azimuth, and elevation measurements
- **MDATA = −1** error analysis, assuming range measurement only.

The SIGMA array contains the rms errors associated with the measurement vector.
SIGMA(1) = rms range measurement error  
SIGMA(2) = rms azimuth measurement error (in radians)  
SIGMA(3) = rms elevation measurement error (in radians)  
SIGMA(4) = rms range rate (Doppler) measurement (if used)  
SIGMA(5) = rms azimuth rate measurement error (if used)  
SIGMA(6) = rms elevation rate measurement error (if used)  
SIGMA(7) : not used  

The COVAR array contains the rms errors associated with the initial estimate of the state vector. (At the start of a run, an initial estimate of the state vector, along with rms errors associated with it, must be given.) The elements of the COVAR array may be in radar-centered rectangular or polar coordinates and are listed below.

COVAR(1) = rms error in initial estimate of x (or range)  
COVAR(2) = rms error in initial estimate of y (or azimuth)  
COVAR(3) = rms error in initial estimate of z (or elevation)  
COVAR(4) = rms error in initial estimate of \( \dot{x} \) (or range rate)  
COVAR(5) = rms error in initial estimate of \( \dot{y} \) (or azimuth rate)  
COVAR(6) = rms error in initial estimate of \( \dot{z} \) (or elevation rate)  
COVAR(7) = rms error in initial estimate of \( \alpha \) = drag parameter = \( 1/\beta \)

The specification of the coordinate system of COVAR is indicated by the value of the parameter COORD.

COORD = ±1.0  COVAR in rectangular coordinates  
COORD = ±2.0  COVAR in polar coordinates  

A selection of printed output is available by specification of the parameter PRNT.

PRNT < 0. No printed output from ESTMAT  
PRNT = 0. The covariance matrix of the estimated state vector is printed for each point processed. It is given in rectangular coordinates along with a "hybrid" matrix in polar coordinates. In the hybrid matrix, the diagonal terms are rms, rather than mean-square, errors while off-diagonal terms are correlation coefficients of the mean-square errors. Mathematically,

\[
\begin{align*}
    c_{ii} &= \sqrt{\sigma^2_i} \quad \text{(diagonal terms)} \\
    c_{ij} &= \frac{\sigma^2_{ij}}{\sqrt{\sigma^2_{ii} \sigma^2_{jj}}} \quad \text{(off-diagonal terms)}
\end{align*}
\]

where

\( \sigma^2 \) = mean square covariance in polar coordinates  
\( c \) = hybrid matrix element  

PRNT > 0. Covariance matrices for predicted as well as estimated state vectors are printed for each data point in the same formats described above. In addition, the partial derivative matrix \( H \), the weighting matrix \( W \) (see Sec. V. B), and the estimated state vector in rectangular coordinates are printed.
Other quantities in COMMON storage are:

\( NN \) = number of the data point currently being processed.

\( KLAMP \) = clamping, or memory, time of process note: \( KLAMP = 0 \) is used to indicate that no clamping is desired. The same effect may be obtained by making \( KLAMP \) greater than the total number of points to be processed.

\( DLAT \) = latitude of radar (reference) site, in degrees (not used by ESTMAT)

B. Mathematical Discussion

As stated in part (A) of this section, ESTMAT estimates the true value of a state vector by combining a predicted value of the state vector with a noisy measurement vector. The mathematical formula is given below:

\[
\hat{x}_k = x_{k, k-1} + W [r_k - h(x_{k, k-1})]
\]

where

\[
W = (S_k^{-1} + H^T R^{-1} H)^{-1} H^T R^{-1}
\]

\[
\hat{x}_k = \text{estimate of state vector } x \text{ at time } t_k
\]

\( x_{k, k-1} \) = predicted state vector \( x \) at time \( t_k \), based on the estimate of \( x \) at time \( t_{k-1} \)

\( r_k \) = measurement vector at time \( t_k \)

\( S_k \) = mean-square error covariance matrix of \( \hat{x}_k \)

where \( (S_k^{-1})_{ij} = E\{[(\hat{x}_k)_i - (\hat{x}_k)_j] [(\hat{x}_k)_j - (\hat{x}_k)_i]^T\} \)

and \( \hat{x}_k \) = nominal (noiseless) value of \( x \).

\( S_{k, k-1} \) = mean-square error covariance matrix of \( x_{k, k-1} \)

\( R \) = mean-square error covariance matrix of measurement vector \( r \).

\( H \) = partial derivative matrix \( \partial r / \partial x \)

where \( r \) = vector in measurement coordinates

\( x \) = vector in estimation coordinates

\( r = h(x) \)

If \( r \) has \( m \) components and \( x \) has \( n \) components \( H \) will have \( m \) rows and \( n \) columns. If \( r \) and \( x \) are in the same coordinate system, \( H_{ij} = 16 (i - j) \)

Equation (1) is approximately equivalent to

\[
\hat{x}_k \approx S_k^{-1} [S_k^{-1} x_{k, k-1} + H^T R^{-1} r_k]
\]

Equation (3) shows that the prediction \( x_{k, k-1} \) and the measurement \( r_k \) are each weighted by the inverse of their respective covariance function. The matrix \( S_k \) is related to the other matrices by the relation
\[ S_k = \left( S_{k-1}^{-1} + H^T R^{-1} H \right)^{-1} \]  

and \( S_{k-1} \) is related to the covariance matrix of the estimate at the previous point by

\[ S_{k-1} = \Phi_{k-1} S_{k-1} \Phi_{k}^T \]

where

\[ \Phi_{k-1} = \frac{\partial x_{k-1}}{\partial S_{k-1}} \]

The matrix \( \Phi \) is called the transition matrix and is calculated in subroutine TRAJGIX.

Figure 5 shows a flow chart of version I of ESTMAT, based on Eqs. (1) and (2).

Emphasis is on the inverse covariance matrix \( S^{-1} \), which is continually updated by the equation

\[ S_{k-1}^{-1} = \Phi_{k-1}^{-1} S_{k-1}^{-1} \Phi_{k}^{-1} \]

Matrix inversion is performed to obtain \( \Phi_{k-1}^{-1} \) for use in Eq. (7) and to calculate the weighting matrix \( W \) of Eq. (2). Since \( x \) is a seven-component vector, both of these inversions involve \( 7 \times 7 \) matrices.

V. O. Mowery has presented some alternate equations, mathematically identical with Eqs. (2) and (4), which involve inversion of lesser order matrices. Replacing Eqs. (2) and (4), we have

\[ \mathbf{W} = S_{k-1} H^T (R_k + HS_{k-1} H^T)^{-1} \]

\[ S_k = (I - \mathbf{WH}) S_{k-1} \]

where

\[ I = \text{identity matrix} \]

The matrix to be inverted is now \( n \times n \) where \( n \), the dimension of the measurement vector \( \mathbf{r} \), is practically always less than 7. Version II of ESTMAT, using Mowery's equations, is shown in Fig. 6.

In performing error analyses, all matrices are evaluated along the nominal trajectory, with the covariance matrices being updated by Eqs. (7) and (4), or (5) and (9) depending on which version of ESTMAT was used.

VI. SUBROUTINE DENS

A. Description

Subroutine DENS computes, in double precision, the atmospheric density at any altitude. The subroutine may be entered through two points, DENS and ATM. The calling statements for each entry point are given below with explanation.

The statement CALL DENS must be used to enter the program initially, before any atmospheric density data are required. It sets up arrays of up to 100 discrete altitudes in km and kft, and arrays of their corresponding atmospheric densities in kg/meter\(^3\) and lb/ft\(^3\), respectively.*

Input $x_k$ = measurement vector
$s_{x,k-1}$ = predicted state vector
$\delta s_{x,k-1} = transition$ matrix $\delta s_{x,k-1}/\delta x_{k-1}$

1st data point

No

Initialize arrays and matrices

Yes

Update indices

PRINT > 0?

Print out transition matrix $F_{k,k-1}$

$F_{k,k-1} = \frac{1}{\delta s_{x,k-1}}$

Call XCOVTR to generate matrix $H = 2\delta x_k F_{k,k-1}$

$H = \frac{1}{\delta x_k}$

Calculate matrix $C_{k,k-1}^T$ $C_{k,k-1}$

Calculate weighting matrix $\delta s_k H^T H^{-1}$

$\delta s_k H^T H^{-1}$

MDATA > 0?

$\delta s_k = \frac{1}{\delta s_k}$

Pivot diagnostic:

$\delta x_k = \frac{1}{\delta s_k}$

RETURN

Fig. 5. Flow chart of subroutine ESTMAT.
Calculate estimate $\hat{x} = \hat{x}_{k-1} + WRx$

Call XCOVTR to convert $S_x$ to $S_x$ in radar polar coordinates

Loop $J = 1, 7$

Loop $K = 1, 7$

Test that diagonal terms are positive.

Print diagnostic

Calculate correlation coefficients of hybrid matrix $(h_{k,j})_{j,j} = \left(\frac{\hat{x}_{k,j}}{S_x^{1/2}}\right)$

Test that magnitudes of correlation coefficients do not exceed 1.

Print estimate $\hat{x}_k$

Store standard deviations of estimate $\text{DEVX}(j) = \left(\frac{h_{k,j}}{S_x^{1/2}}\right)$

Print estimate $\hat{x}_k$

Print correlation matrix $S_x$

Print covariance matrix $h_k$

Print diagnostic

$K = K + 1$

$K = 7$

$J = J + 1$

$J = 7$

Print out $Re$

$J = 7$

$J = 7$

Print $Re$

Print $Re$

Calculates diagonal elements of hybrid matrix $(h_{k,j})_{j,j} = \left(\frac{\hat{x}_{k,j}}{S_x^{1/2}}\right)$

Store determinant of $S_x$

Return

Fig. 5. Continued.
Fig. 6. Flow chart of subroutine ESTMAT (Mowery method).
Fig. 6. Continued.
(As presently written, only the first 61 elements of each array are filled in, up to an altitude of about 200 km. At higher altitudes the atmospheric density is set equal to zero.) Control is then returned to the main program.

The statement CALL ATM(HKMFT, RHO) is used when atmospheric density data are required. The arguments are

- HKMFT = altitude in km or kft
- RHO = density in kg/m$^3$ or lb/ft$^3$

The units used will depend on the value of the parameter COORD, in COMMON storage. English units are used if COORD > 0, metric units if COORD < 0.

The following COMMON statement is also required in the calling program:

```
COMMON/FCOM/COORD, DLAT, PRNT
```

The other quantities in COMMON storage are not used by this subroutine.

B. Mathematical Discussion

The given altitude is compared with those stored by the subroutine, and the desired density is computed as follows:

If

$$h_j < H < h_{j+1}$$

then

$$A = \frac{(H - h_j)}{(h_{j+1} - h_j)}$$

and

$$\rho = \rho_j \left(\frac{\rho_{j+1}}{\rho_j}\right)^A$$

where

- $H$ = input altitude, in kilometers or kilofeet
- $h_j$ = $j^{th}$ altitude in stored array
- $\rho_j$ = $j^{th}$ atmospheric density in stored array corresponding to $h_j$
- $\rho$ = atmospheric density at altitude $H$

If $H > h_{j_{\text{max}}}$ where $h_{j_{\text{max}}}$ = maximum altitude stored in array, then $\rho = 0$. If $H < 0$, the program prints out EARTH IMPACT.

VII. SUBROUTINE XYTOR

Subroutine XYTOR converts position and velocity given in radar-centered rectangular coordinates into radar-centered polar coordinates. The following calling statement is required:

```
CALL XYTOR(X, Y, Z, XDOT, YDOT, ZDOT, R, A, E, RDOT, ADOT, EDOT)
```
The input arguments are:

- X = position component in easterly direction
- Y = position component in northerly direction
- Z = position component in vertical direction
- \( \dot{X} = \frac{dX}{dt} \)
- \( \dot{Y} = \frac{dY}{dt} \)
- \( \dot{Z} = \frac{dZ}{dt} \)

The output arguments are:

- R = range
- A = azimuth (in radians)
- E = elevation (in radians)
- \( \dot{R} = \frac{dR}{dt} \)
- \( \dot{A} = \frac{dA}{dt} \)
- \( \dot{E} = \frac{dE}{dt} \)

The following equations are used in the calculations:

- \( R = \left( X^2 + Y^2 + Z^2 \right)^{1/2} \)
- \( A = \tan^{-1}(X/Y) \)
- \( E = \sin^{-1}(Z/R) \)
- \( \dot{R} = (XX + YY + ZZ)/R \)
- \( \dot{A} = \frac{YX + XY}{X^2 + Y^2} \)
- \( \dot{E} = \frac{RZ - ZR}{[R(R^2 - Z^2)^{1/2}]} \)

VIII. SUBROUTINE RTOXY

Subroutine RTOXY converts position and velocity given in radar-centered polar coordinates to radar-centered rectangular coordinates. The following calling statement is required:

```
CALL RTOXY(R, A, E, RDOT, ADOT, EDOT, X, Y, Z, XDOT, YDOT, ZDOT)
```
The output arguments are:

- **X** = position component in easterly direction
- **Y** = position component in northerly direction
- **Z** = position component in vertical direction

- **XDOT** = \( \dot{X} = \frac{dX}{dt} \)
- **YDOT** = \( \dot{Y} = \frac{dY}{dt} \)
- **ZDOT** = \( \dot{Z} = \frac{dZ}{dt} \)

The following equations are used:

- \( X = R \cos E \sin A \)
- \( Y = R \cos E \cos A \)
- \( Z = R \sin E \)
- \( XDOT = \frac{XR}{R} - ZE \sin A + YA \)
- \( YDOT = \frac{YR}{R} - ZE \cos A - XA \)
- \( ZDOT = \frac{ZR}{R} + RE \cos E \)

IX. SUBROUTINES XCOVTR AND RCOVTRX

A. Description of XCOVTR

Subroutine XCOVTR converts a 7 x 7 mean-square error covariance matrix in radar-centered rectangular coordinates to one in polar coordinates. It also calculates a partial derivative matrix \( H = \frac{\partial \mathbf{r}}{\partial \mathbf{x}} \) where \( \mathbf{r} \) and \( \mathbf{x} \) are the state vectors in radar-centered polar and rectangular coordinates, respectively.

The calling statement is

\[
\text{CALL XCOVTR(XCOV, XVCTR, RCOV, NCODE)}
\]

The input arguments are:

- **XCOV** = 7 x 7 covariance matrix in rectangular coordinates
- **XVCTR** = seven-component state vector used in evaluating matrices
- **NCODE** = operation selector code
  - **NCODE > 0**: subroutine calculates covariance matrix in polar coordinates
  - **NCODE < 0**: subroutine calculates partial derivative matrix \( H = \frac{\partial \mathbf{r}}{\partial \mathbf{x}} \)

The output argument is:

- **RCOV** = 7 x 7 mean-square error covariance in radar-centered polar coordinates
  - or
  - partial derivative matrix \( H = \frac{\partial \mathbf{r}}{\partial \mathbf{x}} \)
The following statement is also required in the calling program:

```plaintext
DIMENSION XCOV(7, 7), RCOV(7, 7), XVCTR(7)
```

Subroutines XYTOR and DMTMUL are required by this subroutine.

**B. Description of RCOVTX**

Subroutine RCOVTX converts a 7 x 7 mean-square error covariance matrix in radar-centered polar coordinates to one in rectangular coordinates. The calling statement is:

```plaintext
CALL RCOVTX(COVR, XSTATE, COVX)
```

The input arguments are:

- **COVR** = 7 x 7 mean-square error covariance matrix in polar coordinates
- **XSTATE** = seven-component state vector used in evaluating matrices

The output argument is:

- **COVX** = 7 x 7 mean-square error covariance matrix in radar-centered rectangular coordinates

The following statement is also required in the calling program:

```plaintext
DIMENSION COVR(7, 7), COVX(7, 7), XSTATE(7)
```

Subroutine XYTOR is required by this subroutine.

**C. Mathematical Discussion**

Consider the state vectors $\mathbf{x}$ and $\mathbf{r}$ in radar-centered rectangular and polar coordinates, respectively.

- $x_1$ = x easterly component
- $x_2$ = y northerly component
- $x_3$ = z vertical component
- $x_4$ = $\dot{x}$
- $x_5$ = $\dot{y}$
- $x_6$ = $\dot{z}$
- $x_7 = \alpha = 1/\beta = \text{reciprocal of ballistic coefficient}$

- $r_1 = R = \text{range}$
- $r_2 = A = \text{azimuth}$
- $r_3 = E = \text{elevation}$
- $r_4 = \dot{R}$
- $r_5 = \dot{A}$
- $r_6 = \dot{E}$

The covariance matrices are

$$
\Sigma_{\mathbf{x}} = E \left[ (\mathbf{x} - \mathbf{x}^*) (\mathbf{x} - \mathbf{x}^*)^T \right] \text{ in rectangular coordinates}
$$

$$
\Sigma_{\mathbf{r}} = E \left[ (\mathbf{r} - \mathbf{r}^*) (\mathbf{r} - \mathbf{r}^*)^T \right] \text{ in polar coordinates}
$$

where

- $E$ indicates average or expected value
- $\mathbf{x}^*$ = nominal value of $\mathbf{x}$
- $\mathbf{r}^*$ = nominal value of $\mathbf{r}$
The covariance matrices $\Sigma_x$ and $\Sigma_r$ are related by the equations

$$
\Sigma_r = H \Sigma_x H^T
$$

$$
\Sigma_x = B \Sigma_r B^T
$$

$$
H = \delta r / \delta x
$$

$$
H_{ij} = \delta r_i / \delta x_j
$$

$$
B = \delta x / \delta r = H^{-1}
$$

$$
B_{ij} = \delta x_i / \delta r_j
$$

where the subscripts indicate the component (or element) of the vector (or matrix) involved.

