Reports of the Department of Geodetic Science
Report Number 88

# LEAST SQUARES ADJUSTMENT OF SATELLITE OBSERVATIONS FOR SIMULTANEOUS DIRECTIONS OR RANGES

## PART 3 of 3: SUBROUTINES

by


Edward J. Krakiwsky
Jack Ferrier
James P. Reilly



Prepared for
National Aeronautics and Space Administration
Washington, D. C.
Contract No. NSR 36-008-033
OSURF Project No. 1997



The Ohio State University
Research Foundation
Columbus, Ohio 43212


December, 1967


GPO PRICE $ _____
CSFTI PRICE(S) $ _____
Hard copy (HC) 3.00
Microfiche (MF) .65
ff 653 July 65

FACILITY FORM 602
68 18724 (ACCESSION NUMBER)
(THRU)
54 (PAGES)
1 (CODE)
CR-93424 (NASA CR OR TMX OR AD NUMBER)
(CATEGORY)

MAR 1968

Reports of the Department of Geodetic Science

REPORT NUMBER 88

# LEAST SQUARES ADJUSTMENT

# of

# SATELLITE OBSERVATIONS

# for

# SIMULTANEOUS DIRECTIONS OR RANGES

## Part 3 of 3: Subroutines

by

Edward J. Krakiwsky
Jack Ferrier
James P. Reilly

Prepared for

National Aeronautics and Space Administration
Washington D. C.

Contract No. NSR 36-008-033
OSURF Project No. 1997

The Ohio State University
Research Foundation
Columbus, Ohio 43212

December 1967

## PREFACE

This project is under the direction of Professor Ivan I. Mueller of the Department of Geodetic Science, The Ohio State University. Project manager is Jerome D. Rosenberg, Geodetic Satellites, Code SAG, NASA Headquarters, Washington, D. C. The contract is administered by the office of Grants and Research Contracts, Office of Space Science and Applications, NASA Headquarters, Washington, D. C.

The report was written by James P. Reilly, graduate research assistant. Computer programming was done by Messrs. Ferrier, technical assistant, Krakiwsky, graduate research associate, and Reilly.

Reports related to NASA Contract No. NSR 36-008-033 and published to date are the following:

The Determination and Distribution of Precise Time, Report No. 70 of the Department of Geodetic Science, The Ohio State University.

Proposed Optical Network for the National Geodetic Satellite Program, Report No. 71 of the Department of Geodetic Science, The Ohio State University.

Preprocessing Optical Satellite Observations, Report No. 82 of the Department of Geodetic Science, The Ohio State University.

Least Squares Adjustment of Satellite Observations for Simultaneous Directions or Ranges, Part 1 of 3: Formulation of Equations, Report No. 86 of the Department of Geodetic Science, The Ohio State University.

Least Squares Adjustment of Satellite Observations for Simultaneous Directions or Ranges, Part 2 of 3: Computer Programs, Report No. 87 of the Department of Geodetic Science, The Ohio State University.

Least Squares Adjustment of Satellite Observations for Simultaneous Directions or Ranges, Part 3 of 3: Subroutines, Report No. 88 of the Department of Geodetic Science, The Ohio State University.

Quarterly Progress Reports, Numbers 1, 2, 3, 4, 5, 6, 7, 8.

ABSTRACT

The purpose of the report is to present the documentation of the subroutines used in the "Least Squares Adjustment of Satellite Observations for Simultaneous Directions or Ranges" computer programs.

Each write-up is self-contained, there are no references made to other subroutine write-ups. However, in many cases, the logic can be fully comprehended only by seeing the subroutine within the framework of the complete computer program.

TABLE OF CONTENTS


| | Page |
|---|---|
| PREFACE | iii |
| ABSTRACT | iv |
| SUBROUTINE DOCUMENTATION | |
| ANRAD (function) | 1 |
| CARGEO | 2 |
| CHCON | 4 |
| FMMJD (function) | 6 |
| GEOCAR | 7 |
| HRRAD (function) | 8 |
| INVPR | 9 |
| MULTM3 | 11 |
| MULTV3 | 12 |
| NUT | 13 |
| POLE | 14 |
| RADAN | 15 |
| RADHR | 16 |
| RBRAN | 17 |
| RCOMP | 19 |
| ROTATE | 21 |
| SIG | 23 |
| SYINVT | 25 |
| TRANSP | 26 |
| APPENDIX | |

TITLE: ANRAD (Function)

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Converts from degrees, minutes, and seconds into radians.

RESTRICTIONS/
ASSUMPTIONS: Only one conversion per call.

METHOD: This function converts an angle to radians by first converting the angle to degrees and decimal degrees, and then dividing this by 57.29577951308232.

CALLING SEQUENCE: FUNCTION (G) = ANRAD. (I,M,N,C ) +

| | | |
|---|---|---|
| where | G = angle in radians | RDF * |
| | I = sign of angle | RSI ** |
| | M = degrees | RSI |
| | N = minutes | RSI |
| | C = seconds | RDF |

* Real Double Precision Floating

** Real Single Precision Integer

TITLE: CARGEO

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: Converts Cartesian coordinates to geodetic coordinates.

RESTRICTIONS/ ASSUMPTIONS: Only one conversion per call. The values for 'A', 'EZ', and 'PI' ($a$, $e^2$, $\Pi$) must be defined in main program.

METHOD: Given $X$, $Y$, $Z$, $a$, $e^2$, and $\Pi$,

$$D = \frac{Z}{(X^2+Y^2)^{\frac{1}{2}}} \tag{1}$$

$$\text{Tan}\ \phi' = D\ (1 - e^2) \tag{2}$$

$$N' = \frac{a}{(1-e^2 \text{Sin}^2 \phi')^{\frac{1}{2}}} \tag{3}$$

$$\text{Tan}\ \phi = D\ \left(\frac{1+e^2 N' \text{Sin}\phi'}{Z}\right) \tag{4}$$

$$N = \frac{a}{(1-e^2 \text{Sin}^2 \phi)^{\frac{1}{2}}} \tag{5}$$

If $(\phi - \phi')$ is greater than $.5 \times 10^{-10}$, the value of $\phi'$ is replaced by $\phi$, and equations (4) and (5) are iterated until $(\phi - \phi')$ is less than $.5 \times 10^{-10}$.

Then,

$$\lambda = \tan^{-1} \frac{Y}{X} \tag{6}$$

$$H = \frac{(X^2+Y^2)^{\frac{1}{2}}}{\cos\phi} - N \tag{7}$$

where

$\phi$ = geodetic latitude of stations, in radians RDF

$\lambda$ = geodetic longitude of stations, in radians RDF

H = elevation of station above the ellipsoid, in meters RDF

CALLING SEQUENCE: CALL SUBROUTINE (C, G, H) = CARGEO. (X, Y, Z)+ where

X, Y, and Z = the Cartesian coordinates of the station, in meters RDF

C = geodetic latitude, in radians RDF

G = geodetic longitude, in radians RDF

H = elevation above ellipsoid, in meters RDF

TITLE: CHCON

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: This subroutine computes the chord distance between two stations along with the 3 - 3x3 matrices required to constrain this distance in the solution of Normal Equations.

RESTRICTIONS/ ASSUMPTIONS: Only one computation per call.

METHOD: Given the two station Numbers (for identification) and the X, Y, Z coordinates for the two stations,

$$RR = \left( (X_2-X_1)^2 + (Y_2-Y_1)^2 + (Z_2-Z_1)^2 \right)^{\frac{1}{2}}$$

The direction cosines of the vector between the two stations are computed from each station

$$T1\ (0) = -\frac{(X_2-X_1)}{R}$$

$$T1\ (1) = -\frac{(Y_2-Y_1)}{R}$$

$$T1\ (2) = -\frac{(Z_2-Z_1)}{R}$$

$$T2\ (0) = \frac{X_2-X_1}{R}$$

$$T2\ (1) = \frac{Y_2-Y_1}{R}$$

$$T2\ (2) = \frac{Z_2-Z_1}{R}$$

Three 3x3 matrices are computed as follows:

$N_{11}(i,j) = T1\ (i) * P * T1\ (j)$

$N_{22}(i,j) = T2\ (i) * P * T2\ (j)$

$N_{12}(i,j) = T1\ (i) * P * T2\ (j)$

where
P = weight = 1000.

