SATELLITE ALTIMETRY APPLICATIONS FOR MARINE GRAVITY: SOFTWARE

Nick Christou
Alfred Kleusberg
John Mantha
Spiros Pagiatakis

Department of Surveying Engineering
University of New Brunswick
P.O. Box 4400
Fredericton, N.B.
Canada
E3B 5A3

February 1987

TABLE OF CONTENTS

INTRODUCTION

This Technical Memorandum describes the software packages that were developed in the Department of Surveying Engineering at the University of New Brunswick during the research done for the Bedford Institute of Oceanography, Energy, Mines and Resources Canada, under DSS contract number 0SC84-00472. The theoretical background and results of this research are found in the final contract report entitled *Satellite Altimetry Applications for Marine Gravity*, prepared for the Scientific Authority, Dr. J. Woodside, and dated June 1986.

The software described herein has been mainly written on the UNB IBM mainframe computer in FORTRAN 77. In addition, individual computer programs have been developed and written in FORTRAN 77 on the Department-owned HP-1000 computer.

This Technical Memorandum contains information regarding the general description of the developed programs, the general purpose of the algorithms used, general limitations and restrictions and general descriptions of I/O operations. It also contains detailed information on the use of the individual programs regarding handling of input values/files. In addition, compilation listings are provided together with the program listings.

1. GENERAL DESCRIPTION OF PROGRAMS

1.1 Program GRIDALT

1.1.1 General Description

This program is designed to perform a bilinear surface interpolation of adjusted satellite altimetry data and create a regular grid of satellite altimetry heights, based on moving data windows with variable width.

1.1.2 Input Parameters

(a) The boundaries of the area for which the regular grid will be generated (in decimal degrees).

(b) The grid interval in latitude and longitude (in minutes of arc) respectively.

(c) The maximum, intermediate, and minimum half-widths of the working windows in the latitude and longitude directions, respectively (in decimal degrees).

1.1.3 Output Parameters

The coordinates of the generated points with the associated interpolation value of sea surface height and its estimated standard deviation.

1.1.4 Documentation

The description of the surface modelling function is given in the final contract report, section 3.2.5.

1.1.5 Limitations and Restrictions

(a) The program has not the diagnostic capability to distinguish between grid points in the sea and grid points on near-shore and island masses. Thus the result is grid interpolated values over land in those areas mentioned. A certain masking procedure must follow thereafter to eliminate the artifacts of the interpolation scheme. Reference to it is also given in the final contract report, section 3.2.5.

(b) Other limitations of the program involve the size of the arrays that hold the pertinent 'raw' altimetry data points. Currently the maximum size of input data array is set to 50 000 elements (one-dimensional arrays), but it depends on the density of altimetry data points and the maximum half-width parameter of the working window in the latitude direction and the length of the data strip formed.

1.2 Program LEVITUS

1.2.1 General Description

This program generates worldwide $1° \times 1°$ matrix representations of dynamic topography maps with respect to different reference levels.

1.2.2 Input Parameters

No external input parameters are necessary. The selection of the desired reference level is done inside the program using a DATA statement.

1.2.3 Output Parameters

The generated grid node geographical coordinates and the associated dynamic topography value.

1.2.4 Documentation

Details on Levitus's dynamic topography maps are given in the final contract report, section 3.4.1.

1.2.5 Limitations and Restrictions

There is no limitation embedded in this program. The program has the versatility to generate a portion of the selected dynamic topography map through appropriately placed (but generally flagged) IF statements, when the desired matrix is built.

1.3 Program LEVINT

1.3.1 General Description

This program was specifically designed to generate a finer grid of dynamic topography (finer than the $1° \times 1°$ grid produced by the program LEVITUS). It uses linear interpolation in one dimension and, if run twice for both longitude and latitude directions, it produces a $10' \times 10'$ grid of dynamic topography. It is not a general purpose program.

1.3.2 Input Parameters

No external input is necessary.

1.3.3 Output Parameters

In two steps, the longitude and latitude (or vise versa) interpolated lines, given in terms of a matrix of grid points with their geographical coordinates and the interpolated values of dynamic topogrpahy on the finer grid.

1.3.4 Documentation

Some reference on this program's function is given in the final contract report, section 3.4.1.

Essentially, the program performs linear interpolation between two consecutive grid nodes of the $1° \times 1°$ field, by determining the height difference and then the slope in a local Cartesian coordinate system and generating interpolated values at the desired finer grid spacing.

When one of the two consecutive grid nodes used has a default value for dynamic topography, then the program does not perform any interpolation. It just fills in default values.

1.3.5 Limitations and Restrictions

(a) Since this program is a special purpose one, it has inherent limitations, such as array size (depending on the extent of the $1° \times 1°$ field to be interpolated).

(b) The same program can be run for both longitude and latitude directions with minor changes. Actually both 'codes' appear in the program. However, the statements for the run in the latitude direction are commented cards. An exchange in the comment cards between longitude and latitude is very easy to perform.

(c) This program is useful when a fast sorting routine is available to order the data points of the input file(s) in respective order (i.e., all longitude points within one latitude for the longitude run or all latitude points within one longitude for the latitude run, respectively).

(d) Because of the nature of the problem for which this program was developed, certain lines of the code conform specifically to the structure of the file produced by program LEVITUS. Thus caution is needed to use the two programs in a consistent way. One example is the handling of the default values in the $1° \times 1°$ file when processed with program LEVINT.

1.4 Program ALTINT

1.4.1 General Description

This program performs a two-dimensional numerical integration over an altimetry field of residual geoidal heights using a specially developed kernel. It operates either in a point or grid mode.

1.4.2 Input Parameters

(a) mode of operation (point or grid mode)
(b) number of computation points (if in point mode)
(c) geographical coordinates of computation points (if in point mode)
(d) thickness of cocentric rings formed and used in the numerical integration scheme
(e) maximum radius of circular integration area and equivalent radius of Stokes's cap
(f) the number of kernel values and the values to be used in the integration
(g) boundaries of available grid of altimetry residual geoidal heights
(h) boundaries of area to perform computations and step (if in grid mode).

1.4.3 Output Parameters

Geographical coordinates of computation point and integration result with its associated standard deviation.

1.4.4 Documentation

The details of the algorithm used in this program are given in the final contract report, section 5.4.

1.4.5 Limitations and Restrictions

(a) The built-in restriction of this program is the use of a DIRECT ACCESS file that contains the altimetry data input (in the form of grid values and associated standard deviations). This in effect minimizes the search time for the relevant input data when computations are performed, especially in the point mode.

(b) The limitation of the program regarding the chosen thickness of rings is dictated by the density of the input grid of data. As a rule of thumb, the thickness of rings should be compatible with the grid step of available data. A small deviation from this rule of thumb is permissible. If the rule of thumb is not respected, then error messages will appear in the form "one or more rings have no data points."

(c) The geographical area of computations is always smaller than the area of available data. Depending on the selected 'Stokes's' radius of integration (i.e., on the kernel to be used) there is a different amount of margins between the area of computations and the area of data coverage. The larger the 'Stokes's' radius the larger the margin will be. Currently the program has an 'error' detection capability that can accommodate most of the kernels that can be used. The 'error' detection criterion may have to be altered when very large 'Stokes's' radii are used (say, larger than 3° or 4°).

(d) When, during the integration process for a specific computation point, default values in the input data set occur, the program sets these default values to ZERO, so there is no contribution from these input points to the final integration result.

1.5 Program GPOT

1.5.1 General Description

This program computes geoid-related parameters, such as geoidal height, gravity disturbance (or

gravity anomaly), and deflections of the vertical for any point on the earth, from a given data set of spherical harmonics (potential coefficients). The program can be used either in point or grid mode.

1.5.2 Input Parameters

(a) The geographical coordinates of the computation point.

(b) The file containing the set of geopotential coefficients. This must be always in input logical unit 12.

(c) The desired degree of expansion of the spherical harmonics (in the BLOCK data statement).

1.5.3 Output Parameters

(a) The geographical coordinates of the computation point.

(b) The geoidal height, gravity disturbance (or gravity anomaly), and the deflections of the vertical.

1.5.4 Documentation

Details of the algorithm to compute the geoidal parameters are given in *Manuscripta Geodaetica*, Vol. 8, 1983, pp. 249-272. This paper includes the computer programs as well.

1.5.5 Limitations and Restrictions

(a) The most notable limitation of the program is the currently set maximum degree of expansion of the gravity field that can be used: It is 180. Therefore, only gravity fields (or portions thereof) up to 180×180 can be actually computed.

(b) The normal gravity field (U) used for the computation of the disturbing potential (T) ($T = W - U$) currently is the one generated by the Geodetic Reference System 1980 (GRS'80).

(c) Any input data set of spherical harmonics must be stored as follows:

for each degree all the orders; i.e.,

```
.
.
.
2   0
2   1
2   2
3   0
3   1
3   2
3   3
.
.
.
```

(d) For efficient storage, the spherical harmonics are in binary format. This is the way the subroutine that loads the harmonics (LOADCS) works at the moment.

(e) The potential coefficient data set must be in the form of fully normalized spherical harmonics. If not, then changes in the BLOCK DATA statement must be made to accommodate data sets that are non-normalized. Currently, the parameter in the BLOCK DATA statement is set for fully normalized spherical harmonics.

(f) The current maximum dimensions of certain arrays used in the program are based on the maximum allowed degree of expansion 180.

(g) This version of the program (see Documentation) has a maximized efficiency for grid geoid computations. It exploits the benefit of computing only once the Legendre polynomials for all computation points having the same latitude. Appropriate changes in the main program can be made to accommodate point mode computations.

1.6 Program SGSINT

1.6.1 General Description

This program performs Stokes's integration over a data set of sparse point gravity anomalies. It can operate either in point or in grid mode.

1.6.2 Input Parameters

(a) The mode of operation (point or grid)

(b) The Stokes's integration radius.

(c) The geographical coordinates of the boundaries of a sub-area where data is available (i.e.,

block A).

(d) The geographical coordinates of the boundaries of the computation area and the grid size when in grid mode.

(e) The number of computation points (if in point mode).

(f) The geographical coordinates of the computation points (if in point mode).

1.6.3 Output Parameters

(a) The computation point coordinates.

(b) The integration result and its associated standard deviation.

1.6.4 Documentation

The details of the algorithm used are given in the final contract report, sections 5.1 and 5.2.

1.6.5 Limitations and Restrictions

(a) The user of the program must know a priori the geographical area of data coverage of the input point gravity anomaly file.

(b) Based on this information, the user can decide upon the boundaries of the sub-area (i.e., block A) that will be read in, and which therefore have to conform to both overall data coverage and desired computation area.

In general, block A boundaries have to be at least a distance 2 * RHO (RHO is the Stokes cap radius) inside the data file boundaries, and at least by the same amount larger than the desired area of computations.

All these precautions are taken to ensure that when the point gravity anomaly data are loaded into the arrays belonging to block A, we do not exceed the maximum array size (set to 50 000 elements), especially for areas of very dense coverage, such as the Labrador coast.

(c) Another built-in restriction to this program is the decision of whether or not to perform the integration. The decision is based on the fact that: If there are more than 1 data points and all lie inside a cap of radius RHO/3, then assign default values for the integration at that computation point. This is based on the fact that certain data distribution patterns cause ill-conditioning of the system of normal equations when the polynomial fit to gravity anomalies is done.

1.7 Programs for the Computation of the "Truncation" Kernel

1.7.1 General Description

There are two interactive programs: Program COEFF and program FPHIO. Program COEFF produces a number of numerical values for the "truncation" kernel for equidistant arguments. The range of the argument varies in steps of 0.1 degree in the interval $0 < \psi < 15$ degrees. Program FPHIO computes the optimal truncation radius for the integration over altimetric geoidal heights.

These two main programs call three subroutines: TCPAL for the computation of the so-called truncation coefficients using M. Paul's algorithm [*Bulletin Géodésique*, 1973, pp. 413-425], LGNDR for the recursive computation of Legendre's polynomials, and DOPRO for the evaluation of the scalar product of two vectors.

1.7.2 Input Parameters

Both programs prompt the user to give appropriate input data through the terminal keyboard. Hard copies of the terminal screen for sample runs are included in section 2.7.

1.7.3 Output Parameters

The output of program COEFF is a series of ψ and $T(\psi)$ in steps of 0.1 degrees. These values are written to the terminal screen and to an output disk file chosen by the user. This file is part of the input for program ALTINT (see section 1.4). A printout of the file generated by the sample run is included in section 2.7.

The output of program FPHIO consists of a single number, written to the terminal screen: the radius ϕ_0 for the "truncation" integration.

1.7.4 Documentation

Sections 4 and 5 of the contract report.

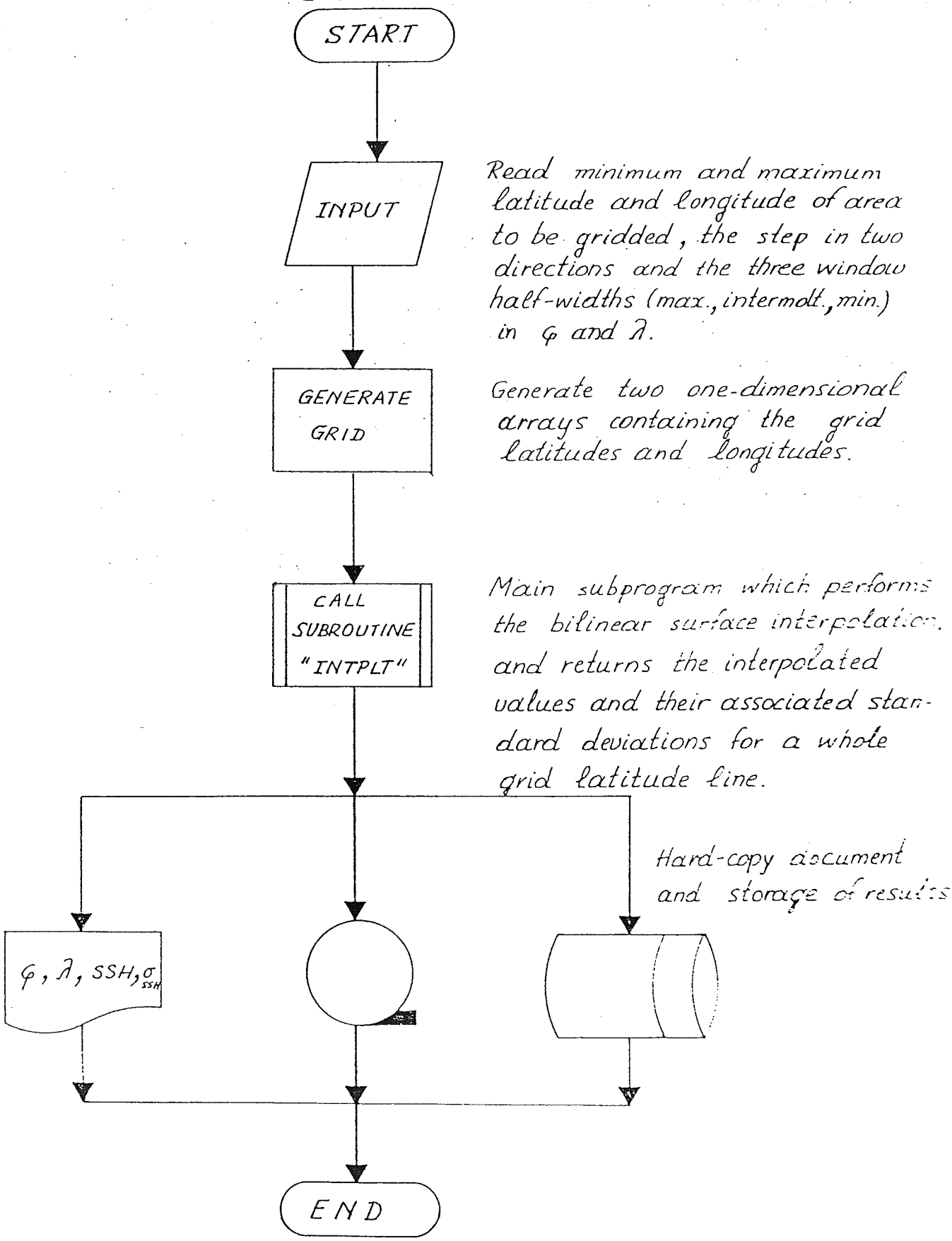
1.7.5 Limitations

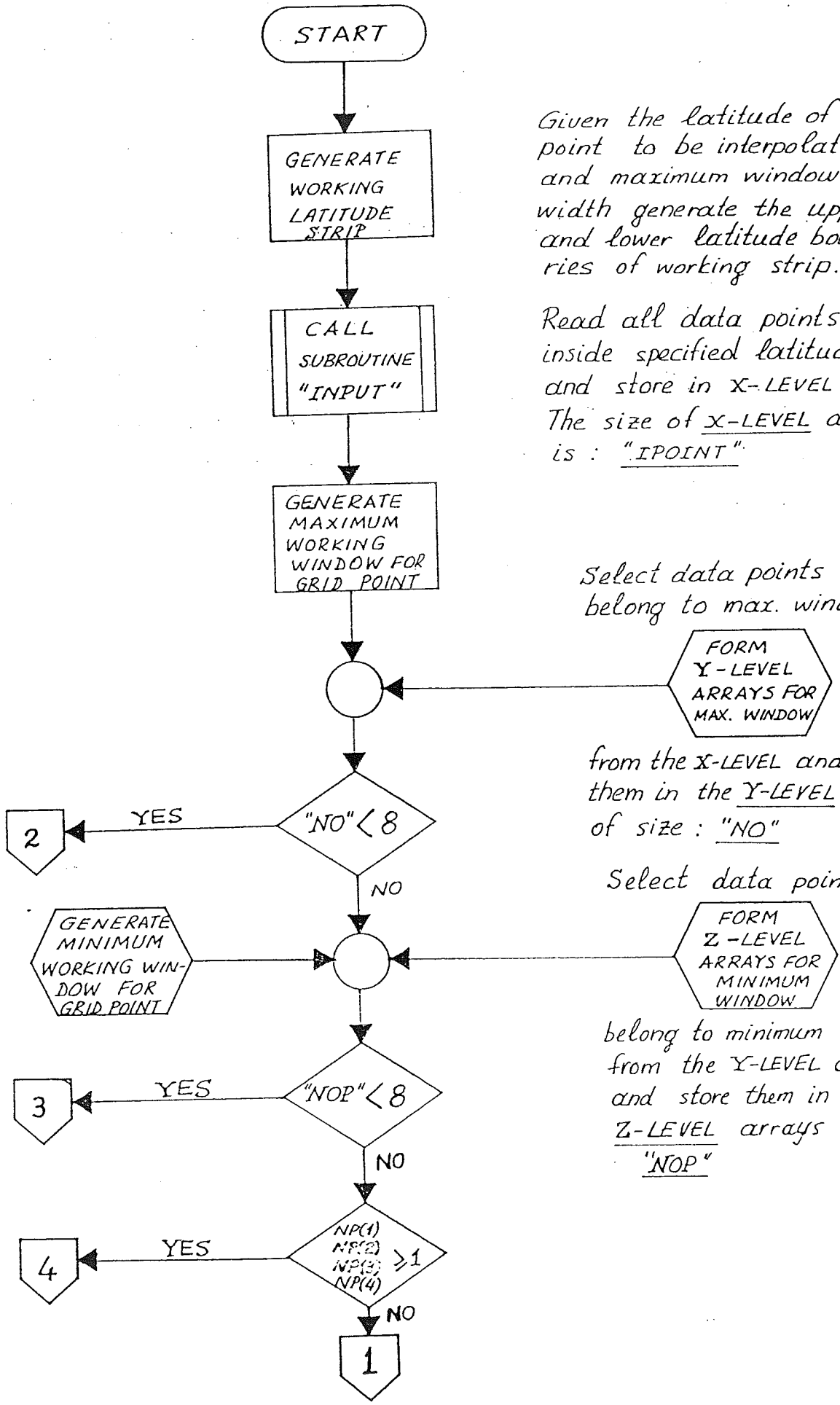
None.

2. PROGRAM LISTINGS

2.1 Program GRIDALT

MAIN PROGRAM "GRIDALT"



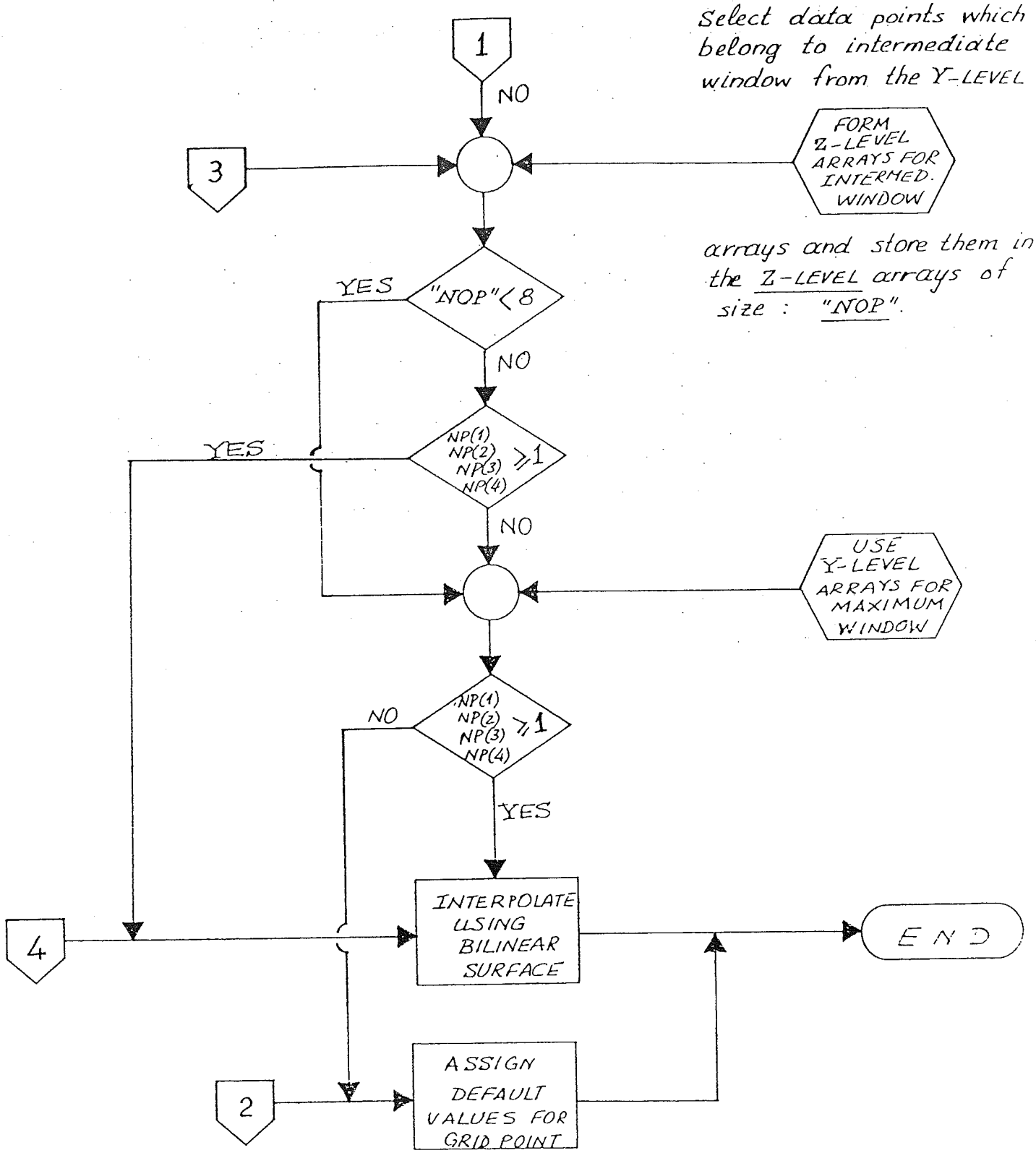


Given the latitude of the point to be interpolated and maximum window half-width generate the upper and lower latitude boundaries of working strip.