The first six components of the $x$ and $r$ vectors are related by the equations given with subroutines XYTOR and RTOXY, while $x_7 = r_7$. The various elements of the $H$ and $B$ matrices are given below:

$$
H_{11} = \delta R / \delta x = x/R
$$

$$
H_{12} = \delta R / \delta y = y/R
$$

$$
H_{13} = \delta R / \delta z = z/R
$$

$$
H_{14} = \text{through } H_{17} = 0
$$

$$
H_{21} = \delta A / \delta x = \frac{y}{x^2 + y^2}
$$

$$
H_{22} = \delta A / \delta y = -\frac{x}{x^2 + y^2}
$$

$$
H_{23} = \delta A / \delta z = 0
$$

$$
H_{24} = \text{through } H_{27} = 0
$$

$$
H_{31} = \delta E / \delta x = -\frac{x z}{R^2 \sqrt{x^2 + y^2}}
$$

$$
H_{32} = \delta E / \delta y = -\frac{y z}{R^2 \sqrt{x^2 + y^2}}
$$

$$
H_{33} = \delta E / \delta z = \frac{\sqrt{x^2 + y^2}}{R^2}
$$

$$
H_{34} = \text{through } H_{37} = 0
$$

$$
H_{41} = \delta R / \delta x = \frac{R x - R x}{R^2}
$$
\[ H_{61} = \frac{\partial E}{\partial x} = -\frac{x E}{(x^2 + y^2)} - \frac{2R}{R} H_{31} - \frac{z x}{R^2 \sqrt{x^2 + y^2}} \]
\[ H_{62} = \frac{\partial E}{\partial y} = -\frac{y E}{(x^2 + y^2)} - \frac{2R}{R} H_{32} - \frac{z y}{R^2 \sqrt{x^2 + y^2}} \]
\[ H_{63} = \frac{\partial E}{\partial z} = -\frac{z E}{R^2} - \frac{R}{R} H_{33} \]
\[ H_{64} = H_{31} \]
\[ H_{65} = H_{32} \]
\[ H_{66} = H_{33} \]
\[ H_{67} = 0 \]
\[ H_{71} \text{ through } H_{76} = 0 \]
\[ H_{77} = \partial \alpha / \partial \alpha = 1 \]

\[ B_{11} = \partial x / \partial R = \cos E \sin A \]
\[ B_{12} = \partial x / \partial A = R \cos E \cos A = y \]
\[ B_{13} = \partial x / \partial E = -R \sin E \sin A = -z \sin A \]
\[ B_{14} \text{ through } B_{17} = 0 \]
\[ B_{21} = \partial y / \partial R = \cos E \cos A \]
\[ B_{22} = \partial y / \partial A = -R \cos E \sin A = -x \]
\[ B_{23} = \partial y / \partial E = -R \sin E \cos A = z \cos A \]
\[ B_{24} \text{ through } B_{27} = 0 \]
\[ B_{31} = \partial z / \partial R = \sin E = z / R \]
\[ B_{32} = \partial z / \partial A = 0 \]
\[ B_{33} = \partial z / \partial E = R \cos E \]
\[ B_{34} \text{ through } B_{37} = 0 \]
\[ B_{41} = \partial \dot{x} / \partial R = \dot{A} \cos E \cos A - \dot{E} \sin E \sin A \]
\[ B_{42} = \partial \dot{x} / \partial A = \dot{R} \cos E \cos A - \dot{ER} \sin E \cos A - \dot{AR} \cos E \sin A \\
= \dot{R} \cos E \cos A - z \dot{E} \cos A - x \dot{A} \]
\[ B_{43} = -\dot{R} \sin E \sin A - \dot{\dot{E}} \cos E \sin A - \dot{\dot{R}} \sin E \cos A \]
\[ = -\dot{R} \sin E \sin A - \dot{x}_E - \dot{z}_A \cos A \]

\[ B_{44} = \frac{\partial x}{\partial \dot{R}} = \frac{\partial x}{\partial \dot{\dot{R}}} = B_{11} \]
\[ B_{45} = \frac{\partial x}{\partial \dot{A}} = \frac{\partial x}{\partial \dot{\dot{A}}} = B_{12} \]
\[ B_{46} = \frac{\partial x}{\partial \dot{E}} = \frac{\partial x}{\partial \dot{\dot{E}}} = B_{13} \]
\[ B_{47} = 0 \]

\[ B_{51} = \frac{\partial y}{\partial \dot{R}} = -\dot{E} \sin E \cos A - \dot{A} \cos E \sin A \]
\[ B_{52} = \frac{\partial y}{\partial \dot{A}} = -\dot{R} \cos E \sin A + \dot{\dot{E}} \sin A - \dot{\dot{R}} \cos E \cos A \]
\[ = -\dot{R} \cos E \sin A + z_E \sin A - y_A \]
\[ B_{53} = \frac{\partial y}{\partial \dot{E}} = -\dot{R} \sin E \cos A - \dot{\dot{E}} \cos E \cos A + \dot{\dot{R}} \sin E \sin A \]
\[ = -\dot{R} \sin E \cos A - y_E + z_A \sin A \]

\[ B_{54} = \frac{\partial y}{\partial \dot{R}} = \frac{\partial y}{\partial \dot{\dot{R}}} = B_{21} \]
\[ B_{55} = \frac{\partial y}{\partial \dot{A}} = \frac{\partial y}{\partial \dot{\dot{A}}} = B_{22} \]
\[ B_{56} = \frac{\partial y}{\partial \dot{E}} = \frac{\partial y}{\partial \dot{\dot{E}}} = B_{23} \]
\[ B_{57} = 0 \]

\[ B_{61} = \frac{\partial z}{\partial \dot{R}} = \dot{E} \cos E \]
\[ B_{62} = \frac{\partial z}{\partial \dot{A}} = 0 \]
\[ B_{63} = \frac{\partial z}{\partial \dot{E}} = \dot{R} \cos E - \dot{\dot{E}} \sin E \]
\[ = \dot{R} \cos E - z_E \]
\[ B_{64} = \frac{\partial z}{\partial \dot{R}} = \frac{\partial z}{\partial \dot{\dot{R}}} = B_{31} \]
\[ B_{65} = \frac{\partial z}{\partial \dot{A}} = \frac{\partial z}{\partial \dot{\dot{A}}} = B_{32} \]
\[ B_{66} = \frac{\partial z}{\partial \dot{E}} = \frac{\partial z}{\partial \dot{\dot{E}}} = B_{33} \]
\[ B_{67} = 0 \]

\[ B_{71} \text{ through } B_{76} = 0 \]
\[ B_{77} = \frac{\partial \alpha}{\partial \alpha} = 1 \]

X. SUBROUTINE GAUSSN

A. Description

Subroutine GAUSSN generates random numbers with a Gaussian distribution of zero mean and standard deviation SIGMA. The program is called by
CALL GAUSSN(NS, RN, SIGMA)

where

NS = number of samples desired
RN = array name of output
SIGMA = desired standard deviation

B. Mathematical Discussion

The subroutine makes use of the library subroutines RAN and ALOG. The procedure used to obtain a Gaussian number, RN, is as follows:

Let \( y = -\ln V_1 \)
\[ z = -\ln V_2 \]

where \( V_1 \) and \( V_2 \) are random numbers obtained from the RAN subroutine.

The quantity \((y - 1)^2\) is compared with \(2z\). If \((y - 1)^2 > 2z\), two new random numbers are selected and the above comparison repeated. If \((y - 1)^2 ≤ 2z\), a third random number \(S\) is selected to determine the sign of the desired output number. If

\[ S < 0.5, \quad RN = (y) \times (SIGMA) \]
\[ S > 0.5, \quad RN = -(y) \times (SIGMA) \]
\[ S = 0.5, \quad \text{another number } S \text{ is chosen and the above test for sign is repeated.} \]

XI. SUBROUTINE DMTMUL

Subroutine DMTMUL is a double-precision matrix multiplication subroutine. It is called by

CALL DMTMUL(A, B, AB, NRA, NCA, NCB).

The input arguments are:

\[
\begin{align*}
A & \quad \text{input matrices to be multiplied} \\
B & \\
NRA & = \text{number of rows in matrix } A \\
NCA & = \text{number of columns in matrix } A \\
NCB & = \text{number of columns in matrix } B
\end{align*}
\]

The output argument is

\[
AB = \text{matrix product } (A)(B)
\]

The matrices A, B, and AB must have the same dimensions in both the calling program and the subroutine. When used with the TRAP program described here, A, B, and AB are dimensioned \(7 \times 7\). Smaller matrices may also be multiplied by DMTMUL, even though they may not fill the complete array, by proper specification of NRA, NCA, and NCB.

XII. SUBROUTINE MINV

Subroutine MINV is a matrix inversion subroutine from the IBM System/360 Scientific Subroutine Package. It uses the standard Gauss-Jordan method and is available in both single and
double precision. (Our applications use the double-precision version.) The subroutine is called by the statement:

\[
\text{CALL MINV}(A, N, D, L, M)
\]

where

- \( A \) = input matrix, destroyed in computation and replaced by resultant inverse
- \( N \) = order of matrix
- \( D \) = resultant determinant
- \( L, M \) = work vectors of length \( N \).

The one-dimensional array variables \( A, L \) and \( M \) must be suitably dimensioned in the calling program.

The matrix to be inverted may be stored in a one-dimensional array of size \( N^2 \) before calling MINV. A series of FORTRAN statements to do this are given below:

\[
\text{DO 1 } J = 1, N \\
\text{DO 1 } K = 1, N \\
L = N \times (J - 1) + K \\
1 A(L) = \text{ARRAY}(J, K)
\]

where \( \text{ARRAY} \) is the two-dimensional array of the matrix to be inverted.

Subroutine MINV may now be called to invert the matrix stored in the one-dimensional array \( A \). After matrix inversion, the resulting inverse, stored in the one-dimensional array \( A \), may be placed in a two-dimensional array by the following statements:

\[
\text{DO 2 } L = 1, LMAX \\
J = (L - 1)/N + 1 \\
K = L - N \times (J - 1) \\
2 \text{ARRAY}(J, K) = A(L)
\]

where \( LMAX = N^2 \) and \( \text{ARRAY} \) is the \( N \times N \) array of the inverted matrix.
ACKNOWLEDGMENT

The author wishes to thank Mr. M. Gruber, Dr. J. A. Tabaczynski, and Dr. R. P. Wishner for helpful discussions and suggestions.

REFERENCES


APPENDIX A
SOME RELATIONS USED IN CALCULATION OF DERIVATIVES
FOR A MATRIX $\frac{\partial f(x)}{\partial x}$

$$P = -\frac{\rho V_D}{2}$$

$$\rho = \rho_0 e^{-\gamma h} = \text{atmospheric density}$$

$$h = [x^2 + y^2 + (z + R_E)^2]^{1/2} - R_E = \text{altitude}$$

$$R_E = \text{radius of earth}$$

$$V_D = (\dot{x}^2 + \dot{y}^2 + \dot{z}^2)^{1/2} = \text{drag velocity}$$

$$R = [x^2 + y^2 + (z + R_E)^2]^{1/2} = \text{distance from center of earth to object}$$

$$\frac{\partial P}{\partial x} = -P \frac{\dot{y}}{R}$$

$$\frac{\partial P}{\partial y} = -P \frac{\dot{x}}{R}$$

$$\frac{\partial P}{\partial z} = -P \frac{\dot{y}(z + R_E)}{R}$$

$$\frac{\partial P}{\partial \dot{x}} = P \dot{x} \frac{V_D^2}{2}$$

$$\frac{\partial P}{\partial \dot{y}} = P \dot{y} \frac{V_D^2}{2}$$

$$\frac{\partial P}{\partial \dot{z}} = P \dot{z} \frac{V_D^2}{2}$$

$$B = \omega^2 - \frac{G_M}{R^3}$$

$$G = \omega^2 \sin^2(DLAT) - \frac{G_M}{R^3}$$

$$N = \omega^2 \cos^2(DLAT) - \frac{G_M}{R^3}$$

$$\frac{\partial B}{\partial \dot{x}} = \frac{\partial N}{\partial \dot{x}} = 3xG_M/R^5$$

$$\frac{\partial B}{\partial \dot{y}} = \frac{\partial N}{\partial \dot{y}} = 3yG_M/R^5$$

$$\frac{\partial B}{\partial \dot{z}} = \frac{\partial N}{\partial \dot{z}} = 3(z + R_E)G_M/R^5$$

37
TABLE B-I
MAIN PROGRAM (VERSION I)

```
C MAIN PROGRAM
IMPLICIT REAL*8(A-H,O-Z)
COMMON /ACCM/CCVAR(7),SIGMA(7)/FCOM/COORD,DLAT,PRNT/ICOM/KLAMP,MDATA
C
DIMENSION GNCISE(7),RGN(54),AZN(54),ELN(54),RDPN(54),BETAH(54),
  RHAT(54),AHAT(54),EMAT(54),PHAT(54),EPHAT(54),STAT
2EM(7),STATEP(7),PHIMAT(7,7),FNOISE(7),ALPHA(54),DALPHA(54)
C DIMENSION PHINV(7,7),PROD(7,7),STATE(7),STATE(7),X(54),Y(54),Z(54),
  XP(54),YP(54),ZP(54),DX(54),DY(54),DZ(54),DYP(54),DP(54),DZP(54),
  XHAT(54),YHAT(54),ZHAT(54),XPHAT(54),YPHAT(54),ZPHAT(54),STATEP(7),
  ASPECT(7,7),SIGR(54),SIGA(54),SIGE(54),SIGR(54),SIGP(54),SIGY(54),
  SIGA(54),SIGE(54),SIGA(54),SIGP(54),SIGY(54),SIGA(54),SIGP(54),
  DIMENSION GN(7),LABEL(18),RA(54),AZ(54),EL(54),AZP(54),ELP(54),
  HEIGHT(54),TIME(54),RAP(54),BATA(54),ERRMAX(7),STDEVR(7),STDEV
2X(7)
C CALL DENS TO SET UP ATMOSPHERIC DENSITY TABLES
C CALL DENS
C READ INPUT DATA CARDS
C 1 READ(5,2,END=9000)(LABEL(I),I=1,18)
  2 FORMAT (1BA4)
  3 READ(5,3)(STATE(I),I=1,7)
  4 FORMAT (5F10.3,F5.3,3I5)
  READ(5,3)(SIGMA(I),I=1,6),COORD
  READ(5,3)(CCVAR(J),J=1,7)
  READ(5,3)(FNCISE(J),J=1,7)
  READ(5,3)(ERRMAX(JJ),JJ=1,7)
C C LABEL= IDENTIFYING COMMENTS TO APPEAR WITH DATA PRINT-OUT
C TZERC= INITIAL TIME
C TINT= INTERVAL BETWEEN DESIRED DATA POINTS (SEC.)
C TINC= INTEGRATION STEP SIZE (SEC.)
C DLAT= LATITUDE OF RADAR SITE (IN DEGREES)
C NDATA= NUMBER OF DESIRED DATA POINTS
C PRNT= PRINT-OUT SELECTOR
C PRNT= 1. MINIMUM PRINT-OUT, TABULATED SUMMARY ONLY
C PRNT= 0. PRINT-OUT OF COVARIANCE MATRICES + TABULATED SUMMARY
C PRNT= 1. FULL PRINT-OUT
C KLAMP= MEMORY OF ALGORITHM, EXPRESSED AS NUMBER OF DATA POINTS
C MDATA= MC13 SELECTOR
C MDATA GREATER THAN 0. TRAJECTORY PARAMETER ESTIMATION
C MDATA = 0. NDISELESS TRAJECTORY GENERATION ONLY
C MDATA LESS THAN = 0. ERROR ANALYSIS
C CCCORD = CCORDINATE AND UNIT SELECTOR
C CLORD=2. PCLAR COORDINATES, METRIC UNITS
C CCLORD=-1. RECTANGULAR COORDINATES, METRIC UNITS
C CCCORD=1. RECTANGULAR COORDINATES, ENGLISH UNITS
C CCCORD=2. PCLAR COORDINATES, ENGLISH UNITS
C SIGMA = RMS NOISE COMPONENTS OF MEASUREMENT VECTOR
C CCVAR = RMS NOISE COMPONENTS OF INITIAL STATE VECTOR
```
TABLE B-I (Continued)

C  FNCISE = NCISE SAMPLES ASSOCIATED WITH INITIAL STATE VECTOR
C  ERRMAX = TRAJECTORY INTEGRATION CONVERGENCE ERRORS (IN RC-
C  TANGULAR COORDINATES)
C
9 IMAX=50
CCCRDM=DABS(CCCRC)
RTCEG=180. / 3.14159265
C
C  SETUP INITIAL STATE IN BOTH RADAR POLAR AND XYZ COORDINATES
C
IF(CCCRDM-1.)400,400,300
300 RA(1)=STATE(1)
AZ(1)=STATE(2)
EL(1)=STATE(3)
RAP(1)=STATE(4)
AZP(1)=STATE(5)
ELP(1)=STATE(6)
CALL RTOXY(RA(1),AZ(1),EL(1),RAP(1),AZP(1),ELP(1),X(1),Y(1),Z(1),X
YP(1),ZP(1))
GC TO 401
400 X(1)=STATE(1)
Y(1)=STATE(2)
Z(1)=STATE(3)
XP(1)=STATE(4)
YP(1)=STATE(5)
ZP(1)=STATE(6)
CALL XYTCR(X(1),Y(1),Z(1),XP(1),YP(1),ZP(1),RA(1),AZ(1),EL(1),RAP
11),AZP(1),ELP(1))
401 BETA=1./STATE(7)
BETA1=BETA
BETA1=BETA1
C
C  CALCULATE ASPECT ANGLE
C
ASP=(X(1)*XP(1)+Y(1)*YP(1)+Z(1)*ZP(1))/(RA(1)*DSCRT(
1XP(1)**2+YP(1)**2+ZP(1)**2))
ASP(1)=ASP
ASP(1)=ASP
C
C  PRINT HEADER PAGE OF OUTPUT LISTING
C
WRITE (6,10)(LABEL(1),I=1,18)
10 FCRMAT('*1',20X,'OUTPUT LISTING'/18A4//'/
WRITE(6,11)
11 FCRMAT('/5X,'INITIAL CONDITIONS//'/
12 FCRMAT('/5X,'METRIC UNITS(METERS,KILOGRAMS,RADIANS,SECONDS) USED TH
13 FCRMAT('/5X,'ENGLISH UNITS(Feet,Pounds,RADIANS,SECONDS) USED THROUG
14 RE=2.0925738E7/3.2808333
WRITE(6,15)
15 FCRMAT('/5X,'CUT PROGRAM'//)
GC TC 16
16 RE=2.0925738C7
WRITE(6,17)
17 FCRMAT('/5X,'ENGLISH UNITS(Feet,Pounds,RADIANS,SECONDS) USED THROUG
18 CC 19 J=1,7
19 STATE(J)=0.
X1=X(1)
Y1=Y(1)

42
### TABLE B-1 (Continued)

<table>
<thead>
<tr>
<th>Column A</th>
<th>Column B</th>
<th>Column C</th>
<th>Column D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z1=Z(1)</td>
<td>XGCTI=XP(1)</td>
<td>YGCTI=YP(1)</td>
<td>ZGCTI=ZP(1)</td>
</tr>
<tr>
<td>HEIGHT(1)=CSGRT(X1<strong>2+Y1</strong>2+(Z1+RE)**2)-RE</td>
<td>WRITE(6,50)TIME(INDEX),RA(INDEX),AZ(INDEX),EL(INDEX),RA(_INDEX),AZ (INDEX),EL(_INDEX),X1,Y1,Z1,XDOT1,YDOT1,ZDOT1,HEIGHT(INDEX),BETA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE(6,500)ASPECT(1)</td>
<td>WRITE(6,500)ASPECT(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500 FORMAT(5X,*ASPECT ANGLE='F10.2,' DEGREES/')</td>
<td>WRITE(6,110)ILAT,NCATA,TINT,TINCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>110 FORMAT(5X,*RACAR LATITUDE='F10.5,2X,*DEGREES'/5X,16,5X,'</td>
<td>DATA POINTS SPACED',3X, F10.6 ,2X,'SEC. APART, INTEGRATION STEP',2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2X,F10.6,2X,'SEC./')</td>
<td>IF(MDATA)70,74,72</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70 WRITE(6,71)</td>
<td>71 FORMAT(5X,'ERROR ANALYSIS/')</td>
<td></td>
<td></td>
</tr>
<tr>
<td>198 CC 201 J=1,7</td>
<td>201 FNCISE(J)=0.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GC TC 75</td>
<td>74 WRITE(6,80)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80 FORMAT(5X,'TRAJECTORY GENERATION/')</td>
<td>TAU=TINT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GC TC 81</td>
<td>72 WRITE(6,73)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73 FORMAT(5X,'TRAJECTORY PARAMETER ESTIMATION FROM SIMULATED NOISY D</td>
<td>75 IF(KLAMP)760,760,751</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1ATA/')</td>
<td>751 WRITE(6,752) KLAMP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>752 FORMAT(5X,'MEMCRY TIME =','16,' DATA POINTS/')</td>
<td>TSTEP=TINCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>760 WRITE(6,76) CELT</td>
<td>76 FORMAT(5X,'TIME OF INITIAL MEASUREMENT WITH RESPECT TO TIME OF IN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>761 WRITE(6,77)</td>
<td>IITIAL ESTIMATE =','F10.3,3X,*SEC./')</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77 FORMAT(5X,'RMS NOISE LEVELS ASSOCIATED WITH MEASUREMENT VECTOR/')</td>
<td>WRITE (6,77)(SIGMA(J),J=1,6)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 FORMAT(5X,'SIGMA(R)=','F10.2,5X,'SIGMA(AZ)=','F10.6,5X,'SIGMA(EL)= '</td>
<td>1,F10.6,5X,'SIGMA(RECT)=','F10.2/5X,'SIGMA(AZDOT)=','F10.6,5X,'SIGMA(</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2ELDOT)=','F10.6,6/)</td>
<td>13 FORMAT(5X,'ONLY THE 1ST',13,1X,'QUANTITIES OF THE VECTOR'/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15X,'CONSISTING OF R,AZ,EL,RCOT,AZDOT,ELDOT,1.,/BETA, ARE MEASURED')</td>
<td>IF(CCCORD=1,780,780,782</td>
<td></td>
<td></td>
</tr>
<tr>
<td>780 WRITE(6,781)</td>
<td>781 FORMAT(5X,'INITIAL STATE VECTOR READ IN RADAR-CENTERED RECTANGULA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>782 WRITE(6,783)</td>
<td>18 COORDINATES/')</td>
<td></td>
<td></td>
</tr>
<tr>
<td>783 FORMAT(5X,'INITIAL STATE VECTOR READ IN RADAR-CENTERED POLAR COOR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>784 WRITE(6,784)</td>
<td>DINATES/')</td>
<td></td>
<td></td>
</tr>
<tr>
<td>784 CC VAR(J),J=1,7</td>
<td>78 FORMAT(5X,'RMS NOISE LEVELS ASSOCIATED WITH INITIAL STATE VECTOR*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/7C17.0/')</td>
<td>WRITE(6,799)(FNCISE(J),J=1,7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>799 FORMAT(5X,'NCISE SAMPLES AT INITIAL STATE*/7D17.8///)</td>
<td>81 WRITE(6,12)((ERRMAX(JJ),JJ=1,7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 FORMAT(5X,'MAXIMUM INTEGRATION CONVERGENCE ERRORS(IN RECTANGULAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18COORDINATES*/7C17.8///)</td>
<td>L=2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TABLE B-I (Continued)