CALLING SEQUENCE: CALL SUBROUTINE (N11, N22, N12, RR) = CHCON. (ICC1, ICC2, XX1, XX2, YY1, YY2, ZZ1, ZZ2)+

where

| | | |
|---|---|---|
| N11, N22, N12 are defined under METHOD | | RSF* |
| RR = | Computed chord distance between the two stations | RDF |
| ICC1 = | Identification number of first station | RSI |
| ICC2 = | Identification number of second station | RSI |
| XX1, YY1, ZZ1 = | X, Y, Z coordinates of first station | RDF |
| XX2, YY2, ZZ2 = | X, Y, Z coordinates of second station | RDF |

* Real Single Precision Flating

TITLE: FMMJD (FUNCTION)

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Calculates the number of Julian days that have ellapsed since Jan. 1.0, 1950.

RESTRICTIONS/
ASSUMPTIONS: Only one conversion per call.

METHOD: The function tests the 'month', and calculates the number of days in the year prior to that month. A test is made for a leap year, and if it is a leap year the extra day is added.

The hours, minutes, and seconds are converted to the decimal part of the day. Then the number of Julian days that have ellapsed since Jan. 1.0, 1950, is computed.

CALLING SEQUENCE: FUNCTION (AMJD) = FMMJD. (LHR, LMN, ASC, LMO, LDA, LYR) +
where

| | | |
|---|---|---|
| LHR = | Hours | RSI |
| LMN = | Minutes | RSI |
| ASC = | Seconds | RDF |
| LMO = | Month | RSI |
| LDA = | Day | RSI |
| LYR = | Year | RSI |
| AMJD = | Number of Julian Days since Jan. 1.0, 1950 | RDF |

TITLE: GEOCAR

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: Converts geodetic coordinates to Cartesian coordinates.

RESTRICTIONS/ ASSUMPTIONS: Only one conversion per call. The values for 'A', and 'EZ' ($a$, $e^2$) must be defined in main program.

METHOD: If 'a' is the semi-major axis of the ellipsoid, and '$e^2$' is the eccentricity of the ellipsoid, then

$$N = \frac{a}{(1-e^2 \mathrm{Sin}^2\phi)^{\frac{1}{2}}}$$

$$X = (N+H)\ \mathrm{Cos}\phi\ \mathrm{Cos}\lambda$$

$$Y = (N+H)\ \mathrm{Cos}\phi\ \mathrm{Sin}\lambda$$

$$Z = (N\ (1-e^2) + H)\ \mathrm{Sin}\phi$$

where
$\phi$ = latitude of station
$\lambda$ = longitude of station
$H$ = height of station above the ellipsoid

CALLING SEQUENCE: CALL SUBROUTINE (X, Y, Z) = GEOCAR. ($\phi$, $\lambda$, H) +

| | | |
|---|---|---|
| where $\phi$ = | latitude in radians | RDF |
| $\lambda$ = | longitude in radians | RDF |
| H = | elevation above ellipsoid in meters, | RDF |
| X, Y, & Z = | Cartesian coordinates of station, in meters | RDF |

TITLE: HRRAD (Function)

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Calculates radians from hours, minutes and seconds.

RESTRICTIONS/
ASSUMPTIONS: Only one conversion per call.

METHOD: This subroutine converts hours, minutes and seconds into radians by using the standard procedures.

CALLING SEQUENCE: FUNCTION (B) = HRRAD. (I, J, A) +
where

| | | |
|---|---|---|
| I | = Hours | RSI |
| J | = Minutes | RSI |
| A | = Seconds | RDF |
| B | = Output value, in radians | RDF |

TITLE: INVPR

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Converts Cartesian coordinates to geodetic coordinates.

RESTRICTIONS/
ASSUMPTIONS: Only one conversion per call. This subroutine should be used only for getting 'rough estimates' of $\phi$, $\lambda$, H.

METHOD: Given X, Y, Z, a, $e^2$

$$D = \frac{Z}{(X^2+Y^2)^{\frac{1}{2}}}$$

$$\tan \phi' = D\ (1 - e^2)$$

$$N = \frac{a}{(1-e^2 \mathrm{Sin}^2 \phi')^{\frac{1}{2}}}$$

$$\mathrm{Tan}\ \phi = D\ \left(\frac{1-e^2 N \mathrm{Sin}\phi'}{Z}\right)$$

$$H = \frac{Z}{\mathrm{Sin}\phi} - N\ (1-e^2)$$

$$\lambda = \tan^{-1} \frac{Y}{X}$$

CALLING SEQUENCE: CALL SUBROUTINE (FF, XLAM, H) = INVPR. (A,EE,X,Y,Z) +

where

| | | |
|---|---|---|
| FF = | latitude of station, in radians | RSF |
| XLAM = | longitude of station, in radians | RSF |
| H = | elevation above ellipsoid, in meters | RSF |
| A = | semi-major axis of ellipsoid | RSF |

INVPR (Contd)

| | | |
|---|---|---|
| EE | = eccentricity of ellipsoid | RSF |
| X, Y, Z | = Cartesian coordinates of station, in meters | RSF |

TITLE: MULTM3

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Multiplies two 3x3 matrices.

RESTRICTIONS/
ASSUMPTIONS: The elements of the A and B matrix must be defined prior to calling the subroutine.

METHOD: Given two 3x3 matrices, A and B.

```
R (1,1) = A (1,1) * B (1,1) + A (1,2) *
          B (2,1) + A (1,3) + B (3,1).
```

This process is repeated for all nine (9) elements of the 'R' matrix.

CALLING SEQUENCE: CALL SUBROUTINE (R) = MULTM3. (A,B)+

```
where A = 3x3 matrix                RDF
      B = 3x3 matrix                RDF
      R = Resulting 3x3 matrix      RDF
```

TITLE: MULTV3

PROGRAMMING LANGUAGE: SCARTAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Multiplies a 3x3 Matrix by a 3x1 vector.

RESTRICTIONS/
ASSUMPTIONS: The vector will be pre-multiplied by the matrix. The vector cannot be post-multiplied by the matrix.

METHOD: Given the matrix A (3,3), and the vector B (3,1). The subroutine multiplies A times B to give the resulting vector R (3,1).

CALLING SEQUENCE: CALL SUBROUTINE (R) = MULTV3. (A,B) +
where

| | | |
|---|---|---|
| A = | 3x3 Matrix | RDF |
| B = | 3x1 vector | RDF |
| R = | Resulting 3x1 vector | RDF |

TITLE: NUT

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: This subroutine computes the nutation in longitude and obliquity of the ecliptic to be used in computing the Greenwich apparent sidereal time.

RESTRICTIONS/ ASSUMPTIONS: Only one computation per call. This subroutine calls the system subroutine DMTMPY, which must be in the library.

METHOD: The subroutine is entered with the time of the event expressed as the number of days and decimal days since 1900.0

The theory of the computations is explained in full detail in the text of 'Part I: Theory. The output of the subroutine is the nutation in longitude and nutation in obliquity.

CALLING SEQUENCE: CALL SUBROUTINE (C,E) = NUT. (DD) +

where

| | | |
|---|---|---|
| C | = Nutation in longitude | RDF |
| E | = Nutation in obliquity | RDF |
| DD | = Number of days and decimal days since 1900.0 | RDF |

TITLE: POLE

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: This subroutine interpolates, from data in computer storage, the motion of the pole (x and y) for any instant of time from 1958.0 to 1967.0.

RESTRICTIONS/ ASSUMPTIONS: Only one interpolation per call.

METHOD: The values for the motion of the pole are tabulated from Bessellian date 1958.0 to Bessellian date 1967.0, the tabular interval being 0.05 years.

The input to the program is the date of interest, expressed as the number of Julian days and decimal days that have ellapsed since January 1.0, 1950 Greenwich universal time. To this is added the Julian Date of January 1.0, 1950.

Julian Date = Time of interest +
+ 2433282.500

where
2433282.500 = Julian date of January 1.0, 1950 U T

The time of event is then converted to Bessellian date by the following conversion

$$B.D. = \frac{J.D. - 1721060.133}{365.2422}$$

The table of polar motion values are searched using the Bessellian date of the event as argument. A second order Bessillian interpolation is performed to arrive at the values for x and y.