Read all data points falling inside specified latitude strip and store in X-LEVEL arrays. The size of X-LEVEL arrays is : "IPOINT".

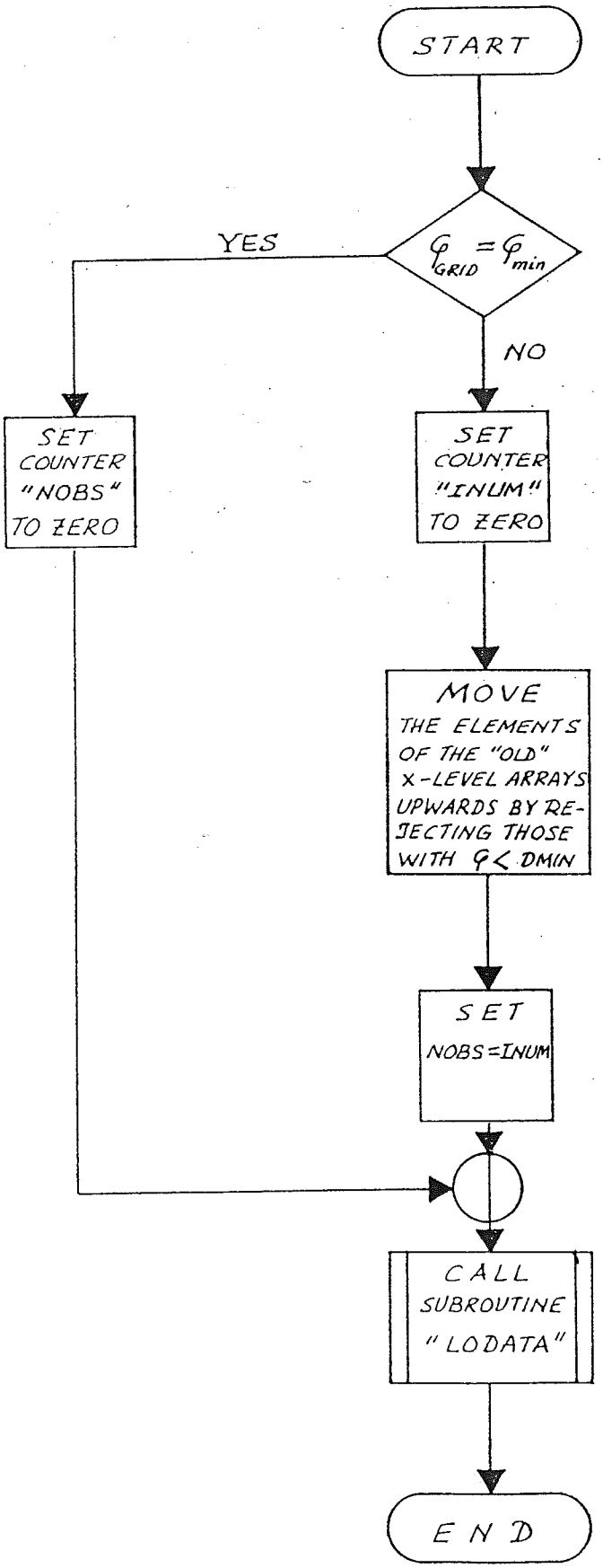
Select data points which belong to max. window from the X-LEVEL and store them in the Y-LEVEL arrays, of size : "NO"

Select data points which belong to minimum window from the Y-LEVEL arrays and store them in the Z-LEVEL arrays of size: "NOP"



Select data points which belong to intermediate window from the Y-LEVEL arrays and store them in the Z-LEVEL arrays of size : "NOP".

SUBPROGRAM " INPUT "

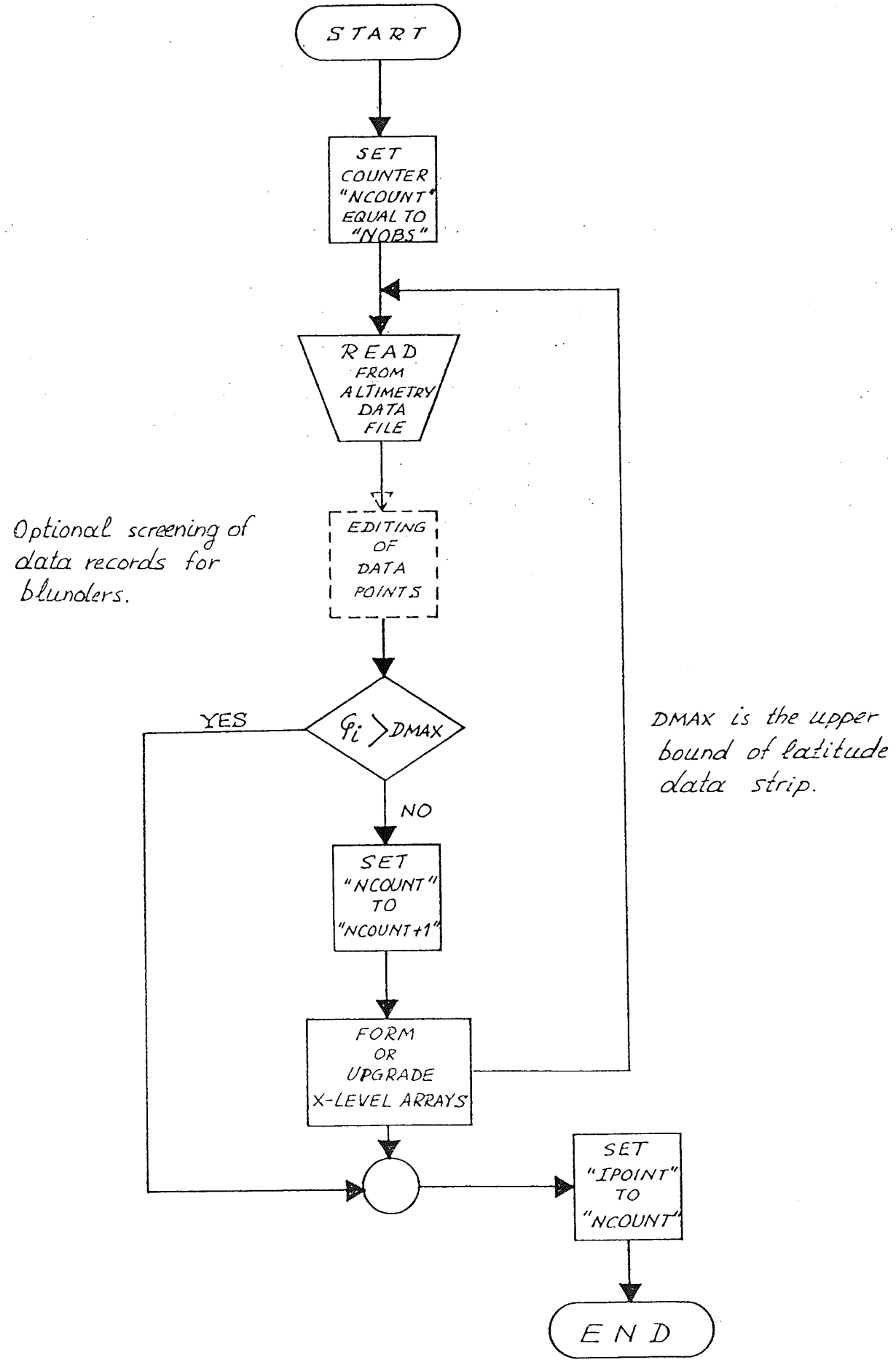


If the grid latitude is the first latitude line, then set number of data points in latitude strip to zero and load x-LEVEL arrays. Otherwise set new counter to zero.

For every subsequent latitude grid line update the elements of x-LEVEL arrays by using every common point in "new" and "old" latitude strip and rejecting the others. This is a partially updated x-LEVEL array. DMIN is the lower bound of new latitude strip(data strip

Read from file either for the first time, or, for subsequent times data points and form, or, upgrade x-LEVEL arrays, respectively.

SUBPROGRAM "LODATA"



```
//OPT30SU JOB NOTIFY=6461
/*SETUP    SLOT=P0147    VOLUME=NL6250    NOWRITE
/*SERVICE  DEFERRED
/*JOBPARM T=80,L=999,R=4096,PRINT=ALL
//STEP1   EXEC FORTVCLG,REGION=4096K,PARM.FORT='OPTIMIZE(3)'
//FORT.SYSIN   DD  *

C-----------------------------------------------------------------------
C***********************************************************************
C                                                                      *
C  PROGRAM NAME :    GRIDALT                                           *
C  FUNCTION     :  GENERATION OF A REGULAR GRID OF ADJUSTED            *
C                  SEA SURFACE HEIGHTS FROM SEASAT ADJUSTED           *
C                  ALTIMETRY DATA.                                    *
C COMPILER      :  VS FORTRAN VERSION 2                               *
C AUTHOR        :  NICK CHRISTOU                                      *
C HISTORY       :  JULY   13, 1985 - VERSION 1.0                      *
C                  AUGUST 20, 1985 - VERSION 2.0  (OPTIMIZED)         *
C REFERENCE     :  U.N.B.  TECHNICAL REPORT  #                        *
C                                                                     *
C EXTERNALS     :  INTPLT , INPUT , LODATA , SPIN .                   *
C***********************************************************************
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION GPHI(300),FVALUE(600),SDEVF(600)
      DIMENSION XLAT(40000),XLON(40000),XASSH(40000),XSD(40000)
C
      COMMON /STRIP/XLAT,XLON,XASSH,XSD
      COMMON GLON(600),PHIMIN,PHIMAX,DLAMIN,DLAMAX,HWMAX1,HWMAX2,
     @                 HWINT1,HWINT2,HWMIN1,HWMIN2,IPOINT,NOBS,JJ
C
      READ(5,1000) PHIMIN,DLAMIN,PHIMAX,DLAMAX,INCPHI,INCLAM
 1000 FORMAT(4F9.4,2I3)

C-----------------------------------------------------------------------
C            PHIMIN :
C            PHIMAX :   REPRESENT THE BOUNDARIES OF THE AREA FOR WHICH
C            DLAMIN :   THE REGULAR GRID WILL BE GENERATED (IN DEC. DEG.)
C            DLAMAX :
C
C            INCPHI :   GRID INTERVAL IN LATITUDE  (IN MINUTES)
C            INCLAM :   GRID INTERVAL IN LONGITUDE (IN MINUTES)
C-----------------------------------------------------------------------

      READ(5,1001) HWMAX1,HWMAX2,HWINT1,HWINT2,HWMIN1,HWMIN2
 1001 FORMAT(6F5.2)

C-----------------------------------------------------------------------
C  HWMAX1  :              MAXIMUM
C  HWINT1  : ARE THE  INTERMEDIATE  HALF-WIDTHS OF THE WORKING WINDOWS
C  HWMIN1  :              MINIMUM           IN THE LATITUDE DIRECTION
C
C  HWMAX2  :              MAXIMUM
C  HWINT2  : ARE THE  INTERMEDIATE  HALF-WIDTHS OF THE WORKING WINDOWS
C  HWMIN2  :              MINIMUM           IN THE LONGITUDE DIRECTION
C-----------------------------------------------------------------------
C    \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
C                 GENERATE  NOW  THE  DESIRED  GRID .
C    ////////////////////////////////////////////////////////////////
C-----------------------------------------------------------------------
```

```
      DELPHI = PHIMAX - PHIMIN
      DELLAM = DLAMAX - DLAMIN
C
        DPHI = INCPHI/60.0D0
      DLAMDA = INCLAM/60.0D0
C
          II = DELPHI/DPHI + 1.0D0
          JJ = DELLAM/DLAMDA + 1.0D0

C-----------------------------------------------------------------------
C          II : NUMBER OF GRID POINTS IN LATITUDE (DEFINES DIM. OF GPHI)
C          JJ :  -"-    -"-     -"-     LONGITUDE (DEFINES DIM. OF GLON)
C-----------------------------------------------------------------------

     GPHI(1) = PHIMIN
     GLON(1) = DLAMIN
C
      DO   11  I=1,(II-1)
         GPHI(I+1)=GPHI(1)+DPHI*I
   11 CONTINUE
      DO   22  J=1,(JJ-1)
         GLON(J+1)=GLON(1)+DLAMDA*J
   22 CONTINUE

C-----------------------------------------------------------------------
C  THE GENERATED GRID IS CONTAINED IN THE ARRAYS GPHI(I) AND GLON(I)
C=======================================================================
C                          NEXT   STEP
C=======================================================================
C
C      GENERATION OF THE FUNCTIONAL VALUES ON THE REGULAR GRID
C          ( HERE THE FUNCTION IS THE SEA SURFACE HEIGHT )
C-----------------------------------------------------------------------

      DO  88  K=1,II
                    GLAT=GPHI(K)
        CALL INTPLT(GLAT,FVALUE,SDEVF)

        WRITE(6,9000) (GLAT,GLON(L),FVALUE(L),SDEVF(L),L=1,JJ)

   88 CONTINUE
 9000 FORMAT(F8.4,F10.4,F8.2,F6.2)
      STOP
      END

      SUBROUTINE INTPLT(GLAT,FVALUE,SDEVF)
C-----------------------------------------------------------------------
C***********************************************************************

      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER NP(4)
C
      DIMENSION XLAT(40000),XLON(40000),XASSH(40000),XSD(40000)
      DIMENSION YLAT(6000),YLON(6000),YVAL(6000),YDEV(6000)
      DIMENSION ZLAT(6000),ZLON(6000),ZVAL(6000),ZDEV(6000)
C
      DIMENSION X(6000),Y(6000),Z(6000),W(6000)
      DIMENSION FF(6000,4),DD(10,10),CC(4),XX(4)
C
      DIMENSION FVALUE(600),SDEVF(600)
```

```
C
      COMMON /STRIP/ XLAT,XLON,XASSH,XSD
      COMMON GLON(600),PHIMIN,PHIMAX,DLAMIN,DLAMAX,HWMAX1,HWMAX2,
     @                HWINT1,HWINT2,HWMIN1,HWMIN2,IPOINT,NOBS,JJ
C
C
          PI = 3.14159265358979300

C-----------------------------------------------------------------------
C  HWMAX1  :              MAXIMUM
C  HWINT1  : ARE THE  INTERMEDIATE  HALF-WIDTHS OF THE WORKING WINDOWS
C  HWMIN1  :              MINIMUM          IN THE LATITUDE DIRECTION
C
C  HWMAX2  :              MAXIMUM
C  HWINT2  : ARE THE  INTERMEDIATE  HALF-WIDTHS OF THE WORKING WINDOWS
C  HWMIN2  :              MINIMUM          IN THE LONGITUDE DIRECTION
C
C=======================================================================
C
C     FORM  THE WORKING STRIP OF LATITUDE BASED ON THE LATITUDE OF
C                THE COMPUTATION POINT : I.E. ,  GLAT .
C
C     DMIN :         LOWER BOUNDARY OF THE LATITUDE STRIP
C     DMAX :         UPPER    -"-      -"-    -"-    -"- .
C-----------------------------------------------------------------------

      DMIN = GLAT - HWMAX1
      DMAX = GLAT + HWMAX1

C-----------------------------------------------------------------------
C     READ ALL THE ALTIMETRY DATA POINTS WHICH FALL INSIDE THE
C     ABOVE SPECIFIED LATITUDE STRIP AND STORE THEM IN ARRAYS :
C
C     XLAT(I) :                       LATITUDE
C     XLON(I) :                       LONGITUDE
C             :  CONTAINS THE,                        ,OF EACH DATA POINT
C     XASSH(I):                ADJ. SEA SURF. HEIGHT
C     XSD(I)  :                  ST. DEV. OF ASSH
C
C  THE TOTAL NUMBER OF POINTS FALLEN INSIDE THE STRIP IS: *** IPOINT ***
C-----------------------------------------------------------------------

      CALL INPUT(GLAT,DMIN,DMAX)

C-----------------------------------------------------------------------
C     FORM NOW THE MAXIMUM WORKING WINDOW AROUND EACH COMPUTATION POINT
C     BASED ON THE LONGITUDE OF THE GRID NODE :   I.E. , GLON(KK) .
C
C     DLMIN :   THE LEFT  BOUNDARY OF THE MAXIMUM WORKING WINDOW (KK).
C     DLMAX :   THE RIGHT    -"-      -"-   -"-     -"-     -"-  (KK) .
C-----------------------------------------------------------------------

      DO 66 KK=1,JJ
            DLMIN = GLON(KK) - HWMAX2
            DLMAX = GLON(KK) + HWMAX2

C-----------------------------------------------------------------------
C     LOAD NOW THOSE DATA POINTS FROM THE LATITUDE STRIP WHICH BELONG
C     TO THE MAXIMUM WINDOW (KK) , AND PERFORM THE PRELIMINARY TEST
C     ON THE NUMBER OF POINTS , NO , FALLEN INSIDE THIS WINDOW :
C                        (  I.E.,  NO >= 8 )
```

```
C-----------------------------------------------------------------------
      ICOUNT=0
          DO  77  I=1,IPOINT
               IF(XLON(I).GT.DLMAX.OR.XLON(I).LT.DLMIN) GO TO 77
                  ICOUNT = ICOUNT+1
             YLAT(ICOUNT) = XLAT(I)
             YLON(ICOUNT) = XLON(I)
             YVAL(ICOUNT) = XASSH(I)
             YDEV(ICOUNT) = XSD(I)
   77 CONTINUE
             NO = ICOUNT
          IF(NO.LT.8)  GO TO 7777

C-----------------------------------------------------------------------
C NOTE 1 :   IF THE NUMBER OF POINTS , NO , FOUND IN THE MAXIMUM WINDOW
C            IS LESS THAN 8 , THEN THE PROGRAMME IS DIRECTED TO
C            STATEMENT  7777  WHICH ASSIGNS A DEFAULT VALUE OF HEIGHT
C            AND STANDARD DEVIATION TO THIS GRID NODE .
C
C NOTE 2 :   THE DATA POINTS BELONGING TO THE MAXIMUM WINDOW ARE
C            CONTAINED IN THE ARRAYS :
C            YLAT(I) , YLON(I) , YVAL(I) , YDEV(I) .
C-----------------------------------------------------------------------
C
C=======================================================================
C     GENERATE NOW THE MINIMUM WORKING WINDOW TO BE USED FOR THE
C     INTERPOLATION OF THE HEIGHT FUNCTION ON THE (KK) GRID NODE.
C
C     FMIN   :  THE LOWER (LATITUDE) BOUNDARY OF THE MINIMUM WINDOW
C     FMAX   :  THE UPPER     -"-         -"-      -"-   -"-    -"-
C     FLMIN  :  THE LEFT (LONGITUDE) BOUNDARY OF THE MINIMUM WINDOW
C     FLMAX  :  THE RIGHT     -"-         -"-      -"-   -"-    -"-
C=======================================================================

       FMIN = GLAT - HWMIN1
       FMAX = GLAT + HWMIN1
      FLMIN = GLON(KK) - HWMIN2
      FLMAX = GLON(KK) + HWMIN2
C
      LCOUNT=0
         DO 111  I=1,NO
                 IF(YLAT(I).GT.FMAX.OR.YLAT(I).LT.FMIN) GO TO 111
                 IF(YLON(I).GT.FLMAX.OR.YLON(I).LT.FLMIN) GO TO 111
                 LCOUNT = LCOUNT+1
            ZLAT(LCOUNT)=YLAT(I)
            ZLON(LCOUNT)=YLON(I)
            ZVAL(LCOUNT)=YVAL(I)
            ZDEV(LCOUNT)=YDEV(I)
  111 CONTINUE
             NOP=LCOUNT
          IF(NOP.LT.8) GO TO 3333

C-----------------------------------------------------------------------
C NOTE 3 :   IF THE NUMBER OF POINTS , NOP , IN THE MINIMUM WINDOW
C            IS LESS THAN 8 , THEN THE PROGRAMME IS DIRECTED TO
C            STATEMENT  3333  WHICH ASSIGNS THE NEW WINDOW BOUNDARIES
C            DEFINED BY HWINT1 AND HWINT2 , AND REPEATS THE SAME TEST.
C
C NOTE 4 :   THE DATA POINTS BELONGING IN THE MINIMUM WINDOW ARE
C            CONTAINED IN THE ARRAYS :
```

```
C           ZLAT(I) , ZLON(I) , ZVAL(I) , ZDEV(I) .
C
C NOTE 5 :  THE NEXT TEST PERFORMED IS RELATED TO DATA-POINT
C           DISTRIBUTION INSIDE THE MINIMUM WINDOW .
C
C           NP(1) , NP(2) , NP(3) , NP(4) :
C           THEY ARE ARRAY ELEMENTS
C
C           REPRESENTING THE NUMBER OF POINTS
C
C           IN EACH QUQDRANT OF THE MINIMUM
C
C           WINDOW IN THE FOLLOWING MANNER ;
C
C  THE TOTAL NUMBER OF DATA POINTS IN THIS WINDOW IS : ****  NOP   ****
C-----------------------------------------------------------------------
```