<table>
<thead>
<tr>
<th>C</th>
<th>DATA LCOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>CO 1000 NN=1,NDATA</td>
</tr>
<tr>
<td></td>
<td>NN=NN</td>
</tr>
<tr>
<td>C</td>
<td>IF(NN=1)21,20,21</td>
</tr>
<tr>
<td>20</td>
<td>IF(MDATA=202,22,202</td>
</tr>
<tr>
<td>C</td>
<td>ACG NOISE CF RMS LEVEL COVAR TO NOMINAL INITIAL STATE</td>
</tr>
<tr>
<td></td>
<td>(NCISE=C, FCR ERROR ANALYSIS CASE)</td>
</tr>
<tr>
<td>C</td>
<td>SET INITIAL STATE ESTIMATE = INITIAL NOISY STATE</td>
</tr>
<tr>
<td>C</td>
<td>202 IF(CDCRDM-1.)203,203,204</td>
</tr>
<tr>
<td>203</td>
<td>XHAT(1) =X(1)+FNCISE(1)</td>
</tr>
<tr>
<td></td>
<td>YHAT(1) =Y(1)+FNCISE(2)</td>
</tr>
<tr>
<td></td>
<td>ZHAT(1) =Z(1)+FNCISE(3)</td>
</tr>
<tr>
<td></td>
<td>PXHAT(1)=XP(1)+FNCISE(4)</td>
</tr>
<tr>
<td></td>
<td>PYHAT(1)=YP(1)+FNCISE(5)</td>
</tr>
<tr>
<td></td>
<td>PZHAT(1)=ZP(1)+FNCISE(6)</td>
</tr>
<tr>
<td></td>
<td>CALL XYTXR(XHAT(1),YHAT(1),ZHAT(1),XPHAT(1),YPHAT(1),ZPHAT(1),RHAT</td>
</tr>
<tr>
<td></td>
<td>1(1),AHAT(1),EHAT(1),RHAT(1),APHAT(1),EPHAT(1))</td>
</tr>
<tr>
<td>GC TO 206</td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>RHAT(1)=RA(1)+FNOISE(1)</td>
</tr>
<tr>
<td></td>
<td>AHAT(1)=A(1)+FNCISE(2)</td>
</tr>
<tr>
<td></td>
<td>EHAT(1)=EL(1)+FNCISE(3)</td>
</tr>
<tr>
<td></td>
<td>RPXHAT(1)=RPX(1)+FNCISE(4)</td>
</tr>
<tr>
<td></td>
<td>APXHAT(1)=APX(1)+FNCISE(5)</td>
</tr>
<tr>
<td></td>
<td>EPXHAT(1)=EJPX(1)+FNCISE(6)</td>
</tr>
<tr>
<td></td>
<td>CALL RTXYR(RHAT(1),AHAT(1),EHAT(1),RPXHAT(1),APXHAT(1),EPXHAT(1),XHAT</td>
</tr>
<tr>
<td></td>
<td>1(1),YHAT(1),ZHAT(1),XPHAT(1),YPHAT(1),ZPHAT(1))</td>
</tr>
<tr>
<td>206</td>
<td>ESTATE(7)=STATE(1)+FNCISE(7)</td>
</tr>
<tr>
<td></td>
<td>ESTATE(1)=XHAT(1)</td>
</tr>
<tr>
<td></td>
<td>ESTATE(2)=YHAT(1)</td>
</tr>
<tr>
<td></td>
<td>ESTATE(3)=ZHAT(1)</td>
</tr>
<tr>
<td></td>
<td>ESTATE(4)=XPXHAT(1)</td>
</tr>
<tr>
<td></td>
<td>ESTATE(5)=YPXHAT(1)</td>
</tr>
<tr>
<td></td>
<td>ESTATE(6)=ZPXHAT(1)</td>
</tr>
<tr>
<td></td>
<td>ALPHA(1)=ESTATE(7)</td>
</tr>
<tr>
<td>205</td>
<td>BETAH(1)=1./ESTATE(7)</td>
</tr>
<tr>
<td></td>
<td>TAL=DELT</td>
</tr>
<tr>
<td>C</td>
<td>COMPUTE INITIAL ERRORS = (ESTIMATE)-(NOMINAL STATE)</td>
</tr>
<tr>
<td>C</td>
<td>CR(1)=RHAT(1)-RA(1)</td>
</tr>
<tr>
<td></td>
<td>CA(1)=AHAT(1)-A(1)</td>
</tr>
<tr>
<td></td>
<td>CE(1)=EHAT(1)-EL(1)</td>
</tr>
<tr>
<td></td>
<td>CRP(1)=RPXHAT(1)-RPX(1)</td>
</tr>
<tr>
<td></td>
<td>CAP(1)=APXHAT(1)-APX(1)</td>
</tr>
<tr>
<td></td>
<td>CEP(1)=EPXHAT(1)-EJPX(1)</td>
</tr>
<tr>
<td></td>
<td>CB(1)=BETAH(1)-BETA(1)</td>
</tr>
<tr>
<td></td>
<td>CALPH(1)=ESTATE(7)-1./BETA</td>
</tr>
<tr>
<td>CX(1)=XHAT(1)-X(1)</td>
<td></td>
</tr>
<tr>
<td>CY(1)=YHAT(1)-Y(1)</td>
<td></td>
</tr>
<tr>
<td>CZ(1)=ZHAT(1)-Z(1)</td>
<td></td>
</tr>
<tr>
<td>CXP(1)=XPXHAT(1)-XPX(1)</td>
<td></td>
</tr>
<tr>
<td>CYP(1)=YPXHAT(1)-YJPX(1)</td>
<td></td>
</tr>
<tr>
<td>CZP(1)=ZPXHAT(1)-ZPX(1)</td>
<td></td>
</tr>
<tr>
<td>CC TC 22</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>SET UP ARGUMENTS FCR TRAJECT TO INTEGRATE NOMINAL TRAJECTORY TO</td>
</tr>
</tbody>
</table>
### TABLE B-I (Continued)

<table>
<thead>
<tr>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C NEXT POINT C</td>
<td>21 X1=X2 Y1=Y2 Z1=Z2 XDC1=XDC2 YDC1=YDC2 ZDC1=ZDC2 ETA1=ETA2 TAU=TINT C</td>
</tr>
<tr>
<td>C UPDATE SUBSCRIPTS C</td>
<td>22 K=MCD(NN,IMAX) NXT=K+1 INDEX=NXT IF(K123,23,24 K=IMAX C</td>
</tr>
<tr>
<td>C CALL TRAJGXR TO GENERATE NEXT DATA POINT OF NOMINAL TRAJECTORY C</td>
<td>24 TIME(NXT)=TLAST+TAU TLAST=TIME(INDEX) CALL TRAJGXR(X1,Y1,Z1,XOCT1,YOCT1,ZOCT1,ETA1,TAU,TINCR,ERRMAX, ( X2,Y2,Z2,XOCT2,YOCT2,ZOCT2,ETA2,PHI1IMAX )) X(NXT)=X2 Y(NXT)=Y2 Z(NXT)=Z2 XP(NXT)=XOCT2 YP(NXT)=YOCT2 ZP(NXT)=ZOCT2 ETA(NXT)=ETA2 HEIGHT(NXT)=CLSRT(X2<strong>2+Y2</strong>2+Z2**2)+2-RF C</td>
</tr>
<tr>
<td>C CONVERT TO RADAR POLAR COORDINATES C</td>
<td>CALL XICK(X2,Y2,Z2,XOCT2,YOCT2,ZOCT2,RA(NXT),AZ(NXT),EL(NXT),RAP(INXT),AZP(INXT),ELP(INXT)) C</td>
</tr>
<tr>
<td>C CALCULATE ASPECT ANGLE C</td>
<td>ASP=(X(NXT)*XP(NXT)+Y(NXT)*YP(NXT)+Z(NXT)*ZP(NXT))/(RA(INXT)<em>CLSRT(XP(NXT)**2+YP(NXT)**2+ZP(NXT)**2)) ASPECT(INXT)=180.-KTEEK</em>DARCOS(ASP) C</td>
</tr>
<tr>
<td>C WRITE NEXT NOMINAL DATA POINT C</td>
<td>IF(IPRINT1)502,48,48 WRITE(10,49) 49 FCMAT('1','5X','NOMINAL DATA POINT')// WRITE(10,50)TIME(INDEX),RA(INDEX),AZ(INDEX),EL(INDEX),RAP(INDEX),AZL(INDEX),ELP(INDEX),X2,Y2,Z2,XOCT2,YOCT2,ZOCT2,HEIGHT(INDEX),ETA2 50 FCMAT('5X','TIME='1,F10.4,'5X','TRA='1,F15.2,3X,'AZ='9,F10.6,3X,'EL='1,F10.6,'3X','AZP='1,F10.6,'3X','ELP='1,F10.6,'3X','X='1,F10.6,'3X','Y='1,F10.6,'3X','Z='1,F10.6,3X,'XP='1,F10.6,3X,'YP='1,F10.6,3X,'ZP='1,F10.6,3X,'HEIGHT='1,F10.6,3X,'BETA='1,F10.6,3X,'F10 2.2,'3X,'F10.2,'5X,'HEIGHT='1,F10.2,3X,'BETA='1,F10.2,5X,'F10.2/' WRITE(10,500)ASPECT(INXT) C</td>
</tr>
<tr>
<td>C WRITE TRANSITION MATRIX FOR NOMINAL DATA POINT C</td>
<td>45</td>
</tr>
</tbody>
</table>
```
501 WRITE(6,51)((PHIMAT(JJ,KK),KK=1,7),JJ=1,7)
51 FORMAT(5x,'TRANSITION MATRIX PHIMAT(I,J)')
502 IF(MDATA)5800,25,2400
C
ERRCR ANALYSIS
C
5800 STATEP(1)=X2
STATEP(2)=Y2
STATEP(3)=Z2
STATEP(4)=XCT2
STATEP(5)=YCT2
STATEP(6)=ZCT2
STATEP(7)=1./BETA2
CALL ESTMAT(STATEP,PHIMAT,ESTATE,STDEVX,STDEVX)
C
STORE EXPECTED RMS ERRORS
C
SIGR (NX1)=STDEVX(1)
SIGA (NX1)=STDEVX(2)
SIGE (NX1)=STDEVX(3)
SIGG (NX1)=STDEVX(4)
SIGAP(X1)=STDEVX(5)
SIGEP(X1)=STDEVX(6)
SIGX (NX1)=STDEVX(7)
SIGY (NX1)=STDEVX(8)
SIGZ (NX1)=STDEVX(9)
SIGXP(X1)=STDEVX(10)
SIGYP(X1)=STDEVX(11)
SIGZP(X1)=STDEVX(12)
SIGNALP(NX1)=STDEVX(13)

CC TO 25
C
TRACKING AND ESTIMAT..C
C
CALL GAUSSN(X1,CMISE(J),SIGNA(J))
RGN(NX1)=RGN(NX1)+CMISE(1)
AZN(NX1)=AZN(NX1)+CMISE(2)
ELN(NX1)=ELN(NX1)+CMISE(3)
RAPM(NX1)=RAPM(NX1)+CMISE(4)
STATEP(1)=RGN(NX1)
STATEP(2)=AZN(NX1)
STATEP(3)=ELN(NX1)
STATEP(4)=RAPM(NX1)
IF(FRAT)59,57,57
C
PRINT OUT SIMULATED NOISY MEASUREMENT
C
57 WRITE(6,58) TIME(NX1),RGN(NX1),AZN(NX1),ELN(NX1),RAPM(NX1)
58 FORMAT('TIME=',F10.5,'RGN=',F10.3,'AZN=',F10.3,'ELN=',F10.3,'RAPM=',F10.2)
C
CALL TRAJEX TO OBTAIN NEXT PREDICTED DATA POINT
C
59 CALL TRAJEX(ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTATE(6),
ETAP(K),TAU,TINC,ERKMAX,STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PEETA,PROD)
C
```
TABLE B-I (Continued)

CCNVERT NEXT PREDICTED DATA POINT TO RADAR POLAR COORDINATES

STATEP(7) = 1./PBETA
IF (PRINT) 5702, 5700, 5700

5700 CALL XYTCR(STATEP(1), STATEP(2), STATEP(3), STATEP(4), STATEP(5), STATEP(6), PRA, PAZ, PEL, PRAP, PAZP, PELP)

PH = DSQRRT(STATEP(1)**2 + STATEP(2)**2 + (STATEP(3) + RE)**2) - RE
WRITE (6, 570)

570 FORMAT (/5X, **PREDICTED STATE BASED ON PAST DATA ONLY**)
WRITE (6, 50) TIME(NXT), PRA, PAZ, PEL, PRAP, PAZP, PELP, STATEP(1), STATEP(2), STATEP(3), STATEP(4), STATEP(5), STATEP(6), PH, PBETA

CALL ESTMAT TO COMBINE SIMULATED MEASUREMENT WITH PREDICTED POINT TO OBTAIN FINAL ESTIMATE OF NEXT DATA POINT

5702 CALL ESTMAT(STATEM, STATEP, PROD, ESTATE, STDEVX, STDEVR)

STCRF EXPECTED RMS ERRORS

SIGR(NXT) = STCRF(1)
SIGA(NXT) = STCRF(2)
SIGE(NXT) = STCRF(3)
SIGRP(NXT) = STCRF(4)
SIGAP(NXT) = STCRF(5)
SIGEP(NXT) = STCRF(6)
SIGX(NXT) = STCRFVX(1)
SIGY(NXT) = STCRFVX(2)
SIGZ(NXT) = STCRFVX(3)
SIGXP(NXT) = STCRFVX(4)
SIGYP(NXT) = STCRFVX(5)
SIGZP(NXT) = STCRFVX(6)
SIGCALP(NXT) = STCRFVX(7)

579 CONTINUE

CCNVERT ESTIMATED STATE TO RADAR POLAR COORDINATES

STORE ESTIMATED STATE VECTOR IN OUTPUT ARRAYS

5793 CALL XYTCR(ESTATE(1), ESTATE(2), ESTATE(3), ESTATE(4), ESTATE(5), ESTATE(6), RHAHAT(NXT), AHAHAT(NXT), EHAHAT(NXT), RPHAHAT(NXT), APHAHAT(NXT), EPHAHAT(NXT))

XHAT(NXT) = ESTATE(1)
YHAT(NXT) = ESTATE(2)
ZHAT(NXT) = ESTATE(3)
XPHAT(NXT) = ESTATE(4)
YPHAT(NXT) = ESTATE(5)
ZPHAT(NXT) = ESTATE(6)

ALPHAHAT(NXT) = ESTATE(7)
EETHAHAT(NXT) = 1./ESTATE(17)

XHAT = DSQRRT(ESTATE(1)**2 + ESTATE(2)**2 + (ESTATE(3) + RE)**2) - RE
IF (PRINT) 581, 5794, 5794

C PRINT CUT ESTIMATED STATE

5794 WRITE(6, 500)

580 FORMAT (/5X, **ESTIMATED POINT**)
WRITE (6, 50) TIME(NXT), RHAHAT(NXT), AHAHAT(NXT), EHAHAT(NXT), RPHAHAT(NXT), APHAHAT(NXT), EPHAHAT(NXT), ESTATE(1), ESTATE(2), ESTATE(3), ESTATE(4), ESTATE(5), ESTATE(6), NXT, EETHAHAT(NXT)
TABLE B-1 (Continued)

**COMPUTE DIFFERENCES BETWEEN ESTIMATE AND NOMINAL DATA POINT**

- **CR(NXT)** = RHAT(NXT) - R'NXT
- **CA(NXT)** = AHAT(NXT) - AZ(NXT)
- **CE(NXT)** = EHAT(NXT) - EL(NXT)
- **CRP(NXT)** = RPHAT(NXT) - RAP(NXT)
- **CAP(NXT)** = APATH(NXT) - AZP(NXT)
- **CEP(NXT)** = EPHAT(NXT) - ELP(NXT)
- **CB(NXT)** = BETAH(NXT) - BATA(NXT)
- **CALPH(NXT)** = ESTATE(7) - 1 / BATA(NXT)
- **CX(NXT)** = XHAT(NXT) - X(NXT)
- **CY(NXT)** = YHAT(NXT) - Y(NXT)
- **CZ(NXT)** = ZHAT(NXT) - Z(NXT)
- **CXP(NXT)** = XPHAT(NXT) - XP(NXT)
- **CYP(NXT)** = YPHAT(NXT) -YP(NXT)
- **CZP(NXT)** = ZPHAT(NXT) - ZP(NXT)

**TEST FOR LAST DATA POINT OF RUN**

- IF(NN-NDATA)<27,26,26
- KMAX=INDEX
- GC TC 40
- IF(INDEX-IMAX)<10CC,28,26
- KMAX=IPAX
- WRITE OUTPUT

- WRITE(6,10)(LABEL(1),I=1,18)
- WRITE(6,41)
- FORMAT(/5X,'NOMINAL TRAJECTORY IN RADAR COORDINATES'//
  1,3X,'TIME',8X,'RANGE',7X,'RACCT',7X,'AZIM',8X,'AZDOT',7X
  2,'HEIGHT',6X,'BETA',//)
  WRITE(6,42)(TIME(KK),R,A(KK),RAP(KK),AZ(KK),AZP(KK),EL(KK),ELP(KK),
  1,HEIGHT(KK),BATA(KK),KK=1,KMAX)
  42 FORMAT(3X,F10.3,2X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X)
  WRITE(6,10)(LABEL(1),I=1,18)
  WRITE(6,41)
  410 FORMAT(5X,'NOMINAL TRAJECTORY IN XYZ COORDINATES'//)
  WRITE(6,41)
  411 FORMAT(3X,'TIME',8X,'X',11X,'Y',11X,'Z',11X,'XDOT',8X,'YDOT',8X,'ZDCT',8X,'BETA',8X,'ASPECT ANGLE'//)
  WRITE(6,412)(TIME(J),X(J),Y(J),Z(J),X(J),Y(J),Z(J),BATA(J),ASPECT
  1,ICT(J),J=1,KMAX)
  412 FORMAT(3X,F10.3,2X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X)
  WRITE(6,10)(LABEL(1),I=1,18)
  WRITE(6,41)
  420 FORMAT(6,10)(LABEL(1),I=1,18)
  WRITE(6,43)
  43 FORMAT(5X,'MEASUREMENT DATA'//6X,'TIME',8X,'RANGE',7X,'RACCT',17X
  1,'AZIM',20X,'ELLEV'//)
  WRITE(6,44)(TIME(KK),RGN(KK),RAPN(KK),AZN(KK),ELN(KK),KK=1,KMAX)
  WRITE (6,10)(LABEL(1),I=1,18)
  WRITE(6,60)
  60 FORMAT(5X,'ESTIMATE VALUES'//)
  WRITE(6,61)
  61 FORMAT(3X,'TIME',8X,'RANGE',7X,'RACCT',7X,'AZIM',8X,'AZDOT',7X,'EL
  1,LEV',8X,'ELLEV',7X,'BETA',8X,'ALPHA'//)
  WRITE(6,62)(TIME(J),RHAT(J),RPHAT(J),AHAT(J),APHAT(J),EHA(J),EPAH