CALLING SEQUENCE: CALL SUBROUTINE (FUNX, FUNY) = POLE. (TIEV) +
where

FUNX = the x motion of the pole RDF
FUNY = the y motion of the pole RDF
TIEV = time of event, which is equal to the number of days and decimal days that have ellapsed since Jan. 1.0, 1950. RDF

TITLE: RADAN

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: This subroutine converts radians into degrees, minutes and seconds.

RESTRICTIONS/
ASSUMPTIONS: Only one conversion per call.

METHOD: This subroutine multiplies the radian value by 57.29577951308232 to get degrees and decimal degrees. The decimal portion of the degrees is multiplied by 60 to get minutes and decimal minutes. The decimal portion of the minutes is multiplied by 60 to get seconds and decimal seconds.

CALLING SEQUENCE: CALL SUBROUTINE (I,M,N,C) = RADAN. (T) +
where

| | | |
|---|---|---|
| I | = sign of angle | RSI |
| M | = degrees | RSI |
| N | = minutes | RSI |
| C | = seconds | RDF |
| T | = radians | RDF |

TITLE: RADHR

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Calculates hours, minutes, seconds from radians.

RESTRICTIONS/
ASSUMPTIONS: Only one conversion per call.

METHOD: This subroutine converts radians to hours, minutes and seconds by using the standard procedures.

CALLING SEQUENCE: CALL SUBROUTINE (I, J, A) = RADHR. (THETA)+
where

| | | |
|---|---|---|
| I | = Hours | RSI |
| J | = Minutes | RSI |
| A | = Seconds | RDF |
| THETA | = input value, in radians | RDF |

TITLE: RBRAN

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: This subroutine performs a least square iteration adjustment for the X, Y, Z coordinates (terrestrial or geodetic system) of a satellite.

RESTRICTIONS/ ASSUMPTIONS: Subroutines GEOCAR and SYINVT must be included with this subroutine.

METHOD: This subroutine performs a least square adjustment for the X, Y, Z coordinates of a satellite by holding the ground stations fixed and applying the residuals to the ranges. The approximation for the parameters (XS, YS, ZS) is obtained by converting the so-called approximate geodetic coordinates of the satellite to Cartesian coordinates. The approximation is obtained by meaning the latitudes and longitudes of the ground stations and using 1.6 megameters for the ellipsoidal height. (Note that 1.6 is constant, and for higher satellites this value is too small; however, in view of the rapid convergence, it is at this time not deemed necessary to make this a variable with each satellite).

The following formulation is used:

$$R = [\Delta X^2+\Delta Y^2+\Delta Z^2]^{\frac{1}{2}}$$

$$A = \frac{\partial R}{\partial \text{ Parameters}}$$

$$V = AX + L\ (R_{comp}-R_{obs})$$

where

X = corrections to parameters
V = residuals to ranges

The convergence criterion of the XS, YS, ZS is 1 cm. Upon convergence the coefficient matrix "A" and the residuals "V" are output.

CALLING SEQUENCE: CALL SUBROUTINE (ACOEF, XS, YS, ZS, FL, V, K) = RBRAN. (NS, FLAT, FLON, STAXYZ, NO , RO) +

where

| | | | |
|---|---|---|---|
| ACOEF | = | A (defined in METHOD) | RSF |
| XS, YS, ZS | = | cartesian coordinates of satellite | RDF |
| FL | = | $R_{comp}-R_{obs} = L$ (topocentric range) | RDF |
| V | = | AX+L (residuals to ranges) | RDF |
| K | = | Number of iteration (maximum) | RSI |
| NS | = | Number of stations involved in the simultaneous event | RSI |
| FLAT | = | an array containing the latitudes (in radians) of the stations involved | RDF |
| FLON | = | an array containing the longitudes (in radians) of the stations involved | RDF |
| STAXYZ | = | a matrix containing the Cartesian coordinates of all the ground stations in the <u>whole</u> net. | RDF |
| NO | = | an array containing the assigned numbers of stations involved in the simultaneous event | RSI |
| RO | = | an array of observed ranges | RDF |

TITLE: RCOMP.

PROGRAMMING LANGUAGE: SCARTAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: This subroutine computes the X, Y, Z of the satellite from the $\alpha$, $\delta$ from the two ground stations.

RESTRICTIONS/ ASSUMPTIONS: One calculation per call.

METHOD: Given $\alpha$ & $\delta$ from the two ground stations to the satellite,

$$a_i = \cos \alpha_i \cos\delta_i$$

$$b_i = \sin \alpha_i \cos\delta_i \quad \text{(true celestial coordinate system)}$$

$$c_i = \sin \delta_i$$

a rotation is performed to the vector into the terrestrial coordinate system.

$$\begin{vmatrix} a_i' \\ b_i' \\ c_i' \end{vmatrix} = R \begin{vmatrix} a_i \\ b_i \\ c_i \end{vmatrix}$$

where the rotation matrix R has been computed prior to calling the subroutine.

Since the two ground stations have known coordinates, the direction cosines of the unit vector between the two stations is computed. The three vectors then represent the three sides of a triangle, and the vectors are dotted to get the angles of the triangle. The scale is introduced by the computed distance between the two ground stations; finally, the X, Y, and Z of the satellite is computed using the laws of trigonometry.

CALLING SEQUENCE: CALL SUBROUTINE (XS, YS, ZS) = RCOMP.
(NO, STAXYZ, RA, DEC, R3) +

where

| | | | |
|---|---|---|---|
| XS, YS, ZS | = | Cartesian coordinates of satellite in the terrestrial or geodetic system | RSF |
| NO | = | Vector containing the numbers of the two ground stations | RSI |
| STAXYZ | = | matrix containing the X, Y, Z of the ground stations | RDF |
| RA | = | vector containing the right ascensions (radians) from the two ground stations (in the same order as NO). | RDF |
| DEC | = | same as RA, but for the declinations | RDF |
| R3 | = | rotation matrix to rotate into the terrestrial system | RSF |

TITLE: ROTATE

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Forms a rotation matrix which is the product of any number of single-axis rotations through any angles.

RESTRICTIONS/
ASSUMPTIONS: The rotation angles must be in the array 'C', and the rotation axes must be in the array 'L' prior to calling the subroutine.

METHOD: This subroutine forms a rotation matrix which is the product of any number of single axis rotations through any angles. The input to the subroutine is:

1. The rotation angles, which are given in the array 'C'. The order of the angles in the array will be the order in which the rotations are performed.

2. The axis about which the rotations are to take place are given in the array 'L'. The value 1 refers to a rotation about the X axis, 2 about the Y axis, 3 about the Z axis, and a number larger than 3 in the array is to indicate that all rotations have been made, and to transfer out of the subroutine.

The subroutine looks at the value in L (o). If it is a 1, it will perform a rotation about the X axis by the amount of the angle given in C (o). If L (o) is a 2, a rotation will be performed about the Y axis by the amount of the angle in C (o). If L (o) is a 3, a rotation will be performed about the Z axis by the amount of the angle in C (o).

The rotation matrix will be as follows:

$$\text{If } L(o) = 1,\ R = \begin{vmatrix} 1 & 0 & 0 \\ 0 & \cos C(o) & \sin C(o) \\ 0 & -\sin C(o) & \cos C(o) \end{vmatrix}$$

$$\text{If } L(o) = 2,\ R = \begin{vmatrix} \cos C(o) & 0 & -\sin C(o) \\ 0 & 1 & 0 \\ \sin C(o) & 0 & \cos C(o) \end{vmatrix}$$

$$\text{If } L(o) = 3,\ R = \begin{vmatrix} \cos C(o) & \sin C(o) & 0 \\ -\sin C(o) & \cos C(o) & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

The subroutine then looks at L (1), performs the rotation through the angle located in C (1), and then multiplies this new rotation matrix by the previous matrix. This process is repeated until a number larger than '3' appears as one of the elements in the 'L' array.