NP(2)	NP(1)
NP(4)	NP(3)

```
        NP(1)=0
        NP(2)=0
        NP(3)=0
        NP(4)=0
     DO  222  I=1,NOP
            N=0
              IF(ZLAT(I).LT.GLAT)  N=2
              IF(ZLON(I).LT.GLON(KK))  N=N+2
              IF(ZLON(I).GT.GLON(KK))  N=N+1
           NP(N)=NP(N)+1
 222 CONTINUE

     IF(NP(1).GE.1.AND.NP(2).GE.1.AND.
    &   NP(3).GE.1.AND.NP(4).GE.1) GO TO 999

C-----------------------------------------------------------------------
C NOTE 6 :  IF THE NUMBER OF DATA POINTS INSIDE THE MINIMUM WINDOW
C           IS GREATER THAN  8  AND IN ADDITION THERE IS AT LEAST  1
C           DATA POINT IN EACH QUADRANT OF THE WINDOW THEN THE PROGRAMME
C           IS DIRECTED TO STATEMENT  999  WHICH IS THE FIRST
C           STATEMENT OF THE INTERPOLATION ALGORITHM .
C
C NOTE 7 :  IF EITHER OF THE ABOVE CONDITIONS IS NOT SATISFIED THEN
C           THE PROGRAMME IS AGAIN DIRECTED TO STATEMENT  3333  WHICH
C           WILL ASSIGN THE NEXT WINDOW BOUNDARIES ( IN THIS CASE
C           INTERMEDIATE SIZE WINDOW ) , AND REPEAT ONCE MORE THE SAME
C           OPERATIONS AS BEFORE .
C-----------------------------------------------------------------------

 3333  FMIN = GLAT - HWINT1
       FMAX = GLAT + HWINT1
      FLMIN = GLON(KK) - HWINT2
      FLMAX = GLON(KK) + HWINT2
C
      JCOUNT=0
           DO  333  I=1,NO
                IF(YLAT(I).GT.FMAX.OR.YLAT(I).LT.FMIN)  GO  TO 333
                IF(YLON(I).GT.FLMAX.OR.YLON(I).LT.FLMIN) GO TO 333
                  JCOUNT=JCOUNT+1
              ZLAT(JCOUNT)=YLAT(I)
              ZLON(JCOUNT)=YLON(I)
              ZVAL(JCOUNT)=YVAL(I)
              ZDEV(JCOUNT)=YDEV(I)
  333 CONTINUE
```

```
      NOP=JCOUNT
      IF(NOP.LT.8) GO TO 4444

C-----------------------------------------------------------------------
C NOTE  8 :  IF THE NUMBER OF POINTS , NOP , IN THE INTERMEDIATE
C            WINDOW IS LESS THAN  8  , THEN THE PROGRAMME IS DIRECTED
C            TO STATEMENT  4444  WHICH IS GOING TO CHECK ONLY ABOUT
C            THE DISTRIBUTION OF DATA POINTS IN THE MAXIMUM WINDOW,
C            SINCE THE PRELIMINARY TEST ON THE TOTAL NUMBER OF POINTS
C            IN THE MAX. WINDOW HAS ALREADY BEEN DONE .
C
C NOTE  9 :  THE ARRAYS :  ZLAT(I) , ZLON(I) , ZVAL(I) , ZDEV(I)
C            CONTAIN NOW THE DATA POINTS FOUND IN THE INTERMEDIATE
C            WORKING WINDOW .
C
C NOTE 10 :  THE NEXT TEST PERFORMED IS RELATED TO THE DATA-POINT
C            DISTRIBUTION WITHIN THE INTERMEDIATE WINDOW .
C            NP(1) , NP(2) , NP(3) , NP(4) REPRESENT NOW THE NUMBER OF
C            POINTS FOUND IN EACH QUADRANT OF THE INTERMEDIATE WINDOW
C            AND THEY HAVE THE SAME DESIGNATION AS IN THE FIGURE OF
C            NOTE # 5 .
C
C   THE TOTAL NUMBER OF POINTS IS DESIGNATED BY :  ****   NOP   ****
C-----------------------------------------------------------------------

         NP(1)=0
         NP(2)=0
         NP(3)=0
         NP(4)=0
      DO  444  I=1,NOP
            N=0
                IF(ZLAT(I).LT.GLAT)  N=2
                IF(ZLON(I).LT.GLON(KK))  N=N+2
                IF(ZLON(I).GT.GLON(KK))  N=N+1
            NP(N)=NP(N)+1
  444 CONTINUE

      IF(NP(1).GE.1.AND.NP(2).GE.1.AND.
     &   NP(3).GE.1.AND.NP(4).GE.1) GO TO 999

C-----------------------------------------------------------------------
C NOTE 11 :  IF THE NUMBER OF DATA POINTS IN THE INTERMEDIATE WINDOW
C            IS LARGER THAN  8  AND IN ADDITION THERE IS AT LEAST  1
C            DATA POINT IN EACH QUADRANT THEN THE PROGRAMME IS
C            DIRECTED TO STATEMENT  999  WHICH IS THE FIRST STATEMENT
C            OF THE INTERPOLATION ALGORITHM .
C
C NOTE 12 :  IF BOTH THE ABOVE CONDITIONS ARE NOT SATISFIED ,THEN THE
C            PROGRAMME WILL BE DIRECTED TO STATEMENT  4444 WHICH WILL
C            TEST THE DATA DISTRIBUTION INSIDE THE MAXIMUM WINDOW .
C
C NOTE 13 :  AT THIS STAGE THE ELEMENTS OF THE Y-DESIGNATED ARRAYS ARE
C            TRANSFERRED TO THE Z-DESIGNATED ARRAYS .
C-----------------------------------------------------------------------

 4444 KCOUNT=0
           DO  555  I=1,NO
                    KCOUNT=KCOUNT+1
                ZLAT(KCOUNT)=YLAT(I)
                ZLON(KCOUNT)=YLON(I)
                ZVAL(KCOUNT)=YVAL(I)
```

```
                ZDEV(KCOUNT)=YDEV(I)
  555 CONTINUE
      NOP=KCOUNT
C
         NP(1)=0
         NP(2)=0
         NP(3)=0
         NP(4)=0
      DO  666  I=1,NOP
            N=0
              IF(ZLAT(I).LT.GLAT)  N=2
              IF(ZLON(I).LT.GLON(KK))  N=N+2
              IF(ZLON(I).GT.GLON(KK))  N=N+1
            NP(N)=NP(N)+1
  666 CONTINUE

      IF(NP(1).GE.1.AND.NP(2).GE.1.AND.
     &   NP(3).GE.1.AND.NP(4).GE.1) GO TO 999
C
      GO TO 7777

C-------------------------------------------------------------------------
C NOTE 14 :  IF THE DISTRIBUTION OF DATA POINTS INSIDE THE MAXIMUM
C            WINDOW IS SATISFACTORY (I.E., ONE DATA POINT IN EACH
C            QUADRANT) THEN THE PROGRAMME CARRIES ON WITH THE
C            INTERPOLATION OF THE HEIGHT FUNCTION ON THE GRID NODE .
C
C NOTE 15 :  IF THE DISTRIBUTION OF DATA POINTS IN THE MAXIMUM
C            WINDOW FAILS THE TEST THEN THE PROGRAMME QUITS TRYING
C            AND IS DIRECTED TO STATEMENT  7777  WHICH ASSIGNS THE
C            CHOSEN DEFAULT VALUES FOR HEIGHT AND STANDARD DEVIATION
C            TO THAT GRID POINT .
C-------------------------------------------------------------------------

  999 DO  998  I=1,NOP
       X(I)=69.041*(ZLAT(I)-GLAT)
       YY=69.041*DCOS((GLAT*PI/180.0D0))
       Y(I)=YY*(ZLON(I)-GLON(KK))
       Z(I)=ZVAL(I)
       W(I)=1.0D0/(ZDEV(I)*ZDEV(I))
  998 CONTINUE
      DO    5  I=1,NOP
         FF(I,1)=1.0D0
         FF(I,2)=X(I)
         FF(I,3)=Y(I)
    5    FF(I,4)=Y(I)*X(I)
C   5 CONTINUE
      DO    6  I=1,4
      DO    6  J=1,4
         DD(I,J)=0.0D0
      DO    6  M=1,NOP
    6    DD(I,J)=DD(I,J)+FF(M,I)*FF(M,J)*W(M)
C   6 CONTINUE
      CALL SPIN(DD,4,10,DET,IDEXP)
      QQ=DD(1,1)
      DO    7  I=1,4
         CC(I)=0.0D0
      DO    7  J=1,NOP
    7    CC(I)=CC(I)+FF(J,I)*W(J)*Z(J)
      DO    8  I=1,4
         XX(I)=0.0D0
```

```
      DO     8  J=1,4
    8    XX(I)=XX(I)+DD(I,J)*CC(J)
C
      FVALUE(KK)=XX(1)
C
      PVV=0.0D0
      DO     9  I=1,NOP
         POLYNO=0.0D0
      DO    10  J=1,4
   10    POLYNO=POLYNO+FF(I,J)*XX(J)
    9    PVV=PVV+W(I)*(Z(I)-POLYNO)*(Z(I)-POLYNO)
      SIGMA0=DSQRT(PVV/DFLOAT(NOP-4))
C
      SDEVF(KK)=SIGMA0*DSQRT(QQ)
C
      GO TO 66
C
 7777 FVALUE(KK)=0.0D0
       SDEVF(KK)=50.0D0
   66 CONTINUE
      RETURN
      END

      SUBROUTINE INPUT(GLAT,DMIN,DMAX)
C-----------------------------------------------------------------------
C***********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION XLAT(40000),XLON(40000),XASSH(40000),XSD(40000)
C
      COMMON /STRIP/ XLAT,XLON,XASSH,XSD
      COMMON GLON(600),PHIMIN,PHIMAX,DLAMIN,DLAMAX,HWMAX1,HWMAX2,
     @                HWINT1,HWINT2,HWMIN1,HWMIN2,IPOINT,NOBS,JJ

C-----------------------------------------------------------------------
C NOTE  1 :  IF THE INPUT GRID-LATITUDE (GLAT) IS THE FIRST LATITUDE
C            LINE ALONG WHICH THE INTERPOLATION OF THE ( HEIGHT )
C            FUNCTION H(PHI,LAMDA) WILL BE PERFORMED , THEN THE
C            PROGRAMME IS DIRECTED TO STATEMENT  120  WHICH LOADS THE
C            DATA POINTS (FROM A TAPE FILE) THAT FALL INSIDE THE
C            FIRST FORMED LATITUDE STRIP.
C            THE BOUNDARIES OF THE STRIP ARE INDICATED BY THE VARIABLES:
C            DMIN , DMAX .
C
C NOTE  2 :  THE DATA POINTS LOADED FROM THE TAPE FILE ARE STORED IN
C            THE ARRAYS :
C            XLAT(I) , XLON(I) , XASSH(I) , XSD(I) ;
C            THEY CONTAIN   ****   IPOINT   *****  NUMBER OF ELEMENTS .
C
C NOTE  3 :  WHEN THE PROGRAMME HAS COMPLETED THE PROCESSING OF THE
C            FIRST LATITUDE-GRID-LINE , THE CONTROL OF LOADING DATA
C            WILL BE TRANSFERRED TO STATEMENT  121  .
C            AT THIS STAGE, A NEW LATITUDE STRIP WILL HAVE BEEN FORMED .
C            THE OLD DATA POINTS THAT ARE COMMON IN THE NEW LATITUDE
C            STRIP WILL BE RETAINED , HOWEVER THEY WILL RESUME NEW
C            POSITIONS AS ELEMENTS OF THE X-DESIGNATED ARRAYS .
C-----------------------------------------------------------------------

        IF(GLAT.EQ.PHIMIN) GO TO 120
      GO TO 121
  120 NOBS=0
```

```
      GO TO 122
  121 INUM=0
        DO  123  I=1,IPOINT
          IF(XLAT(I).LT.DMIN)  GO TO 123
              INUM=INUM+1
          XLAT(INUM)=XLAT(I)
          XLON(INUM)=XLON(I)
         XASSH(INUM)=XASSH(I)
          XSD(INUM)=XSD(I)
  123 CONTINUE
      NOBS = INUM

C-----------------------------------------------------------------------
C NOTE  4 :  THE DATA POINTS COMMON TO THE OLD AND NEW LATITUDE STRIPS
C            HAVE NOW BEEN RELOCATED IN THE X-DESIGNATED ARRAYS .
C            THEY CONTAIN NOW   ****   NOBS   ****  NUMBER OF ELEMENTS:
C                           NOBS  <  OR  =  IPOINT(OLD)
C
C NOTE  5 :  THE NEXT STEP IS TO UPDATE THE X-ARRAYS WITH ADDITIONAL
C            DATA POINTS (IF ANY) THAT MAY LIE BETWEEN THE UPPER
C            BOUNDARIES OF THE OLD AND NEW LATITUDE STRIPS.
C            THE TOTAL NUMBER OF OBSERVATIONS WILL THEN BECOME :
C                    IPOINT(NEW)  >  OR  =  NOBS
C-----------------------------------------------------------------------

  122       CALL LODATA(DMIN,DMAX)
       RETURN
      END

      SUBROUTINE LODATA(DMIN,DMAX)
C-----------------------------------------------------------------------
C***********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
               REAL*4 XXLAT,XXLON
            INTEGER*4 REVNO,MJD
          CHARACTER*7 XXSSH
C
      DIMENSION XLAT(40000),XLON(40000),XASSH(40000),XSD(40000)
C
      COMMON /STRIP/ XLAT,XLON,XASSH,XSD
      COMMON GLON(600),PHIMIN,PHIMAX,DLAMIN,DLAMAX,HWMAX1,HWMAX2,
     @                HWINT1,HWINT2,HWMIN1,HWMIN2,IPOINT,NOBS,JJ

C-----------------------------------------------------------------------
C NOTE 1 :  WHENEVER THE AREA FOR WHICH THE REGULAR GRID OF THE HEIGHT
C           FUNCTION WILL BE CONSTRUCTED IS SMALLER THAN THE GENERAL
C           AREA OF DATA COVERAGE, THEN IT IS DESIRABLE TO CONFINE OUR
C           INTEREST TO A SUB-AREA CONTAINING ONLY THE RELEVANT DATA .
C           ONLY THE LONGITUDINAL BOUNDARIES ARE NEEDED TO BE SET ,
C           SINCE THE LATITUDINAL ONES ARE CONTROLLED THROUGH THE
C           VARIABLES  DMIN  AND  DMAX .
C
C    THE OSU SEASAT ADJUSTED ALTIMETRY DATA SET COVERS THE AREA :
C                 35.0 N  =<  LATITUDE   =<   72.1667 N
C                260.0 E  =<  LONGITUDE  =<  350.1667 E
C
C NOTE 2 :  DLAM1 :    REPRESENT THE LONGITUDINAL BOUNDARIES
C           DLAM2 :    OF THE SUB-AREA OF RELEVANT DATA POINTS.
C
C    IF THIS IS THE CASE THEN REMOVE THE COMMENT CARD FROM THE   !!!!!
C    STATEMENTS BELOW AND THE COMMENT CARD FROM THE IF STATEMENT !!!!!
```

```
C
C NOTE 3 :  THE DATA POINTS ARE READ FROM A TAPE FILE .
C           THE DATA ON THE TAPE ARE SORTED IN INCREASING LONGITUDE
C           WITHIN INCREASING LATITUDE .
C
C NOTE 4 :  DATA POINTS WITH ZERO STAND. DEV. FOR HEIGHT ARE REJECTED.
C
C NOTE 5    DATA POINTS WITH ASSH ABSOLUTE VALUE >=70.0M ARE REJECTED.
C
C NOTE 6 :  THE RELEVANT DATA POINTS ARE STORED IN THE ARRAYS :
C           XLAT(I) , XLON(I) , XASSH(I) , XSD(I) .
C-----------------------------------------------------------------------

C     DLAM1 = DLAMIN - HWMAX2
C     DLAM2 = DLAMAX + HWMAX2

          NCOUNT=NOBS
 2221 READ(20,1002,END=2222) XXLAT,XXLON,SD,IT,GEM9,ASSH,SN,XTIDE,XSSH
 1002 FORMAT(2F10.5,F5.2,I10,2F7.2,I10,2F8.2)
C
      IF(XXLAT.LT.DMIN.OR.SD.EQ.0.0E0) GO TO 2221
      IF(DABS(ASSH).GT.70.0D0) GO TO 2221
C
C     IF(XXLON.LT.DLAM1.OR.XXLON.GT.DLAM2) GO TO 2221
C
      IF(XXLAT.GT.DMAX)  GO TO 2222
C
      NCOUNT=NCOUNT+1
         XLAT(NCOUNT)=XXLAT
         XLON(NCOUNT)=XXLON
         XASSH(NCOUNT)=ASSH
         XSD(NCOUNT)=SD
      GO TO 2221
 2222 CONTINUE
      IPOINT=NCOUNT
      RETURN
      END
C*********************************************************************
      SUBROUTINE SPIN(Q,N,MM,DET,IDEXP)
C-----------------------------------------------------------------------
C  FUNCTION  :
C     SUBROUTINE SPIN IS A MATRIX INVERSION ROUTINE FOR SYMMETRIC
C     POSITIVE-DEFINITE MATRICES.  THE MATRIX INVERTED IS THE UPPER
C     N BY N PORTION OF THE MATRIX Q WHICH IS DIMENSIONED MM BY MM
C     IN THE CALLING ROUTINE.
C  AUTHOR    :   R.R.STEEVES
C  HISTORY   :   SEPTEMBER 1979
C-----------------------------------------------------------------------
C  INPUT:
C       Q - THE MATRIX DIMENSIONED MM BY MM WHICH CONTAINS THE
C           MATRIX TO BE INVERTED.
C
C       N - THE DIMENSION OF THE ACTUAL PART( UPPER LEFT CORNER)
C           OF Q WHICH IS TO BE INVERTED. ( N MAY BE EQUAL BUT MUST
C           NOT BE LARGER THAN MM) .
C       MM- DIMENSIONED SIZE OF Q IN THE CALLING ROUTINE.
C
C
C  OUTPUT:
C
C       Q - THE UPPER LEFT N BY N PORTION CONTAINS THE INVERSE OF
```

```
C           THE INPUT UPPER LEFT N BY N PORTION.
C
C       DET - THE NON-EXPONENT PORTION OF THE DETERMINANT OF THE
C             INPUT N BY N (UPPER LEFT PORTION OF Q) MATRIX. SEE
C             IDEXP  BELOW.
C
C       IDEXP - THE EXPONENT (OF 10) PART OF THE DETERMINANT DESCRIBED
C               UNDER DET ABOVE. THUS THE DETERMINANT IS RETURNED IN
C               TWO PARTS CORRESPONDING TO
C                    DETERMINANT = DET * 10 ** IDEXP .
C               THIS IS DONE TO AVOID UNDER OR OVERFLOW IN THE
C               COMPUTATION OF THE DETERMINANT.  TO PRINT THE DETERM-
C               INANT THE USER SHOULD PRINT BOTH NUMBERS AS FOLLOWS;
C               (FOR EXAMPLE)
C                  PRINT 10,DET,IDEXP
C               10 FORMAT(' ','DETERMINANT= ',F7.4,'D',I4)
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      REAL*8 Q(MM,MM),DET,SUM,DSQRT,DABS,RPART,APART
      DET=0.D0
      DO 4 J=1,N
      DO 4 I=1,J
      IF(I.EQ.1) GO TO 2                                                  20
      M=I-1
      SUM=0.0D0
      DO 1 K=1,M
1     SUM=SUM+Q(K,I)*Q(K,J)
      Q(I,J)=Q(I,J)-SUM
2     IF(I.EQ.J) GO TO 3
      Q(I,J)=Q(I,J)/Q(I,I)
      GO TO 4
3     CONTINUE
      DET=DET+DLOG10(Q(I,I))
      Q(I,I)=DSQRT(Q(I,I))
4     CONTINUE
      IDEXP=DET
      RPART=DET-IDEXP
      APART=DABS(RPART)
      IF(APART.LT.1.D-20) DET=1.D0
      IF(APART.LT.1.D-20) GO TO 10
      DET=10**RPART
   10 CONTINUE
      DO 7 J=1,N
      DO 7 I=1,J
      IF(I.LT.J) GO TO 5
      Q(J,J)=1.0D0/Q(J,J)
      GO TO 7
5     SUM=0.0D0
      M=J-1
      DO 6 K=I,M
6     SUM=SUM-Q(I,K)*Q(K,J)
      Q(I,J)=SUM/Q(J,J)
7     CONTINUE
      DO 9 J=1,N
      DO 9 I=1,J
      SUM=0.0D0
      DO 8 K=J,N
8     SUM=SUM+Q(I,K)*Q(J,K)
      Q(I,J)=SUM
      IF(I.EQ.J) GO TO 9
      Q(J,I)=SUM
```

```
9      CONTINUE
       RETURN
       END
//GO.SYSIN      DD  *
  35.0000 260.0000  72.0000 350.0000 10 10
 2.00 2.00 1.00 1.00 0.50 0.50
//GO.FT20F001  DD UNIT=TAPE6250,VOL=SER=P0147,
//             LABEL=(1,NL),DISP=OLD,
//             DCB=(RECFM=FB,LRECL=75,BLKSIZE=32700)
//
```

2.2 Program LEVITUS

```
//LEVITUS JOB NOTIFY=6461
/*SETUP   SLOT=P5271      VOLUME=NC5271          NOWRITE
/*SERVICE NONPRIME
/*JOBPARM S=349,L=999,R=1024,PRINT=ALL
//STEP1   EXEC FORTVCLG,REGION=1024K
//FORT.SYSIN   DD  *
C---------------------------------------------------------------------
C*********************************************************************
C                                                                     *
C  PROGRAM NAME :    LEVITUS                                          *
C  FUNCTION     :  GENERATION OF A PARTICULAR MAP OF DYNAMIC          *
C                  TOPOGRAPHY  OUT  OF  THE  32  AVAILABLE           *
C                  REFERENCE  LEVELS.                                *
C COMPILER      :  VS FORTRAN VERSION 2                              *
C AUTHOR        :  NICK CHRISTOU                                     *
C HISTORY       :  OCTOBER 3, 1985 - VERSION 1.0                     *
C*********************************************************************
C---------------------------------------------------------------------
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C*                                                                    *
C*      SPECIFY   BELOW   WHICH   LEVEL   YOU   WANT                  *
C*                                                                    *
C*        TO  BE  PLOTTED  OUT  OF  THE  32                           *
C*                                                                    *
C*   I.E.:   LEVEL = 26                                               *
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 DTOP
      DIMENSION D(64800)
      DATA LEVEL/26/
C---------------------------------------------------------------------
C   FIND THE LAST AND FIRST RECORD NUMBER OF THE LEVEL SPECIFIED
C   ABOVE IN THE FILE CONTAINING THE DYNAMIC TOPOGRAPHY FOR ALL
C   32 REFERENCE LEVELS.
C---------------------------------------------------------------------
      LL = LEVEL * 64802
       N =(( LEVEL - 1 ) * 64802) + 2
C---------------------------------------------------------------------
C   PUT THE DYNAMIC TOPOGRAPHY 1X1 DEGREE VALUES OF THE
C   SPECIFIED LEVEL IN THE DOUBLE PRECISION ARRAY D(K)
C   READ FROM UNIT 12
C---------------------------------------------------------------------
      DO 10 L=1,LL
         READ(21,1000) DTOP
          IF(L.LE.N)  GO TO 10
            K = L - N
          IF(DTOP.LT.-1000.0) DTOP=-9.9
         D(K)=DBLE(DTOP)
   10 CONTINUE
C---------------------------------------------------------------------
C GENERATE THE GRID COORDINATES AND ASSOCIATE THE ELEMENTS
C  OF THE D(K) ARRAY TO THE CORRESPONDING GRID NODE
C---------------------------------------------------------------------
      DO 30 J=1,180
         DLAT=-89.5D0+DFLOAT((J-1))
           DO 20 I=1,360
              DLON=0.5D0+DFLOAT((I-1))
                M=I+((J-1)*360)
C---------------------------------------------------------------------
C  IF YOU WISH TO OBTAIN THE DYNAMIC TOPOGRAPHY OF A SMALLER
C  AREA REMOVE COMMENT CARDS AND SPECIFY LATITUDE AND
```

```
C  LONGITUDE BOUNDARIES IN THE TWO STATEMENTS BELOW
C-------------------------------------------------------------
C---------
C                IF(DLAT.LT.35.0D0.OR.DLAT.GT.70.0D0) GO TO 20
C                IF(DLON.LT.260.0D0.OR.DLON.GT.350.0D0) GO TO 20
C---------
              WRITE(6,9000) DLAT,DLON,D(M)
   20       CONTINUE
   30 CONTINUE
C-------------------------------------------------------------
 1000 FORMAT(E12.4)
 9000 FORMAT(3F12.4)
      STOP
      END
//GO.FT21F001  DD  UNIT=3480,DSN=LEV32.GRID1X1.DEG,VOL=SER=NC5271,
//             LABEL=(5,SL),DISP=OLD,
//             DCB=(RECFM=FB,LRECL=12,BLKSIZE=32640)
//GO.SYSIN     DD  *
//
```