48
TABLE B-I (Continued)

1T(J),BETAH(J),ALPHA(J),J=1,KMAX
62 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F1
10.6,2X,F10.2,2X,F12.8)
   WRITE(6,10)(LABEL(I),I=1,18)
   WRITE(6,60)
   WRITE(6,413)
413 FORMAT(3X,'TIME',8X,'X',11X,'Y',11X,'Z',11X,'XDOT',8X,'YDOT',8X,'Z
   DCT',8X,'BETA',8X,'ALPHA'/)
   WRITE(6,414)(TIME(J),XHAT(J),YHAT(J),ZHAT(J),XPHAT(J),YPHAT(J),ZPH
   LAT(J),BETAH(J),ALPHA(J),J=1,KMAX)
10.2,2X,F10.2,2X,F12.8)
   WRITE (6,10)(LABEL(I),I=1,18)
   WRITE(6,63)
63 FORMAT(5X,'(ERRORS)=(ESTIMATED VALUES)-(NOMINAL VALUES)'/)
   WRITE(6,61)
   WRITE(6,62)(TIME(J),DR(J),DRP(J),DA(J),DE(J),DEP(J),DB(J),D
   ALPH(J),J=1,KMAX)
   WRITE(6,10)(LABEL(I),I=1,18)
   WRITE(6,63)
   WRITE(6,413)
   WRITE(6,414)(TIME(J),DX(J),DY(J),DZ(J),DXP(J),DYP(J),DZP(J),DB(J),D
   ALPH(J),J=1,KMAX)
900 WRITE(6,10)(LABEL(I),I=1,18)
   WRITE(6,901)
901 FORMAT(5X,'EXPECTED RMS ERRORS'//)
   WRITE(6,61)
   WRITE(6,602)(TIME(J),SIGR(J),SIGRP(J),SIGA(J),SIGAP(J),SIGE(J),SIG
   E(J),SIGEP(J),SIGALP(J),J=1,KMAX)
902 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F1
10.6,2X,F12.8)
   WRITE(6,10)(LABEL(I),I=1,18)
   WRITE(6,901)
   WRITE(6,413)
   WRITE(6,403)(TIME(J),SIGX(J),SIGY(J),SIGZ(J),SIGXP(J),SIGYP(J),SIG
   ZP(J),SIGALP(J),J=1,KMAX)
10.2,14X,F12.8)
   L=1
1000 CONTINUE
1001 GC TC 1
9000 RETURN
EAC
49
TABLE B-II  
MAIN PROGRAM (VERSION II)

C MAIN PROGRAM
IMPLICIT REAL*8(A-H,O-Z)
CCMN /ACC/CCVAR(7),SIGMA(7)/FCOM/COORD,DLAT,PRNT/ICCM/MDA
ITAN
DIMENSION GNCISE(7),RGN(54),AZN(54),ELN(54),RAPN(54),BETAM(54),
1 KHAT(54),AHAT(54),EFAT(54),RPHAT(54),APHAT(54),EPHAT(54),STAT
2EM(7),STATE(7),PHS(7,7),NOISE(7),ALPHA(54),DILPHA(54)
DIMENSION PHINV(7,7),PROC(7,7),ESTATE(7),STATE(7),X(54),Y(54),Z(5
14),XP(54),YP(54),ZP(54),CX(54),DY(54),DZ(54),DXP(54),DYP(54),DZP(5
24),XHAT(54),YHAT(54),ZHAT(54),YPHAT(54),XPHAT(54),ZPHAT(54),ASPECT
3(54),SIGR(54),SIGY(54),SIGR(54),SIGXP(54),SIGYP(54),SIGZP(54),SIGX
4(54),SIGY(54),SIGZ(54),SIGXP(54),SIGY(54),SIGZP(54),SIGXP(54),SIGYP(54)
DIMENSION CR(54),DE(54),DA(54),DRP(54),DAP(54),DEP(54),DB(54)
CALL DENS TC
SET ATOMOSPHERIC DENSITY TABLES
CALL DENS REAC INPUT PARAMETERS AND PRINT LISTING
1 REAC(5,2,ENC=9C0C)(LABEL(I),I=1,18)
2 FCRMAT (18A4)
3 REAC(5,4)TZERC,TINT,TINCR,DLAT,PRNT,NDATA,KLAMP,MDATA
FCRMAT(4F10.3,F5.3,315)
4 REAC(5,3)
5 SIGMA(J),J=1,6),COORD
FCRMAT7F10.3)
6 REAC(5,3)
7 ERRMAX(JJ),JJ=1,7)
8 IMAX=50
9 CCCCRC=DOABS(CCCC)
HTEG=180.3/14\18
C CCCCRC=DOABS(CCCC)
HTEG=180.3/14\18
C WRITE (1,E10) (LABEL(I),I=1,18)
10 FCRMAT(1",2CX,"OUTPUT LISTING"/18A4/)
WRITE(6,E11C)CLAT,NCATA,TINT,TINCR
11 FCRMAT(5X,"RADAR LATITUDE",3X,F10.5,2X,"DEGREES"/5X,16,5X,
11 DATA PRINTS SPACEC3X,F10.6,2X,"SEC. APART, INTEGRATION STEP",2
2X,F10.6,2X,"SEC."/)}
TABLE B-II (Continued)

14 IF(CCOORD)14,16,16
14 RE=2.0925738C7/3.2808333
15 WRITE(6,15)
15 FORMAT(/5X,'METRIC UNITS(METERS,KILOGRAMS,RADIANS,SECONDS) USED THROUGHOUT PROGRAM///)
16 GO TO 18
16 RE=2.0925738C7
17 WRITE(6,17)
17 FORMAT(/5X,'ENGLISH UNITS(FEET,POUNDS,RADIANS,SECONDS) USED THROUGHOUT PROGRAM///)
18 CC 19 J=1,7
19 STATEM(J)=0.
75 IF(KLAMP)81,81,751
75 WRITE(6,7521)
752 FORMAT(/5X,'MEMORY TIME =',16,' DATA POINTS///)
81 WRITE(6,12)(ERRMAX(JJ),JJ=1,7)
12 FORMAT(/5X,'MAXIMUM INTEGRATION CONVERGENCE ERRORS(IN RECTANGULAR
1CCCRUCINATES///7C17.8///)
L=2
LST=0
C
C DATA LCGP
C
CC 1000 NN=1,NCATA
NN=NN
NN1=NN+1
C
C TEST FCR 1ST TIME THROUGH
C
C IF(NN-1)21,2C,21
C
C REAC INPUT DATA CARCS
C
20 REAC(5,30) TIME(1),RGN(1),AZN(1),ELN(1),RAPN(1)
30 FORMAT(15.2,4C15.6)
30 IF(TIME(1)=TZERO)20,200,200
C
C SET INITIAL STATE ESTIMATE = INITIAL NOISY STATE
C
200 RPHAT(1)=RGN(1)
APHAT(1)=AZN(1)
EPHAT(1)=ELN(1)
E=TIME(2)-TIME(1)
201 REAC(5,30) TIME(2),RGN(2),AZN(2),ELN(2),RAPN(2)
E=TIME(2)-TIME(1)
201 IF(E(TINT)201,202,202
202 APHAT(1)=(AZN(2)-AZN(1))/E
EPHAT(1)=(ELN(2)-ELN(1))/E
CCVAR(1)=SIGMA(1)
CCVAR(2)=SIGMA(2)
CCVAR(3)=SIGMA(3)
CCVAR(5)=1.414*SIGMA(2)/E
CCVAR(6)=1.414*SIGMA(3)/E
CCVAR(7)=1.
203 CCVAR(4)=SIGMA(1)/E
C
203 WRITE(6,21C)
210 FORMAT(/5X,*** THIS MAIN PROGRAM REQUIRES MDATA = 3 CR 4, QUIT*)
RETURN
204 RPHAT(1)=(RGN(2)-RGN(1))/E
CCVAR(4)=1.414*SIGMA(1)/E
GC TC 206

51
TABLE B-II (Continued)

205 \text{RPHAT(1)=RAPN(1)}
\text{C}
\text{CCVAR(4)=SIGMA(4)}
206 \text{ESTATE(7)=1.-C-4}
\text{BETAH(1)=1./ESTATE(7)}
\text{C}
\text{PRINT HEADER PAGE OF OUTPUT LISTING}
\text{C}
\text{WRITE(6,13) MDATA}
13 \text{FORMAT(/5X,'ONLY THE 1ST',13,1X,'QUANTITIES OF THE STATE VECTOR'/}
\text{15X,'CONSISTING OF R,AZ,EL,ROOT,AZDOT,ELDOT,1./BETA, ARE MEASURED')}
\text{WRITE(6,77)}
77 \text{FORMAT(/5X,'RMS NOISE LEVELS ASSOCIATED WITH MEASUREMENT VECTOR')}
\text{WRITE (6,7)(SIGMA(J),J=1,6)}
7 \text{FORMAT(/5X,'SIGMA(1/BETA)=',F10.3//)}
\text{WRITE(6,78)}
78 \text{FORMAT(/5X,'RMS NOISE LEVELS ASSOCIATED WITH INITIAL STATE VECTOR')}\n\text{1/}
\text{WRITE(6,7)(CCVAR(J),J=1,6)}
\text{WRITE(6,79) COVAR(7)}
79 \text{FORMAT(5X,'SIGMA(1/BETA)=',F10.3//)
\text{SET UP INITIAL STATE IN BOTH RADAR POLAR AND XYZ COORDINATES}
\text{CALL RT0XY(RHAT(1),AHAT(1),EHAT(1),RPHAT(1),APHAT(1),EPHAT(1),ESTA}
\text{SITE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTATE(6))
\text{XHAT(1)=ESTATE(1)}
\text{YHAT(1)=ESTATE(2)}
\text{ZHAT(1)=ESTATE(3)}
\text{XPFAT(1)=ESTATE(4)}
\text{YPHAT(1)=ESTATE(5)}
\text{ZPHAT(1)=ESTATE(6)}
\text{ALPHA(1)=ESTATE(7)}
\text{HEIGHT(1)=XHAT(1)}
\text{NXT=1}
\text{WRITE(6,49) NN}
\text{WRITE(6,50) TIME(NXT),RPHAT(NXT),AHAT(NXT),EHAT(NXT),RPHAT(NXT),APHA}
\text{LTA(NXT),EPHAT(NXT),ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTA}
\text{TE(6),BETAH(NXT))}
\text{K=1}
\text{NXT=2}
\text{INDEX=2}
\text{GO TO 31}
\text{UPDATE SUBSCRIPTS}
\text{21 K=K+1}
\text{INDEX=NN}
\text{IF(K)<IMAX}
\text{K=1}
\text{READ NEXT DATA POINT}
\text{24 READ(5,30,ENC=8000)TIME(NXT),RGN(NXT),AZN(NXT),ELN(NXT),RAPN(NXT)}
\text{CT=TIME(NXT)-TLAST}
\text{IF(CT<TINT)24,31,31}
\text{31 TLAST=TIME(NXT)}
TAU=CT

CALL TRAJGX TO OBTAIN NEXT PREDICTED DATA POINT

CALL TRAJGX(ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTATE(6),BETAH(K),TAU,TINC,ERRMAX,STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PBETA,PROD)

CCONVERT NEXT PREDICTED DATA POINT TO RADAR POLAR COORDINATES

CALL XTYTCR(STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PRA,PAZ,PEL,PKAP,PAZP,PELP)

PH=CSCRT(STATEP(1)**2+STATEP(2)**2+(STATEP(3)+RE)**2)-RE

STATEP(7) = 1./PBETA

STATEM(1) = RGMNXT

STATEM(2) = AZN(NXT)

STATEM(3) = ELN(NXT)

STATEM(4) = RAPN(NXT)

IF(PRNT)570,569,569

PRINT NCISY MEASUREMENT

PRINT PREDICTED STATE

WRITE(6,569)NAP1

49 FCRMAT('1',5X,15,' THE DATA POINT')/

WRITE(6,558)TIME(NXT),RGN(NXT),AZN(NXT),ELN(NXT),RAPN(NXT)

50 FCRMAT('5X','NCISY DATA'/5X,' TIME =',F10.4,5X,' RGN =',F10.4,5X,' AZN =

1',F10.6,3X,' ELN =',F10.6,3X,' RAPN =',F10.2//)

WRITE(6,570)

51 FCRMAT('5X','PREDICTED STATE BASED ON PAST DATA ONLY')

WRITE(6,550)TIME(NXT),PRA,PAZ,PEL,PRAP,PAZP,PELP,STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PH, PBETA

WRITE(6,570,5702,569,569)

WRITE TRANSITION MATRIX

WRITE(6,51)(PRCC(JJ,KK),KK=1,7),JJ=1,7)

CALL ESTMAT TO COMBINE MEASUREMENT WITH PREDICTION TO OBTAIN

ESTIMATE OF STATE VECTOR

WRITE(6,50)TIME(NXT),PRA,PAZ,PEL,PRAP,PAZP,PELP,STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PBETA

WRITE(6,570,5702,570,5702)

WRITE TRANSITION MATRIX

51 FCRMAT('5X','TRANSITION MATRIX'/(7D17.8))

CALL ESTMAT TO COMBINE MEASUREMENT WITH PREDICTION TO OBTAIN

ESTIMATE OF STATE VECTOR

WRITE(6,50)TIME(NXT),PRA,PAZ,PEL,PRAP,PAZP,PELP,STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PBETA

WRITE(6,570,5702,570,5702)

WRITE TRANSITION MATRIX

51 FCRMAT('5X','TRANSITION MATRIX'/(7D17.8))

CALL ESTMAT TO COMBINE MEASUREMENT WITH PREDICTION TO OBTAIN

ESTIMATE OF STATE VECTOR

WRITE(6,50)TIME(NXT),PRA,PAZ,PEL,PRAP,PAZP,PELP,STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PBETA

WRITE(6,570,5702,570,5702)

WRITE TRANSITION MATRIX

51 FCRMAT('5X','TRANSITION MATRIX'/(7D17.8))
TABLE B-II (Continued)

<table>
<thead>
<tr>
<th>C</th>
<th>CCNVERT ESTIMATED STATE TO RADAR POLAR COORDINATES</th>
<th>C</th>
<th>STCRE ESTIMATED STATE VECTOR IN OUTPUT ARRAYS</th>
</tr>
</thead>
<tbody>
<tr>
<td>5793</td>
<td>CALL XYTCR(ESTATE(1), ESTATE(2), ESTATE(3), ESTATE(4), ESTATE(5), ESTATE(6), RPHAT(NXT), APHAT(NXT), EPHAT(NXT), RPHAT(NXT), APHAT(NXT), EPHAT(NXT))</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>XHAT(NXT) = ESTATE(1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>YHAT(NXT) = ESTATE(2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZHAT(NXT) = ESTATE(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>XPAT(NXT) = ESTATE(4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>YPHAT(NXT) = ESTATE(5)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZPHAT(NXT) = ESTATE(6)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALPHA(NXT) = ESTATE(7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BETA(NXT) = 1./ESTATE(7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>NHT = CSQRT(ESTATE(1)**2 + ESTATE(2)**2 + (ESTATE(3) + RE)**2) - RE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HEIGHT(NXT) = FNXT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IF(PRNT)581,579,579</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PRINT LUT ESTIMATED STATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>579</td>
<td>WRITE(6,580)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>580</td>
<td>FORMAT(5X,'ESTIMATED POINT/')</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WRITE(6,50)</td>
<td>TIME(NXT), RPHAT(NXT), APHAT(NXT), EPHAT(NXT), RPHAT(NXT), APHAT(NXT), EPHAT(NXT), ESTATE(1), ESTATE(2), ESTATE(3), ESTATE(4), ESTATE(5), ESTATE(6), NHT, BETA(NXT)</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>FORMAT(5X,<em>TIME</em>='F10.4/5X,<em>RA</em>='F15.2,3X,<em>AZ</em>='F10.6,3X,<em>EL</em>='F10.6,3X,<em>XP</em>='F10.2,3X,<em>YP</em>='F10.2,3X,<em>ZP</em>='F10.2,3X,<em>HEIGHT</em>='F10.2,5X,<em>BETA</em>='F10.2,5X,*F10.2//'</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CCMPUTE DIFFERENCES BETWEEN ESTIMATE AND MEASURED DATA POINT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>581</td>
<td>CR(NXT) = RPHAT(NXT) - RGN(NXT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CA(NXT) = APHAT(NXT) - AZN(NXT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CE(NXT) = EPHAT(NXT) - ELN(NXT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CRP(NXT) = RPHAT(NXT) - RPN(NXT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>IF(NN-NCATA)27,2e,2e</td>
<td></td>
</tr>
<tr>
<td></td>
<td>26</td>
<td>KMAX=INDEX</td>
<td></td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>IF(INDEX-IMAX)1CCG,28,28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>KMAX=IMAX</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WRITE COUTPUT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>WRITE(6,10)(LABEL(I), I=1,18)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>420</td>
<td>WRITE(6,43)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>43</td>
<td>FORMAT(5X,<em>MEASUREMENT DATA</em>//6X,<em>TIME</em>,'RX','RANGE',7X,'RAC0T',17X,1,'AZIM*',20X,'ELEV//')</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WRITE(6,44)(TIME(KK), RGN(KK), RPN(KK), AZN(KK), ELN(KK), KK=1,KMAX)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.6,14X,F10.6,14X,F10.6)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WRITE(6,10)(LABEL(I), I=1,18)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>WRITE(6,60)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>FORMAT(</td>
<td>5X,<em>ESTIMATED VALUES</em>'//)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WRITE(6,41)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FORMAT(3X,<em>TIME</em>,'RX','RANGE',7X,'RAC0T',7X,<em>AZIM</em>',8X,<em>AZDOT',7X,1,'ELEV</em>',8X,*ELECT',7X,*HEIGHT',6X,*BETA//)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WRITE(6,62)(TIME(J), RHAT(J), RPAT(J), AHAT(J), APAT(J), EAHAT(J), J=1,KMAX)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>62</td>
<td>FORMAT(3X,F10.3,2X,F10.2,1X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6)</td>
<td></td>
</tr>
</tbody>
</table>