CALLING SEQUENCE: CALL SUBROUTINE (R) = ROTATE. (C,L) +
where

| | | |
|---|---|---|
| C = | rotation angles array, single subscript (0 to (n-1)) | RDF |
| L = | rotation axes array, single subscript (0 to N) | RSI |
| R = | rotation matrix array, double subscript (0,0 to 2,2). | RDF |
| N = | number of single axis rotations | RSI |

L (M) $\neq$ 1, 2, or 3

TITLE: SIG

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/ TYPE: IBM 7094

DESCRIPTION: This subroutine computes the variance-covariance matrix of $\phi$, $\lambda$, H from the variance-covariance matrix of X, Y, Z.

RESTRICTIONS/ ASSUMPTIONS This subroutine calls two system subroutines, MTINV and MTMPY. These subroutines must be in the library.

METHOD: Given the variance-covariance matrix of X, Y, and Z, and also given $\phi$, $\lambda$, H, the subroutine computes

$$G = \begin{vmatrix} -\sin\phi \cos\lambda & -\cos\phi & \sin\lambda & \cos\phi & \cos\lambda \\ -\sin\phi \sin\lambda & \cos\phi & \cos\lambda & \cos\phi & \sin\lambda \\ \cos\phi & 0 & & \sin\phi & \end{vmatrix}$$

$\Sigma_{\substack{x\\y\\z}}$ is the variance-covariance matrix of X, Y, Z.

$$\Sigma_{\substack{\phi\\\lambda\\h}} (\text{Meters})^2 = G^{-1} \Sigma_{\substack{x\\y\\z}} (G')^{-1}$$

where $\Sigma_{\substack{\phi\\\lambda\\h}}$ is the variance-covariance of $\phi$, $\lambda$, H (in meters$^2$) .

The subroutine also computes $\Sigma_{\substack{\phi\\\lambda\\h}}[\text{meters}^2, \text{sec}^2]$ which is the variance-covariance matrix of $\phi$, $\lambda$, H in mixed units by converting the units of $\phi$ and $\lambda$ to seconds$^2$.

CALLING SEQUENCE: CALL SUBROUTINE (SIGL, SIGM) = SIG. (SIGX, FLAT, FLON, H) +
where

| | | |
|---|---|---|
| SIGX = | $\Sigma_{\substack{X\\Y\\Z}}$, the variance-covariance matrix of X, Y, Z | RSF |
| FLAT = | Latitude | RSF |
| FLON = | Longitude | RSF |
| H = | Height above ellipsoid | RSF |
| SIG L = | $\Sigma_{\substack{\phi\\\lambda\\H}}^{\text{Meters}^2}$, the variance-covariance matrix of $\phi$, $\lambda$, H, in meters$^2$ | RSF |
| SIGM = | $\Sigma_{\substack{\phi\\\lambda\\H}}$ the variance-covariance matrix of $\phi$, $\lambda$, H, in mixed units. | RSF |

TITLE: SYINVT

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Inverts a symmetric 3x3 matrix.

RESTRICTIONS/
ASSUMPTIONS: Only one matrix inversion per call.

METHOD: This subroutine inverts a symmetric 3x3 matrix by forming the matrix of cofactors, and dividing each element of the cofactor matrix by the determinate of the original matrix. The result is the inverse matrix.

CALLING SEQUENCE: CALL SUBROUTINE (B) = SYINVT. (A,S) +
where

A = name of the double subscripted array that is to be inverted. (array starts at (0,0)) RDF
B = name of double subscripted array that is the inverse of A (array starts at (0,0)) RDF
S = single subscripted variable with dimension at least 10, used for intermediate storage. S(9) will be the determinate of 'A' RDF

TITLE: TRANSP

PROGRAMMING LANGUAGE: SCATRAN

MACHINE CONFIGURATION/
TYPE: IBM 7094

DESCRIPTION: Takes the transpose of a square matrix.

RESTRICTIONS/
ASSUMPTIONS: Only one matrix transposed per call.

METHOD: Direct application of the definition.

CALLING SEQUENCE: CALL SUBROUTINE (G) = TRANSP. (N) +
where

G = the matrix to be transposed. It will also be the name of the transposed matrix. (Double subscripted array variable starting at (0,0)) RDF