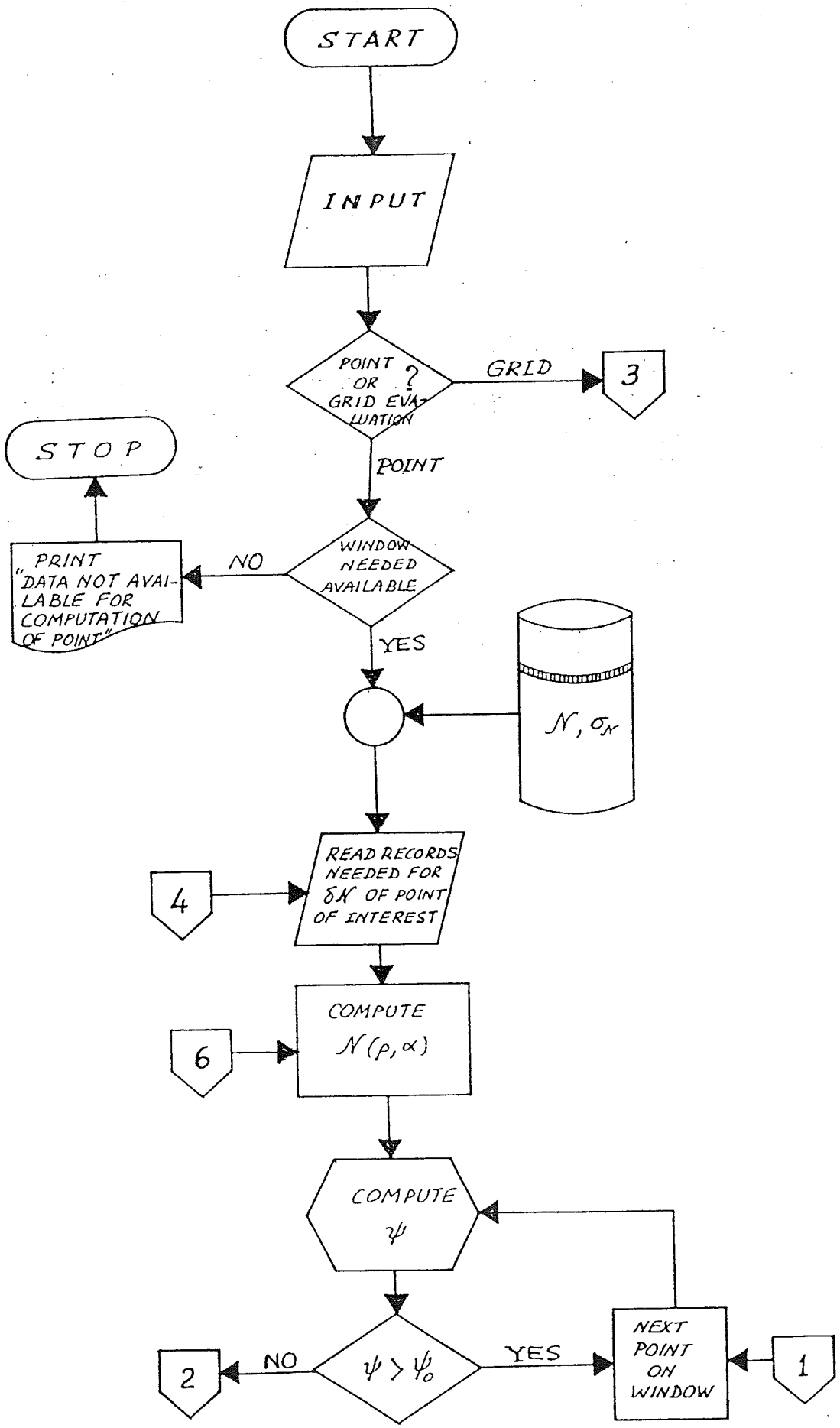
2.3 Program LEVINT

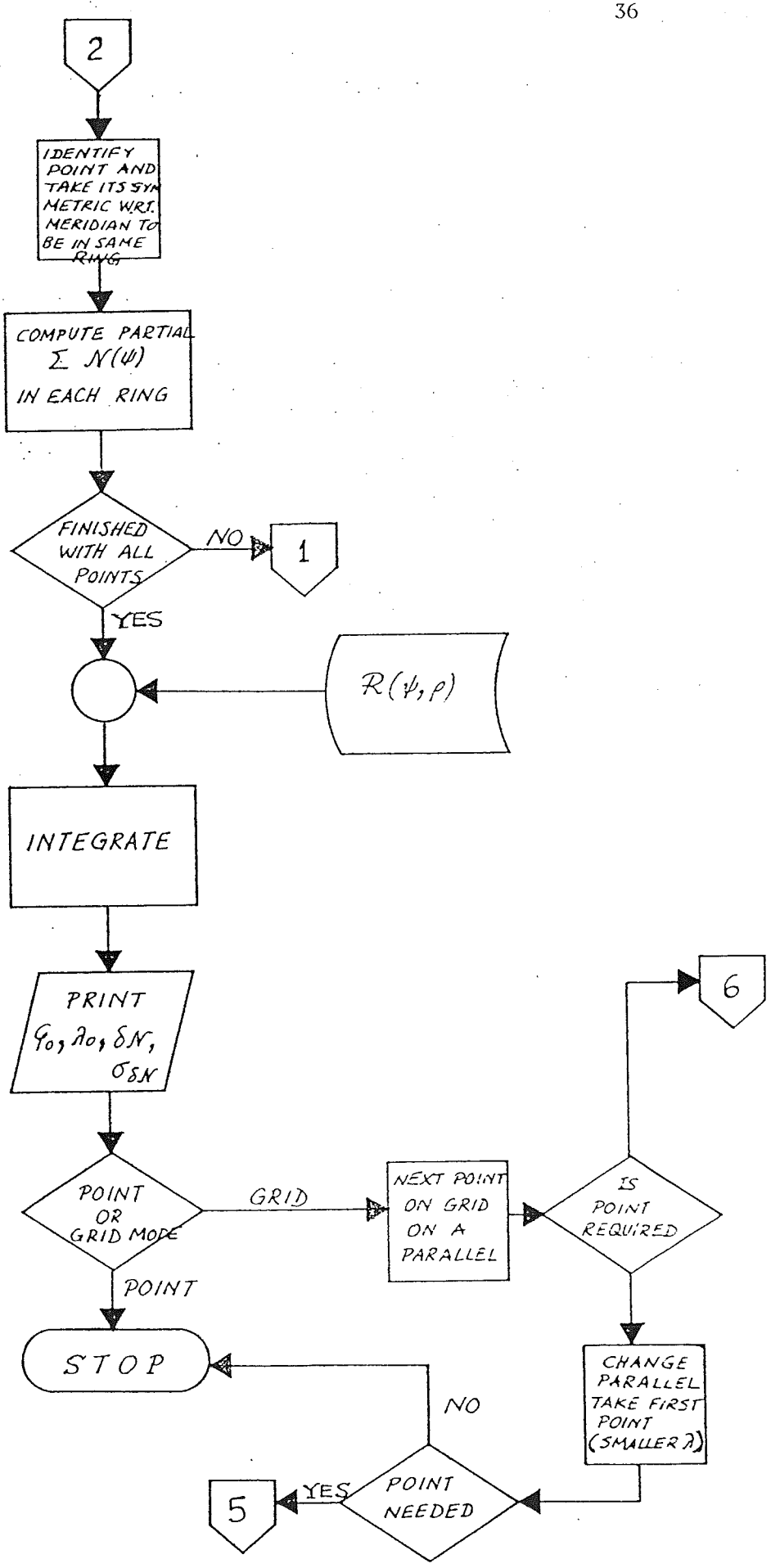
```
//LEVINT  JOB NOTIFY=6461
/*SERVICE NONPRIME
/*JOBPARM S=149,L=199,R=2048,PRINT=ALL
//STEP1   EXEC FORTVCLG,REGION=2048K
//FORT.SYSIN   DD  *
C---------------------------------------------------------------------
C*********************************************************************
C                                                                     *
C  PROGRAM NAME :    LEVINT                                           *
C  FUNCTION     :  INTERPOLATION OF 1X1 DEGREE LEVITUS DYNAMIC       *
C                  TOPOGRAPHY DATA ON 10 X 10 MINUTE GRID USING     *
C                  LINEAR INTERPOLATION IN LATITUDE AND LONGITUDE   *
C                  DIRECTIONS SUCCESSIVELY                          *
C COMPILER       :  VS FORTRAN VERSION 2                            *
C AUTHOR         :  NICK CHRISTOU                                   *
C HISTORY        :  OCTOBER 23, 1985 - VERSION 1.0                  *
C*********************************************************************
C---------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DLAT(92),DLON(92),DTOP(92)
C-------------------------------------------------------------
C  THE SETUP OF THE PROGRAM IS TO PERFORM LINEAR INTERPOLATION
C  IN THE LONGITUDE DIRECTION FIRST.
C  ALL THE STATEMENTS WITH C2-- CARDS ARE RESERVED FOR THE
C  LATITUDE INTERPOLATION (SECOND RUN OF THE PROGRAM).
C  IN THE SECOND RUN THE "LONGITUDE" STATEMENTS SHOULD NOT BE
C  COMMENTED WHILE THE "LATITUDE" ONES SHOULD
C  THE OUTPUT FILE OF THE FIRST RUN  SHOULD BE SORTED IN SUCH
C  A WAY AS TO HAVE ALL THE LATITUDE NODES FOR ONE LONGITUDE
C-------------------------------------------------------------
      PI=3.14159265358979D0
C-------------------------------------------------------------
C  SUPPOSE THERE ARE 37 LATITUDE LINES              (1ST RUN)
C-------------------------------------------------------------
      DO  100  J = 1,37
C2--  DO  100  J = 1,546
C-------------------------------------------------------------
C  SUPPOSE THERE ARE 92 GRID NODES IN LONGITUDE    (1ST RUN)
C  READ FROM UNIT 11 ONE LATITUDE LINE OF THE 1 X 1 DEGREE FILE
C-------------------------------------------------------------
        READ(11,1000) (DLAT(I),DLON(I),DTOP(I),I=1,92)
C2--    READ(11,1000) (DLAT(I),DLON(I),DTOP(I),I=1,37)
C-------------------------------------------------------------
C  FIND THE LONGITUDE DIFFERENCE BETWEEN TWO SUCCESSIVE GRID NODES
C  NODES AND CONVERTED IN A LOCAL CARTESIAN COORDINATE SYSTEM
C-------------------------------------------------------------
        DELLAM = DLON(2) - DLON(1)
            DD = 69.041D0 * DCOS((DLAT(1)*PI/180.0D0))
            DX = DD * DELLAM
C2--    DELPHI = DLAT(2) - DLAT(1)
C2--        DY = 69.041D0 * DELPHI
C-------------------------------------------------------------
C  START DO LOOP FOR LINEAR INTERPOLATION IN THE LONGITUDE
C  DIRECTION , FIND THE HEIGHT DIFFERENCE BETWEEN TWO CONSEQUTIVE
C  GRID NODES , DETERMINE SLOPE , AND PERFORM LINEAR INTERPOLATION
C  THE DEFAULT VALUE OF -9.90 HAS BEEN SET IN THE PROGRAM "LEVITUS"
C-------------------------------------------------------------
        DO  10  K = 1,91
C2--    DO  10  K = 1,36
              DH = DTOP(K+1) - DTOP(K)
        IF(DTOP(K).EQ.-9.90D0.OR.DTOP(K+1).EQ.-9.90D0) DH = 0.0D0
```

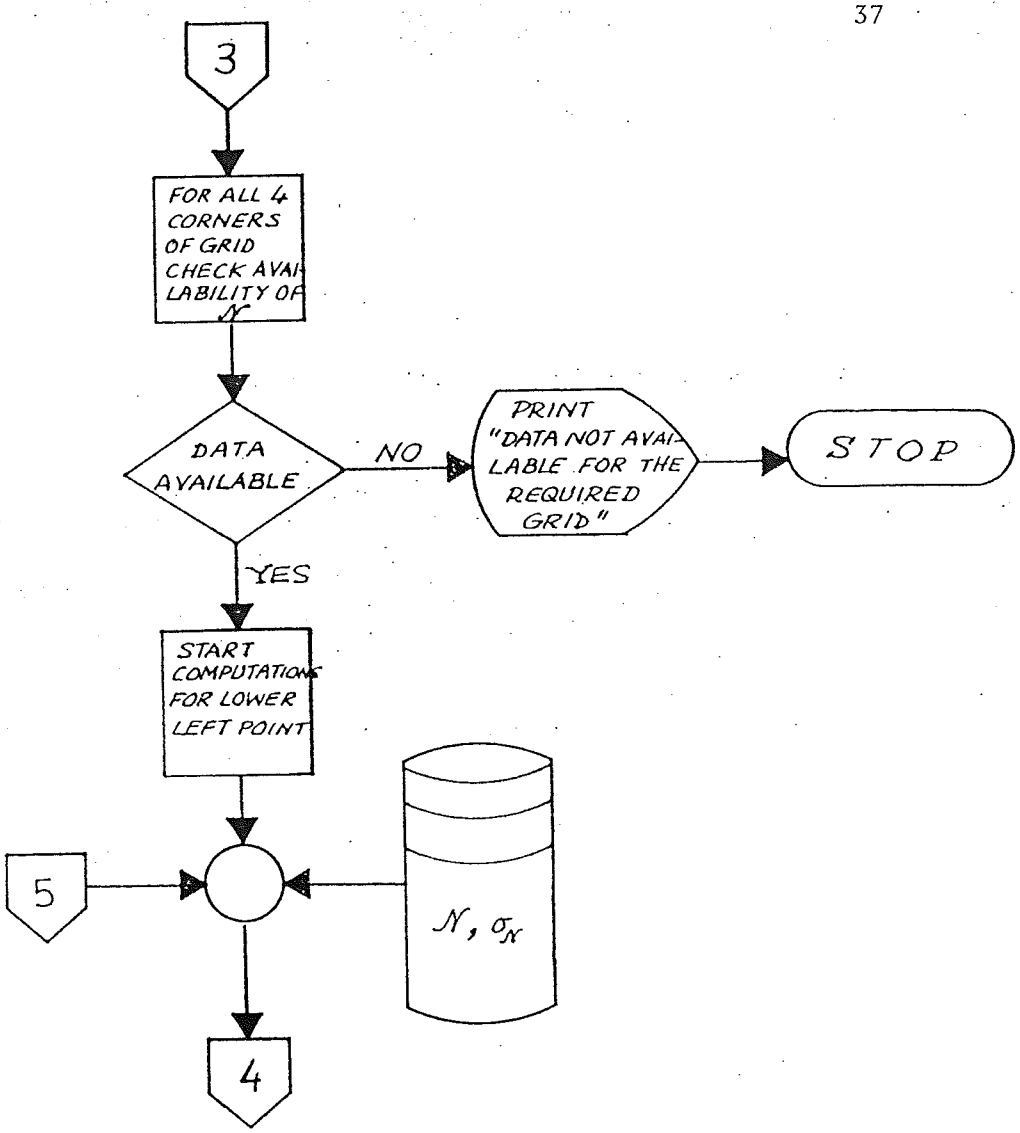
```
            SLOPE = DH / DX
C2--        SLOPE = DH / DY
C-------------------------------------------------------------
C  DO LOOP FOR 10 X 10 MINUTE GRID, I.E. 6 POINTS IN 1 DEGREE
C-------------------------------------------------------------
               DO  40  KK = 1,6
                   DLAM = DLON(K) + ((1.0D0/6.0D0)*(KK-1))
C2--               FLAT = DLAT(K) + ((1.0D0/6.0D0)*(KK-1))
                    TOP = SLOPE * ((DX/6.0D0) * (KK-1))
C2--                TOP = SLOPE * ((DY/6.0D0) * (KK-1))
                   HTOP = DTOP(K) + TOP
                     WRITE(6,1000) DLAT(1),DLAM,HTOP
C2--                 WRITE(6,1000) FLAT,DLON(1),HTOP
   40          CONTINUE
   10    CONTINUE
  100 CONTINUE
C-------------------------------------------------------------
 1000 FORMAT(3F12.4)
      STOP
      END
//GO.FT11F001  DD  DSN=A.M12126.LEV25,DISP=SHR
//GO.SYSIN  DD  *
//
```

2.4 Program ALTINT

PROGRAM " ALTINT"




2
IDENTIFY POINT AND TAKE ITS SYMMETRIC W.R.T. MERIDIAN TO BE IN SAME RING
COMPUTE PARTIAL $\Sigma N(\psi)$ IN EACH RING
FINISHED WITH ALL POINTS
NO
1
YES
$R(\psi, \rho)$
INTEGRATE
PRINT $\varphi_0, \lambda_0, \delta N, \sigma_{\delta N}$
POINT OR GRID MODE
GRID
NEXT POINT ON GRID ON A PARALLEL
IS POINT REQUIRED
6
POINT
STOP
CHANGE PARALLEL TAKE FIRST POINT (SMALLER λ)
NO
POINT NEEDED
YES
5


3
FOR ALL 4 CORNERS OF GRID CHECK AVAILABILITY OF N
DATA AVAILABLE
NO
PRINT "DATA NOT AVAILABLE FOR THE REQUIRED GRID"
STOP
YES
START COMPUTATIONS FOR LOWER LEFT POINT
5
N, σN
4

```
//ALTINT   JOB NOTIFY=6461
/*SERVICE -4
/*JOBPARM T=10,L=99,R=4096,PRINT=ALL
//S1 EXEC FORTVCLG,REGION=4096K,PARM.FORT='OPTIMIZE(3)'
//FORT.SYSIN DD *
C-------------------------------------------------------------------------
C*************************************************************************
C                                                                        *
C  PROGRAM NAME :    ALTINT                                              *
C  FUNCTION     :  TWO-DIMENSIONAL NUMERICAL INTEGRATION OVER            *
C                  RESIDUAL SEA SURFACE HEIGHTS USING DIFFERENT          *
C                  ALTIMETRIC TRUNCATION KERNELS                         *
C COMPILER      :  VS FORTRAN VERSION 2                                  *
C AUTHOR        :  SPIROS PAGIATAKIS                                     *
C HISTORY       :  FEBRUARY 4, 1986 - VERSION 1.0                        *
C REFERENCE     :  U.N.B.  TECHNICAL REPORT  #                           *
C*************************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL *8 KER(100),KERNEL(200)
      CHARACTER *5 MODE
      DIMENSION ALATO(16), ALONO(16), RING(200), RRING(200), X(500),
     *          B(500), C(500), D(500)
      DO = 3.14159265DO / 180.DO
C
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C INPUT: ALATO  = NORTH LATITUDE OF POINT OF INTEREST IN DECIMAL DEGREES
C        ALONO  = EAST LONGITUDE OF POINT OF INTEREST IN DECIMAL DEGREES
C        TRING  = THICKNESS OF RINGS IN DECIMAL MINUTES OF ARC
C        PSIMAX = MAXIMUM PSI FOR INTEGRATION IN DECIMAL DEGREES
C        RHO    = RADIUS OF STOKES' RING IN DECIMAL DEGREES
C        PHIMIN = LATITUDE OF SOUTH-WEST CORNER OF AVAILABLE GRID IN
C                 DECIMAL DEGREES
C        ALAMIN = EAST LONGITUDE OF SOUTH-WEST CORNER OF AVAILABLE GRID
C                 IN DECIMAL DEGREES
C        PHIMAX = LATITUDE OF NORTH-EAST CORNER OF AVAILABLE GRID
C                 IN DECIMAL DEGREES
C        ALAMAX = EAST LONGITUDE OF NORTH-EAST CORNER OF AVAILABLE GRID
C                 IN DECIMAL DEGREES
C        MODE   = POINT OR GRID COMPUTATION (MODE=POINT OR MODE=GRID)
C        W1     = LATITUDE OF SOUTH-WEST CORNER OF GRID OF INTEREST
C        W2     = EAST LONGITUDE OF SOUTH-WEST CORNER OF GRID OF INTEREST
C        W3     = LATITUDE OF NORTH-EAST CORNER OF GRID OF INTEREST
C        W4     = EAST LONGITUDE OF NORTH-EAST CORNER OF GRID OF INTEREST
C        STEP   = GRID STEP IN DECIMAL MINUTES OF ARC
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C READ MODE OF COMPUTATION
C
      READ (5,1000) MODE
C
C READ NUMBER OF POINTS OF INTEREST
C
      READ (5,1001) NPNTS
C
C READ LATITUDE AND LONGITUDE OF POINTS OF INTEREST AND THICKNESS
C OF RINGS
C
      DO 10 I = 1, NPNTS
        READ (5,1002) ALATO(I), ALONO(I)
        IF(ALONO(I) .GT. 180) ALONO(I) = 180.DO - ALONO(I)
```