54
TABLE B-II (Continued)

```plaintext
WRITE(6,10)(LABEL(I),I=1,18)
WRITE(6,60)
WRITE(6,413)
WRITE(6,414)(TIME(J),XHAT(J),YHAT(J),ZHAT(J),XPHAT(J),YPHAT(J),ZPHAT(J),J=1,KMAX)
WRITE(6,10)(LABEL(I),I=1,18)
WRITE(6,63)
63 FORMAT(5X,'RESIDUALS=(ESTIMATED VALUES)-(NOISY DATA)'//)
WRITE(6,64)
64 FORMAT(6X,'TIME',8X,'RANGE',7X,'RADOT',17X,'AZIM',20X,'ELEV'//)
WRITE(6,44)(TIME(J),DR(J),DRP(J),DA(J),DE(J),J=L,KMAX)
900 WRITE(6,10)(LABEL(I),I=1,18)
901 WRITE(6,401)
901 FORMAT(5X,'EXPECTEC RMS ERRORS'//)
WRITE(6,61)
61 FORMAT(3X,'TIME',8X,'RANGE',7X,'RADOT',7X,'AZIM',8X,'AZDCT',7X,'EL',16X,8X,'ELDCT',7X,'BETA',8X,'ALPHA'//)
WRITE(6,902)(TIME(J),SIGR(J),SIGRP(J),SIGA(J),SIGAP(J),SIGE(J),SIG1P(J),SIGALP(J),J=L,KMAX)
901 WRITE(6,10)(LABEL(I),I=1,18)
901 WRITE(6,413)
413 FORMAT(3X,'TIME',8X,'Y',11X,'Z',11X,'XDOT',8X,'YDOT',8X,'Z',10X,'BETA',8X,'ALPHA'//)
WRITE(6,903)(TIME(J),SIGX(J),SIGY(J),SIGZ(J),SIGXP(J),SIGYP(J),SIGZP(J),SIGALP(J),J=L,KMAX)
901 L=1
IF(LAST)9000,100C,9000
1000 CONTINUE
1001 GC TC 1
8CC0 IF(NXT-1)9000,900C,8001
8C01 KMAX=NXT-1
8C01 LAST=NXT-1
GC TC 40
9000 RETURN
END
```

55
**TABLE B-III**

SUBROUTINE TRAJGX

```fortran
SUBROUTINE TRAJGX(XI,YI,ZI,XDOT1,YDOT1,ZDOT1,BETA1,TAU,TSTEP,ERRM
1AX,X2,Y2,Z2,XDOT2,YDOT2,ZDOT2,BETA2,PHIMAT)
IMPLICIT REAL*8(A-H,O-Z)
REAL*8 TAU,TSTEP
REAL*8 J,K,L,M,N,X1,Y1,Z1,XDOT1,YDOT1,ZDOT1,BETA1,TINT,TINC,ERRMA
1X,X2,Y2,Z2,XDOT2,YDOT2,ZDOT2,BETA2,PHIMAT
COMMON /FCOM/CCORD,DLAT,PRNT/ICOM/KLAMP,MDATA,NN
DIMENSION XVCTR(7,3),BVCTR(7,3),XI(7,3),ERR(7,3),AMTRX(7,7,3),ASQ(17,7,3),AB(7,3),PHI(7,7,3),UNIT(7,7),ERRMAG(7),ERRMAX(7),PHIMAT(7,7,2)
C
C CHECK THAT DATA INTERVAL,TAU, AND INTEGRATION STEP,TSTEP, HAVE
C THE SAME SIGN
C
TPROD=TAU*TSTEP
IF(TPROD)1,2,2
1 TINCRC=-TSTEP
WRITE(6,I1)
11 FORMAT(5X,'DATA INTERVAL,TAU, AND INTEGRATION STEP,TSTEP, HAD OP-
1POSITE SIGN.'/5X,'LATTER SIGN WAS CHANGED TO FORCE AGREEMENT'//)
GO TO 5
2 TINC=TSTEP
5 TM=OABS(TAU)
TSTM=DABS(TSTEP)
IF(TM-TSTM)3,4,4
3 TINC=TAU
4 TINT=TAU
C
C CHECK FOR 1ST DATA POINT
C
IF(NN -1)100,100,89
C
SET UP CONSTANTS AND ARRAYS
C
100 PI=3.14159265
RTDEG=1.8D2/PI
OMEGA=7.27D-5
RLAT=DLAT/RTDEG
OMC=OMEGA*CCOS(RLAT)
CMS=OMEGA*DSIN(RLAT)
IF(COOKD)1000,1001,1001
1000 RE=2.092573807/3.2808333
GM=1.40763488016/
I
3.2808333)**3
GAMMA=.304800/7.D3
GO  TO 1002
1001 RE=2.092573807
GM=1.40763488016
GAMMA=.304800/7.D3
1002 DO 101 JJ=1,7
DO 101 KK=1,7
101 UNIT(JJ,KK)=0.
DO 102 JJ=1,7
ERR(JJ,1)=0.
ERR(JJ,2)=0.
ERR(JJ,3)=0.
102 UNIT(JJ,JJ)=1.
89 IF(TAU)103,90,103
C DATA INTERVAL,TAU, = 0.
C SET OUTPUT STATE = INPUT STATE, TRANSITION MATRIX=IDENTITY MATRIX
C
90 X2=X1
Y2=Y1
```

56
TABLE B-III (Continued)

\[ Z_2 = z_1 \]
\[ \Omega_2 = \Omega_1 \]
\[ \Omega_2 = \Omega_1 \]
\[ \Omega_2 = \Omega_1 \]
\[ \beta_2 = \beta_1 \]

**CO 91 JJ=1,7**
**CO 91 KK=1,7**

\[ \phi_{\text{MATUJ,KK}} = \text{UNIT}(JJ, KK) \]

**RETURN**

**SET INDICES**

**103**

**NP0INT=1**
**K0UNT=1**

**NEXT=2**
**NLAST=3**

**T=0.**

**K0ELT=0**

**PHSKIP=0.**

**TSC=TINCR**

**CALCULATE DRAG VELOCITY AND HEIGHT**

\[ V_D = \sqrt{\Omega_2(4, NP0INT)^2 + \Omega_2(5, NP0INT)^2 + \Omega_2(6, NP0INT)^2} \]

\[ R = \sqrt{\Omega_2(1, NP0INT)^2 + \Omega_2(2, NP0INT)^2 + \Omega_2(3, NP0INT)^2 + \Omega_2(4, NP0INT)^2 + \Omega_2(5, NP0INT)^2 + \Omega_2(6, NP0INT)^2} \]

**IF(HT)3001,3003,3003**

**NEGATIVE HEIGHT, APPARENT EARTH IMPACT, RETURN TO CALLING PROGRAM**

**3001**

**WRITE(6,3002)HT,T,(\Omega_2(JJ,NP0INT)\text{JJ=1,7})**

**3002**

**FORMAT(4X,'HEIGHT=',F10.2,'AFTER ',F10.5,' SEC. OF INTEGRATION**

**1-- APPARENT EARTH IMPACT*/5X,'STATE VECTOR=','/7017.8/ 5X,'RETURNED**

**2TO CALLING PROGRAM//*/**

**NEXT=NP0INT**

**GO TO 600**

**3003**

**HKMFT=HT/1.03**

**RHO=0.**

**CALL ATM(HKMFT,RHO)**

**ALPHA=\Omega_2(7,NPCINT)**

**SET UP EQUATIONS OF MOTION**

\[ A = -RHO*V_0/2.00 \]

\[ W = GM/R**3 \]

\[ B = \Omega_2**2 - W \]

\[ C = 2.00*OMS \]
TABLE B-III (Continued)

\[
\begin{align*}
E &= -2.00*OMC \\
G &= OMS*2-W \\
J &= J*RE \\
K &= J*RE \\
N &= OMC*2-W \\
C &= N*RE \\
XPP &= A*XVCTR(7,NPOINT)+B*XVCTR(1,NPOINT)+C*XVCTR(5,NPOINT)+E*XVCTR(6,NPOINT) \\
YPP &= -C*XVCTR(4,NPOINT)+G*XVCTR(2,NPOINT)+A*XVCTR(7,NPOINT)*XVCTR(5,NPOINT)+J*RE \\
ZPP &= -E*XVCTR(4,NPOINT)+J*XVCTR(3,NPOINT)+K \\
&= J*XVCTR(2,NPOINT)+N*XVCTR(3,NPOINT)+A*XVCTR(7,NPOINT)*XVCTR(6,NPOINT)+Q \\
\end{align*}
\]

FILL IN STATE VECTOR MATRICES

\[
\begin{align*}
BVCTR(1,NPOINT) &= XVCTR(4,NPOINT) \\
BVCTR(2,NPOINT) &= XVCTR(5,NPOINT) \\
BVCTR(3,NPOINT) &= XVCTR(6,NPOINT) \\
BVCTR(4,NPOINT) &= XVCTR(7,NPOINT) \\
BVCTR(5,NPOINT) &= YPP \\
BVCTR(6,NPOINT) &= ZPP \\
BVCTR(7,NPOINT) &= 0. \\
\end{align*}
\]

CALCULATE SYSTEM MATRIX A

\[
\begin{align*}
&\text{DO 301 JJ=1,3} \\
&\text{DO 301 KK=1,7} \\
301 &\text{AMTXX(JJ,KK,NPOINT)=0.} \\
&\text{AMTXX(1,4,NPOINT)=1.} \\
&\text{AMTXX(2,5,NPOINT)=1.} \\
&\text{AMTXX(3,6,NPOINT)=1.} \\
\end{align*}
\]

CALCULATE DERIVATIVES OF COEFFICIENTS

\[
\begin{align*}
&\text{DA=-AGAMMA/R} \\
&\text{CAXX=DA*XVCTR(1,NPOINT)} \\
&\text{LAXY=DA*XVCTR(2,NPOINT)} \\
&\text{LAXZ=DA*XVCTR(3,NPOINT)+RE} \\
&\text{IF(VD)701,702,701} \\
701 &\text{CAP=A/VD**2} \\
&\text{GO TO 703} \\
702 &\text{CAP=A/1.0-4} \\
703 &\text{CONTINUE} \\
&\text{CADXP=CAP*XVCTR(4,NPOINT)} \\
&\text{CAXY=CAP*XVCTR(5,NPOINT)} \\
&\text{CAXZ=CAP*XVCTR(6,NPOINT)} \\
&\text{DB=-3.00**W/R**3} \\
&\text{DBDX=DB*XVCTR(1,NPOINT)} \\
&\text{DBDY=DB*XVCTR(2,NPOINT)} \\
&\text{DBDZ=DB*XVCTR(3,NPOINT)+RE} \\
&\text{EGOX=EBDX} \\
&\text{EGDY=EBDY} \\
&\text{EGDZ=EBDZ} \\
&\text{ECSX=ECBD} \\
&\text{ECSY=ECBD} \\
&\text{ECSZ=ECBD} \\
&\text{FOR CONSTANT BETA} \\
&\text{CALFDX=0.} \\
\end{align*}
\]
TABLE B-III (Continued)

DALFDY=0.
DALFDZ=0.

C

DALADX=ALPHA*GADX+A*DALFDX
DALADY=ALPHA*GADY+A*DALFDY
DALADZ=ALPHA*GAOZ+A*DALFDZ

AMTRX(4,1,NPOINT)=B*XVCTR(1,NPOINT)*DBX*XVCTR(4,NPOINT)*DALADX
AMTRX(4,2,NPOINT)=XVCTR(1,NPOINT)*DBDY*XVCTR(4,NPOINT)*DALADY
AMTRX(4,3,NPOINT)=XVCTR(1,NPOINT)*DBDZ*XVCTR(4,NPOINT)*DALADZ
AMTRX(4,4,NPOINT)=ALPHA*(A*XVCTR(4,NPOINT)*DALADP)
AMTRX(4,5,NPOINT)=XVCTR(4,NPOINT)*ALPHA*DADYP+C
AMTRX(4,6,NPOINT)=XVCTR(4,NPOINT)*ALPHA*DAOZP+E

AMTRX(4,7,NPOINT)=A*XVCTR(4,NPOINT)

AMTRX(5,1,NPOINT)=XVCTR(2,NPOINT)*GQOX*XVCTR(5,NPOINT)*DALADX
AMTRX(5,2,NPOINT)=G*XVCTR(2,NPOINT)*GQOX*XVCTR(5,NPOINT)*DALADY
AMTRX(5,3,NPOINT)=J*XVCTR(2,NPOINT)*GQDZ*XVCTR(5,NPOINT)*DALADZ
AMTRX(5,4,NPOINT)=XVCTR(5,NPOINT)*ALPHA*DADXP-C
AMTRX(5,5,NPOINT)=ALPHA*(A*XVCTR(5,NPOINT)*DALADP)
AMTRX(5,6,NPOINT)=XVCTR(5,NPOINT)*ALPHA*DAOZP
AMTRX(5,7,NPOINT)=A*XVCTR(5,NPOINT)

AMTRX(6,1,NPOINT)=DQDX*XVCTR(3,NPOINT)*DNDX*XVCTR(6,NPOINT)*DALADX
AMTRX(6,2,NPOINT)=J+DQDY*XVCTR(3,NPOINT)*DNDY*XVCTR(6,NPOINT)*DALADY
AMTRX(6,3,NPOINT)=J+DQDY*XVCTR(3,NPOINT)*DNDY*XVCTR(6,NPOINT)*DALADY

CY

AMTRX(6,4,NPOINT)=N*XVCTR(3,NPOINT)*DNDZ*GQDZ*XVCTR(6,NPOINT)*DALADZ

1C2

AMTRX(6,5,NPOINT)=XVCTR(6,NPOINT)*ALPHA*DADYP-E
AMTRX(6,5,NPOINT)=XVCTR(6,NPOINT)*ALPHA*DADYP
AMTRX(6,6,NPOINT)=ALPHA*(A*XVCTR(6,NPOINT)*DALADP)
AMTRX(6,7,NPOINT)=A*XVCTR(6,NPOINT)
AMTRX(7,1,NPOINT)=DALFDX
AMTRX(7,2,NPOINT)=DALFDY
AMTRX(7,3,NPOINT)=DALFDZ

CO 3010 JJ=4,7

3010 AMTRX(7, JJ,NPOINT)=0.

CO 302 JJ=1,7
AB(JJ,NPOINT)=0.

CO 302 LL=1,7

302 AB(JJ,NPOINT)=AB(JJ,NPOINT)+AMTRX(JJ,LL,NPOINT)*BVCTR(LL,NPOINT)

IF(PHISP)400,303,400

C

C CALCULATE TRANSITION MATRIX

C

303 CO 304 JJ=1,7

304 ASG(JJ,KK,NPCINT)=0.

304 ASG(JJ,KK,NPCINT)=ASG(JJ,KK,NPCINT)+AMTRX(JJ,LL,NPOINT)*AMTRX(LL,

KK,NPOINT)

305 PHI(JJ,KK,KPCINT)=UNIT(JJ,KK)+TINCR*AMTRX(JJ,KK,NPOINT)+TSQ*ASQ(JJ

1,KK,NPOINT)/2.D0

C IF(KGUNT>320,320,330

306 PHI(JJ,KK,NEXT)=PHI(JJ,KK,NEXT)+PHI(JJ,LL,KPCINT)*PHI(LL,KK,NPOINT

1)

C

C INTEGRATE OVER INTERVAL TINCR

C

IF(KGUNT<KGUNT)320,320,330
TABLE B-III (Continued)

320 CO 321 JJ=1,7
321 XI(JJ,NEXT)=XVCTR(JJ,NPOINT)*TINCR*BVCTR(JJ,NPOINT)+TSQ*AB(JJ,NPOINT)/2.00
GO TO 350
330 DO 331 JJ=1,7
331 XI(JJ,NEXT)=XVCTR(JJ,NPOINT)+2.00*TINCR*BVCTR(JJ,NPOINT)+2.00/3.00*T5S*AB(JJ,NPOINT)
350 NPT=NPOINT
NPOINT=NEXT
CO 351 JJ=1,7
351 XVCTR(JJ,NPOINT)=XI(JJ,NEXT)
PHSKIP=1.
GO TO 300
400 CO 4000 JJ=1,7
400 XVCTR(JJ,NEXT)=XVCTR(JJ,NPOINT)+.500*TINCR*BVCTR(JJ,NPOINT)+BVCTR(JJ,NPOINT)*TINCR
ERR(JJ,NEXT)=XI(JJ,NEXT)-XVCTR(JJ,NEXT)
ERRMAG(JJ)=DABS(ERR(JJ,NEXT))
4000 CONTINUE
C TEST INTEGRATION CONVERGENCE ERRORS
C
CO 401 JJ=1,7
401 IF(ERRE1MAG(JJ)-ERRMAX(JJ))401,500,500
CONTINUE
T=T+TINCR
TREM=TAU-T
4010 TRENAG=MIN(TREM)
401 IF(TRENAG<TSTM)401,402,402
405 TINCR=TREM
TSC=TINCR**2
KOUNF=0.
C UPDATE SUBSCRIPTS
C
402 NEXT=MOD(NEXT,3)+1
N1AST=NPT
KDEL=F+KDEL+1
PHSKIP=1.
GO TO 300
C CONVERGENCE ERRORS TOO LARGE -- PERFORM ITERATION
C
500 CO 501 KK=1,7
501 XI(KK,NEXT)=XVCTR(KK,NEXT)
PHSKIP=1.
GO TO 300
C SET UP OUTPUT STATE VECTOR AND TRANSITION MATRIX
C
600 X2=XVCTR(1,NEXT)
Y2=XVCTR(2,NEXT)
Z2=XVCTR(3,NEXT)
XDEL2=XVCTR(4,NEXT)
YDEL2=XVCTR(5,NEXT)
ZDEL2=XVCTR(6,NEXT)
BETA2=1./XVCTR(7,NEXT)
CO 601 JJ=1,7
CO 601 KK=1,7
601 PHI1MAT(JJ,KK)=PHI(JJ,KK,NEXT)
RETURN
ENC
**TABLE B-IV**

**SUBROUTINE ESTMAT (VERSION I)**

```plaintext
SUBROUTINE ESTMAT(STATEM, STATEP, PHIMTX, ESTATE, DEVX, DEVR)

IMPLICIT REAL*8(A-H, O-Z)
REAL*8 STATEM, STATEP, PHIMTX, ESTATE
COMMON /ACCM/CCVAR(7), SIGMA(7)/FCOM/COORD, DLAT, PRNT/ICCM/KLAMP, MDA
ITAI,N
DIMENSION PHIMTX(7,7), CUMMY(7,7), W(7,7), DX(7), STATEP(7)
DIMENSION STATEM(7), ESTATE(7), H(7, 7), R(7, 7), S(7,7, 2)
DIMENSION DEVX(7), DEVR(7)
DIMENSION U77(7,7), SR(7,7,2), PHIT(7,7), UNIT(7,7)
DIMENSION HT(7,7), RSTATE(7), MVI(7), MV2(7), ARRAYU(49)
NZERO=0
CCCRM=DAABS(COCR)

TEST FCR 1ST TIME THROUGH
IF(0N-1)1, 1, 100

1 NSTART=1
   IF(MDATA)2, 200, 1
2 MD=MDATA
   CC TC 4
3 MD=MDATA
4 MPI=MC+1
   LMAX=MC**2
   NK=1
   NKM1=2
   CC 7 J=1,7
   CC 7 K=1,7
   UNIT(J,K)=0.
   H(J,K)=0.
   DX(J)=0.
   F(J,K)=0.
   R(J,K)=0.
   SR(J,K,1)=0.
   7 S(J,K,1)=0.
   CC 6 J=1, MC
   F(J,J)=1.
   8 R(J,J)=SIGMA(J)**2
   88 CC H J=1, 7
   8 UNIT(J,J)=1.
   IF(CCCRMK-1.)90, 90, 80
   90 CC 91 J=1, 7
   91 S(J,1,1)=CCVAR(J)**2
   CC TC 1302
   80 CC 81 J=1, 7
   81 SR(J,J)=CCVAR(J)**2
   IF(PRNT)10,9,9
   9 WRITE(6,1180)(SR(J,K, NK), K=1, 7), J=1, 7)
1180 FORMAT(*5X,'APRICPI COVARIANCE MATRIX IN RADAR COORDINATES'/(7D17.
   18))
10 CC 1181 J=1, 7
   CC 1181 K=1, 7
1181 U77(J,K)=SR(J,K, NK)
CALL RCCVTX(U77, ESTATE, CUMMY)
```

```c