N = number of elements in any row or column RSI

# APPENDIX

Listing of Subroutines

```
 FUNCTION (G)=ANRAD.(I,M,N,C)-
 PRECISION (2,G,C,RHO,D)-
 LITERALS (RHO,57.29577951308232)-
D=60.-
G=(M+N/D+C/(D*D))/RHO-
 PROVIDED (I.LQ.$-      $),G=-G-
 NORMAL EXIT -
 END SUBPROGRAM -
```

```
        SUBROUTINE(N11,N22,N12,RR)=CHCON.(ICC1,ICC2,XX1,XX2,YY1,YY2,
        ZZ1,ZZ2)-
        PRECISION (2,XX1,YY1,ZZ1,XX2,YY2,ZZ2,RR ) -
        PRECISION(2,DSQRT.) -
        FLOATING(N11,N22,N12)-
        DIMENSION(          N11(0,L),N22(0,L),N12(0,L),T1(3),T2(3))-
        P=1000.-
        L=3-
        DX=XX2-XX1-
        DY=YY2-YY1-
        DZ=ZZ2-ZZ1-
        RR = DSQRT.((XX2 - XX1).P.2 + (YY2 - YY1).P.2 + (ZZ2 - ZZ1)
         .P.2 ) -
        R =  SQRT.(DX.P.2+DY.P.2+DZ.P.2) -
        T1(0)=-DX/R-
        T1(1)=-DY/R-
        T1(2)=-DZ/R-
        DOTHROUGH(NEG),I=0,1,I.L.3-
NEG     T2(I)=-T1(I)-
        DOTHROUGH(ONE),I=0,1,I.L.3-
        DOTHROUGH(ONE),J=0,1,J.L.3-
ONE     N11(I,J)=T1(I)*P*T1(J)-
        DOTHROUGH(TWO),I=0,1,I.L.3-
        DOTHROUGH(TWO),J=0,1,J.L.3-
TWO     N22(I,J)=T2(I)*P*T2(J)-
        DOTHROUGH(THR),I=0,1,I.L.3-
        DOTHROUGH(THR),J=0,1,J.L.3-
THR     N12(I,J)=T1(I)*P*T2(J)-
        NORMALEXIT-
        ENDSUBPROGRAM-
```

```
          SUBROUTINE (C,G,H)=CARGEO.(X,Y,Z)-
          UNIVERSAL (A,E2)-
          LITERALS (PI,3.1415926535897 93)-
          PRECISION (2,D,C,G,H,X,Y,Z,CO,DATAN.,P,PI)-
          PRECISION (2,TANC,TANCO,DSIN.,DCOS.,F,BI,DSQRT.)-
         F=DSQRT.(X*X+Y*Y)-
         D=Z/F-
         TANCO=D/(1.-E2)-
         CO=DATAN.(TANCO)-
         BI=DSIN.(CO)-
         P=A/DSQRT.(1.-E2*BI*BI)-
HERM1    TANC=D*(1.+E2*P*BI/Z)-
         C=DATAN.(TANC)-
         BI=DSIN.(C)-
         P=A/DSQRT.(1.-E2*BI*BI)-
          PROVIDED (.ABS.(C-CO).LE..5.X.-10), TRANSFER TO (HERM2)-
         CO=C-
          TRANSFER TO (HERM1)-
HERM2    G=DATAN.(Y/X)-
          PROVIDED (X.L.0.0),G=G+PI-
          PROVIDED (X.GE.0.0.AND.Y.L.0.0),G=G+2.*PI-
         H=F/DCOS.(C)-P-
          NORMAL EXIT -
          END SUBPROGRAM -
```

```
          FUNCTION (AMJD)=FMMJD.(LHR,LMN,ASC,LMO,LDA,LYR)-
          PRECISION (2,ASC,AMJD)-
          CONDITIONAL (MON)-
          PROVIDED (LMO.LQ.$JAN    $) OTHERWISE (MON1)-
         KMO=0-
MON1      OR PROVIDED (LMO.LQ.$FEB    $) OTHERWISE (MON2)-
         KMO=31-
MON2      OR PROVIDED (LMO.LQ.$MAR    $) OTHERWISE (MON3)-
         KMO=59-
MON3      OR PROVIDED (LMO.LQ.$APR    $) OTHERWISE (MON4)-
         KMO=90-
MON4      OR PROVIDED (LMO.LQ.$MAY    $) OTHERWISE (MON5)-
         KMO=120-
MON5      OR PROVIDED (LMO.LQ.$JUN    $) OTHERWISE (MON6)-
         KMO=151-
MON6      OR PROVIDED (LMO.LQ.$JUL    $) OTHERWISE (MON7)-
         KMO=181-
MON7      OR PROVIDED (LMO.LQ.$AUG    $) OTHERWISE (MON8)-
         KMO=212-
MON8      OR PROVIDED (LMO.LQ.$SEP    $) OTHERWISE (MON9)-
         KMO=243-
MON9      OR PROVIDED (LMO.LQ.$OCT    $) OTHERWISE (MON10)-
         KMO=273-
MON10     OR PROVIDED (LMO.LQ.$NOV    $) OTHERWISE (MON11)-
         KMO=304-
          END CONDITIONAL -
MON11    KMO=334-
MON      M=LYR/4-
          PROVIDED (LYR-4*M.E.0.AND.KMO.L.59),KMO=KMO-1-
         AMJD=-3663.0+(LYR-40)*365+KMO+LDA+M+(LHR+(LMN+ASC/60.)/60.)
        /24.-
          NORMAL EXIT -
          END SUBPROGRAM -
```

```
 SUBROUTINE (X,Y,Z)=GEOCAR.(C,G,H)-
 UNIVERSAL (A,E2)-
 PRECISION (Z,X,Y,Z,C,G,H,P,DSIN.,DCOS.,BI)-
BI=DSIN.(C)-
P=A/(1.-E2*BI.P.2).P..5-
X=(P+H)*DCOS.(C)-
Y=X*DSIN.(G)-
X=X*DCOS.(G)-
Z=(P*(1.-E2)+H)*BI-
 NORMAL EXIT -
 END SUBPROGRAM -
```

```
 FUNCTION (B)=HRRAD.(I,J,A)-
 LITERALS (T,0.2617993877991494)-
 PRECISION (2,B,A,T)-
B=I*T+J*T/60.+A*T/3600.-
 NORMAL EXIT -
 END SUBPROGRAM -
```

```
          SUBROUTINE (FF,XLAM,H)=INVPR.(A,EE,X,Y,Z)-
         RO=57.29578-
         PI=3.1415927-
         TLAM=Y/X-
         XLAM=ATAN.(TLAM)-
          PROVIDED (Y.GE.0.), TRANSFER (L10)-
          PROVIDED (X.GE.0.),XLAM=XLAM+2.*PI-
          PROVIDED (X.L.0.),XLAM=XLAM+PI-
          TRANSFER (L20)-
L10       PROVIDED (X.GE.0.),XLAM=XLAM-
          PROVIDED (X.L.0.),XLAM=XLAM+PI-
L20      XLAM=XLAM*RO-
         TERM3=SQRT.(X*X+Y*Y)-
         TERM1=Z/TERM3-
         TERM2=TERM1/(1-EE)-
         FF=ATAN.(TERM2)-
         SFI=SIN.(FF)-
         XN=A/SQRT.(1.-EE*SFI*SFI)-
         TFI=TERM1*(1.+EE*XN*SFI/Z)-
         FI1=ATAN.(TFI)-
         FI1D=FI1*RO-
         SFI=SIN.(FI1)-
         H=Z/SFI-XN*(1.-EE)-
         FF=FI1D-
          NORMAL EXIT -
          END SUBPROGRAM -
```

```
            SUBROUTINE (R)=MULTM3.(A,B)-
            PRECISION (2,A,B,R)-
            LITERALS (KDM,3)-
            DIMENSION (A(0,KDM),B(0,KDM),R(0,KDM))-
            DO THROUGH (MULT),I=0,1,I.L.3-
            DO THROUGH (MULT),J=0,1,J.L.3-
MULT       R(I,J)=A(I,0)*B(0,J)+A(I,1)*B(1,J)+A(I,2)*B(2,J)-
            NORMAL EXIT -
            END SUBPROGRAM -
```

```
            SUBROUTINE (R)=MULTV3.(A,B)-
            PRECISION (2,A,B,R)-
            LITERALS (KDM,3)-
            DIMENSION (A(0,KDM))-
            DO THROUGH (MULT),I=0,1,I.L.3-
MULT       R(I)=A(I,0)*B(0)+A(I,1)*B(1)+A(I,2)*B(2)-
            NORMAL EXIT -
            END SUBPROGRAM -
```

```
  SUBROUTINE (C,E)=NUT.(DD)-
  DIMENSION (A(5,L),AM(345,LL),CS(69),DCS(69),CC(69),DCC(69)
,AMA(69,L)-
  PRECISION (2,MM,M,TCS,TCC,DCC)-
  DIMENSION (TCS(69),TCC(69))-
  PRECISION (2,RHO,A,AM,AMA,DD,DII,DMTMPY.,DSIN.,DCOS.,E,C,C
S,CC,DCS,DCI)-
  LITERALS (AM,0.,0.,0.,0.,1.,0.,0.,0.,0.,2.,-2.,0.,2.,0.,1.
,2.,0.,-2.,0.,0.,0.,-2.,2.,-2.,1.,-2.,0.,2.,0.,2.,1.,-1.,0.,