* THIS PROGRAM ALSO REQUIRES FUNCTION "SEVALD" WHICH IS GIVEN IN ADDENDUM AT BACK OF THIS VOLUME

```
   10 CONTINUE
      READ (5,1003) TRING
C
C READ MAXIMUM PSI FOR INTEGRATION, RHO OF CONSTANT RING
C
      READ (5,1004) PSIMAX, RHO
      WRITE(6,1009) PSIMAX, RHO, TRING
      TRING = TRING / 60.DO
C
C READ NUMBER OF KERNEL VALUES , AND KERNEL
C
      READ (5,1005) NK
      DO 20 I = 1, NK
        READ (5,1006) X(I), KERNEL(I)
   20 CONTINUE
C
C READ AREA BOUNDARIES OF AVAILABLE GRIDDED DATA
C
      READ (5,1007) PHIMIN, ALAMIN, PHIMAX, ALAMAX
C
C READ WINDOW OF GRID REQUESTED AND STEP
C
      READ (5,1008) W1, W2, W3, W4, STEP
      STEP = STEP / 60.DO
C
C CHECK IF DATA ON DISK IS AVAILABLE FOR POINTS OF INTEREST
C
      DO 30 I = 1, NPNTS
        CALL PSIO(I, ALATO, ALONO, ALATO, ALAMIN, PSI)
        IF(PSI .LT. PSIMAX) CALL ERROR (3)
        CALL PSIO(I, ALATO, ALONO, PHIMAX, ALONO, PSI)
        IF(PSI .LT. PSIMAX) CALL ERROR(3)
        CALL PSIO(I, ALATO, ALONO, ALATO, ALAMAX, PSI)
        IF(PSI .LT. PSIMAX) CALL ERROR(3)
        CALL PSIO(I, ALATO, ALONO, PHIMIN, ALONO, PSI)
        IF(PSI .LT. PSIMAX) CALL ERROR(3)
   30 CONTINUE
C
C CHECK IF REQUESTED GRID FALLS WITHIN AVAILABLE DATA GRID
C
      W11 = W1 - PSIMAX - 1.0DO
      W22 = W2 - PSIMAX - 1.0DO
      W33 = W3 + PSIMAX + 1.0DO
      W44 = W4 + PSIMAX + 1.0DO
      IF(W11 .LT. PHIMIN) CALL ERROR(1)
      IF(W22 .LT. ALAMIN) CALL ERROR(1)
      IF(W33 .GT. PHIMAX) CALL ERROR(1)
      IF(W44 .GT. ALAMAX) CALL ERROR(1)
C
C CHECK IF DIMENSIONS OF GRID ARE INTEGER MULTIPLES OF STEP
C
      D1 = DMOD ((W3-W1), STEP)
      D2 = DMOD ((W4-W2), STEP)
C
C COMPUTE OUTER AND MEAN RADIUS OF INNERMOST RING
C
      DR = RHO - TRING / 2.DO
      DM = DMOD(DR,TRING)
      DM = DABS(DM - TRING)
      IF(DM .LT. 0.000001DO) GO TO 40
      IRING = DR / TRING
```

```
      RING(1) = DR - TRING * IRING
C                        WRITE(6,*) 'RING(1)= ',RING(1)
      RRING(1) = RING(1) / 2.DO
      GO TO 50
   40 RING(1) = TRING
      RRING(1) = TRING / 2.DO
C
C COMPUTE OUTER AND MEAN RADII OF INTERMEDIATE RINGS
C IDENTIFY RING THAT IS IDENTICAL WITH STOKES' RING
C
   50 NRINGS = PSIMAX / TRING + 2.DO
C
      DO 60 I = 2, NRINGS
        RING(I) = RING(1) + (I-1) * TRING
        IF(RING(I) .GE. PSIMAX) GO TO 70
        RRING(I) = RING(I) - TRING/2.DO
        DRING = DABS(RRING(I) - RHO)
        IF(DRING .LT. 0.000001D0) KK = I
   60 CONTINUE
C
C COMPUTE RADIUS OF THE OUTERMOST RING
C
   70 RING(I) = PSIMAX
      RRING(I) = RING(I-1) + (PSIMAX - RING(I-1)) / 2.DO
      NRINGS = I
C
C COMPUTE CUBIC SPLINE FOR KERNEL
C
      NI = NK / 3 + 1
C
      DO 80 II = 1,NI
        B(II) = 0.DO
        C(II) = 0.DO
        D(II) = 0.DO
   80 CONTINUE
      CALL SPLIND (NK, X, KERNEL, B, C, D)
C
C COMPUTE KERNEL VALUES FOR EACH OF THE RINGS
C
      U = RRING(1)
      SEV = SEVALD(NK, U, X, KERNEL, B, C, D)
      KER(1) = SEV*DSIN(U*D0)*2.D0*U*D0/(DCOS(U*D0)-DCOS(RHO*D0))
      DO 90 I = 2, NRINGS-1
        IF(I .EQ. KK) GO TO 90
        U = RRING(I)
        SEV = SEVALD(NK, U, X, KERNEL, B, C, D)
        KER(I) = SEV*DSIN(U*D0)*TRING*D0/(DCOS(U*D0)-DCOS(RHO*D0))
   90 CONTINUE
      U = RRING(NRINGS)
      SEV = SEVALD(NK, U, X, KERNEL, B, C, D)
      KER(NRINGS) = SEV*DSIN(U*D0)*(PSIMAX - RING(NRINGS-1))*D0
     *              /(DCOS(U*D0)-DCOS(RHO*D0))
C
C OPEN DIRECT ACCESS FILE ON DISK (THIS IS AN IBM DIRECT ACCESS FILE)
C
      OPEN (1,STATUS='OLD',ACCESS='DIRECT',FORM='UNFORMATTED',
     *      RECL=4088)
C
      IF(MODE .EQ. 'POINT') CALL POINTM(NPNTS,NRINGS,RING,
     * PHIMIN,ALAMIN,PSIMAX,ALATO,ALONO,KER,KK)
C
```

```
      IF(MODE .EQ. 'GRID ') CALL GRIDM (NRINGS,RING,
     * PSIMAX,W1,W2,W3,W4,KER,KK,STEP,PHIMIN,ALAMIN)
C
 1000 FORMAT (5A)
 1001 FORMAT (I4)
 1002 FORMAT (F9.5,1X,F9.5)
 1003 FORMAT (F8.4)
 1004 FORMAT (F9.5,1X,F9.5)
 1005 FORMAT (I4)
 1006 FORMAT (F13.7,F17.10)
 1007 FORMAT (4(F9.5,1X))
 1008 FORMAT (5(F9.5,1X))
 1009 FORMAT (1X,'MAXIMUM PSI    = ',F10.5,/,1X,'RHO            = ',
     *        F10.5,/,1X,'RING THICKNESS = ',F10.5,
     *        ///,2X,'POINT',2X,'LATITUDE',2X,'LONGITUDE',1X,
     *        ' DN(M)',1X,'SIGMA(M)')
      STOP
      END
C*********************************************************************
      SUBROUTINE PSIO (I,LATA, LONA, LAT, LON, PSI)
C---------------------------------------------------------------------
C     SUBROUTINE PSIO DETERMINES THE SPHERICAL ANGLE PSI, DEFINED BY THE
C     THE POINT OF INTEREST AND THE DUMMY POINT.
C
C     INPUT
C     =====
C     LATA  = LATITUDE OF THE POINT OF INTEREST (NORTH OR SOUTH)
C     LONA  = LONGITUDE OF THE POINT OF INTEREST (EAST OR WEST)
C     LAT   = LATITUDE OF THE DUMMY POINT
C     LON   = LONGITUDE OF THE DUMMY POINT
C
C     OUTPUT
C     ======
C     PSI  = GEOCENTRIC SPHERICAL ANGLE IN DEGREES.
C
C     NOTE:
C     =======
C     ALL THE INPUT ANGLES AS WELL AS THE VARIABLES ARE IN DEGREES.
C
C     NORTH LATITUDE AND EAST LONGITUDE ARE CONSIDERED POSITIVE WHILE
C     SOUTH LATITUDE AND WEST LONGITUDE ARE CONSIDERED NEGATIVE.
C---------------------------------------------------------------------
      IMPLICIT REAL *8 (A-H,O-Z)
      REAL *8    LATA(16), LONA(16), LAT, LON
      DO = 3.1415926D0 / 180.D0
      P1 = DSIN(LATA(I)*DO) * DSIN(LAT*DO)
      P2 = DCOS(LATA(I)*DO) * DCOS(LAT*DO) * DCOS((LON-LONA(I))*DO)
      PSI = DARCOS(P1 + P2) / DO
      RETURN
      END
C*********************************************************************
      SUBROUTINE ERROR(IER)
      INTEGER          IER,IPR
      DATA             IPR /6/
C---------------------------------------------------------------------
C FUNCTION:  ERROR DETECTS WHETHER ERROR HAS OCCURED
C
C ARGUMENT:  IER = ERROR INDEX
C---------------------------------------------------------------------
      IF(IER .EQ. 1) GO TO 1
      IF(IER .EQ. 2) GO TO 2
```

```
      IF(IER .EQ. 3) GO TO 3
      IF(IER .EQ. 4) GO TO 4
      IF(IER .EQ. 5) GO TO 5
      IF(IER .EQ. 6) GO TO 6
    1 WRITE (6,1001)
      STOP
    2 WRITE (6,1002)
      STOP
    3 WRITE (6,1003)
      STOP
    4 WRITE (6,1004)
      STOP
    5 WRITE (6,1005)
      STOP
    6 WRITE (6,1006)
      STOP
 1001 FORMAT (' ',///,1X,'*** ERROR ***',//,1X,'REQUESTED GRID IS OUT OF
     * RANGE OF AVAILABLE ALTIMETRY DATA')
 1002 FORMAT (' ',///,1X,'*** ERROR ***',//,1X,'REQUESTED  DIMENSIONS OF
     * GRID ARE NOT INTEGER MULTIPLES OF STEP')
 1003 FORMAT (' ',///,1X,'*** ERROR ***',//,1X,'ALTIMETRY DATA NOT AVAIL
     *ABLE. CHECK POSITION OF POINTS OF INTEREST')
 1004 FORMAT (' ',///,1X,'*** ERROR ***',//,1X,'THERE  ARE NO  POINTS IN
     * ONE OR MORE RINGS')
 1005 FORMAT (' ',///,1X,'*** ERROR ***',//,1X,'FILE NUMBER NOT AVAILABL
     *E ON DISK')
 1006 FORMAT (' ',///,'*** ERROR ***')
      END
C*********************************************************************
      SUBROUTINE POINTM (NPNTS,NRINGS,RING,PHIMIN,ALAMIN,
     *                   PSIMAX,ALATO,ALONO,KER,KK)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL *8 KER(100),NUM(200)
      REAL *4 AN(212,541),SIGMAN(212,541)
      DIMENSION ALATO(16), ALONO(16), RING(200),
     *          NNN(200),VAR(200), SAN(200), STD(200)
      DO = 3.14159265DO / 180.DO
C
C READ RECORDS FROM DISK
C
      DO 130 MM = 1, NPNTS
        ALMIN = 180.DO - ALONO(MM) - PSIMAX - 3.0DO
        ALMAX = 180.DO - ALONO(MM) + PSIMAX + 3.0DO
        N = 6.DO * (ALATO(MM) - PSIMAX - PHIMIN)
        NREC = 12.DO * PSIMAX + 8.DO
        NREC = NREC + N
        DO 10 I = N, NREC
          READ (1,REC=I) (AN(I,J), SIGMAN(I,J), J = 1,541)
          IF(AN(I,J) .GT. 100.0) AN(I,J) = 0.0
   10   CONTINUE
C
C INITIALIZE VARIABLES
C COMPUTE N(RHO,ALFA) AND N(PSI,ALFA)
C
        K = 0
        SUM = 0.DO
        DO 20 I = 1, NRINGS
          SAN(I) = 0.DO
          VAR(I) = 0.DO
          NUM(I) = 0.DO
          NNN(I) = 0
```

```
   20   CONTINUE
C
        DO 90 I = N, NREC
          PHI = PHIMIN + (I - 1) / 6.DO
          DO 80 J = 1, 541
            ALAM = ALAMIN + (J - 1) / 6.DO
            IF(ALAM .LT. ALMIN .OR. ALAM .GT. ALMAX) GO TO 80
            IF(ALAM .GT. 180.DO) ALAM = 180.DO - ALAM
            P1 = DSIN(ALATO(MM)*DO) * DSIN(PHI*DO)
            P2 = DCOS(ALATO(MM)*DO) * DCOS(PHI*DO)
     *           * DCOS((ALAM-ALONO(MM))*DO)
            PSI = DARCOS(P1+P2) / DO
            IF(PSI .GT. PSIMAX) GO TO 80
C
   30       DO 40 K = 2, NRINGS
              K1 = NRINGS - K + 1
              K2 = K1 + 1
              IF(PSI .GT. RING(K1) .AND. PSI .LE. RING(K2)) GO TO 50
   40       CONTINUE
C
            GO TO 60
   50       NUM(K2) = NUM(K2) + DCOS(PHI*DO)
            NNN(K2) = NNN(K2) + 1
            SAN(K2) = SAN(K2) + DBLE(AN(I,J))*DCOS(PHI*DO)
            VAR(K2) = VAR(K2) + DBLE(SIGMAN(I,J) * SIGMAN(I,J))
            GO TO 80
   60       IF(PSI .LE. RING(1)) GO TO 70
            GO TO 80
   70       NUM(1) = NUM(1) + DCOS(PHI*DO)
            NNN(1) = NNN(1) + 1
            SAN(1) = SAN(1) + DBLE(AN(I,J))*DCOS(PHI*DO)
            VAR(1) = VAR(1) + DBLE(SIGMAN(I,J) * SIGMAN(I,J))
   80     CONTINUE
   90   CONTINUE
C
        DO 100 IJ = 1 ,NRINGS
          IF(NUM(IJ) .LE. 0.000001DO) CALL ERROR(4)
          WRITE(6,*) 'RING',IJ,'# POINTS', NNN(IJ),'SUM N',SAN(IJ)
  100   CONTINUE
C
        SUM = SAN(KK) / NUM(KK)
        STDSUM = DSQRT(VAR(KK)) / NUM(KK)
        SAN(1) = SAN(1) / NUM(1)
        STD(1) = DSQRT(VAR(1)) / NUM(1)
        SAN(1) = (SAN(1)-SUM) * KER(1)
        STD(1) = KER(1) * DSQRT(STD(1)*STD(1) + STDSUM*STDSUM)
C
        DO 110 L = 2, NRINGS
          IF(L .EQ. KK) GO TO 110
          SAN(L) = SAN(L) / NUM(L)
C         WRITE(6,*) 'RING',L,'AVERAGE N', SAN(L)
          SAN(L) = (SAN(L)-SUM) * KER(L)
          STD(L) = KER(L) * DSQRT(STD(L)*STD(L) + STDSUM*STDSUM)
  110   CONTINUE
C
        DN = 0.DO
        VARDN = 0.DO
        DO 120 M = 1, NRINGS
          IF(M .EQ. KK) GO TO 120
          DN = DN + SAN(M)
C         VARDN = VARDN + (SEV*SEV) * (STD(M)*STD(M))
```

```
  120   CONTINUE
C
        DN = 0.5D0 * DN
        DNT=SAN(1)-DN
        STDDN = 0.5D0 * DSQRT(VARDN)
        IF(ALONO(MM) .LT. 0) ALONO(MM) = 180.D0 - ALONO(MM)
C
C PRINT RESULTS
C
      WRITE(6,1001)  MM, ALATO(MM), ALONO(MM), DNT, STDDN
  130 CONTINUE
 1001 FORMAT (' ',2X,I3,1X,F10.5,1X,F10.5,1X,F6.3,1X,F6.3)
      STOP
      END
C*********************************************************************
      SUBROUTINE GRIDM(NRINGS,RING,PSIMAX,W1,W2,W3,W4,
     *                 KER,KK,STEP,PHIMIN,ALAMIN)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL *8 KER(100),NUM(200)
      REAL *4 AN(211,511),SIGMAN(211,511)
      DIMENSION RING(200),
     *          NNN(200),VAR(200), SAN(200), STD(200)
      DO = 3.14159265358979D0 / 180.DO
C
C READ RECORDS FROM DISK
C
        N = 6.DO * (W1 - PSIMAX - PHIMIN)
        NREC = 6.DO * (2.DO * PSIMAX + W3 - W1) + 8.DO
        NREC = NREC + N
C
        DO 10 I = N, NREC
          READ (1,REC=I) (AN(I,J), SIGMAN(I,J), J = 1,511)
         IF(ABS(AN(I,J)).GT.100.0) AN(I,J)= 1.D12
   10   CONTINUE
C
C INITIALIZE VARIABLES
C COMPUTE N(RHO,ALFA) AND N(PSI,ALFA)
C
        GLAT = W1
        GLON = W2
   15   ALMIN = GLON -(PSIMAX/DCOS(GLAT*DO)+.2D-3)
        ALMAX = GLON +(PSIMAX/DCOS(GLAT*DO)+.2D-3)
        CLM1 = GLAT - PSIMAX
        CLM2 = GLAT + PSIMAX
        IF(GLON .GE. 180.DO) GLON = 180.DO - GLON
C
        DO 20 I = 1, NRINGS
          SAN(I) = 0.DO
          VAR(I) = 0.DO
          NUM(I) = 0.DO
          NNN(I) = 0
   20   CONTINUE
C
        DO 90 I = N, NREC
          PHI = PHIMIN + (I - 1) / 6.DO
          IF(PHI .LT. CLM1 .OR. PHI .GT. CLM2) GO TO 90
          DO 80 J = 1, 511
            ALAM = ALAMIN + (J - 1) / 6.DO
            IF(ALAM .LT. ALMIN .OR. ALAM .GT. ALMAX) GO TO 80
            IF(ALAM .GT. 180.DO) ALAM = 180.DO - ALAM
            P1 = DSIN(GLAT*DO) * DSIN(PHI*DO)
```

```
            P2 = DCOS(GLAT*DO) * DCOS(PHI*DO)
     *           * DCOS((ALAM-GLON)*DO)
            PSI = DARCOS(P1+P2) / DO
            IF(PSI .GT. PSIMAX)  GO TO 80
            DO 40 K = 2, NRINGS
              K1 = NRINGS - K + 1
              K2 = K1 + 1
              IF(PSI .GT. RING(K1) .AND. PSI .LT. RING(K2)) GO TO 50
   40       CONTINUE
            GO TO 60
   50       NUM(K2) = NUM(K2) + DCOS(PHI*DO)
            NNN(K2) = NNN(K2) + 1
            SAN(K2) = SAN(K2) + DBLE(AN(I,J))*DCOS(PHI*DO)
            VAR(K2) = VAR(K2) + DBLE(SIGMAN(I,J) * SIGMAN(I,J))
            GO TO 80
   60       IF(PSI .LE. RING(1)) GO TO 70
            GO TO 80
   70       NUM(1) = NUM(1) + DCOS(PHI*DO)
            NNN(1) = NNN(1) + 1
            SAN(1) = SAN(1) + DBLE(AN(I,J))*DCOS(PHI*DO)
            VAR(1) = VAR(1) + DBLE(SIGMAN(I,J) * SIGMAN(I,J))
   80     CONTINUE
   90   CONTINUE
C
       DO 100 IJ = 1 ,NRINGS
         IF(NUM(IJ) .LE. 0.000001D0) CALL ERROR(4)
  100  CONTINUE
C
       SUM = SAN(KK) / NUM(KK)
       STDSUM = DSQRT(VAR(KK)) / NUM(KK)
       SAN(1) = SAN(1) / NUM(1)
C
       STD(1) = DSQRT(VAR(1)) / NUM(1)
       SSS = SAN(1)
       SAN(1) = (SAN(1)-SUM) * KER(1)
       STD(1) = KER(1) * DSQRT(STD(1)*STD(1) + STDSUM*STDSUM)
C
       DO 110 L = 2, NRINGS
         IF(L .EQ. KK) GO TO 110
         SAN(L) = SAN(L) / NUM(L)
         STD(L) = DSQRT(VAR(L)) / NUM(L)
         SAN(L) = (SAN(L)-SUM) * KER(L)
         STD(L) = KER(L) * DSQRT(STD(L)*STD(L) + STDSUM*STDSUM)
  110  CONTINUE
C
       DN = 0.D0
       VARDN = 0.D0
       DO 120 M = 1, NRINGS
         IF(M .EQ. KK) GO TO 120
         DN = DN + SAN(M)
         VARDN = VARDN +  (STD(M)*STD(M))
  120  CONTINUE
C
       DN = 0.5D0 * DN
       DNT=SSS-DN
       STDDN = 0.5D0 * DSQRT(VARDN)
       IF(GLON .LT. 0) GLON = 180.D0 - GLON
C
C PRINT RESULTS
C
     WRITE(6,1001)  GLAT, GLON, DNT, STDDN
```

```
      GLON = GLON + STEP
      IF(GLON .GT. W4) GO TO 130
      GO TO 15
  130 GLAT = GLAT + STEP
      IF(GLAT .GT. W3) GO TO 140
      GLON = W2
      GO TO 15
 1001 FORMAT (' ',6X,F10.5,1X,F10.5,1X,F10.3,1X,F6.3)
  140 STOP
      END
C*********************************************************************
      SUBROUTINE SPLIND(N,X,Y,B,C,D)
      INTEGER N
      REAL*8 X(N),Y(N),B(N),C(N),D(N)
      INTEGER NM1,IB,I
      REAL*8 T
      NM1=N-1
      IF(N.LT.2) RETURN
      IF (N.LT.3) GO TO 50
      D(1)=X(2)-X(1)
      C(2)=(Y(2)-Y(1))/D(1)
      DO 10 I=2,NM1
        D(I)=X(I+1)-X(I)
        B(I)=2.*(D(I-1)+D(I))
        C(I+1)=(Y(I+1)-Y(I))/D(I)
        C(I)=C(I+1)-C(I)
   10 CONTINUE
      B(1)=-D(1)
      B(N)=-D(N-1)
      C(1)=0.
      C(N)=0.
      IF (N.EQ.3) GO TO 15
      C(1)=C(3)/(X(4)-X(2))-C(2)/(X(3)-X(1))
      C(N)=C(N-1)/(X(N)-X(N-2))-C(N-2)/(X(N-1)-X(N-3))
      C(1)=C(1)*D(1)**2/(X(4)-X(1))
      C(N)=-C(N)*D(N-1)**2/(X(N)-X(N-3))
   15 DO 20 I=2,N
        T=D(I-1)/B(I-1)
        B(I)=B(I)-T*D(I-1)
        C(I)=C(I)-T*C(I-1)
   20 CONTINUE
      C(N)=C(N)/B(N)
      DO 30 IB=1,NM1
        I=N-IB
        C(I)=(C(I)-D(I)*C(I+1))/B(I)
   30 CONTINUE
      B(N)=(Y(N)-Y(NM1))/D(NM1)+D(NM1)*(C(NM1)+2.*C(N))
      DO 40 I=1,NM1
        B(I)=(Y(I+1)-Y(I))/D(I)-D(I)*(C(I+1)+2.*C(I))
        D(I)=(C(I+1)-C(I))/D(I)
        C(I)=3.*C(I)
   40 CONTINUE
      C(N)=3.*C(N)
      D(N)=D(N-1)
      RETURN
   50 I=1
      B(I)=(Y(2)-Y(1))/(X(2)-X(1))
      C(1)=0.
      D(1)=0.
      B(2)=B(1)
      C(2)=0.
```

```
      D(2)=0.
      RETURN
      END
//GO.FT01F001 DD DSN=A.M66774.SW35265.NE70350.REDUCED.DAFILE,
//            DISP=SHR,LABEL=(,,,IN)
//GO.SYSIN DD *
GRID
0004
050.00000 330.00000
050.50000 330.00000
051.00000 330.00000
051.00000 331.00000
 10.0000
002.98497 000.50000
0035
     .0500000        -.6352274155
     .1500000        -.6097598817
     .2500000        -.5937992761
     .3500000        -.6137125445
     .4500000        -.6524700664
     .5500000        -.6641680044
     .6500000        -.6191783843
     .7500000        -.5284651491
     .8500000        -.4251898481
     .9500000        -.3313138102
    1.0500000        -.2463315798
    1.1500000        -.1616584493
    1.2500000        -.0757554642
    1.3500000         .0067780342
    1.4500000         .0840416263
    1.5500000         .1587937783
    1.6500000         .2329758779
    1.7500000         .3047420898
    1.8500000         .3716928187
    1.9500000         .4341187881
    2.0500000         .4934516996
    2.1500000         .5492422862
    2.2500000         .5995406127
    2.3500000         .6435180810
    2.4500000         .6819575638
    2.5500000         .7152043921
    2.6500000         .7421347971
    2.7500000         .7616109284
    2.8500000         .7738343657
    2.9500000         .7795058968
    3.0500000         .7783813695
    3.1500000         .7695661770
    3.2500000         .7528953127
    3.3500000         .7291113778
    3.4500000         .6986867250
035.00000 265.00000 070.00000 350.00000
053.00000 300.00000 054.00000 301.00000 010.00000
//
```

INTERPOLATION

SPLINS
SPLIND
SEVALS
SEVALD

Purpose: To evaluate the cubic spline function

$$F(X)=Y(I)+B(I)(X-X(I))+C(I)(X-X(I))^2+D(I)(X-X(I))^3$$

for X(I) .LE. X(I+1) for the range of all the data points, to calculate the best (smoothest) fitting cubic polynomial through the data.