```
CC 1182 J=1,7
CC 1182 K=1,7
1182 S(J,K,NK)=CUMMY(J,K)
CC TC 1302
C
C UPDATE INCIICES NK,NKM1
C
100 IF(NK=111,102,103
102 NK=2
NK1=1
GC TC 104
103 NK=1
NK1=2
C
C COMPUTE CCVARIANCE MATRIX S(K,K-1) IN XYZ COORDINATES
C
104 CC 105 J=1,7
CC 105 K=1,7
PHIT(J,K)=PHIMTX(K,J)
105 EUMMY(J,K)=S(J,K,NKM1)
CALL DMPMUL(EUMMY,PHIT,UL77,7,7,7)
CALL DMTMUL(PHIMTX,UL77,EUMMY,7,7,7)
START=0
106 GC 106 J=1,7
GC 106 K=1,7
106 S(J,K,NKM1)=EUMMY(J,K)
IF(PRNT)1060,1C6C,1C58
PRINT CUT TRANSITION MATRIX
PRINT CUT CCVARIANCE MATRIX S(K,K-1) OF PREDICTED STATE
C
1058 WRITE(6,300)((PHIMTX(J,K),K=1,7),J=1,7)
300 FORMT('/5X,'TRANSITION MATRIX USED IN ESTIMATION'/7D17.8))
WRITE(6,1059)
1059 FORMT('/5X,'COVARIANCE MATRICES S(K,K-1'))
WRITE (6,1183)((S(J,K,NKM1),K=1,7),J=1,7)
C
C CALCULATE F-YERIC CCVARIANCE MATRIX OF PREDICTED STATE
C
CALL XCCVTRI(EUMMY,STATEP,DU77,7)
CC 2016 J=1,7
CC 2015 K=1,7
2015 H(J,K)=DU77(J,K)/CSCRT(DU77(J,J)*DU77(K,K))
2016 H(J,K)=CSCRT(DU77(J,J))
C
C PRINT CUT F-YERIC CCVARIANCE MATRIX OF PREDICTED STATE
C
WRITE(6,1301)((H(J,K),K=1,7),J=1,7)
C
C COMPUTE WEIGHTING MATRIX W
C
1060 CALL XLUVTRI(EUMMY,STATEP,H,-1)
CC 100 J=1,7
CC 107 K=1,7
107 F(J,K)=H(K,J)
CALL DMTMUL(EUMMY,F1,DU77,7,7,MD)
CALL DMTMUL(F,EU77,EUMMY,MC,7,MD)
CC 100 J=1,ME
CC 108 K=1,MC
108 EU77(J,K)=EUMMY(J,K)*R(J,K)
TABLE B-IV (Continued)

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>109 J=MP1,7</td>
<td>CC 109 K=MP1,7</td>
</tr>
<tr>
<td>1090 L=MC*(J-1)+K</td>
<td>CALL M1NVI(ARRAY, MC, CETERM, MV1, MV2)</td>
</tr>
<tr>
<td>110 J=1,7</td>
<td>CALL CMTMUL(HT, DL77, CUMMY, 7, MD, MD)</td>
</tr>
<tr>
<td>111 J=1,7</td>
<td>CALL CMTMUL(CU77, CUMMY, W, 7, MD)</td>
</tr>
<tr>
<td>112 CALL XYTCR(STATEP(1), STATEP(2), STATEP(3), STATEP(4), STATEP(5), STATEP(6))</td>
<td></td>
</tr>
<tr>
<td>113 CUMMY(J,1)=0.</td>
<td></td>
</tr>
<tr>
<td>114 CUMMY(J,1)=CUMMY(J,1)+h(J,K)*DX(K)</td>
<td></td>
</tr>
</tbody>
</table>

CC 109 J=MP1,7
CC 109 K=MP1,7
1090 ARRAY(L)=CU77(J,K)
CALL MINV(ARRAY, MC, CETERM, MV1, MV2)
CC 1091 L=1,LMAX
J=(L-1)/MC+1
K=L-MC*(J-1)
10910 CU77(J,K)=ARRAY(L)
CALL CMTMUL(CU77, CUMMY, 7, MC, MD)
CC 110 J=1,7
CC 110 K=1,7
110 CU77(J,K)=5(J,K,NKM1)
CALL CMTMUL(CU77, CUMMY, W, 7, MD)
IF(PRINT)1102,1102,2219

C PRINT CUT PARTIAL DERIVATIVE MATRIX H=CR/DX, AND WEIGHTING MATRIX C EVALUATED AT PREDICTED STATE
2219 WRITE(6,2220)((H(J,K),K=1,7),J=1,7)
2220 FFORMAT(5X,*' MATRIX (CR/DX) EVALUATED AT PREDICTED STATE'/(7D17.8 11))
1101 WRITE(6,1111)((W(JJ,KK),KK=1,7),JJ=1,7)
1111 FFORMAT(5X,*'WEIGHTING MATRIX W'/(7D17.8))

C CALCULATE COVARIANCE MATRIX S(K) OF ESTIMATED STATE
1102 CALL CMTMUL(h,F,CUMMY,7,MC,7)
CC 117 J=1,7
CC 117 K=1,7
117 CUMMY(J,K)=UNIT(J,K)-CUMMY(J,K)
CC 118 J=1,7
CC 118 K=1,7
S(J,K,NK)=0.
CC 118 L=1,7
118 S(J,K,NK)=5(J,K,NK)+CUMMY(J,L)*S(L,K,NKM1)
IF(MDA1A)1056,2C, 112
1056 CC 1057 J=1,7
1057 ESTATE(J)=STATEP(J)
CC TC 1000
C CCMPUT: NEW ESTIMATE OF STATE VECTOR C
1112 CALL XYTCR(STATEP(1), STATEP(2), STATEP(3), STATEP(4), STATEP(5), STATEP(6))
STATE(7)=STATEP(7)
CC 113 K=1,MC
113 EX(K)=STATEP(K)-RSTATE(K)
IF(PRNT)1132,1132,1130
1130 WRITE(6,1131)((EX(J),J=1,7)
1131 FFORMAT(5X,*'Y-H(X)'/(7D17.8)
1132 CC 114 J=1,7
CUMMY(J,1)=0.
CC 114 K=1,MC
114 CUMMY(J,1)=CUMMY(J,1)+h(J,K)*DX(K)
CC 115 K=1,7
115 ESTATE(K)=STATEP(K)+CUMMY(K,1)

63
TABLE B-IV (Continued)

C CALCULATE HYBRID COVARIANCE MATRIX OF ESTIMATED STATE
C
1000 CC 1190 J=1,7
CC 1190 K=1,7
1190 CUMMY(J,K)=S(J,K,NK)
   CALL XCOVTR(DUMMY,ESTATE,DU77,7)
CC 1300 J=1,7
CC 1297 K=1,7
IF(DU77(J,J))1291,1291,1290
1290 IF(DU77(K,K))1291,1291,1290
1291 WRITE(6,1292)
1292 FORMAT(/5X,'***NEGATIVE DIAGONAL TERMS OF COVARIANCE MATRIX IN RAD
   1AR PLCLAR COORDINATES***'/8X,'DISREGARD HYBRID MATRIX PRINTED BELOW
   2'//)
   PRNT=1.
CC TC 1300
1294 CONTINUE
   CUMMY(J,K)=DU77(J,K)/DSQRT(DU77(J,J)*DU77(K,K))
   CORR=ABS(CUMMY(J,K))
   IF(CORR<1.00001)1297,1297,1298
1298 WRITE(6,1299)
1299 FORMAT(/5X,'***** CORRELATION COEFFICIENT EXCEEDS 1. *****'//)
   PRNT=1.
1297 CONTINUE
1300 CUMMY(J,J)=DSQRT(DU77(J,J))
C C STCRE DIAGCNAL ELEMENTS OF COVARIANCE MATRICES
C
1399 CC 1400 J=1,7
CC 1399 CEVR(J)=CUMMY(J,J)
1400 CEVX(J)=DSQRT(S(J,J,NK))
   IF(PRNT)1302,1401,1401
C C PRINT CUT STATE ESTIMATE ANC COVARIANCE MATRICES
C
1401 WRITE(6,1150) (ESTATE(J),J=1,7)
1150 FORMAT(/5X,'ESTIMATE OF STATE VECTOR'/'7D17.8)
1001 WRITE(6,1179)
1179 FORMAT(/5X,'FINAL ESTIMATE OF COVARIANCE MATRICES')
   WRITE(6,1183)((S(J,K,NK),K=1,7),J=1,7)
1183 FORMAT(/5X,'COVARIANCE MATRIX IN XYZ COORDINATES'/'7D17.8)
1300 WRITE(6,1301)((CUMMY(J,K),K=1,7),J=1,7)
1301 FORMAT(/5X,'HYBRID MATRIX IN RADAR COORDINATES'/'5X, 'DIAGCNAL TERMS
   1 ARE STANARC DEVIATIONS, OFF-DIAGONAL TERMS ARE CORRELATION CCEFF
   2NCIENTS'/'7D17.8)
C C TEST FCR MEMCRY CLAMPING
C
1302 IF(KLAMP)1055,1055,119
119 IF(AN-KLAMP)1055,1191,1191
1191 CC 1192 J=1,7
CC 1192 K=1,7
L=7*(J-1)+K
1192 ARRAY(L)=S(J,K,NK)
   CALL MINV(ARRAY,7,CEV1,CEV2)
   IF(AN-KLAMP)1055,1193,1194
1193 CELAST=DET
   GC TC 1055
C
64
### TABLE B-IV (Continued)

<table>
<thead>
<tr>
<th>C</th>
<th>SCALE COVARIANCE MATRIX</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SCALE=(DELAST/DET)**.142857143</td>
</tr>
<tr>
<td></td>
<td>DO 1195 J=1,7</td>
</tr>
<tr>
<td></td>
<td>DO 1195 K=1,7</td>
</tr>
<tr>
<td>1195</td>
<td>S(J,K,NK)=SCALE*S(J,K,NK)</td>
</tr>
<tr>
<td>1055</td>
<td>IF(NSTART)100,1200,100</td>
</tr>
<tr>
<td>1200</td>
<td>RETURN</td>
</tr>
<tr>
<td>200</td>
<td>WRITE (6,201)</td>
</tr>
<tr>
<td>201</td>
<td>FORMAT(//'****** MDATA=0, THIS SUBROUTINE SHOULD NOT HAVE BEEN CALLED ******///)</td>
</tr>
<tr>
<td></td>
<td>RETURN</td>
</tr>
<tr>
<td></td>
<td>END</td>
</tr>
</tbody>
</table>

65
SUBROUTINE ESTMAT(STATEM, STATEP, PHIMTX, ESTATE, DEVX, DEVR)

THIS VERSION OF ESTMAT INVERTS 7X7 COVARIANCE MATRICES IN THE COURSE OF CALCULATIONS

IMPLICIT REAL*8(A-H,O-Z)
REAL*8 STATEM, STATEP, PHIMTX, ESTATE
COMMON /ACCM/CCVAR(7), SIGMA(7)/FCOM/COORD, DLAT, PRNT/ICCM/KLAMP, MDALITAN
DIMENSION PHIMTX(7,7), DUMMY(7,7), DUM(7,7), DX(7), STATEP(7)
DIMENSION STATEM(7), ESTATE(7), H(7,7), R(7,7), S(7,7,2), RSTATE(7)
DIMENSION RINV(7,7), SI(7,7,2), HT(7,7), HTBACK(7,7)
DIMENSION CU77(7,7), MV1(7), MV2(7), ARRAY(49), SR(7,7,2), PHIT(7,7)
DIMENSION CEVX(7), CEVR(7)

C TEST FOR 1ST TIME THROUGH
C
IF(NN-1)1,100

1 NSTART=1
IF(MDATA)2,500,3
2 MD=-MDATA
GO TO 4
3 MD=MDATA
4 MPI=MD+1
5 NK=1
6 NK1=2
7 CC 7 J=1,7
8 CX(7)=0.
9 CC 7 K=1,7
10 CUMMY(J,K)=0.
11 CU77(J,K)=0.
12 W(J,K)=0.
13 H(J,K)=0.
14 HTBACK(J,K)=0.
15 HTBACK(J,K)=0.
6 R(J,K)=0.
7 SR(J,K,1)=0.
8 RINV(J,K)=0.
9 SI(J,K,1)=0.
10 SI(J,K,2)=0.
11 S(J,K,1)=0.
12 CC 6 J=1,MC
13 RINV(J,K)=1./SIGMA(J)**2
14 R(J,K)=SIGMA(J)**2
15 IF(COCU=1.)111,11,61
61 CC 8 J=1,7
62 CUMMY(J,J)=1./COVAK(J)**2
63 SR(J,J)=COVAR(J)**2
64 CALL XCOVTH(DUMMY, STATE, H, -1)
65 CC 9 J=1,7
66 CC 9 K=1,7
67 FT(K,J)=F(K,J)
68 CALL CMTMUL(FT, DUMMY, CU77, 7, 7, 7)
69 CALL CMTMUL(CU77, FT, DUMMY, 7, 7, 7)

100 RETURN

}
TABLE B-V (Continued)

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>10</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>10</td>
<td>SI</td>
<td>(J,K,1)=DUMMY(J,K)</td>
</tr>
<tr>
<td>100</td>
<td>WRITE</td>
<td>(6,1180)</td>
</tr>
<tr>
<td>1180</td>
<td>FORMAT</td>
<td>//5X,'CCVARIANCE MATRIX IN RADAR COORDINATES'/(7D17.8))</td>
</tr>
<tr>
<td>1180</td>
<td>WRITE</td>
<td>(6,2011)</td>
</tr>
<tr>
<td>11</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>11</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>12</td>
<td>S</td>
<td>S(J,J,1)=CCVAR(J)**2</td>
</tr>
<tr>
<td>12</td>
<td>SI</td>
<td>SI(J,J,1)=1./S(J,J,1)</td>
</tr>
<tr>
<td>13</td>
<td>CALL</td>
<td>XCCVTR(ESTATE,DUMMY,7)</td>
</tr>
<tr>
<td>14</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>14</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>14</td>
<td>S</td>
<td>SRIJ,K,1)=CUMMY(J,K)</td>
</tr>
<tr>
<td>14</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>14</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>14</td>
<td>S</td>
<td>SI(J,K,NK),K=1,7),J=1,7</td>
</tr>
<tr>
<td>16</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>16</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>16</td>
<td>S</td>
<td>S(J,J,1)=CCVAR(J)**2</td>
</tr>
<tr>
<td>16</td>
<td>SI</td>
<td>SI(J,J,1)=1./S(J,J,1)</td>
</tr>
<tr>
<td>20</td>
<td>CALL</td>
<td>XCCVTR(ESTATE,DUMMY,7)</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>20</td>
<td>S</td>
<td>SRIJ,K,1)=CUMMY(J,K)</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>20</td>
<td>S</td>
<td>SI(J,K,NK),K=1,7),J=1,7</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>20</td>
<td>S</td>
<td>SRIJ,K,1)=CUMMY(J,K)</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>J=1,7</td>
</tr>
<tr>
<td>20</td>
<td>CC</td>
<td>K=1,7</td>
</tr>
<tr>
<td>20</td>
<td>S</td>
<td>SI(J,K,NK),K=1,7),J=1,7</td>
</tr>
</tbody>
</table>

C UPDATE INDICES NK,NKM1

100 IF(NK-1)1102,103
102 AK=2
102 AKM1=1
103 NK=1
103 AKM1=1
104 ASTART=0
104 IF(PRNT)106,105,105
C PRINT CUT TRANSITION MATRIX

105 WRITE(6,200) | (PHIMTX(J,K),K=1,7),J=1,7 |
200 FORMAT'/5X,TRANSITION MATRIX USED IN ESTIMATION/(7D17.8))
C COMPUTE INVERSE CCVARIANCE MATRIX SII(K,K-1) IN XYZ COORDINATES

106 CC 2001 J=1,7
106 CC 2001 K=1,7
106 L=7*(J-1)+K
2001 ARRAY(L)=PHIMTX(J,K)
2001 CALL MINV(ARRAY,7,LETTER,MV1,MV2)
2001 CC 2002 L=1,49
2001 J=(L-1)/7+1
2001 K=L-7*(J-1)
2002 PHIMTX(J,K)=ARRAY(L)
2002 CC 201 J=1,7
2002 CC 201 K=1,7
2002 PHIT(J,K)=PHIMTX(K,J)
2002 CUMMY(J,K)=SI(J,K,NKM1)
2002 CALL DMTMUL(DUMMY,PHIMTX,DU77,7,7,7)
2002 CALL DMTMUL(PHIT,DU77,CUMMY,7,7,7)
2030 CALL XCCVTR(RINV,STATEP,+,-1)
2030 CC 203 J=1,7
2030 CC 203 K=1,7
2030 HT(J,K)=H(K,J)
2030 CALL DMTMUL(H+,RINV,+,7,7,7)
2030 IF(PRNT)2039,2035,2010

67
TABLE B-V (Continued)

C C CALCULATE HYBRID COVARIANCE MATRIX OF PREDICTED STATE
C
2010 DO 202 J=1,7
202   DU77(J,K)=DUMMY(J,K)
DO 203 J=1,7
203   CO 2003 K=1,7
L=7*(J-1)*K
2003 ARRAY(L)=DU77(J,K)
CALL MINV(ARRAY,7,DETERM,MV1,MV2)
DO 2004 L=1,49
2004   J=(L-1)/7+1
2004   K=L-7*(J-1)
2004   CALL XCOVTR(DU77,STATEP,HBACK,7)
DO 2015 J=1,7
2015   K=1,7
2015   HTBACK(J,K)=HBACK(J,K)/SQRT(HBACK(J,J)*HBACK(K,K))
2016   HTBACK(J,J)=SQRT(HBACK(J,J))
C C PRINT CUT COVARIANCE MATRIX S(K,K-1) OF PREDICTED STATE
C AND ITS INVERSE
C C PRINT CUT HYBRID COVARIANCE MATRIX OF PREDICTED STATE
C C PRINT CUT PARTIAL DERIVATIVE MATRIX H=DR/DX, AND HTRANS*CINV
EVALUATED AT PREDICTED STATE
C
WRITE (6,1059)
1059 FORMAT(/5X,'COVARIANCE MATRICES S(K,K-1)')
WRITE(6,2011)((DUMMY(J,K),K=1,7),J=1,7)
2011 FORMAT(/5X,'INVERSE MATRIX IN XYZ COORDINATES'/(7D17.8))
WRITE(6,1183)((DU77(J,K),K=1,7),J=1,7)
WRITE(6,1301)((HTBACK(J,K),K=1,7),J=1,7)
WRITE(6,2220)((H(J,K),K=1,7),J=1,7)
WRITE(6,2221)((W(J,K),K=1,7),J=1,7)
2220 FORMAT(/5X,'H MATRIX (ER/DX) EVALUATED AT PREDICTED STATE'/(7D17.8))
2221 FORMAT(/5X,'W MATRIX = HTRANS*CINV'/(7D17.8))
C C CALCULATE COVARIANCE MATRIX S(K) OF ESTIMATED STATE
C
2039 CALL DMULTM(W,X,DU77,7,7,7)
204   J=1,7
DO 205 K=1,7
205   DUMMY(J,K)=DUMMY(J,K)+DU77(J,K)
DO 204 J=1,7
2005 ARRAY(L)=DUMMY(J,K)
CALL MINV(ARRAY,7,DETERM,MV1,MV2)
DO 2006 L=1,49
2006   J=(L-1)/7+1
2006   K=L-7*(J-1)
2006   S(J,K,NK)=ARRAY(L)
2006   DUMMY (J,K)=ARRAY(L)
2006   CALL DMULTM(DUMMY,W,DUM,7,7,7)
IF(MDATA)1056,50C,1058