-1.,0.,0.,0.,2.,-2.,2.,0.,1.,0.,0.,0.,0.,1.,2.,-2.,2.,0.,-1.
,2.,-2.,2.,0.,0.,2.,-2.,1.,2.,0.,0.,-2.,0.,0.,0.,2.,-2.,0.,0
.,2.,0.,0.,0.,0.,1.,0.,0.,1.,0.,2.,2.,-2.,2.,0.,-1.,0.,0.,1.
)-
  LITERALS (AM(90),-2.,0.,0.,2.,1.,0.,-1.,2.,-2.,1.,2.,0.,0.
,-2.,1.,0.,1.,2.,-2.,1.,1.,0.,0.,-1.,0.,0.,0.,2.,0.,2.,1.,0.
,0.,0.,0.,0.,0.,2.,0.,1.,1.,0.,2.,0.,2.,1.,0.,0.,-2.,0.,-1.,
0.,2.,0.,2.,0.,0.,0.,2.,0.,1.,0.,0.,0.,1.,-1.,0.,0.,0.,1.,-1
.,0.,2.,2.,2.,1.,0.,2.,0.,1.0,0.,0.,2.,2.,2.,2.,0.,0.,0.,0.)
-
  LITERALS (AM(180),1.,0.,2.,-2.,2.,2.,0.,2.,0.,2.,0.,0.,2.,
0.,0.,-1.,0.,2.,0.,1.,-1.,0.,0.,2.,1.,1.,0.,0.,-2.,1.,-1.,0.
,2.,2.,1.,1.,1.,0.,-2.,0.,0.,1.,2.,0.,2.,1.,0.,0.,2.,0.,0.,0
.,0.,2.,1.,0.,-1.,2.,0.,2.,1.,0.,2.,2.,2.,2.,0.,2.,-2.,2.,0.
,0.,0.,-2.,1.,0.,0.,2.,2.,1.,1.,0.,2.,-2.,1.,0.,0.,0.,1.,0.)
-
  LITERALS (AM(270),0.,1.,0.,-2.,0.,1.,-1.,0.,0.,0.,1.,0.,-2
.,0.,0.,2.,0.,2.,0.,1.,1.,0.,2.,0.,0.,1.,1.,0.,0.,0.,1.,-1.,
2.,0.,2.,-2.,0.,0.,0.,1.,-1.,0.,2.,-2.,1.,2.,0.,0.,0.,1.,-1.
,-1.,2.,2.,2.,0.,-1.,2.,2.,2.,1.,0.,0.,0.,2.,1.,1.,2.,0.,2.,
3.,0.,2.,0.,2.)-
  LITERALS (CS,-172327.,2088.,45.,10.,-4.,-3.,-2.,-12729.,12
61.,-497.,214.,124.,45.,-21.,16.,-15.,-15.,-10.,-5.,-5.,4.,3
.,-3.,-2037.,675.,-342.,-261.,-149.,114.,60.,58.,-57.,-52.,-
44.,-32.,28.,26.,-26.,25.,19.,14.,-13.,-9.,-7.,7.,6.,-6.,-6.
,-6.,6.,-5.,-5.,5.,-4.,-4.,4.,4.,-4.,3.,-3.,-3.,-2.,-2.,2.,-
2.,-2.,-2.,2.,-2.)-
  LITERALS (CC,92100.,-904.,-24.,0.,2.,2.,0.,5522.,0.,216.,-
93.,-66.,0.,0.,0.,8.,7.,5.,3.,3.,-2.,-2.,0.,884.,0.,183.,113
.,0.,-50.,0.,-31.,30.,22.,23.,14.,0.,-11.,11.,0.,-10.,-7.,7.
,5.,0.,-3.,0.,3.,3.,3.,-2.,3.,3.,-3.,0.,0.,0.,0.,2.)-
  BOOLEAN (B)-
 L=1-
 LL=5-
 K=69-
  LITERALS (RHO,57.29577951308232)-
 DII=DD/10000-
```

```
          T=DD/36525-
          A(0)=296.104608+13.0649924465*DD+0.0006890*DII*DII+0.000000
         295*DII*DII*DII-
          A(1)=358.475833+0.9856002669*DD-0.0000112*DII*DII-0.0000000
         68*DII*DII*DII-
          A(2)=11.250889+13.2293504490*DD-0.0002407*DII*DII-0.0000000
         07*DII*DII*DII-
          A(3)=350.737486+12.1907491914*DD-0.0001076*DII*DII+0.000000
         039*DII*DII*DII-
          A(4)=259.183275-0.0529539222*DD+0.0001557*DII*DII+0.0000000
         46*DII*DII*DII-
           DO THROUGH (TWO),I=0,1,I.LE.4-
          MM=A(I)/360-
TWO       A(I)=A(I)-MM*360.-
          C=0-
          E=0-
          DCS(0)=-173.7*T-
          DCS(1)=0.2*T-
          DCS(7)=-1.3*T-
          DCS(8)=-3.1*T-
          DCS(9)=1.2*T-
          DCS(10)=-0.5*T-
          DCS(11)=0.1*T-
          DCS(14)=-0.1*T-
          DCS(16)=0.1*T-
          DCS(23)=-0.2*T-
          DCS(24)=0.1*T-
          DCS(25)=-0.4*T-
          DCC(0)=9.1*T-
          DCC(1)=0.4*T-
          DCC(7)=-2.9*T-
          DCC(9)=-0.6*T-
          DCC(10)=0.3*T-
          DCC(23)=-0.5*T-
          DCC(26)=-0.1*T-
           CALL SUBROUTINE (B,AMA)=DMTMPY.(AM,A,K,LL,L)-
           DO THROUGH (ABLE),I=0,1,I.L.69-
          M=AMA(I)/360-
          AMA(I)=AMA(I)-M*360.-
          TCS(I)=CS(I)+DCS(I)-
          TCC(I)=CC(I)+DCC(I)-
          C=C+TCS(I)*DSIN.(AMA(I)/RHO)-
ABLE      E=E+TCC(I)*DCOS.(AMA(I)/RHO)-
          E=E/10000.-
          C=C/10000.-
           NORMAL EXIT -
           END SUBPROGRAM -
```

```
      SUBROUTINE (FUNX,FUNY)=POLE.(TIEV) -
   WHEN MORE VALUES OF X AND Y ARE NEEDED,          -
 CHANGE THE FOLLOWING DIMENSION CARD -
      DIMENSION (T(180),X(180), Y(180)) -
      PRECISION(2,DEL0X,DEL1X,DEL2X,DELL0X,DELL1X,FUNX,
       DEL0Y,DEL1Y,DEL2Y,DELL0Y,DELL1Y,FUNY,TIEV) -
      LITERALS(T(0),0.,1.,2.,3.,4.,5.,6.,7.,8.,9.,10.,11.,12.,13.,
      14.,15.,16.,17.,18.,19.,20.,21.,22.,23.,24.,25.,26.,27.,28.,
      29.,30.,31.,32.,33.,34.,35.,36.,37.,38.,39.,40.,41.,42.,43.,
      44.,45.,46.,47.,48.,49.,50.,51.,52.,53.,54.,55.,56.,57.,58.,
      59.,60.,61.,62.,63.,64.,65.,66.,67.,68.,69.,70.,71.,72.,73.,
      74.,75.,76.,77.,78.,79.,80.,81.,82.,83.,84.,85.,86.,87.,88.,
      89.)-
      LITERALS(T(90),90.,91.,92.,93.,94.,95.,96.,97),98.,99.,100.,
      101.,102.,103.,104.,105.,106.,107.,108.,109.,110.,111.,112.,
      113.,114.,115.,116.,117.,118.,119.,120.,121.,122.,123.,124.,
      125.,126.,127.,128.,129.,130.,131.,132.,133.,134.,135.,136.,
      137.,138.,139.,140.,141.,142.,143.,144.,145.,146.,147.,148.,
      149.,150.,151.,152.,153.,154.,155.,156.,157.,158.,159.,160.,
      161.,162.,163.,164.,165.,166.,167.,168.,169.,170.,171.,172.,
      173.,174.,175.,176.,177.,178.,179.)-
      LITERALS(X(0),-.173,-.215,-.235,-.237,-.218,-.162,-.097,-.03
      2,.036,.111,.188,.257,.348,.398,.398,.368,.330,.280,.218,.14
      4,.069,.000,-.062,-.112,-.140,-.153,-.151,-.126,-.086,-.037,
      .026,.092,.161,.223,.272,.299,.308,.296,.261,.202,.135,.073,
      .046,.035,.013,-.026,-.072,-.096,-.107,-.103,-.087,-.039,.00
      4,.040,.070,.080,.109,.117,.117,.109)-
      LITERALS(X(60),.092,.074,.065,.064,.062,.057,.046,.034,.030,
      .032,.040,.043,.042,.041,.039,.028,.019,-.010,-.027,-.021,-.
      009,.008,.027,.047,.071,.095,.120,.144,.162,.173,.171,.157,.
      128,.094,.056,.017,-.019,-.054,-.086,-.110,-.121,-.119,-.105
      ,-.076,-.038,.009,.070,.134,.191,.239,.274,.301,.281,.237,.1
      76,.112,.048,-.011,-.069,-.122)-
      LITERALS(X(120),-.171,-.206,-.194,-.169,-.139,-.101,-.055,.0
      04,.074,.164,.214,.240,.241,.239,.255,.250,.219,.161,.099,.0
      42,-.012,-.067,-.120,-.160,-.185,-.196,-.194,-.174,-.130,-.0
      72,-.003,.071,.127,.168,.201,.221,.227,.220,.194,.138,.075,.
      033,.000,-.029,-.058,-.086,-.105,-.116,-.119,-.115,-.104,-.0
      86,-.057,-.010,.052,.096,.117,.125,.123,.115)-