How to Use: Refer to the table to determine which subroutine to use.

To evaluate	Use the calling sequence	With variables typed
REAL*4 FUNCTION	CALL SPLINS(N,X,Y,B,C,D) S=SEVALS(N,U,X,Y,B,C,D)	REAL X,Y,B,C,D,U,S INTEGER N
REAL*8 FUNCTION	CALL SPLIND(N,X,Y,B,C,D) S=SEVALD(N,U,X,Y,B,C,D)	REAL*8 X,Y,B,C,D, U,S,SEVALD INTEGER N

where:

N = the number of data points (N.GE.2)
X = the array of abscissas of the data points in strictly increasing order
Y = the array of the ordinates of the data points
B,C,D = arrays of the coefficients of the spline polynomials for each interval. B(I),C(I),D(I) for I=1,2,...,N
U = the abscissa point at which the cubic spline is to be evaluated
S = the value of the spline at point U

In addition to the call statements, the calling program must have at least the following statements:

- appropriate type statements
- initialization of N and values for X and Y

Algorithm: Given a number of data points, a cubic polynomial is calculated for each of the intervals insuring that they are the cubics which will fit the equation of the entire set of data the best. A value is returned calculated using this function at the requested point.

Accuracy: The end point conditions are chosen such that the natural cubic spline, the smoothest or best curve which interpolates the data is used. The more data points supplied, assuming small intervals occur, causes the cubic spline to converge to the true function fitting the data exactly.

Availability: FORTRAN H - UNB1.FORTRAN

WATFIV - UNB1.WATFIV.UNBLIB

Core Requirements:

Routine / Version	SPLINS	SPLIND	SEVALS	SEVALD
FORTRAN H	1790	1818	640	656
WATFIV	2960	2992	920	928

References:

Forsythe, Malcolm, and Moler, Computer Methods for Mathematical Computations, Prentice-Hall, New Jersey, 1977
Subroutines- SPLINE and SEVAL p. 77
Coded by - D. Goguen

2.5 Program GPOT

```
//GPOT     JOB  NOTIFY=6461
/*SERVICE -4
/*JOBPARM L=19,S=29,R=2048,PRINT=ALL
//STEP1    EXEC FORTVCLG,REGION=2048K
//FORT.SYSIN DD *
C-----------------------------------------------------------------------
C    ******************    PROGRAM GPOT    ************************
C      TEST PROGRAM USING TSCHERNING'S HARMONIC MODEL SUBROUTINE
C      PROGRAM TO COMPUTE GEOID UNDULATIONS , DEFLECTIONS OF THE
C      VERTICAL AND GRAVITY DISTURBANCES FROM POTENTIAL COEFFICIENTS
C***********************************************************************
C
C    REFERENCE :  MANUSCRIPTA GEODAETICA VOL. 8 (1983) 249-272
C
C***********************************************************************
C
C                        GRID  GEOID  COMPUTATIONS
C
C      INPUT: LAT1,LAT2 - MAXIMUM AND MINIMUM LATITUDE OF THE AREA
C                         OF INTEREST (IN INTEGER DEGREES)
C             LON1,LON2 - MAXIMUM AND MINIMUM LONGITUDE OF THE AREA
C                         OF INTEREST (IN INTEGER DEGREES)
C                   INT - GRID INTERVAL (IN INTEGER MINUTES OF ARC)
C                  NMAX - DEGREE AND ORDER OF SPHERICAL HARMONIC
C                         EXPANSION DESIRED
C
C     OUTPUT:        UND - GEOIDAL HEIGHT
C                 XI, ETA - N-S AND E-W DEFLECTIONS OF THE VERTICAL
C                      DG - GRAVITY DISTURBANCE
C
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8(A-H,O-Z)
C-----------------------------------------------------------------------
C    INPUT LOGICAL UNIT FOR EXTERNAL INPUT IS :           5
C    INPUT LOGICAL UNIT FOR POTENTIAL COEFFICIENTS IS :  12
C-----------------------------------------------------------------------
      ICR = 5
C-----------------------------------------------------------------------
C    READ THE MAXIMUM AND MINIMUM LATITUDE AND LONGITUDE
C    OF THE AREA OF GEOID COMPUTATION IN INTEGER DEGREES
C    READ THE GRID INTERVAL IN INTEGER MINUTES OF ARC
C-----------------------------------------------------------------------
      READ(ICR,100) LAT1,LAT2,LON1,LON2,INT
  100 FORMAT(5I4)
C
      LATR =((LAT1-LAT2)*(60/INT))+1
      LONR =((LON1-LON2)*(60/INT))+1
      WRITE(6,100) LAT1,LAT2,LON1,LON2,LATR,LONR
      DINCR=INT/60.0D0
C
      LATST = 1
      DO 20 LAT=LATST,LATR
           PHI = DFLOAT(LAT2)+((LAT-1)*DINCR)
C
      DO 20 LON=1,LONR
          DLAM = DFLOAT(LON2)+((LON-1)*DINCR)
C
      CALL POT1(PHI,DLAM,0.D0,UN1,XI1,ETA1,DG1)
C-----------------------------------------------------------------------
C  IF YOU WISH YOU CAN CONVERT THE GRAVITY DISTURBANCE INTO
```

```
C  GRAVITY ANOMALY USING THE FOLLOWING FORMULA:
C              DG1=DG1-0.3086D0*UN1
C-----------------------------------------------------------------------
      WRITE (6,111) PHI,DLAM,UN1,DG1
  111 FORMAT(4F10.4)
C
   20 CONTINUE
      STOP
      END
C*********************************************************************
      BLOCK DATA
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL INIT
      LOGICAL FIRST
      INTEGER OLDORD
      REAL*4 C,CO
      COMMON/GPOTCM/OLDT,OLDR,IZ,FIRST,OLDORD,I1,I2,I3,I4,
     +              I5,I6,I7,I8,I9,NMAXSV
      COMMON/POT1CM/SU(1810),DJN(20),GM,FLAT,AE,OMEGA,INIT,IORDER,NMAX,
     *              NEGN
      COMMON/PIDTR/PI,DTR
      COMMON/TRANCM/TOL,MAXIT
      COMMON/CM/C20IN,G1(3),G2(3,3),CM3,CM2,CM1,CO,C(32760)
      DATA IZ/0/
      DATA FIRST/.FALSE./
      DATA OLDT/0.D0/,OLDR/0.D0/
      DATA OLDORD,I1,I2,I3,I4,I5,I6,I7,I8,I9/10*0/
      DATA NMAXSV/0/
      DATA DJN/20*0.D0/
      DATA INIT/.TRUE./
      DATA IORDER/1/
      DATA NMAX/180/
      DATA PI/3.14159265358979D0/
      DATA DTR/0.1745329251994330D-1/
      DATA TOL/1.D-14/,MAXIT/10/
      END
C*********************************************************************
      SUBROUTINE POT1(PHI,DLON,HT,UN,XI,ETA,DIST)
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 C,CO
      LOGICAL INIT
      DIMENSION P(6)
      COMMON/PIDTR/PI,DTR
      COMMON/POT1CM/SU(1810),DJN(20),GM,FLAT,AE,OMEGA,INIT,IORDER,NMAX,
     &       NEGN
      COMMON/CM/C20IN,G1(3),G2(3,3),CM3,CM2,CM1,CO,C(32760)
C*****************************************************************
C                                                                *
C  INPUT                                                         *
C                                                                *
C  PHI  LATITUDE (GEODETIC) IN DEGREES                           *
C  DLON  LONGITUDE (WEST) IN DEGREES                             *
C  HT HEIGHT IN METERS                                           *
C                                                                *
C  OUTPUT                                                        *
C                                                                *
C  UN  HEIGHT ANOMALY IN METERS                                  *
C  XI  N-S DEFLECTION IN SEC OF ARC                              *
C  ETA  E-W DEFLECTION IN SECONDS OF ARC (WEST POSITIVE)         *
C  DIST  GRAVITY DISTURBANCE IN MGALS                            *
```

```
C                                                                      *
C**********************************************************************
      IF(.NOT.INIT) GO TO 500
      INIT=.FALSE.
      NEGN=-NMAX
C
C   SET CONSTANTS (GRS80)
C
      DJN(2)=0.0010826300
      OMEGA=7.292115D-5
      AE=6378137.D0
      GM=3.986005D+14
C
C  OBTAIN FLATTENING OF REFERENCE ELLIPSOID TO BE USED LATER WHEN
C  TRANSFORMING FROM GEODETIC LAT,LON,HT TO ECG X,Y,Z
C
      CALL REFVAL(AE,GM,DJN(2),OMEGA,FLATI)
      FLAT=1.D0/FLATI
C
C       GENERATE NORMAL POTENTIAL VALUES
C
      CAPESQ=AE**2*(2.D0-FLAT)*FLAT
      ESQ=CAPESQ/AE**2
C
C   COMPUTE NORMAL EVEN ZONALS  FROM  4 TO 20
C
      DO 200 N2=4,20,2
      N=N2/2
      DJN(N2)=(-1.D0)**(N+1)*3.D0/(N2+1.D0)*ESQ**N/(N2+3.D0)*
     &        (1.D0-(1.D0-5.D0*DJN(2)/ESQ)*N)
  200 CONTINUE
      CM3=GM
      CM2=AE
      CM1=0.D0
      C0=0.E0
      DO 10 I=1,32760
      C(I)=0.E0
   10 CONTINUE
      CALL LOADCS(NMAX)
      C0=0.D0
C
C     COMPUTE DELC20  IN DOUBLE PRECISION BEFORE STORING IN SINGLE
C     PRECISION RATHER THAN STORING IN SINGLE PRECISION AND THEN
C     DIFFERENCING
C
      DELC20=C20IN+DJN(2)/DSQRT(5.D0)
C
      CALL STORC(0,0,0.D0,0.D0)
      CALL STORC(1,0,0.D0,0.D0)
      CALL STORC(1,1,0.D0,0.D0)
      CALL STORC(2,0,DELC20,0.D0)
C
C     NOW SUBTRACT OFF GRS80 EVEN ZONAL HARMONICS ( J2  ALREADY DONE )
C     TO GET THE ANOMALOUS POTENTIAL FOR HEIGHT ANOMALY AND DERIVATIVE
C     COMPUTATIONS
C
      DO  400 N=4,20,2
C         NORMALIZE ZONALS OF NORMAL POTENTIAL BEFORE SUBTRACTING
      DNJ=DJN(N)/DSQRT(N+N+1.D0)
      CALL MODC(N,0,DNJ,0.D0)
  400 CONTINUE
```

```
C
      CALL SETCM(NMAX)
C
C CONVERT FROM GEODETIC TO EARTH-CENTERED-FIXED X,Y,Z COORDINATES
C
  500 CALL TRANF(1,PHI,DLON,HT,X,Y,Z,AE,FLAT)
C
C     SET UP  INPUT ARRAY TO  GPOTDR
C
      P(1)=DSQRT(X**2+Y**2)
      P(2)=DSQRT(X**2+Y**2+Z**2)
      P(3)=Z/P(2)
      P(4)=P(1)/P(2)
      P(5)=Y/P(1)
      P(6)=X/P(1)
C
      TP=GPOTDR(P,NEGN,IORDER,SU)
C
      CALL NORMAL(GM,DJN,AE,OMEGA,X,Y,Z,UP,GAMMA,GRAD)
C         THE HEIGHT ANOMALY IN METRES IS
C
      UN=TP/GAMMA
C
C         DEFLECTIONS OF THE VERTICAL IN SEC OF ARC
C N-S
      XI =-G1(1)*206264.8D0 / GAMMA
C E-W
      ETA =-G1(2)*206264.8D0 / GAMMA
C         GRAVITY DISTURBANCE IN MILLIGALS
      DIST =-G1(3)*1.D+05
      RETURN
      END
C*********************************************************************
      FUNCTION GPOTDR(PO,NMAX,ORDER,SU)
C
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C  GI REG.NO. 80039  AUTHOR -C.C.TSCHERNING, NOV 1980  IN ALGOL
C                           -C.C.GOAD, MAR 1981 TRANSLATED TO FORTRAN
C
C REFERENCES:
C (1) TSCHERNING, C.C.:ON THE CHAIN-RULE METHOD FOR COMPUTING
C     POTENTIAL DERIVATIVES. MANUSCRIPTA GEODAETICA, VOL.1,
C     PP. 125-141, 1976
C
C (2) GERSTL,M.:VERGLEICH VON ALGORITMEN ZUR SUMMATION VON
C     KUGELFLAECHENFUNKTIONEN. VEROEFFENTL. DER BAYER. KOMM.
C     F.D. INT. ERDMESSUNG DER BAYER. AKADEMIE DER WISSEN.,
C     HEFT NR. 38, PP. 81-88, 1978.
C  THE PROCEDURE COMPUTES THE VALUE AND UP TO THE SECOND ORDER
C DERIVATIVES OF THE POTENTIAL OF THE EARTH (W) OR OF ITS
C CORRESPONDING AMONALOUS POTENTIAL(T). (THE COMPUTATION OF
C THE SECOND ORDER DERIVATIVES HAS NOT YET BEEN IMPLEMENTED).
C
C  THE POTENTIAL IS REPRESENTED BY A SERIES IN SOLID SPHERICAL
C HARMONICS, WITH UN-NORMALIZED OR QUASI-NORMALIZED COEFFICIENTS.
C THE CHAIN-RULE IS USED COMBINED WITH THE CLENSHAW ALGORITHM.
C THE ARRAY C MUST HOLD THE COEFFICIENTS, C(0,0)=1.DO FOR W AND
C 0.0 FOR T, C(1)=C(1,0),C(2)=C(1,1),C(3)=S(1,1) ETC. UP TO
C C((N+1)**2-1) = S(N,N).
C
C
```

```
C PARAMETERS:
C
C (A) INPUT VALUES:
C
C NMAX
C     THE ABSOLUTE VALUES OF NMAX  IS EQUAL TO THE MAXIMAL DEGREE AND
C     ORDER OF THE SERIES. NEGATIVE NMAX INDICATES THAT THE COEFFICIENTS
C     ARE QUASI-NORMALIZED.
C
C IORDER
C     THE MAXIMAL ORDER OF THE DERIVATIVES (< 2 P.T.).
C
C PO
C     ARRAY HOLDING POSITION INFORMATION. PO(6)
C     PO(1)=P, THE DISTANCE FROM THE Z (ROTATION) AXIS,
C     PO(2)=R, THE DISTANCE FROM THE ORIGIN,
C     PO(3),PO(4) COS AND SIN OF THE GEOCENTRIC POLAR ANGLE(COLATITUDE),
C     PO(5),PO(6) SIN AND COS OF THE LONGITUDE.
C
C C
C     C MUST BE DECLARED WITH BOUNDS (-3: (N+1)**2-1) WHEN THE
C     COEFFICIENTS ARE UN-NORMALIZED AND WITH BOUNDS (-3: (N+3)**2-2)
C     WHEN THE COEFFICIENTS ARE QUASI-NORMALIZED. C(1) TO C((N+1)**2-1)
C     CONTAIN THE COEFFICIENTS AND WE MUST HAVE
C     C(-3)=GM
C     C(-2)=A  THE SEMI-MAJOR AXIS OF THE REF ELLIPSOID
C     C(-1)=THE ANGULAR VELOCITY (=0, WHEN DEALING WITH T).
C
C
C SQUARE ROOT ARRAY
C
C     C((N+1)**2+K) = SQRT(K), 0.LE.K.LE.2(ABS(N)+1)-1   WHEN N <0
C
C
C     MOD--APRIL,1981
C     WITH THE USE OF THE TABLE OF SQUARE ROOTS STORED IN ARRAY ROOT,
C     THE DIMENSION OF THE C ARRAY ONLY NEEDS TO BE (N+1)**2-1 FOR BOTH
C     UN-NORMALIZED AND NORMALIZED COEFFICIENTS.  ARRAY ROOT MUST CONTAIN
C     SQUARE ROOT OF 0 TO N+2
C
C
C (B) RETURN VALUES
C
C G
C     THE RESULT IS STORED IN G AS FOLLOWS:
C
C     G1(1)=DW/DX, G1(2)=DW/DY, G1(3)=DW/DZ
C     G(1,1)=DDW/DDX, G(1,2)=G(2,1)=DDW/DXDY,
C     G(1,3)=G(3,1)=DDW/DXDZ, G(2,2)=DDW/DDY,
C     G(2,3)=G(3,2)=DDW/DYDZ AND G(3,3)=DDW/DDZ
C     WHERE W MAY BE INTERCHANGED WITH T AND
C     VARIABLES X, Y, Z ARE THE CARTESIAN COORDINATES
C     IN A LOCAL (FIXED) FRAME WITH ORIGIN IN THE POINT
C     OF EVALUATION, X POSITIVE NORTH, Y POSITIVE EAST,
C     AND Z POSITIVE IN THE DIRECTION OF THE RADIUS
C     VECTOR, (CF. REF.(1),EQ (4) AND (5)).
C     THE VALUES OF W OR T WILL BE RETURNED IN POTCC.
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER CAPN,ORDER,CAPN21,OLDORD
      LOGICAL QUASI,DERIV1,DERIV2,POLE
```

```
      LOGICAL FIRST,NEW,OLD,NPOLE
      REAL*4 C,CO
      REAL*8 M21,M21T,M21U,M21UO
      DIMENSION SML(181),CML(181),SMLP1(182),CMLP1(182),PO(6)
      DIMENSION SU(1810)
      COMMON/SQROOT/DZERO,ROOT(362)
      COMMON/GPOTCM/OLDT,OLDR,IZ,FIRST,OLDORD,I1,I2,I3,I4,
     &              I5,I6,I7,I8,I9,NMAXSV
      COMMON/CM/C20IN,G1(3),G2(3,3),CM3,CM2,CM1,CO,C(32760)
      EQUIVALENCE(SML(1),SMLP1(2)),(CML(1),CMLP1(2))
      IF(NMAXSV.NE.NMAX)FIRST=.FALSE.
      NMAXSV=NMAX
      IF(FIRST) GO TO 100
      FIRST=.TRUE.
      OLDT=2.DO
      J=IABS(NMAX)
      I=J+1
      I1=I+1
      I2=I1+I
      I3=I2+I
      I4=I3+I
      I5=I4+I
      I6=I5+I
      I7=I6+I
      I8=I7+I
      I9=I8+I
  100 CAPN=NMAX
      P=PO(1)
      R=PO(2)
      T=PO(3)
      U=PO(4)
      SL=PO(5)
      CL=PO(6)
      T2=T+T
      POLE=DABS(U).LE.1.D-9
      NEW=DABS(OLDR-R).GT.1.D-3.OR.DABS(OLDT-T).GT.1.D-9.OR.
     &    OLDORD.NE.ORDER.OR.POLE
      OLD=.NOT.NEW
      NPOLE=.NOT.POLE
      IF(OLD) GO TO 200
      OLDR=R
      OLDT=T
      OLDORD=ORDER
C
  200 QUASI=.FALSE.
      IF(CAPN.LT.O)QUASI=.TRUE.
      IF(QUASI)CAPN=-CAPN
      S=CM2/R
      S2=S**2
      CMLP1(1)=1.DO
C     CML(O)=1.DO
      SMLP1(1)=O.DO
C     SML(O)=O.DO
      DERIV1=.FALSE.
      IF(ORDER.GT.O)DERIV1=.TRUE.
      DERIV2=.FALSE.
      IF( ORDER.GT.1)DERIV2=.TRUE.
C
C  SML(M) AND CML(M) ARE THE SINE AND COSINE OF M*LONGITUDE
C
C     WRITE(6,DBUG)
```

```
C
      SML(1)=SL
      CML(1)=CL
C
      M1=1
      DO 300 M=2,CAPN
      SML(M)=SML(M1)*CL+CML(M1)*SL
      CML(M)=CML(M1)*CL-SML(M1)*SL
  300 M1=M
C
      CAPN21=CAPN+CAPN+1
      VM=0.DO
      VXM=0.DO
      VYM=0.DO
      VZM=0.DO
      SQNM1=1.DO
      SQNPM1=1.DO
      IF(.NOT.DERIV2) GO TO 400
      VXXM=0.DO
      VYYM=0.DO
      VZZM=0.DO
      VXYM=0.DO
      VXZM=0.DO
      VYZM=0.DO
  400 KM=(CAPN+1)**2
      MAX2=CAPN21
C
C WE NOW USE THE CLENSHAW ALGORITHM, CF. REF.(2), EQ(9),
C MODIFIED IN AN OBVIOUS WAY FOLLOWING REF.(1).
C
      ITWO=2
      DO 1700 IM=IZ,CAPN
      M=CAPN-IM
      MPLUS1=M+1
      IF(M.EQ.0)ITWO=1
      KM=KM-ITWO
      K=KM
      N21=CAPN21
      VS=0.DO
      VC=0.DO
      VS1=0.DO
      VC1=0.DO
      VXS1=0.DO
      VXC1=0.DO
      VZS=0.DO
      VZC=0.DO
      VZS1=0.DO
      VZC1=0.DO
      VXC=0.DO
      VXS=0.DO
C
      IF(.NOT.DERIV2) GO TO 500
      VXXC=0.DO
      VXXS=0.DO
      VXXC1=0.DO
      VXXS1=0.DO
      VZZC=0.DO
      VZZS=0.DO
      VZZC1=0.DO
      VZZS1=0.DO
      VXZC=0.DO
```

```
      VXZS=0.D0
      VXZC1=0.D0
      VXZS1=0.D0
  500 CM=CMLP1(MPLUS1)
      SM=SMLP1(MPLUS1)
C
      NM1=CAPN-M+2
      N1=CAPN+1
      NPM1=CAPN+M+2
C
      IF(DERIV2)M2=M*M
      IF(OLD) GO TO 1300
C
      N=CAPN+1
      DO 1000 IN=M,CAPN
      N=N-1
      NM2=NM1
      NM1=NM1-1
      NPM1=NPM1-1
      IF(.NOT.QUASI) GO TO 600
      SQNM2=SQNM1
      SQNM1=ROOT(NM1)
      SQNPM2=SQNPM1
      SQNPM1=ROOT(NPM1)
      SQ1=SQNM1*SQNPM1
      A1=S*N21/SQ1
      B2=-S2*SQ1/(SQNM2*SQNPM2)
      GO TO 700
  600 A1=S*N21/NM1
      B2=-(S2*NPM1)/NM2
  700 A1T=A1*T
      A1U=A1*U
      N21=N21-2
      CK=C(K)
      CK1=C(K+1)
      K=K-N21
      V2=VC1
      VC1=VC
      VC=VC1*A1T+V2*B2+CK
      V2=VS1
      VS1=VS
      VS=VS1*A1T+V2*B2+CK1
C
      IF(.NOT.DERIV1) GO TO 1000
      CKZ=CK*N1
      CK1Z=CK1*N1
      V2=VXC1
      VXC1=VXC
      VXC=VXC1*A1T+VC1*A1U+V2*B2
      V2=VXS1
      VXS1=VXS
      VXS=VXS1*A1T+VS1*A1U+V2*B2
      V2=VZC1
      VZC1=VZC
      VZC=VZC1*A1T+V2*B2-CKZ
      V2=VZS1
      VZS1=VZS
      VZS=VZS1*A1T+V2*B2-CK1Z
      N1=N
      IF(.NOT.DERIV2) GO TO 1000
      N2=N+2
```

```
      V2=VZZC1
      VZZC1=VZZC
      VZZC=VZZC1*A1T+V2*B2+N2*CKZ
      V2=VZZS1
      VZZS1=VZZS
      VZZS=VZZS1*A1T+V2*B2+N2*CK1Z
      IF(NPOLE) GO TO 800
      V2=VXXC1
      VXXC1=VXXC
      VXXC=(VXXC1-VC1)*A1T+(A1U+A1U)*VXC1+V2*B2
      V2=VXXS1
      VXXS1=VXXS
      VXXS=(VXXS1-VS1)*A1T+(A1U+A1U)*VXS1+V2*B2
  800 V2=VXZC1
      VXZC1=VXZC
      VXZC=(VXZC1)*A1T+(A1U)*VZC1+V2*B2
      V2=VXZS1
      VXZS1=VXZS
      VXZS=(VXZS1)*A1T+(A1U)*VZS1+V2*B2
 1000 CONTINUE
      SU(M+1)=VC
      SU(M+I1)=VS
      IF(.NOT.DERIV1) GO TO 1500
      SU(M+I2)=VXC
      SU(M+I3)=VXS
      SU(M+I4)=VZC
      SU(M+I5)=VZS
      IF(.NOT.DERIV2) GO TO 1500
      SU(M+I6)=VZZC
      SU(M+I7)=VZZS
      SU(M+I8)=VXZC
      SU(M+I9)=VXZS
      GO TO 1500
 1300 VC=SU(M+1)
      VS=SU(M+I1)
      IF(.NOT.QUASI) GO TO 1400
      SQNPM1=ROOT(MAX2)
      SQNPM2=ROOT(MAX2+1)
 1400 NPM1=MAX2
      MAX2=MAX2-2
      IF(.NOT.DERIV1) GO TO 1500
      VXC=SU(M+I2)
      VXS=SU(M+I3)
      VZC=SU(M+I4)
      VZS=SU(M+I5)
      IF(.NOT.DERIV2) GO TO 1500
      VZZC=SU(M+I6)
      VZZS=SU(M+I7)
      VXZC=SU(M+I8)
      VXZS=SU(M+I9)
C
 1500 U0=U
      IF(M.EQ.0) U0=1.D0
      AUX=NPM1
      IF(QUASI) AUX=SQNPM1/SQNPM2
      M21=S*AUX
      M21U=M21*U
      IF(.NOT.DERIV1) GO TO 1700
      M21T=M21*T
      M21U0=M21*U0
      IF(.NOT.DERIV2) GO TO 1600
```

```
      VZZM=VZZC*CM+VZZS*SM+M21U*VZZM
      IF(M.GT.0) VXYM=M*(VXS*CM-VXC*SM)+M21U*VXYM-M21T*VYM
      VXZM=VXZC*CM+VXZS*SM-M21T*VZM+M21U*VXZM
      VYZM=(VZS*CM-VZC*SM)*M+M21U0*VYZM
      IF(POLE) VXXM=VXXC*CM+VXXS*SM+M21*(U*(VXXM-VM)-T2*VXM)
      IF(NPOLE) VYYM=-(VC*CM+VS*SM)*M2+M21U0*VYYM
 1600 VXM=VXC*CM+VXS*SM-M21T*VM+M21U*VXM
      VYM=M*(VS*CM-VC*SM)+M21U0*VYM
      VZM=(VZC*CM+VZS*SM)+M21U*VZM
 1700 VM=VC*CM+VS*SM+M21U*VM
C
C NOW ADD THE CONTRIBUTIONS FROM THE ROTATIONAL POTENTIAL
C
      OM2=CM1**2
      S=CM3/R
      GPOTDR=S*VM+OM2*P**2*.5D0
      IF(.NOT.DERIV1) RETURN
      S=S/R
      G1(1)=S*VXM-T*P*OM2
      G1(2)=S*VYM
      G1(3)=VZM*S+U**2*OM2*R
      IF(.NOT.DERIV2) RETURN
      S=S/R
      IF(NPOLE) GO TO 1900
      VXXM=VXXM+VZM
      VYYM=-(VXXM+VZZM)
      GO TO 2000
 1900 VYYM=VZM+(VYYM-T*VXM)/U
      VXXM=-(VZZM+VYYM)
 2000 G2(1,1)=VXXM*S+OM2*T**2
      G2(1,2)=S*VXYM
      G2(2,1)=G2(1,2)
      G2(1,3)=S*(VXZM-VXM)-U*T*OM2
      G2(3,1)=G2(1,3)
      G2(2,2)=VYYM*S+OM2
      G2(2,3)=S*(VYZM-VYM)
      G2(3,2)=G2(2,3)
      G2(3,3)=S*VZZM+U**2*OM2
      RETURN
      END
C*********************************************************************
      SUBROUTINE SETCM(CAPN)
C
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER CAPN
      REAL*4 C,CO
      COMMON/SQROOT/DZERO,ROOT(362)
      COMMON/CM/C20IN,G1(3),G2(3,3),CM3,CM2,CM1,CO,C(32760)
C
C THIS ROUTINE SETS THE SQUARE ROOT TABLE IN COMMON
C CM AND CREATES QUASI-NORMALIZED COEFFICIENTS
C FROM NORMALIZED COEFFICIENTS
C
C
C APRIL 1981 CCG
C  THIS VERSION STORES  SQUARE ROOT TABLE IN COMMON SQROOT
C
      DZERO=0.D0
      DO 22 I=1,362
   22 ROOT(I)=DSQRT(DFLOAT(I))
      G1(1)=0.D0
```

```
      G1(2)=0.D0
      G1(3)=0.D0
      SMALLC=1.D0
      IF(C0.NE.0.D0)SMALLC=1.D0/C0
      SQ2=DSQRT(2.D0)
      DO 200 N=1,CAPN
      N2=N+N
      S21=DSQRT(N2+1.D0)
      K=N**2
C
C D IS THE QUASI-NORMALIZATION FACTOR FOR ZONAL TERMS
C
      D=SMALLC*S21
      C(K)=C(K)*D
C
C GG IS THE QUASI-NORMALIZATION FACTOR FOR NON-ZONAL TERMS
C
      GG=D*SQ2
      DO 100 J=1,N
      KJ2=J+J+K
      C(KJ2-1)=C(KJ2-1)*GG
      C(KJ2)=C(KJ2)*GG
  100 CONTINUE
  200 CONTINUE
      RETURN
      END
C*********************************************************************
      SUBROUTINE STORC(N,M,CNM,SNM)
C
C STORE INDIVIDUAL C AND S TERMS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 C,C0
      COMMON/CM/C20IN,G1(3),G2(3,3),CM3,CM2,CM1,C0,C(32760)
C
C SUM OF THE PREVIOUS NUMBER OF TERMS
C
      J=(N-1)*(N+1)
      IF(M.EQ.0) GO TO 10
      K=M+M
      C(J+K)=CNM
      C(J+K+1)=SNM
      RETURN
   10 C(J+1)=CNM
      RETURN
      END
C*********************************************************************
      SUBROUTINE LOADCS(NMAX)
C
      IMPLICIT REAL *8(A-H,O-Z)
      REAL *4 C,C0
      COMMON/CM/C20IN,G1(3),G2(3,3),CM3,CM2,CM1,C0,C(32760)
  100 READ(12,END=200)N,M,CNM,SNM
      IF(N.EQ.2.AND.M.EQ.0) C20IN=CNM
      IF(N.GT.180) GO TO 200
      IF(N.GT.NMAX) GO TO 100
      IF(N.LT.5) WRITE(6,55) N,M,CNM,SNM
   55 FORMAT(1X,4G20.12)
      CALL STORC(N,M,CNM,SNM)
      GO TO 100
  200 CONTINUE
```

```
      REWIND 12
      RETURN
      END
C*****************************************************************
      SUBROUTINE MODC(N,M,CNM,SNM)
C
      IMPLICIT REAL *8(A-H,O-Z)
      REAL *4 C,CO
      COMMON/CM/C2OIN,G1(3),G2(3,3),CM3,CM2,CM1,CO,C(32760)
      J=(N-1)*(N+1)
      IF(M.EQ.0) GO TO 10
      K = M+M
      C(J+K) = C(J+K) + CNM
      C(J+K+1) = C(J+K+1) + SNM
      RETURN
   10 C(J+1) = C(J+1) + CNM
      RETURN
      END
C*****************************************************************
      SUBROUTINE NORMAL(GM,DJN,AE,OMEGA,X,Y,Z,U,GAMMA,GRAD)
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DJN(20)
      PSQ = X**2+Y**2
      RSQ = PSQ + Z**2
      R = DSQRT(RSQ)
      GMOR = GM/R
      AOR = AE/R
      SINE=Z/R
      PNL1 = SINE
      PN=1.5D0*SINE**2-0.5D0
      F = 1.D0
      FP = 0.D0
      FPP = 0.D0
      AORN = 1.D0
      AOR2 = AOR**2
      DO 10 N=2,20,2
      AORN = AORN*AOR2
      TERM=-DJN(N)*AORN*PN
      F = F + TERM
      FP = FP -N*TERM/R
      FPP = FPP+N*(N+1)*TERM/RSQ
      SAVE =(N+N+1.D0)/(N+1.D0)*SINE*PN-N/(N+1.D0)*PNL1
      PNL1 = PN
      PN = SAVE
      SAVE = (N+N+3.D0)/(N+2.D0)*SINE*PN-(N+1.D0)/(N+2.D0)*PNL1
      PNL1 = PN
   10 PN=SAVE
      U = GMOR*F
      GAMMA =GMOR*FP-U/R
      OMEGA2 = OMEGA**2
      GRAD=(U+U)/RSQ-2.D0*GMOR*FP/R+GMOR*FPP+OMEGA2*PSQ/RSQ
      GRAD = DABS(GRAD)
      U = U+0.5D0*OMEGA2*PSQ
      GAMMA = DABS(GAMMA+PSQ/R*OMEGA2)
      RETURN
      END
C*****************************************************************
      SUBROUTINE REFVAL(A,GM,J2,OMEGA,FLATI)
C
      IMPLICIT REAL*8(A-H,O-Z)
```

```
      REAL*8 J2
      PWR=(OMEGA*A)**2*A/GM
      ESQSV=3.D0*J2+PWR
      ESQ = ESQSV
      ITIME = 0
    5 ITIME = ITIME + 1
      EP2 = ESQ/(1.D0-ESQ)
      FACT=1.D0/(1.D0-ESQ)**1.5D0
      DS=1.D0
      TWOQP = 0.D0
      DO 10 N = 1, 20
      TWOQP=TWOQP+DS*4.D0*N/((N+N+1.D0)*(N+N+3.D0))*FACT
      DS = -DS
      FACT = FACT*EP2
   10 CONTINUE
      ESQ=ESQSV+PWR*(4.D0/15.D0/TWOQP-1.D0)
      FLATI=1.D0/(1.D0-DSQRT(1.D0-ESQ))
      TEST = DABS(FLATI-298.257D0)
      IF(TEST.GT.1.D0) GO TO 100
      IF(ITIME.LT.10) GO TO 5
      WRITE(6,20) A,GM,J2,OMEGA
   20 FORMAT('   REFERENCE VALUES OF NORMAL ELLIPSOID A GM J2 OMEGA'/,
     *       3X,4G20.12)
      WRITE(6,30) FLATI
   30 FORMAT('   COMPUTED VALUE OF FLATTENING INVERSE ',G20.12)
      RETURN
  100 WRITE(6,110)
  110 FORMAT('   SOMETHING WRONG IN REFVAL  FLATTENING NOT CONVERGING')
      STOP
      END
C*********************************************************************
      SUBROUTINE TRANF(ISWICH,GLAT,ELON,HT,X,Y,Z,AE,FLAT)
C
      IMPLICIT REAL* 8(A-H,O-Z)
      COMMON/PIDTR/PI,DTR
      COMMON/TRANCM/TOL,MAXIT
      FLATFN=(2.D0-FLAT)*FLAT
      FUNSQ=(1.D0-FLAT)**2
      IF(ISWICH.GT.1) GO TO 1000
      SPHI = DSIN(GLAT*DTR)
      G1=AE/DSQRT(1.D0-FLATFN*SPHI**2)
      G2=G1*FUNSQ+HT
      G1=G1+HT
      X=G1*DCOS(GLAT*DTR)
      Y=X*DSIN(ELON*DTR)
      X=X*DCOS(ELON*DTR)
      Z=G2*SPHI
      RETURN
 1000 RSQ=X**2+Y**2
      R=DSQRT(RSQ)
      E=DATAN2(Y,X)
      IF(E.LT.0.D0) E=E+PI+PI
      ELON=E/DTR
      RHO=DSQRT(Z**2+RSQ)
      SPHI=Z/RHO
      GLATR=DARSIN(SPHI)
      HT=RHO-AE*(1.D0-FLAT*SPHI**2)
      ITER=0
 1100 SPHI=DSIN(GLATR)
      CPHI=DCOS(GLATR)
      G1=AE/DSQRT(1.D0-FLATFN*SPHI**2)
```

```
      G2=G1*FUNSQ+HT
      G1 = G1+HT
      DR=R-G1*CPHI
      DZ=Z-G2*SPHI
      DHT=DR*CPHI+DZ*SPHI
      HT=HT+DHT
      DLATR = (DZ*CPHI-DR*SPHI)/(AE+HT)
      GLATR = GLATR+DLATR
      ITER=ITER+1
      IF(ITER.GT.MAXIT) GO TO 1200
      IF(DABS(DLATR).GT.TOL) GO TO 1100
      IF(DABS(DHT)/(AE+HT).GT.TOL) GO TO 1100
 1200 GLAT=GLATR/DTR
      RETURN
      END
//GO.FT12F001  DD DSN=A.M66774.WENZEL.COEF200.UNFMD,DISP=SHR
//GO.SYSIN DD *
  67  65 290 288  10
//
```