68
TABLE B-V (Continued)

1056 CC 1057 J=1,7
1057 ESTATE(J)=STATEP(J)
CC TC 1148

C
C COMPUTE ESTIMATE OF STATE VECTOR
C

1058 CALL XYTCR(STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATE
1P(6),RSTATE(1),RSTATE(2),RSTATE(3),RSTATE(4),RSTATE(5),RSTATE(6))
RSTATE(7)=STATEP(7)
CC 113 K=1,MC
113 C2(K)=STATEM(K)-RSTATE(K)
IF(PRNT)1132,1132,1130
1130 FORMAT(5X,C*,X-H(X)=*,7D17.8)
1132 CC 2061 J=1,7
1132 CU77(J,1)=C.
CC 2061 L=1,MC
2061 CU77(J,1)=CU77(J,1)+CUM(J,L)*DX(L)
CC 208 J=1,7
208 ESTATE(J)=STATEP(J)+CU77(J,1)

C
C CALCULATE HYBRID COVARIANCE MATRIX OF ESTIMATED STATE
C

1148 CC 1800 J=1,7
1800 CUMMY(J,K)=S(J,K,NK)
CALL XCOVTR(CUMMY,ESTATE,CU77,7)
CC 1300 J=1,7
CC 1297 K=1,7
IF(CU77(J,J)>1.00)1291,1291,1290
1290 IF(CU77(K,K)>1.00)1291,1291,1294
1291 WRITE(6,1291)
1292 FORMAT(15X,**NEGATIVE DIAGONAL TERMS OF COVARIANCE MATRIX IN RAD
1AR POLAR COORDINATES***'/7D17.8)
PRNT=1.
CC TC 1399
1294 CC CONTINUE
1294 CUMMY(J,K)=CU77(J,K)/DSQRT(CU77(J,J)*CU77(K,K))
CRR=ABS(CUMMY(J,K))
IF(CRR<1.00)1297,1297,1298
1298 WRITE(6,1298)
1299 FORMAT(7D17.8)
PRNT=1.
1297 CC CONTINUE
1300 CUMMY(J,J)=DSQRT(CU77(J,J))
C
C STOCH DIAGONAL ELEMENTS OF COVARIANCE MATRICES
C
1399 CC 1400 J=1,7
1400 CEVR(J)=CUMMY(J,J)
1400 CEVX(J)=ESQRT(S(J,J,NK))
IF(PRNT)1402,149,149
C
C PRINT CUT STATE ESTIMATE AND COVARIANCE MATRICES
C
1149 WRITE(6,1152)(CUM(J,K),K=1,7),J=1,7
1152 FORMAT(5X,WEIGHTING MATRIX S(K)*'/7D17.8)
1150 WRITE(6,1151) (ESTATE(J),J=1,7)
1151 FORMAT(7D17.8)

69
TABLE B-V (Continued)

```plaintext
WRITE(6,1179)
1179 FFORMAT(/5X,'FINAL ESTIMATE OF COVARIANCE MATRICES')
WRITE(6,1183)((S(J,K,NK),K=1,7),J=1,7)
WRITE(6,2011)((SI(J,K,NK),K=1,7),J=1,7)
1183 FFORMAT(/5X,'COVARIANCE MATRIX IN XYZ COORDINATES')
WRITE(6,1184) DET
1184 FFORMAT(/5X,'DETERMINANT=17.8')
WRITE(6,1301)((DUMMY(J,K),K=1,7),J=1,7)
1301 FFORMAT(/5X,'HYPERIC MATRIX IN RADAR COORDINATES')
CCVARIANCE MATRIX IN RADAR COORDINATES
DIAGONAL TERMS ARE STANDARD DEVIATIONS, OFF-DIAGONAL TERMS ARE CORRELATION COEFFICIENTS
I
TEST FCR MEMCRY CLAMPING
C
1302 IF(KLAMP)1055,1055,119
119 IF(NN-KLAMP)1055,1193,1194
1193 DELAST=DET
GC TO 1055
C
SCALE CCVARIANCE MATRIX
C
1194 SCALE=(DELAST/CET)**142857143
1195 CC 1195 J=1,7
CC 1195 K=1,7
1195 SI(J,K,NK)=SCALE*SI(J,K,NK)
1000 CONTINUE
1055 IF(NSTART)100,1200,100
1200 RETURN
500 WRITE (6,501)
501 FFORMAT(/5X,'*** MCATA =G, THIS SUBROUTINE SHOULD NOT HAVE BEEN CALLED ***')
RETURN
ENC
```
## Table B-VI

### Subroutine DENS

```plaintext
SUBROUTINE DENS
COMM FTCCM/CORC,CLAT,PRNT
REAL HKMFT, RFC, CTJCRC, DLAT, PRNT
DIMENSION FTKM(100),RHCM(100),HTKFT(100),RHCE(100)

C SET UP ALTITUDE ARRAY HTKM, IN KILOMETERS
CC 3 I=1,51
    HTKM(I)=2.0*FLCAT(I-1)
    CONTINUE
CC 4 J=1,10
    HTKM(J+51)=U0.0+FCAT(J-1)*10.0
    CONTINUE

C SET UP ATMOSPHERIC DENSITY ARRAY RHCM, IN KG/METER**3
C
    RHCM(1) = 1.2250
    RHCM(2) = 1.0066
    RHCM(3) = 8.1935E-1
    RHCM(4) = 6.6011E-1
    RHCM(5) = 5.2579E-1
    RHCM(6) = 4.1351E-1
    RHCM(7) = 3.1194E-1
    RHCM(8) = 2.2786E-1
    RHCM(9) = 1.6647E-1
    RHCM(10) = 1.2165E-1
    RHCM(11) = 8.9106E-2
    RHCM(12) = 6.4510E-2
    RHCM(13) = 4.6938E-2
    RHCM(14) = 3.4257E-2
    RHCM(15) = 2.5076E-2
    RHCM(16) = 1.8410E-2
    RHCM(17) = 1.3556E-2
    RHCM(18) = 9.6874E-3
    RHCM(19) = 7.2578E-3
    RHCM(20) = 5.3666E-3
    RHCM(21) = 3.7957E-3
    RHCM(22) = 2.4946E-3
    RHCM(23) = 2.2584E-3
    RHCM(24) = 1.7141E-3
    RHCM(25) = 1.3187E-3
    RHCM(26) = 1.0269E-3
    RHCM(27) = 8.8970E-4
    RHCM(28) = 6.3137E-4
    RHCM(29) = 4.9762E-4
    RHCM(30) = 3.9086E-4
    RHCM(31) = 3.0592E-4
    RHCM(32) = 2.3531E-4
    RHCM(33) = 1.8837E-4
    RHCM(34) = 1.4713E-4
    RHCM(35) = 1.1395E-4
    RHCM(36) = 8.7835E-5
    RHCM(37) = 6.6593E-5
    RHCM(38) = 5.0151E-5
    RHCM(39) = 3.7363E-5
    RHCM(40) = 2.1705E-5
    RHCM(41) = 1.4954E-5
    RHCM(42) = 1.0525E-5
    RHCM(43) = 7.062E-6
    RHCM(44) = 6.0176E-6
```

71
<table>
<thead>
<tr>
<th>TABLE B-VI (Continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHCM(45)=4.579E-6</td>
</tr>
<tr>
<td>RHCM(46)=3.170E-6</td>
</tr>
<tr>
<td>RHCM(47)=2.137E-6</td>
</tr>
<tr>
<td>RHCM(48)=1.459E-6</td>
</tr>
<tr>
<td>RHCM(49)=1.008E-6</td>
</tr>
<tr>
<td>RHCM(50)=7.044E-7</td>
</tr>
<tr>
<td>RHCM(51)=4.974E-7</td>
</tr>
<tr>
<td>RHCM(52)=9.829E-8</td>
</tr>
<tr>
<td>RHCM(53)=2.436E-8</td>
</tr>
<tr>
<td>RHCM(54)=7.589E-9</td>
</tr>
<tr>
<td>RHCM(55)=3.394E-9</td>
</tr>
<tr>
<td>RHCM(56)=1.836E-9</td>
</tr>
<tr>
<td>RHCM(57)=1.159E-9</td>
</tr>
<tr>
<td>RHCM(58)=8.036E-10</td>
</tr>
<tr>
<td>RHCM(59)=5.858E-10</td>
</tr>
<tr>
<td>RHCM(60)=4.347E-10</td>
</tr>
<tr>
<td>RHCM(61)=3.318E-10</td>
</tr>
</tbody>
</table>

**SET UP ALTITUDE ARRAY HKFT, IN KFT**

<table>
<thead>
<tr>
<th>C</th>
<th>CC</th>
<th>FG</th>
<th>EI=1,51</th>
<th>CONTINUE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>1</td>
<td>I=1,51</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HKFT(I)=6.0*FLOAT(I-1)</td>
<td></td>
</tr>
</tbody>
</table>

**CONTINUE**

<table>
<thead>
<tr>
<th>C</th>
<th>CC</th>
<th>FG</th>
<th>EI=1,10</th>
<th>CONTINUE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
<td>1</td>
<td>J=1,10</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HKFT(J+51)=330.0*FLOAT(J-1)*30.0</td>
<td></td>
</tr>
</tbody>
</table>

**SET UP ATMOSPHERIC DENSITY ARRAY,RHCE, IN LBS/FT**

<table>
<thead>
<tr>
<th>C</th>
<th>CC</th>
<th>FG</th>
<th>EI=7.647E-2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td></td>
<td>RHCE(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.647E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6.3925E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5.3022E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4.3666E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.5531E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.8657E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.2853E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.7010E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.2848E-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9.6701E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.2589E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(12)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5.4485E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4.0624E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(14)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.0368E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.2759E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(16)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.7100E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.2879E-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(18)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9.7241E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.3188E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(20)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5.4944E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(21)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4.5106E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(22)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.1543E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(23)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.4110E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(24)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.8530E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(25)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4317E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(26)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.1119E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(27)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8.6943E-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(28)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6.9261E-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(29)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5.5183E-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(30)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4.4159E-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHCE(31)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.5576E-5</td>
</tr>
</tbody>
</table>

72
<table>
<thead>
<tr>
<th>ENTRY</th>
<th>AT(/HRMFT,RHC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 IF(HKMFT=1.02)11,11,15</td>
<td></td>
</tr>
<tr>
<td>11 L=HKMFT/2.00+1.00</td>
<td></td>
</tr>
<tr>
<td>12 A=(HKMFT-HTKM(L))/(HTKM(L+1)-HTKM(L))</td>
<td></td>
</tr>
<tr>
<td>14 RHO=RHOM(L)*(RHOM(L+1)/RHOM(L))**A</td>
<td></td>
</tr>
<tr>
<td>15 IF(HKMFT=2.02)17,18,16</td>
<td></td>
</tr>
<tr>
<td>17 L=51.000001*(HKMFT-1.02)/1.01</td>
<td></td>
</tr>
<tr>
<td>18 GO TO 12</td>
<td></td>
</tr>
<tr>
<td>18 RHO=RHOM(61)</td>
<td></td>
</tr>
<tr>
<td>19 RETURN</td>
<td></td>
</tr>
</tbody>
</table>

**DATA IN ENGLISH UNITS**

<table>
<thead>
<tr>
<th>ENTRY</th>
<th>AT(/HRMFT,RHC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>21 IF(HKMFT=3.02)21,21,25</td>
<td></td>
</tr>
<tr>
<td>21 L=HKMFT/6.00+1.0000001</td>
<td></td>
</tr>
<tr>
<td>22 A=(HKMFT-HTKFT(L))/(HTKFT(L+1)-HTKFT(L))</td>
<td></td>
</tr>
<tr>
<td>24 RHO=RHOE(L)*(RHOE(L+1)/RHOE(L))**A</td>
<td></td>
</tr>
<tr>
<td>25 IF(HKMFT=6.02)26,27,16</td>
<td></td>
</tr>
<tr>
<td>26 L=51.000001*(HKMFT-3.02)/3.01</td>
<td></td>
</tr>
</tbody>
</table>
TABLE B-VI (Continued)

<table>
<thead>
<tr>
<th>GC</th>
<th>TC</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>RF</td>
<td>=RHC E(61)</td>
</tr>
<tr>
<td>16</td>
<td>RETURN</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>WRITE (6, 130)</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>FCPMAT (5x,***** EARTH IMPACT *****)</td>
<td></td>
</tr>
<tr>
<td>RETURN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENC</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TABLE B-VII

SUBROUTINE XYTOR

SUBROUTINE XYTOR(X, Y, Z, XDOT, YDOT, ZDOT, R, A, E, RDOT, ADOT, EDOT)

SUBROUTINE TC CONVERT FROM XYZ COORDINATES TO RADAR POLAR COORDINATES

REAL*8 X, Y, Z, XDOT, YDOT, ZDOT, R, A, E, RDOT, ADOT, EDOT
IMPLICIT REAL*8 (A-H, O-Z)
R = DSQRT (X**2 + Y**2 + Z**2)
E = DARSIN (Z/R)
A = DATA (X, Y)
RDOT = (X*XDGT + Y*YDOT + Z*ZDOT)/R
ADGT = (Y*XDCT - X*YDGT)/(X**2 + Y**2)
DENOM = DABS (R**2 - Z**2)
EDLT = (Z*ZDCT - Z+KDOT)/(DSQRT (DENOM)*R)
RETURN
END

TABLE B-VIII

SUBROUTINE RTOXY

SUBROUTINE RTOXY (R, A, E, RDOT, ADOT, EDOT, X, Y, Z, XDOT, YDOT, ZDOT)

SUBROUTINE TC CONVERT FROM RADAR COORDINATES TO XYZ COORDINATES

REAL*8 K, A, E, RDOT, ADOT, EDOT, X, Y, Z, XDOT, YDOT, ZDOT
IMPLICIT REAL*8 (A-H, O-Z)
SAZ = DSI(N (A)
SEL = DSI(N (C)
CAZ = DCOS (A)
CEL = DCOS (E)
X = R*CEL*SAZ
Y = R*CEL*CAZ
Z = R*SEL
RPR = RDOT/R
XDOT = RPR*X - ELDOT*Z - SAZ*ADOT*Y
YDOT = RPR*Y - ELDOT*Z - CAZ*ADOT*X
ZDOT = RPR*Z + ELDOT*R*CEL
RETURN
END
<table>
<thead>
<tr>
<th>TABLE B-IX</th>
<th>SUBROUTINE XCOVTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBROUTINE XCOVTR(XCOV,XVCTR,RCOV,NCODE)</td>
<td></td>
</tr>
<tr>
<td>REAL*8 XCOV,XVCTR,RCOV</td>
<td></td>
</tr>
<tr>
<td>IMPLICIT REAL*8(A-H,O-Z)</td>
<td></td>
</tr>
<tr>
<td>DIMENSION XVCTR(7)</td>
<td></td>
</tr>
<tr>
<td>DIMENSION XCOV(7,7),RCOV(7,7),RVCTR(7)</td>
<td></td>
</tr>
<tr>
<td>DIMENSION CONVT(7,7),DUM1(7,7)</td>
<td></td>
</tr>
<tr>
<td>X = XVCTR(1)</td>
<td></td>
</tr>
<tr>
<td>Y = XVCTR(2)</td>
<td></td>
</tr>
<tr>
<td>Z = XVCTR(3)</td>
<td></td>
</tr>
<tr>
<td>XP = XVCTR(4)</td>
<td></td>
</tr>
<tr>
<td>YP = XVCTR(5)</td>
<td></td>
</tr>
<tr>
<td>ZP = XVCTR(6)</td>
<td></td>
</tr>
<tr>
<td>RVCTR(7) = XVCTR(7)</td>
<td></td>
</tr>
<tr>
<td>CALL XTORI(X,Y,Z,XP,YP,ZP,RVCTR(1),RVCTR(2),RVCTR(3),RVCTR(4),RVCTR(5),RVCTR(6))</td>
<td></td>
</tr>
<tr>
<td>RVSQ = RVCTR(1)**2</td>
<td></td>
</tr>
<tr>
<td>RSG = X<strong>2 + Y</strong>2</td>
<td></td>
</tr>
<tr>
<td>RXY = SQRT(RSG)</td>
<td></td>
</tr>
<tr>
<td>DO 1 J = 1, 7</td>
<td></td>
</tr>
<tr>
<td>DO 1 K = 1, 7</td>
<td></td>
</tr>
<tr>
<td>RCOV(J,K) = 0.</td>
<td></td>
</tr>
<tr>
<td>DO 1 CONV(J,K) = 0.</td>
<td></td>
</tr>
<tr>
<td>CONV(1,1) = X/RVCTR(1)</td>
<td></td>
</tr>
<tr>
<td>CONV(1,2) = Y/RVCTR(1)</td>
<td></td>
</tr>
<tr>
<td>CONV(2,1) = Y/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(2,2) = -X/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(3,1) = -Z/(RXY*RVSQ)</td>
<td></td>
</tr>
<tr>
<td>CONV(3,2) = -Y*(RXY*RVSQ)</td>
<td></td>
</tr>
<tr>
<td>CONV(3,3) = -Z/(RXY*RVSQ)</td>
<td></td>
</tr>
<tr>
<td>CONV(4,1) = (RVCTR(1)*XP - RVCTR(4)*X)/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(4,2) = (RVCTR(1)*YP - RVCTR(4)*Y)/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(4,3) = (RVCTR(1)*ZP - RVCTR(4)*Z)/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(5,1) = -(2.00<em>X</em>RVCTR(5)+YP)/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(5,2) = -(XP-2.00<em>Y</em>RVCTR(5))/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(5,3) = -(ZP-2.00<em>X</em>RVCTR(5))/RVSQ</td>
<td></td>
</tr>
<tr>
<td>CONV(6,1) = -X<em>RVCTR(6)/RVSQ - 2.00</em>RVCTR(4)*CONV(3,1)/RVCTR(1)</td>
<td></td>
</tr>
<tr>
<td>CONV(6,2) = -Y<em>RVCTR(6)/RVSQ - 2.00</em>RVCTR(4)*CONV(3,2)/RVCTR(1)</td>
<td></td>
</tr>
<tr>
<td>CONV(6,3) = -Z<em>RVCTR(6)/RVSQ - 2.00</em>RVCTR(4)*CONV(3,3)/RVCTR(1)</td>
<td></td>
</tr>
<tr>
<td>DO 2 J = 4, 6</td>
<td></td>
</tr>
<tr>
<td>DO 2 K = 4, 6</td>
<td></td>
</tr>
<tr>
<td>2 CONV(J,K) = CONV(J-3,K-3)</td>
<td></td>
</tr>
<tr>
<td>IF (NCODE) 100, 20, 20</td>
<td></td>
</tr>
<tr>
<td>20 CONTINUE</td>
<td></td>
</tr>
<tr>
<td>DO 3 J = 1, 7</td>
<td></td>
</tr>
<tr>
<td>DO 3 K = 1, 7</td>
<td></td>
</tr>
<tr>
<td>3 CONV(J,K) = CONV(K,J)</td>
<td></td>
</tr>
<tr>
<td>CALL DMtmul(XCOV,CONVT,DUM1,7,7,7)</td>
<td></td>
</tr>
<tr>
<td>CALL DMtmul(CONV,DUM1,RCOV,7,7,7)</td>
<td></td>
</tr>
<tr>
<td>RETURN</td>
<td></td>
</tr>
<tr>
<td>100 CONTINUE</td>
<td></td>
</tr>
<tr>
<td>DO 101 J = 1, 7</td>
<td></td>
</tr>
<tr>
<td>DO 101 K = 1, 7</td>
<td></td>
</tr>
<tr>
<td>101 RCOV(J,K) = CONV(J,K)</td>
<td></td>
</tr>
<tr>
<td>RETURN</td>
<td></td>
</tr>
<tr>
<td>END</td>
<td></td>
</tr>
<tr>
<td>TABLE B-X</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>ROUTINE RCOVTX</td>
<td></td>
</tr>
</tbody>
</table>

```fortran
SUBROUTINE RCOVTX(COVX, XSTATE, COVR)
REAL*8 COVR, ESTATE, XSTATE, COVX
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION COVR(7,7), COVX(7,7), ESTATE(7), DXDR(7,7), DUMMY(7,7)

CALL XYTOR(XSTATE(1), XSTATE(2), XSTATE(3), XSTATE(4), XSTATE(5), XSTATE(6))
ESTATE(7) = XSTATE(7)

DO 200 J = 1, 7
DO 200 K = 1, 7
200 DXDR(J, K) = 0.
SAZ = DSIN(ESTATE(2))
CAZ = DCOS(ESTATE(2))
SEL = DSIN(ESTATE(3))
CEL = DCOS(ESTATE(3))

CONVERT COVARIANCE MATRIX TO XYZ COORDINATES
DXDR(1, 1) = CEL * SAZ
DXDR(1, 2) = ESTATE(1) * CEL * CAZ
DXDR(1, 3) = -ESTATE(1) * SEL * SAZ
DXDR(2, 1) = CEL * CAZ
DXDR(2, 2) = -ESTATE(1) * CEL * SAZ
DXDR(2, 3) = -ESTATE(1) * SEL * CAZ
DXDR(3, 1) = SEL
DXDR(3, 2) = 0.
DXDR(3, 3) = ESTATE(1) * CEL
DXDR(4, 1) = ESTATE(5) * CEL * CAZ - ESTATE(6) * SEL * SAZ
DXDR(4, 2) = ESTATE(4) * CEL * CAZ - ESTATE(5) * SEL * SAZ
DXDR(4, 3) = ESTATE(4) * SEL * SAZ - ESTATE(5) * CEL + ESTATE(6) * SEL
DXDR(5, 1) = ESTATE(6) * SEL * CAZ
DXDR(5, 2) = ESTATE(5) * SEL * SAZ - ESTATE(6) * CEL
DXDR(5, 3) = ESTATE(5) * SEL * CAZ - ESTATE(6) * CEL
DXDR(6, 1) = ESTATE(6) * CEL
DXDR(6, 2) = 0.
DXDR(6, 3) = ESTATE(4) * CEL - ESTATE(6) * ESTATE(1) * SEL

DO 300 J = 4, 6
DO 300 K = 4, 6
300 DXDR(J, K) = DXDR(J - 3, K - 3)

DXDR(7, 7) = 1.
DO 301 J = 1, 7
DO 301 K = 1, 7
DUMMY(J, K) = 0.
DO 301 L = 1, 7
301 DUMMY(J, K) = DUMMY(J, K) + COVR(J, L) * DXDR(K, L)

DO 302 J = 1, 7
DO 302 K = 1, 7
COVX(J, K) = 0.
DO 302 L = 1, 7
302 COVX(J, K) = COVX(J, K) + DXDR(J, L) * DUMMY(L, K)
RETURN
END
```
### TABLE B-XI
**SUBROUTINE GAUSSN**

```fortran
SUBROUTINE GAUSSN(NS, RN, SIGMA)
REAL*8 RN, SIGMA
DIMENSION RN(1)
CC 9 J=1,NS
  3 V=RAN(X)
    Y=-ALOG(V)
    V=RAN(X)
    Z=-ALOG(V)
    IF((Y-1.)**2-2.*Z)6,6,3
  6 S=RAN(X)
    IF(S<-.5)7,6,8
  7 RN(J)=Y*SIGMA
    CC TO 9
  8 RN(J)=-Y*SIGMA
9 CONTINUE
RETURN
END
```