      LITERALS(Y(0),.022,.098,.187,.265,.328,.389,.443,.478,.493,.
      478,.447,.411,.365,.307,.235,.165,.097,.043,-.007,-.038,-.05
      7,-.064,-.057,-.025,.032,.120,.211,.285,.340,.372,.393,.406,
      .410,.401,.370,.320,.260,.201,.143,.090,.043,.007,-.012,-.00
      7,.025,.059,.094,.123,.153,.182,.209,.238,.263,.288,.300,.30
      6,.301,.288,.271,.249)-
```

```
          LITERALS(Y(60),.220,.189,.161,.150,.151,.158,.161,.160,.155,
          .153,.150,.151,.154,.157,.165,.174,.191,.212,.242,.276,.297,
          .309,.314,.312,.304,.290,.271,.246,.214,.175,.132,.092,.068,
          .060,.067,.083,.104,.128,.160,.200,.248,.295,.329,.356,.376,
          .388,.387,.375,.349,.307,.251,.193,.139,.091,.046,.008,-.020
          ,.005,.041,.078)-
          LITERALS(Y(120),.120,.168,.230,.294,.353,.412,.455,.467,.459
          ,.436,.394,.339,.279,.219,.168,.123,.085,.060,.046,.043,.049
          ,.069,.103,.153,.226,.286,.334,.374,.408,.434,.444,.433,.399
          ,.349,.303,.259,.221,.188,.156,.131,.114,.103,.098,.100,.108
          ,.124,.149,.181,.215,.255,.298,.330,.344,.345,.337,.324,.308
          ,.291,.273,.253)-
           LITERALS (RHOSEC,206264.8062470963) -
          TEV = TIEV + 2433282.500  -
          BY = (TEV - 1721060.133)/365.2422 -
          A = (BY - 1958.0) * 20. -
          L = A + 1. -
          AN = (A - T(L-1)) /T(L) - T(L-0))  -
          B = AN*(AN - 1.)/4. -
          DEL0X = X(L-1) - X(L-2) -
          DEL1X = X(L) - X(L-1) -
          DEL2X = X(L+1) - X(L) -
          DELL0X = DEL1X - DEL0X -
          DELL1X = DEL2X - DEL1X -
          FUNX = X(L-1) + AN*DEL1X + B*(DELL0X + DELL1X) -
          DEL0Y = Y(L-1) - Y(L-2) -
          DEL1Y = Y(L) - Y(L-1) -
          DEL2Y = Y(L+1) - Y(L) -
          DELL0Y = DEL1Y - DEL0Y -
          DELL1Y = DEL2Y - DEL1Y -
          FUNY = Y(L-1) + AN*DEL1Y + B*(DELL0Y + DELL1Y) -
          WRITE OUTPUT,FAT5,(FUNX) -
F  FAT5   (1H0,Q*  X = *,F15.6) -
          WRITE OUTPUT,FAT6,(FUNY) -
F  FAT6   (1H0,Q* Y = *,F15.6) -
          FUNX = FUNX/RHOSEC -
          FUNY = FUNY/RHOSEC -
          WRITE OUTPUT,FAT1,(TIEV) -
F FAT1    (1H1,Q*    TIME OF EVENT = *,DF10.3) -
          WRITE OUTPUT,FAT2,(A,K,P,J,L) -
F  FAT2   (1H0,F10.3,I4,F10.3,2I4) -
          WRITE OUTPUT,FAT3,(AN, B) -
F  FAT3   (1H0,Q* INTERVAL IS *,F10.4,Q*B = *,F10.4) -
          WRITE OUTPUT,FAT4,(DEL0X,DEL1X,DEL2X,DELL0X,DELL1X) -
F  FAT4   (1H0,6F15.6) -
          WRITE OUTPUT,FAT4,(DEL0Y,DEL1Y,DEL2Y,DELL0Y,DELL1Y) -
  FA0     CONTINUE -
          NORMAL EXIT -
          END SUBPROGRAM -
```

```
          SUBROUTINE (I,M,N,C)=RADAN.(T)-
          PRECISION (2,G,T,RHO)-
         G=T-
          LITERALS (RHO,57.29577951308232)-
          PROVIDED (G.L.0.0), TRANSFER TO (PLAC1)-
         I=$+      $-
          TRANSFER TO (PLAC2)-
PLAC1    G=-G-
         I=$-      $-
PLAC2    G=G*RHO-
         M=G-
         G=(G-M)*60.-
         N=G-
         C=(G-N)*60.-
          NORMAL EXIT -
          END SUBPROGRAM -
```

```
 SUBROUTINE (I,J,A)=RADHR.(THETA)-
 PRECISION (2,PI,A,THETA)-
PI=3.14159265358979323 8-
A=THETA*12./PI-
I=A-
A=(A-I)*60.-
J=A-
A=(A-J)*60.-
 NORMAL EXIT -
 END SUBPROGRAM -
```

```
          SUBROUTINE (ACOEE,XS,YS,ZS,FL,V,K)=RBRAN.(NS,FLAT,FLON,STA
        XYZ,NO,RO,YA)-
          DIMENSION (FL(10))-
          PRECISION (2,STAXYZ,RO)-
          DIMENSION (FN(9,L),C(9,L),ACOEE(0,L),STAXYZ(0,L),U(3),D(3)
        ,ST(10),ACOEF(30,L))-
          PRECISION (2,K,XS,YS,ZS,DSQRT.,DX,DY,DZ,GEOCAR.,FLATM,FLON
        M,HS)-
         L=3-
         FLATM=0.-
         FLONM=0.-
          DO THROUGH (SUM),I=0,1,I.L.NS-
         FLATM=FLATM+FLAT(NO(I))-
SUM      FLONM=FLONM+FLON(NO(I))-
         FLATM=FLATM/(1.*NS)-
         FLONM=FLONM/(1.*NS)-
         HS=160000.-
          CALL SUBROUTINE (XS,YS,ZS)=GEOCAR.(FLATM,FLONM,HS)-
         K=0-
          DO THROUGH (ZER1),I=0,1,I.L.3-
          DO THROUGH (ZER1),J=0,1,J.L.3-
         U(I)=0.-
ZER1     C(I,J)=0.-
RCOMP     DO THROUGH (NORMAL),I=0,1,I.L.NS-
         DX=XS-STAXYZ(NO(I),0)-
         DY=YS-STAXYZ(NO(I),1)-
         DZ=ZS-STAXYZ(NO(I),2)-
         R=DSQRT.(DX.P.2+DY.P.2+DZ.P.2)-
         FL(I)=R-RO(I)-
         ACOEE(I,0)=DX/R-
         ACOEE(I,1)=DY/R-
         ACOEE(I,2)=DZ/R-
         ACOEF(I,0)=ACOEE(I,0)*YA(I)-
         ACOEF(I,1)=ACOEE(I,1)*YA(I)-
         ACOEF(I,2)=ACOEE(I,2)*YA(I)-
         C=C+ACOEF(I,0)*ACOEE(I,0)-
         C(1,1)=C(1,1)+ACOEF(I,1)*ACOEE(I,1)-
         C(2,2)=C(2,2)+ACOEF(I,2)*ACOEE(I,2)-
         C(0,1)=C(0,1)+ACOEF(I,0)*ACOEE(I,1)-
         C(1,2)=C(1,2)+ACOEF(I,1)*ACOEE(I,2)-
         C(0,2)=C(0,2)+ACOEF(I,0)*ACOEE(I,2)-
         U(0)=U(0)+ACOEF(I,0)*FL(I)-
         U(1)=U(1)+ACOEF(I,1)*FL(I)-
NORMAL   U(2)=U(2)+ACOEF(I,2)*FL(I)-
          CALL SUBROUTINE (FN)=SYINVT.(C)-
         D(0)=0.-
         D(1)=0.-
         D(2)=0.0-
          DO THROUGH (XVECT),I=0,1,I.L.3-
          DO THROUGH (XVECT),J=0,1,J.L.3-
XVECT    D(I)=D(I)-FN(I,J)*U(J)-
```

```
          XS=XS+D(0)-
          YS=YS+D(1)-
          ZS=ZS+D(2)-
           DO THROUGH (ZERO),I=0,1,I.L.3-
           DO THROUGH (ZERO),J=0,1,J.L.3-
          U(I)=0.-
ZERO      C(I,J)=0.-
          K=K+1-
           TRANSFER TO (RCOMP) PROVIDED (.ABS.D.G.0.01.OR..ABS.D(1).G
         .0.01.OR..ABS.D(2).G.0.01)-
           DO THROUGH (VCOMP),I=0,1,I.L.NS-
VCOMP     V(I)=FL(I)+ACOEE(I,0)*D(0)+ACOEE(I,1)*D(1)+ACOEE(I,2)*D(2)-
           NORMAL EXIT -
           END SUBPROGRAM -
```

```
          SUBROUTINE (XS,YS,ZS)=RCOMP.(NO,STAXYZ,RA,DEC,R3)-
          LITERALS (RHO,57.295779513)-