2.6 Program SGSINT

```
//SQSINT   JOB NOTIFY=6461
/*SERVICE -4
/*JOBPARM T=15,L=99,R=5120,PRINT=ALL
// EXEC FORTVCLG,RC=5120K,RL=2048K,RG=5120K
//FORT.SYSIN DD *
C-----------------------------------------------------------------------
C***********************************************************************
C                                                                      *
C  PROGRAM NAME :    SQSINTT                                           *
C  FUNCTION     :  PARTIAL STOKES INTEGRATION OF SPARSE POINT         *
C                  GRAVITY ANOMALIES .                                *
C COMPILER      :  VS FORTRAN VERSION 2                               *
C AUTHOR        :  JOHN MANTHA                                        *
C HISTORY       :  JULY   26, 1986 - VERSION 1.0                      *
C REFERENCE     :  U.N.B.  TECHNICAL  REPORT  #                       *V
C***********************************************************************
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C  PARAMETERS:   NAME   I/O   DESCRIPTION
C
C               NP       I    NUMBER OF COMPUTATION POINTS
C               PSIO     I    RADIUS OF SPHERICAL CAP
C               ALATSW   I    LAT OF SOUTHWEST CORNER OF BLOCK A
C               ALONSW   I    LONG OF SOUTHWEST CORNER OF BLOCK A
C               ALATNE   I    LAT OF NORTHEAST CORNER OF BLOCK A
C               ALONNE   I    LONG OF NORTHEAST CORNER OF BLOCK A
C               PHISW    I }  THE SOUTH WEST AND NORTH EAST CORNER
C               ALASW    I }  COORDINATES OF THE COMPUTATIONAL
C               PHINW    I }  AREA.
C               ALASE    I }
C               XNTPHI   I    THE GRID INTERVAL IN LAT. DIRECTION
C                             IN MINUTES.
C               XNTALA   I    THE GRID INTERVAL IN LON. DIRECTION
C                             IN MINUTES.
C               PHI      I    CURRENT ARRAYS OF LAT. OF CENTER POINT
C               ALA      I    CURRENT ARRAYS OF LONG. OF CENTER POINT
C               PHIO     D    LAT OF CENTER POINT
C               PHIO1    D    LAT OF NEXT CENTER POINT
C               ALAO     D    LONG OF CENTER POINT
C               DLATSB   D    LAT OF SOUTHERLY BOUND. OF BLOCK B
C               DLONWB   D    LONG OF WESTERLY BOUND. OF BLOCK B
C               DLATNB   D    LAT OF NORTHERLY BOUND. OF BLOCK B
C               DLONEB   D    LONG OF WESTERLY BOUND. OF BLOCK B
C               SLATS    D    LAT OF SOUTHERLY BOUND. OF STRIP
C               SLATN    D    LAT OF NORTHERLY BOUND. OF STRIP
C               SLONE    D    LONG OF EASTERLY BOUND. OF COMP. CAP
C               SLONW    D    LONG OF WESTERLY BOUND. OF COMP. CAP
C                             VALUES STORED IN STRIP
C               SIDG     D    ARRAY OF GRAVITY ANOMALIES
C               SSDG     D    CORRES. STAND. DEV.
C               SLAT     D    CORRES. LATITUDES
C               SLON     D    CORRES. LONGITUDES
C                             VALUES USED IN SUBROUTINE INTGR
C               DELTG    I    ARRAY OF GRAVITY ANOMALIES  (MGAL)
C               SDELG    I    CORRESP. STAND. DEV.        (MGAL)
C               PHIS     I    CORRESP. LATITUDES          (RAD)
C               ALAS     I    CORRESP. LONGITUDES         (RAD)
C               DG       O    ARRAY OF GRAV.ANOM. WITHIN PSIO (MGAL)
C               SDG      O    ARRAY OF CORRESP. STAND. DEV. (MGAL)
C               PSI      O    ARRAY OF CORRESP. SPHERICAL DISTANCES (RAD)
C               NMAX     I    DIMENSION OF THESE ARRAYS AS SPECIFIED
C                             IN CALLING PROGRAM
```

```
C                NTOTAL  I    ACTUAL DIMENSION OF DELTG,SDELG,PHI,ALA
C                DN      O    COMPUTED GEOID UNDULATION (M)
C                SDN     O    CORRESPONDING STAND. DEV. (M)
C
C NOTE:  ALL VALUES ARE ENTERED INTO THE PROGRAM AS DECIMAL DEGREES
C
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      CHARACTER * 5 MODE
      DIMENSION PHI(1891),ALA(1891)
      DIMENSION ALAT(50000),ALON(50000),ADG(50000),ASDG(50000)
      DIMENSION SLAT(50000),SLON(50000),SIDG(50000),SSDG(50000)
      DIMENSION PHIS(10000),ALAS(10000),DELTG(10000),SDELTG(10000)
      DIMENSION DG(10000),SDG(10000),PSI(10000)
         NMAX=10000
C
C NOTE: THE DIMENSION VALUES OF ARRAYS FOR PHI AND ALA CORRESPOND TO
C       THE NUMBER OF COMPUTATION POINTS(NP).
C
C-----------------------------------------------------------------------
C     INITIALIZATIONS OF CONSTANTS
C
      PI=DARCOS(-1.D0)
      DR=PI/180.D0
      NPB=2000
C-----------------------------------------------------------------------
C
C #1) READ IN THE MODE OF OPERATION.POINT MODE OR GRID MODE.
C
       READ(5,1000) MODE
 1000  FORMAT (A5)
C
C #2) READ IN CAPSIZE
C
      READ (5,*)PSIO
C
C #3) READ IN SOUTWEST AND NORTHEAST CORNER OF BLOCK A
C
      READ(5,*)ALATSW,ALONSW,ALATNE,ALONNE
C
C #4) READ IN THE SOUTHWEST AND NORTHEAST CORNER OF GRIDDED BLOCK
C     ALONG WITH THE DESIRED INTERVAL.
C
       READ(5,*)PHISW,ALASW,PHINW,ALASE,XNTPHI,XNTALA
C
C #5) READ IN THE NUMBER OF POINTS
C
      READ(5,*) NP
C
C #5) READ IN THE COORDINATES OF THE CENTER POINTS
C
      DO 11 I=1,NP
        READ(5,*)PHI(I),ALA(I)
 11    CONTINUE
C
         IF(MODE.EQ.'GRID')THEN
C
C-----------------------------------------------------------------------
C                        GRID MODE
```

```
C                    **********
C
C     THE PROGRAM WILL USE THE GRID MODE AND CALCUALTE THE COORDINATES
C     OF THE GRID INTERSECTION LINES THROUGH THE CALL TO THE SUBROUTINE
C     AREA.
C
C   THE AREA:
C           SUBROUTINE GENERATES THE COMPUTATION POINTS FROM MESH (AREA)
C           GIVEN THE BOUNDARIES OF THE GRID AND THE INTERVAL ALONG EACH
C           DIRECTION.
C
C  INPUT:
C  *******PHISW:LATITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C         ALASW:LONGITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C         PHINW:LATITUDE OF THE NORTH WEST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C         ALASE:LONGITUDE OF THE SOUTH EAST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C        XNTPHI:INTERVAL ALONG THE LATITUDINAL DIRECTION IN MINUTES
C        XNTALA:INTERVAL ALONG THE LONGITUDINAL DIRECTION IN MINUTES
C
C   OUTPUT:
C   ******** PHI:ARRAY OF LATITUDES OF COMPUTATION POINTS IN DEGREES.
C            ALA:ARRAY OF LONGITUDES OF COMPUTATION POINTS IN DEGREES.
C              K:NUMBER OF EVALUATION POINTS IN THE AREA.
C             IP:NUMBER OF COMPUTAION POINTS IN THE LATITUDINAL
C                DIRECTION.
C             IL:NUMBER OF COMPUTATION POINTS IN THE LONGITUDINAL
C                DIRECTION.
C
C           THE MAXIMUM NUMBER OF POINTS THAT CAN BE STORED
C           IN THE ARRAYS PHI AND ALA ARE 10,000.
C-----------------------------------------------------------------------
C
      CALL AREA(PHISW,ALASW,PHINW,ALASE,XNTPHI,XNTALA,PHI,ALA,K)
       NP=K
       ENDIF
C
C #6) READ IN THE GRAVITY VALUES FOR BLOCK (A) FROM DISK
C
            I=0
 333     READ(1,111,END=3333)DLAT,DLON,DDG,DSDG,DH,DSH
            IF(DLAT.LT.ALATSW)GO TO 333
            IF(DLON.LT.ALONSW.OR.DLON.GT.ALONNE)GO TO 333
            IF(DLAT.GT.ALATNE)GO TO 3333
              ALAT(I)=DLAT
              ALON(I)=DLON
              ADG(I)=DDG
             ASDG(I)=DSDG
                 I=I+1
                ICOUNT=I
                GO TO 333
3333  CONTINUE
111    FORMAT(2F10.2,4F10.4)  (2F10.4, 2F8.2, 2F7.1)
C
C FORM BOUNDARIES OF INNERBLOCK B,THE DISTANCE BEING PSIO FROM
C       BOUNDARIES OF BLOCK A.
C
C DETEMINE THE CHANGE IN LATITUDE DUE TO CONVERGENCE.
```