### TABLE B-XII
**SUBROUTINE DMTMUL**

```fortran
SUBROUTINE DMTMUL(A, B, AB, NRA, NCA, NCB)
DOUBLE PRECISION A, B, AB
DIMENSION A(7,7), B(7,7), AB(7,7)
CC 1 J=1,NRA
CC 1 K=1,NCB
AB(J,K)=0.
CC 1 L=1,NCA
1 AB(J,K)=AB(J,K)+A(J,L)*B(L,K)
RETURN
END
```
## TABLE B-XIII
### SUBROUTINE MINV

<table>
<thead>
<tr>
<th>C</th>
<th>MINV 001</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MINV 002</td>
</tr>
<tr>
<td></td>
<td>MINV 003</td>
</tr>
<tr>
<td></td>
<td>MINV 004</td>
</tr>
<tr>
<td></td>
<td>MINV 005</td>
</tr>
<tr>
<td></td>
<td>MINV 006</td>
</tr>
<tr>
<td></td>
<td>MINV 007</td>
</tr>
<tr>
<td></td>
<td>MINV 008</td>
</tr>
<tr>
<td></td>
<td>MINV 009</td>
</tr>
<tr>
<td></td>
<td>MINV 010</td>
</tr>
<tr>
<td></td>
<td>MINV 011</td>
</tr>
<tr>
<td></td>
<td>MINV 012</td>
</tr>
<tr>
<td></td>
<td>MINV 013</td>
</tr>
<tr>
<td></td>
<td>MINV 014</td>
</tr>
<tr>
<td></td>
<td>MINV 015</td>
</tr>
<tr>
<td></td>
<td>MINV 016</td>
</tr>
<tr>
<td></td>
<td>MINV 017</td>
</tr>
<tr>
<td></td>
<td>MINV 018</td>
</tr>
<tr>
<td></td>
<td>MINV 019</td>
</tr>
<tr>
<td></td>
<td>MINV 020</td>
</tr>
<tr>
<td></td>
<td>MINV 021</td>
</tr>
<tr>
<td></td>
<td>MINV 022</td>
</tr>
<tr>
<td></td>
<td>MINV 023</td>
</tr>
<tr>
<td></td>
<td>MINV 024</td>
</tr>
<tr>
<td></td>
<td>MINV 025</td>
</tr>
<tr>
<td></td>
<td>MINV 026</td>
</tr>
<tr>
<td></td>
<td>MINV 027</td>
</tr>
<tr>
<td></td>
<td>MINV 028</td>
</tr>
<tr>
<td></td>
<td>MINV 029</td>
</tr>
<tr>
<td></td>
<td>MINV 030</td>
</tr>
<tr>
<td></td>
<td>MINV 031</td>
</tr>
<tr>
<td></td>
<td>MINV 032</td>
</tr>
<tr>
<td></td>
<td>MINV 033</td>
</tr>
<tr>
<td></td>
<td>MINV 034</td>
</tr>
<tr>
<td></td>
<td>MINV 035</td>
</tr>
<tr>
<td></td>
<td>MINV 036</td>
</tr>
<tr>
<td></td>
<td>MINV 037</td>
</tr>
<tr>
<td></td>
<td>MINV 038</td>
</tr>
<tr>
<td></td>
<td>MINV 039</td>
</tr>
<tr>
<td></td>
<td>MINV 040</td>
</tr>
<tr>
<td></td>
<td>MINV 041</td>
</tr>
<tr>
<td></td>
<td>MINV 042</td>
</tr>
<tr>
<td></td>
<td>MINV 043</td>
</tr>
<tr>
<td></td>
<td>MINV 044</td>
</tr>
<tr>
<td></td>
<td>MINV 045</td>
</tr>
<tr>
<td></td>
<td>MINV 046</td>
</tr>
<tr>
<td></td>
<td>MINV 047</td>
</tr>
<tr>
<td></td>
<td>MINV 048</td>
</tr>
<tr>
<td></td>
<td>MINV 049</td>
</tr>
<tr>
<td></td>
<td>MINV 050</td>
</tr>
<tr>
<td></td>
<td>MINV 051</td>
</tr>
<tr>
<td></td>
<td>MINV 052</td>
</tr>
<tr>
<td></td>
<td>MINV 053</td>
</tr>
<tr>
<td></td>
<td>MINV 054</td>
</tr>
<tr>
<td></td>
<td>MINV 055</td>
</tr>
<tr>
<td></td>
<td>MINV 056</td>
</tr>
<tr>
<td></td>
<td>MINV 057</td>
</tr>
<tr>
<td></td>
<td>MINV 058</td>
</tr>
<tr>
<td></td>
<td>MINV 059</td>
</tr>
<tr>
<td></td>
<td>MINV 060</td>
</tr>
<tr>
<td></td>
<td>MINV 061</td>
</tr>
</tbody>
</table>

**SUBROUTINE MINV**

**PURPOSE**

INVERT A MATRIX

**USAGE**

CALL MINVI(A,N,0,L,M)

**DESCRIPTION OF PARAMETERS**

A - INPUT MATRIX, DESTROYED IN COMPUTATION AND REPLACED BY RESULTANT INVERSE.

N - ORDER OF MATRIX A

C - RESULTANT DETERMINANT

L - WORK VECTOR OF LENGTH N

M - WORK VECTOR OF LENGTH N

**REMARKS**

MATRIX A MUST BE A GENERAL MATRIX

**SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED**

NONE

**METHOD**

THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT THE MATRIX IS SINGULAR.

**DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED,** THE C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION STATEMENT WHICH FOLLOWS.

THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS ROUTINE.

THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS. ABS IN STATEMENT 10 MUST BE CHANGED TO DBABS.

**SEARCH FOR LARGEST ELEMENT**

C=1.0

K=1,N

L(K)=K

K(K)=K

78
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
</table>
| KK=KK
BIGA=A(KK)
CC  20 J=K,
IZ=J*(J-1)
CC  20 J=K, |
|      |             |
| 10   | IF(CABS(BIGA)-CABS(A(IJ))) 15,20,20 |
| 15   | BIGA=A(IJ)
L(K)=I
M(K)=J |
| 20   | CC CONTINUE |
| C    | INTERCHANGE ROWS |
|      | J=L(K)
IF(J-K) 35,35,25 |
| 25   | CC  30 I=1,
KI=KI+N
HCLC=-A(KI)
JI=KI-K&J
A(KI)=A(JI) |
| 30   | A(JI) =-HCLC |
| C    | INTERCHANGE COLUMNS |
|      | 35 I=M(K)
IF(I-K) 45,45,38 |
| 38   | JP=N*(I-1)
CC  40 J=1,
JK=JK&J
JI=J&J
HCLC=-A(JK)
A(JK)=A(JI) |
| 40   | A(JI) =HCLC |
| C    | DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS |
|      | CONTAINED IN BIGA) |
|      | 45 IF(BIGA) 48,46,48 |
| 46   | C=0.0
RETURN |
| 48   | CC  55 I=1,
IF(I-K) 50,55,50 |
| 50   | IK=IK&I
A(IK)=A(IK)/(-BIGA) |
| 55   | CC CONTINUE |
| C    | REDUCE MATRIX |
|      | CC  65 I=1,
JK=JK&J
IJ=I&J
EC  65 J=1, |
| 60   | IF(I-K) 60,65,60 |
| 62   | IF(J-K) 62,65,62 |
| 65   | CC CONTINUE |
TABLE B-XIII (Continued)

<table>
<thead>
<tr>
<th>Line</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>C</td>
</tr>
<tr>
<td>5</td>
<td>C</td>
</tr>
<tr>
<td>6</td>
<td>C</td>
</tr>
<tr>
<td>7</td>
<td>C</td>
</tr>
<tr>
<td>8</td>
<td>C</td>
</tr>
<tr>
<td>9</td>
<td>C</td>
</tr>
<tr>
<td>10</td>
<td>C</td>
</tr>
<tr>
<td>11</td>
<td>C</td>
</tr>
<tr>
<td>12</td>
<td>C</td>
</tr>
<tr>
<td>13</td>
<td>C</td>
</tr>
<tr>
<td>14</td>
<td>C</td>
</tr>
<tr>
<td>15</td>
<td>C</td>
</tr>
<tr>
<td>16</td>
<td>C</td>
</tr>
<tr>
<td>17</td>
<td>C</td>
</tr>
<tr>
<td>18</td>
<td>C</td>
</tr>
<tr>
<td>19</td>
<td>C</td>
</tr>
<tr>
<td>20</td>
<td>C</td>
</tr>
<tr>
<td>21</td>
<td>C</td>
</tr>
<tr>
<td>22</td>
<td>C</td>
</tr>
<tr>
<td>23</td>
<td>C</td>
</tr>
<tr>
<td>24</td>
<td>C</td>
</tr>
<tr>
<td>25</td>
<td>C</td>
</tr>
<tr>
<td>26</td>
<td>C</td>
</tr>
<tr>
<td>27</td>
<td>C</td>
</tr>
<tr>
<td>28</td>
<td>C</td>
</tr>
<tr>
<td>29</td>
<td>C</td>
</tr>
<tr>
<td>30</td>
<td>C</td>
</tr>
<tr>
<td>31</td>
<td>C</td>
</tr>
<tr>
<td>32</td>
<td>C</td>
</tr>
<tr>
<td>33</td>
<td>C</td>
</tr>
<tr>
<td>34</td>
<td>C</td>
</tr>
<tr>
<td>35</td>
<td>C</td>
</tr>
<tr>
<td>36</td>
<td>C</td>
</tr>
<tr>
<td>37</td>
<td>C</td>
</tr>
<tr>
<td>38</td>
<td>C</td>
</tr>
<tr>
<td>39</td>
<td>C</td>
</tr>
<tr>
<td>40</td>
<td>C</td>
</tr>
<tr>
<td>41</td>
<td>C</td>
</tr>
<tr>
<td>42</td>
<td>C</td>
</tr>
<tr>
<td>43</td>
<td>C</td>
</tr>
<tr>
<td>44</td>
<td>C</td>
</tr>
<tr>
<td>45</td>
<td>C</td>
</tr>
<tr>
<td>46</td>
<td>C</td>
</tr>
<tr>
<td>47</td>
<td>C</td>
</tr>
<tr>
<td>48</td>
<td>C</td>
</tr>
<tr>
<td>49</td>
<td>C</td>
</tr>
<tr>
<td>50</td>
<td>C</td>
</tr>
<tr>
<td>51</td>
<td>C</td>
</tr>
<tr>
<td>52</td>
<td>C</td>
</tr>
<tr>
<td>53</td>
<td>C</td>
</tr>
<tr>
<td>54</td>
<td>C</td>
</tr>
<tr>
<td>55</td>
<td>C</td>
</tr>
<tr>
<td>56</td>
<td>C</td>
</tr>
<tr>
<td>57</td>
<td>C</td>
</tr>
<tr>
<td>58</td>
<td>C</td>
</tr>
<tr>
<td>59</td>
<td>C</td>
</tr>
<tr>
<td>60</td>
<td>C</td>
</tr>
<tr>
<td>61</td>
<td>C</td>
</tr>
<tr>
<td>62</td>
<td>C</td>
</tr>
<tr>
<td>63</td>
<td>C</td>
</tr>
<tr>
<td>64</td>
<td>C</td>
</tr>
<tr>
<td>65</td>
<td>C</td>
</tr>
<tr>
<td>66</td>
<td>C</td>
</tr>
<tr>
<td>67</td>
<td>C</td>
</tr>
<tr>
<td>68</td>
<td>C</td>
</tr>
<tr>
<td>69</td>
<td>C</td>
</tr>
<tr>
<td>70</td>
<td>C</td>
</tr>
<tr>
<td>71</td>
<td>C</td>
</tr>
<tr>
<td>72</td>
<td>C</td>
</tr>
<tr>
<td>73</td>
<td>C</td>
</tr>
<tr>
<td>74</td>
<td>C</td>
</tr>
<tr>
<td>75</td>
<td>C</td>
</tr>
<tr>
<td>76</td>
<td>C</td>
</tr>
<tr>
<td>77</td>
<td>C</td>
</tr>
<tr>
<td>78</td>
<td>C</td>
</tr>
<tr>
<td>79</td>
<td>C</td>
</tr>
<tr>
<td>80</td>
<td>C</td>
</tr>
<tr>
<td>81</td>
<td>C</td>
</tr>
<tr>
<td>82</td>
<td>C</td>
</tr>
<tr>
<td>83</td>
<td>C</td>
</tr>
<tr>
<td>84</td>
<td>C</td>
</tr>
<tr>
<td>85</td>
<td>C</td>
</tr>
<tr>
<td>86</td>
<td>C</td>
</tr>
<tr>
<td>87</td>
<td>C</td>
</tr>
<tr>
<td>88</td>
<td>C</td>
</tr>
<tr>
<td>89</td>
<td>C</td>
</tr>
<tr>
<td>90</td>
<td>C</td>
</tr>
<tr>
<td>91</td>
<td>C</td>
</tr>
<tr>
<td>92</td>
<td>C</td>
</tr>
<tr>
<td>93</td>
<td>C</td>
</tr>
<tr>
<td>94</td>
<td>C</td>
</tr>
<tr>
<td>95</td>
<td>C</td>
</tr>
<tr>
<td>96</td>
<td>C</td>
</tr>
<tr>
<td>97</td>
<td>C</td>
</tr>
<tr>
<td>98</td>
<td>C</td>
</tr>
<tr>
<td>99</td>
<td>C</td>
</tr>
<tr>
<td>100</td>
<td>K=K-1</td>
</tr>
<tr>
<td>101</td>
<td>IF(K) 150,150,105</td>
</tr>
<tr>
<td>102</td>
<td>105 I=L(K)</td>
</tr>
<tr>
<td>103</td>
<td>IF(I-K) 120,120,108</td>
</tr>
<tr>
<td>104</td>
<td>108 JC=N*(K-1)</td>
</tr>
<tr>
<td>105</td>
<td>JR=N*(I-1)</td>
</tr>
<tr>
<td>106</td>
<td>CC 110 J=1,N</td>
</tr>
<tr>
<td>107</td>
<td>JK=JCLJ</td>
</tr>
<tr>
<td>108</td>
<td>FCLC=A(JK)</td>
</tr>
<tr>
<td>109</td>
<td>J1=JR+J</td>
</tr>
<tr>
<td>110</td>
<td>A(JI)=A(J1)</td>
</tr>
<tr>
<td>111</td>
<td>110 A(JI) =HCLC</td>
</tr>
<tr>
<td>112</td>
<td>120 J=K(K)</td>
</tr>
<tr>
<td>113</td>
<td>IF(J-K) 100,100,125</td>
</tr>
<tr>
<td>114</td>
<td>125 KI=K-N</td>
</tr>
<tr>
<td>115</td>
<td>CC 130 I=1,N</td>
</tr>
<tr>
<td>116</td>
<td>KI=K1&amp;N</td>
</tr>
<tr>
<td>117</td>
<td>HCLC=A(K1)</td>
</tr>
<tr>
<td>118</td>
<td>J1=K1-KLJ</td>
</tr>
<tr>
<td>119</td>
<td>A(K1)=-A(J1)</td>
</tr>
<tr>
<td>120</td>
<td>130 A(J1) =HCLC</td>
</tr>
<tr>
<td>121</td>
<td>150 RETURN</td>
</tr>
<tr>
<td>122</td>
<td>GC TO 100</td>
</tr>
<tr>
<td>123</td>
<td>160 END</td>
</tr>
</tbody>
</table>
A FORTRAN IV program package has been written for the generation, analysis, and estimation of re-entry trajectories. The following functions may be performed by the program:

1. Generation of simulated noiseless trajectory data by integrating the differential equations of motion, using a predictor-corrector method.
2. Error analyses on trajectories generated by the program, in which the effects of hypothetical errors in the initial state and subsequent measurements may be studied.
3. Estimation of state vector quantities, using a recursive Kalman-filter scheme, from
   (a) Simulated noisy data generated by the program.
   (b) Noisy radar measurements, supplied externally to the program.

### KEY WORDS
- Trajectory analysis
- Computer program
- Program package
- FORTRAN IV
- Re-entry trajectories