          DIMENSION (VX(3),VY(3))-
          DIMENSION (R3(0,L))-
          DIMENSION (V1(3),V2(3),V12(3),STAXYZ(0,L))-
          PRECISION (2,R3,DEC,RA,STAXYZ)-
          LITERALS (L,3)-
         C=.HOP.DEC-
         D=.HOP.RA-
         VX=COS.(C)-
         VX(1)=SIN.(D)*VX-
         VX=COS.(D)*VX-
         VX(2)=SIN.(C)-
         C=.HOP.DEC(1)-
         D=.HOP.RA(1)-
         VY=COS.(C)-
         VY(1)=SIN.(D)*VY-
         VY=COS.(D)*VY-
         VY(2)=SIN.(C)-
          DO THROUGH (VZERT),I=0,1,I.L.3-
         V2(I)=0.-
VZERT    V1(I)=0.-
          DO THROUGH (V1ROT),I=0,1,I.L.3-
          DO THROUGH (V1ROT),J=0,1,J.L.3-
V2ROT    V2(I)=V2(I)+.HOP.R3(I,J)*VY(J)-
V1ROT    V1(I)=V1(I)+.HOP.R3(I,J)*VX(J)-
         V12(0)=.HOP.STAXYZ(NO(1),0)-.HOP.STAXYZ(NO(0),0)-
         V12(1)=.HOP.STAXYZ(NO(1),1)-.HOP.STAXYZ(NO(0),1)-
         V12(2)=.HOP.STAXYZ(NO(1),2)-.HOP.STAXYZ(NO(0),2)-
         R12=SQRT.(V12(0).P.2+V12(1).P.2+V12(2).P.2)-
         V12(0)=V12(0)/R12-
         V12(1)=V12(1)/R12-
         V12(2)=V12(2)/R12-
         CAT=-V2*V12-V2(1)*V12(1)-V2(2)*V12(2)-
         SINA2=SQRT.(1.-CAT.P.2)-
         CAT1=V1*V2+V1(1)*V2(1)+V1(2)*V2(2)-
         SINA3=SQRT.(1.-CAT1.P.2)-
         R1S=R12*SINA2/SINA3-
         XS=.HOP.STAXYZ(NO(0),0)+R1S*V1(0)-
         YS=.HOP.STAXYZ(NO(0),1)+R1S*V1(1)-
         ZS=.HOP.STAXYZ(NO(0),2)+R1S*V1(2)-
          NORMAL EXIT -
          END SUBPROGRAM -
```

```
            SUBROUTINE (R)=ROTATE.(C,L)-
            LITERALS (KDM,3)-
            DIMENSION (R(0,KDM),G(9,KDM),Q(2))-
            PRECISION (2,R,G,C,DSIN.,DCOS.,Q)-
           K=0-
INTO        CONDITIONAL (SWING)-
            PROVIDED (L(K).E.1) OTHERWISE (SWING1)-
           G(0,0)=1-
           G(1,1)=DCOS.(C(K))-
           G(1,2)=DSIN.(C(K))-
           G(2,1)=-G(1,2)-
           G(2,2)=G(1,1)-
           G(1)=0.0-
           G(2)=0.0-
           G(3)=0.0-
           G(6)=0.0-
SWING1      OR PROVIDED (L(K).E.2) OTHERWISE (SWING2)-
           G(0,0)=DCOS.(C(K))-
           G(2,0)=DSIN.(C(K))-
           G(1,1)=1-
           G(0,2)=-G(2,0)-
           G(2,2)=G(0,0)-
           G(1)=0.0-
           G(3)=0.0-
           G(5)=0.0-
           G(7)=0.0-
SWING2      OR PROVIDED (L(K).E.3) OTHERWISE (SWING3)-
           G(0,0)=DCOS.(C(K))-
           G(0,1)=DSIN.(C(K))-
           G(1,0)=-G(0,1)-
           G(1,1)=G(0,0)-
           G(2,2)=1-
           G(2)=0.0-
           G(5)=0.0-
           G(6)=0.0-
           G(7)=0.0-
            END CONDITIONAL -
SWING3      NORMAL EXIT -
SWING      K=K+1-
            PROVIDED (K.G.1), TRANSFER TO (COMP)-
            DO THROUGH (GET),I=0,1,I.L.3-
            DO THROUGH (GET),J=0,1,J.L.3-
GET        R(I,J)=G(I,J)-
            TRANSFER TO (INTO)-
COMP        DO THROUGH (MULT),I=0,1,I.L.3-
           Q(0)=R(0,I)-
           Q(1)=R(1,I)-
            DO THROUGH (MULT),J=0,1,J.L.3-
MULT       R(J,I)=Q(0)*G(J,0)+Q(1)*G(J,1)+R(2,I)*G(J,2)-
            TRANSFER TO (INTO)-
            END SUBPROGRAM -
```

```
          SUBROUTINE (SIGL,SIGM)=SIG.(SIGX,FLAT,FLON,H)-
          DIMENSION (GSIGX(9,L))-
          DIMENSION (SIGX(0,L),SIGL(0,L),G(9,L),GT(9,L),SIGM(0,L),T1
        (3),T2(3))-
          LITERALS (R,6370000.,RHOSEC,206264.81)-
         L=3-
         EK1=SIN.(FLAT)-
         EK2=COS.(FLAT)-
         FK1=SIN.(FLON)-
         FK2=COS.(FLON)-
         G(0,0)=-EK1*FK2-
         G(0,1)=-EK2*FK1-
         G(0,2)=EK2*FK2-
         G(1,0)=-EK1*FK1-
         G(1,1)=EK2*FK2-
         G(1,2)=EK2*FK1-
         G(2,0)=EK2-
         G(2,1)=0.-
         G(2,2)=EK1-
          DO THROUGH (TRN),I=0,1,I.L.3-
          DO THROUGH (TRN),J=0,1,J.L.3-
TRN      GT(I,J)=G(J,I)-
          CALL SUBROUTINE (E,D,G)=MTINV.(T1,T2,3,3)-
          CALL SUBROUTINE (E,D,GT)=MTINV.(T1,T2,3,3)-
          CALL SUBROUTINE (E,GSIGX)=MTMPY.(G,SIGX,3,3,3)-
          CALL SUBROUTINE (E,SIGL)=MTMPY.(GSIGX,GT,3,3,3)-
         RHORH=.HOP.RHOSEC/(R+H)-
         SIGM(0,0)=SIGL(0,0)*RHORH.P.2-
         SIGM(1,1)=SIGL(1,1)*RHORH.P.2-
         SIGM(2,2)=SIGL(2,2)-
         SIGM(0,1)=SIGL(0,1)*RHORH.P.2-
         SIGM(1,0)=SIGM(0,1)-
         SIGM(0,2)=SIGL(0,2)*RHORH-
         SIGM(2,0)=SIGM(0,2)-
         SIGM(1,2)=SIGL(1,2)*RHORH-
         SIGM(2,1)=SIGM(1,2)-
          NORMAL EXIT -
          END SUBPROGRAM -
```

```
      SUBROUTINE (A)=SYINVT.(B)-
      DIMENSION (A(0,KDM),B(0,KDM),S(10))-
      LITERALS (KDM,3)-
      DO THROUGH (CH),I=0,1,I.L.3-
     K=2*I-
      DO THROUGH (CH),J=I,1,J.L.3-
CH   S(K+J)=B(I,J)-
     S(5)=S(1)*S(1)-
     S(7)=S(2)*S(2)-
     S(8)=S(4)*S(4)-
     S(9)=S*S(3)*S(6)+2.*S(1)*S(2)*S(4)-S(3)*S(7)-S*S(8)-S(6)*S(
    5)-
     A=(S(3)*S(6)-S(8))/S(9)-
     A(0,1)=(S(2)*S(4)-S(1)*S(6))/S(9)-
     A(0,2)=(S(1)*S(4)-S(2)*S(3))/S(9)-
     A(1,1)=(S*S(6)-S(7))/S(9)-
     A(1,2)=(S(1)*S(2)-S*S(4))/S(9)-
     A(2,2)=(S*S(3)-S(5))/S(9)-
     A(1,0)=A(0,1)-
     A(2,0)=A(0,2)-
     A(2,1)=A(1,2)-
      NORMAL EXIT -
      END SUBPROGRAM -
```

```
            SUBROUTINE (G)=TRANSP.(N)-
            PRECISION (2,G,P)-
            DIMENSION (G(0,N))-
           I=0-
CONJ       J=I+1-
JC         P=G(I,J)-
           G(I,J)=G(J,I)-
           G(J,I)=P-
           J=J+1-
            PROVIDED (J.L.N), TRANSFER TO (JC)-
           I=I+1-
            PROVIDED (I.L.N-1), TRANSFER TO (CONJ)-
            NORMAL EXIT -
            END SUBPROGRAM -
```

For the Department of Geodetic Science

Project Supervisor Ivan I. Mueller Date 12.26.1967

For the Ohio State University Research Foundation

Executive Director Robert C. Stephenson Date 12/28/67