(see errata at back of this volume)

```
C
                  RPSIO=PSIO+DR
                  RPSION=(RPSIO/DCOS(ALATNE+DR))+0.2D-3
                  RPSIOS=RPSIOS/DR
                  RPSION=RPSION/DR
C
C FORMING OF BOUNDARIES OF BLOCK B
C
                BLONWB=ALONSW+RPSION
                BLONEB=ALONNE-RPSION
                BLATSB=ALATSW+PSIO
                BLATNB=ALATNE-PSIO
C
C  DO LOOP BEGINS TO TEST LOCATION OF CENTER POINTS.
C
          DO 13 JJ=1,NP
                PHIO=PHI(JJ)
                ALAO=ALA(JJ)
C
C  TEST TO SEE IF CENTER POINTS LIE OUTSIDE BLOCK B.
C
         IF(PHIO.LT.BLATSB.OR.PHIO.GT.BLATNB)GO TO 555
         IF(ALAO.LT.BLONWB.OR.ALAO.GT.BLONEB)GO TO 555
         IF(JJ.EQ.1)GO TO 777
         PHIO1=PHI(JJ-1)
         CHECK=DABS(PHIO-PHOI1)
         IF(CHECK.LT.0.D-6)GO TO 155
 777     CONTINUE
C
C DETERMINE THE BOUNDARIES OF THE STRIP TO BE LOADED FROM BLOCK B.
C
                   SLATS=PHIO-(PSIO+0.02)
                   SLATN=PHIO+(PSIO+0.02)
C
          DO 99 BER=1,NPB
                  SLAT(BER)=0
                  SLON(BER)=0
                  SIDQ(BER)=0
                  SSDQ(BER)=0
99        CONTINUE
C
                       NPB=0
         DO 15 JJK=1,ICOUNT
             IF(ALAT(JJK).LT.SLATS)GO TO 15
             IF(ALAT(JJK).GT.SLATN)GO TO 155
C
                       NPB=NPB+1
C
                  SLAT(NPB)=ALAT(JJK)
                  SLON(NPB)=ALON(JJK)
                  SIDQ(NPB)=ADQ(JJK)
                  SSDQ(NPB)=ASDQ(JJK)
C
 15      CONTINUE
 155     CONTINUE
C
C DETERMINE THE BOUNDARIES IN THE LONGITUDINAL DIRECTION TO
C FORM THE CAP BOUNDARIES.
C
                   BPSIO=RPSIO/DCOS(PHIO+DR)
                   BPSIO=(BPSIO+0.2D-3)/DR
```

1

(i.e. 1.D-6 rather than 0.D-6)

(see errata at back of this volume)

```
                SLONE=ALAO+BPSIO
                SLONW=ALAO-BPSIO
C
                  NTOTAL=0
          DO 16 KJ=1,NPB
            IF(SLON(KJ).GT.SLONE.OR.SLON(KJ).LT.SLONW)GO TO 16
                  NTOTAL=NTOTAL+1
             PHIS(NTOTAL)=SLAT(KJ)*DR
             ALAS(NTOTAL)=SLON(KJ)*DR
            DELTG(NTOTAL)=SIDG(KJ)
           SDELTG(NTOTAL)=SSDG(KJ)
 16       CONTINUE
C
C CHANGE VALUES TO RADIANS IN ORDER TO INPUT INTO
C SUBROUTINE INTGR.
C
             PHIO=PHIO*DR
             ALAO=ALAO*DR
C
             CALL INTGR (RPSIO,PHIO,ALAO,DELTG,SDELTG,PHIS,ALAS,
     *                    DG,SDG,PSI,NMAX,NTOTAL,NUM,DN,SDN)
C
                 PHIO=PHIO/DR
                 ALAO=ALAO/DR
              SDN1=0.DO
C
        WRITE(6,1002)PHIO,ALAO,DN,SDN1
        WRITE(6,1002)PHIO,ALAO,DN,SDN1
 1002    FORMAT(' ',F10.4,1X,F10.4,1X,F10.3,1X,F8.3)
C
         GO TO 13
 555     CONTINUE
         WRITE(6,*)'POINT LIES OUTSIDE OF COMPUTATION AREA'
 13      CONTINUE
           STOP
           END
C*********************************************************************
      SUBROUTINE AREA(PHISW,ALASW,PHINW,ALASE,XNTPHI,XNTALA,PHI,ALA,K)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION PHI(1891),ALA(1891)
C---------------------------------------------------------------------
C     FUNCTION :    GENERATES THE COMPUTATION POINTS
C                   GIVEN THE BOUNDARIES OF THE GRID AND
C                   AND THE INTERVAL ALONG EACH DIRECTION.
C     AUTHOR    :   HASSAN FASHIR
C     HISTORY   :   FEBRUARY 16, 1985 - VERSION  1.0
C---------------------------------------------------------------------
C INPUT:
C ******
C      PHISW   : LATITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C      ALASW   : LONGITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C      PHINW   : LATITUDE OF THE NORTH WEST CORNER OF GRID TO BE COMPUTED
C      ALASE   : LONGITUDE OF THE SOUTH EAST CORNER OF GRID TO BE COMPUTED
C      XNTPHI  : INTERVAL ALONG THE LATITUDINAL DIRECTION IN MINUTES
C      XNTALA  : INTERVAL ALONG THE LONGITUDINAL DIRECTION IN MINUTES
C
C OUTPUT:
C *******
C      PHI     : ARRAY OF LATITUDES OF COMPUTATION POINTS.
C      ALA     : ARRAY OF LONGITUDES OF COMPUTATION POINTS.
C      K       : NUMBER OF EVALUATION POINTS IN THE AREA
```

```
C       IP      : NUMBER OF COMPUTAION POINTS IN THE LATITUDINAL DIR.
C       IL      : NUMBER OF COMPUTATION POINTS IN THE LONGITUDINAL DIR.
C
C          THE MAXIMUM NUMBER OF POINTS THAT CAN BE STORED
C          IN THE ARRAYS PHI AND ALA IS 10,000.
C-----------------------------------------------------------------------
       XMIN=DFLOAT(60)
       XMINE=(ALASE-ALASW)*XMIN
       IL=(XMINE/XNTALA)+1
       XMINW=(PHINW-PHISW)*XMIN
       IP=(XMINW/XNTPHI)+1
       K=IL*IP
        DO 10 I=1,IP
         DO 11 J=1,IL
          PHI((I-1)*IL+J)=PHISW+DFLOAT(I-1)*XNTPHI/XMIN
          ALA((I-1)*IL+J)=ALASW+DFLOAT(J-1)*XNTALA/XMIN
 11      CONTINUE
 10     CONTINUE
        RETURN
        END
C***********************************************************************
       SUBROUTINE INTGR (PSIO,PHIO,ALAO,DELTG,SDELG,PHI,ALA,
      *                  DG,SDG,PSI,NMAX,NTOTAL,NUM,DN,SDN)
C-----------------------------------------------------------------------
C      FUNCTION   :   ANALYTICAL STOKES INTEGRATION IN A SPHERICAL CAP
C      AUTHOR     :   ALFRED KLEUSBERG
C      HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C-----------------------------------------------------------------------
C   PARAMETERS:     NAME    I/O   DESCRIPTION
C   ***********
C                   PSIO     I    RADIUS OF SPHERICAL CAP (RAD)
C                   PHIO     I    LATITUDE  OF CENTRE POINT    (RAD)
C                   ALAO     I    LONGITUDE OF CENTRE POINT    (RAD)
C                   DELTG    I    ARRAY OF GRAVITY ANOMALIES   (MGAL)
C                   SDELG    I    CORRESP. STAND. DEV.         (MGAL)
C                   PHI      I    CORRESP. LATITUDES           (RAD)
C                   ALA      I    CORRESP. LONGITUDES          (RAD)
C                   DG       O    ARRAY OF GRAV.ANOM. WITHIN PSIO (MGAL)
C                   SDG      O    ARRAY OF CORRESP. STAND. DEV. (MGAL)
C                   PSI      O    ARRAY OF CORRESP. SPHER. DISTANCES (RAD)
C                   NMAX     I    DIMENSION OF THESE ARRAYS AS SPECIFIED
C                                 IN CALLING PROGRAM
C                   NTOTAL   I    ACTUAL DIMENSION OF DELTG,SDELG,PHI,ALA
C                   NUM      O    ACTUAL DIMENSION OF DG,SDG,PSI
C                   DN       O    COMPUTED GEOID UNDULATION (M)
C                   SDN      O    CORRESPONDING STAND. DEV. (M)
C
C
C      EXTERNALS:      CUFIT, STINT
C      **********
C-----------------------------------------------------------------------
       IMPLICIT REAL*8 (A-H,O-Z)
C
       DIMENSION DELTG(NMAX), SDELG(NMAX), PHI(NMAX), ALA(NMAX)
       DIMENSION DG(NMAX),    SDG(NMAX),   PSI(NMAX)
       DIMENSION COEFF(10), SCOEFF(10)
       MAX = 10
C
C      SELECT GRAVITY ANOMALIES WITHIN A SPHERICAL DISTANCE OF PSIO
C      ***********************************************************
       CALL SELCT (PSIO,PHIO,ALAO,DELTG,SDELG,PHI,ALA,NMAX,NTOTAL,
```

```
     *            DG,SDG,PSI,NUM,PSIMIN)
C
            IF(NUM.GT.1.AND.PSIMIN.LT.(PSIO/3.DO))GO TO 1101
              DN=-999.9
              SDN=-999.9
             GO TO 2101
 1101 CONTINUE
C
C     POLYNOMIAL FIT FOR GRAVITY ANOMALIES
C     ####################################
      CALL CUFIT (PSIO,DG,SDG,PSI,NMAX,NUM,COEFF,SCOEFF,MAX,MCOEFF)
C
C     ANALYTICAL STOKES INTEGRATION
C     #############################
      CALL STINT (PSIO,COEFF,SCOEFF,MAX,MCOEFF,DN,SDN)
C
 2101    CONTINUE
      RETURN
      END
C*********************************************************************
      SUBROUTINE CUFIT (PSIO,DG,SDG,PSI,NMAX,NUM,
     *                  COEFF,SCOEF,MAX,MCOEFF)
C---------------------------------------------------------------------
C     FUNCTION  :   POLYNOMIAL FIT TO GRAVITY ANOMALIES
C     AUTHOR    :   ALFRED KLEUSBERG
C     HISTORY   :   JUNE 06, 1986 - VERSION  1.0
C---------------------------------------------------------------------
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C  PARAMETERS:   NAME   I/O    DESCRIPTION
C  ###########
C                PSIO    I     MAXIMUM SPHERICAL DISTANCE (RAD)
C                DG      I     ARRAY OF GRAVITY ANOMALIES (MGAL)
C                SDG     I     ARRAY OF CORRESP. STAND. DEV. (MGAL)
C                PSI     I     ARRAY OF CORRESP. SPHER. DISTANCES (RAD)
C                NMAX    I     DIMENSION OF THESE ARRAYS AS SPECIFIED
C                NUM     I     ACTUAL DIMENSION OF THESE ARRAYS
C                COEFF   O     POLYNOMIAL COFFICIENT ARRAY (MGAL)
C                SCOEF   O     CORRESP. STAND. DEV.        (MGAL)
C                MAX     O     DIMENSION OF THESE ARRAYS AS SPECIFIED
C                MCOEFF  O     ACTUAL DIMENSION OF THESE ARRAYS
C
C  EXTERNALS:    SPIN
C  ##########
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION DG(NMAX), SDG(NMAX), PSI(NMAX)
      DIMENSION COEFF(MAX), SCOEF(MAX)
C
      DIMENSION A(10), ATA(10,10), ATL(10)
      DIMENSION ASIG(10), ASIGA(10,10), SC(10,10)
      ZERO = 0.DO
      MAXI = 10
C
      MCOEFF = MAX
      IF(NUM.LE.MCOEFF) MCOEFF = NUM-1
C
  500 CONTINUE
C
C     INITIALIZE ARRAYS
```

```
C     ******************
      DO 100 I=1,MAXI
      A(I) = ZERO
      ASIG(I) = ZERO
      ATL(I) = ZERO
      DO 100 J=1,MAXI
      ATA(I,J) = ZERO
       IF(I.EQ.J)ATA(I,J)=1.0D-12
      ASIGA(I,J) = ZERO
  100 CONTINUE
C
      DO 150 I=1,NUM
      OBS = DG(I)
      SIG = SDG(I)
C
C     NORMALIZE ARGUMENTS
C     ********************
      PS  = PSI(I)/PSIO
      IF(PS.LT.1.0D-4) PS = ZERO
C
C     DESIGN MATRIX ROW
C     ******************
      A(1) = 1.DO
      ASIG(1) = SIG
      DO 200 K=2,MCOEFF
      A(K)= PS*A(K-1)
      ASIG(K) = A(K)*SIG
  200 CONTINUE
C
C     NORMAL EQUATION MATRIX AND RIGHT HAND SIDE
C     ******************************************
      DO 300 K=1,MCOEFF
      ATL(K)  = ATL(K) + A(K)*OBS
      DO 300 J=1,MCOEFF
      ATA(J,K) = ATA(J,K) + A(K)*A(J)
      ASIGA(J,K) = ASIGA(J,K) + ASIG(K)*ASIG(J)
  300 CONTINUE
C
  150 CONTINUE
C
C     INVERT NORMAL EQUATION MATRIXENTS
C     *********************************
      CALL SPIN(ATA,MCOEFF,MAXI,DET,IDEXP)
C
C     SEE IF INVERSION REGULAR
C     ************************
      IF(IDEXP.GT.-3) GO TO 1000
C
C     ILL CONDITIONED NORMAL EQUATION MATRIX
C     ******************************************
      MCOEFF = MCOEFF - 1
      GO TO 500
C
C     SOLVE FOR POLYNOMIAL COEFFICIENTS
C     *********************************
 1000 CONTINUE
      DO 600 I=1,MCOEFF
      COEFF(I) = ZERO
      DO 600 J=1,MCOEFF
      COEFF(I) = COEFF(I) + ATA(I,J)*ATL(J)
  600 CONTINUE
```

```
C
C      STANDARD DEVIATIONS OF COEFFICIENTS
C      ***********************************
       DO 700 I=1,MCOEFF
       DO 700 J=1,MCOEFF
       SC(I,J) = ZERO
       DO 700 K=1,MCOEFF
       SC(I,J) = SC(I,J) + ATA(I,K)*ASIGA(K,J)
  700  CONTINUE
       DO 800 I=1,MCOEFF
       DO 800 J=1,MCOEFF
       ASIGA(I,J) = ZERO
       DO 800 K=1,MCOEFF
       ASIGA(I,J) = ASIGA(I,J) + SC(I,K)*ATA(K,J)
  800  CONTINUE
       DO 900 I=1,MCOEFF
       SCOEF(I) = DSQRT(ASIGA(I,I))
  900  CONTINUE
C
       RETURN
       END
C*********************************************************************
       SUBROUTINE STINT (PSIO,COEFF,SCOEF,MAX,MCOEFF,DN,SDN)
C---------------------------------------------------------------------
C      FUNCTION   :   ANALYTICAL STOKES INTEGRATION
C      AUTHOR     :   ALFRED KLEUSBERG
C      HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C---------------------------------------------------------------------
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C PARAMETERS:    NAME     I/O     DESCRIPTION
C ***********
C                PSIO      I      RADIUS OF INTEGRATION (RAD)
C                COEFF     I      NORMALIZED POLYN. COEFF. FOR
C                                  GRAVITY ANOMALIES
C                SCOEF     I      CORRESP. STANDARD DEVIATIONS
C                MAX       I      DIMENSIN OF THESE ARRAYS AS SPECIFIED
C                MCOEFF    I      ACTUAL DIMENSION OF THESE ARRAYS
C                DN        O      GEOID UNDULATION
C                SDN       O      CORRESP. STAND. DEV.
C
C EXTERNALS:     NONE
C ***********
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
       IMPLICIT REAL*8 (A-H,O-Z)
C
       DIMENSION COEFF(MAX), SCOEF(MAX)
       DIMENSION CSTOK(7)
C
C      DEFINE POL.COFF. FOR STOKES FUNCTION*SIN(PSI)
C      (NORMALIZED FOR THE INTERVAL PSI<2.5 DEG
C      ****************************************
             CSTOK(1)=1.0D0
             CSTOK(2)=.390334D0
             CSTOK(3)=-0.961691D0
             CSTOK(4)=2.34308D0
             CSTOK(5)=-3.37295D0
             CSTOK(6)=2.48583D0
             CSTOK(7)=-0.726341D0
       TWODEG = 2.5D0/45.D0*DATAN(1.D0)
       ISEVEN = 7
```

```
      R = 6.370D+6
      GAMMA = 981.D+3
C
C     DENORMALIZE STOKES COEFFICIENTS
C     ###############################
      DO 100 I=2,ISEVEN
      CSTOK(I) = CSTOK(I)/TWODEG**(I-1)
  100 CONTINUE
C
C     DENORMALIZE COEFFICIENT FOR GRAVITY ANOMALIES
C     #############################################
      DO 200 I=2,MCOEFF
      COEFF(I) = COEFF(I)/PSIO**(I-1)
      SCOEF(I) = SCOEF(I)/PSIO**(I-1)
  200 CONTINUE
C
C     INTEGRATE
C     #########
      DN  = 0.DO
      SDN = 0.DO
      DO 300 IS = 1,ISEVEN
      DO 300 IC = 1,MCOEFF
      ISC = IS+IC-1
      A   = CSTOK(IS)/ISC*PSIO**ISC
      DN  = DN  + A*COEFF(IC)
      SDN = SDN + (A*SCOEF(IC))**2
  300 CONTINUE
      DN  = DN  * R/GAMMA
      SDN = DSQRT(SDN) * R/GAMMA
C
      RETURN
      END
C*********************************************************************
      DOUBLE PRECISION FUNCTION SPHER (PHI1,ALA1,PHI2,ALA2)
C---------------------------------------------------------------------
C     FUNCTION  :   COMPUTE SPHERICAL DISTANCE FROM SPHERICAL
C                   COORDINATES OF TWO POINTS.
C     AUTHOR    :   ALFRED KLEUSBERG
C     HISTORY   :   JUNE 06, 1986 - VERSION  1.0
C---------------------------------------------------------------------
C
C  PARAMETERS:      NAME      I/O     DESCRIPTION
C  ###########
C                   PHI1       I      LATITUDE  OF POINT #1 (RAD)
C                   ALA1       I      LONGITUDE OF POINT #1 (RAD)
C                   PHI2       I      LATITUDE  OF POINT #2 (RAD)
C                   ALA2       I      LONGITUDE OF POINT #2 (RAD)
C
C  EXTERNALS:       NONE
C  ##########
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
C
      CP =DSIN(PHI1)*DSIN(PHI2)+DCOS(PHI1)*DCOS(PHI2)*DCOS(ALA2-ALA1)
      IF(CP.GT.1.DO) CP = 1.DO
      SPHER = DARCOS(CP)
C
      RETURN
      END
C*********************************************************************
      SUBROUTINE SELCT (PSIO,PHIO,ALAO,DELTG,SDELG,PHI,ALA,
```

```
     *                    NMAX,NTOTAL,DG,SDG,PSI,NUM,PSIMIN)
C-------------------------------------------------------------------
C      FUNCTION   :   SELECT GRAVITY ANOMALIES WHICH FALL WITHIN
C                     A SPECIFIED SPHERICAL DISTANCE.
C      AUTHOR     :   ALFRED KLEUSBERG
C      HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C-------------------------------------------------------------------
C
C PARAMETERS:      NAME     I/O      DESCRIPTION
C ***********
C                  PSIO       I      SPHERICAL DISTANCE (RAD)
C                  PHIO       I      LATITUDE OF CENTRE POINT  (RAD)
C                  ALAO       I      LONGITUDE OF CENTRE POINT (RAD)
C                  DELTG      I      ARRAY OF GRAVITY ANOMALIES
C                  SDELG      I      ARRAY OF CORRESP. STAND. DEV.
C                  PHI        I      ARRAY OF CORRESP. LATITUDES  (RAD)
C                  ALA        I      ARRAY OF CORRESP. LONGITUDES (RAD)
C                  NMAX       I      DIMENSION OF THESE ARRAYS AS SPEC.
C                  NTOTAL     I      ACTUAL DIMENSION OF THESE ARRAYS
C                  DG         O      ARRAY OF SELECTED GRAVITY ANOMALIES
C                  SDG        O      ARRAY OF CORRESP. STAND. DEV.
C                  PSI        O      ARRAY OF CORRESP. SPHER. DISTANCES (RAD)
C                  NUM        O      NUMBER OF SELECTED GRAVITY ANOMALIES
C
C EXTERNALS:       SPHER
C ***********
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION DELTG(NMAX),  SDELG(NMAX),  PHI(NMAX), ALA(NMAX)
      DIMENSION DG(NMAX),     SDG(NMAX),    PSI(NMAX)
C
      ICOUNT = 0
       PSIMIN=PSIO
C
      DO 100 I=1,NTOTAL
      PSI1 = SPHER (PHIO,ALAO,PHI(I),ALA(I))
      IF(PSI1.GT.PSIO) GO TO 100
      IF(PSI1.LT.PSIMIN) PSIMIN=PSI1
      ICOUNT = ICOUNT+1
      DG(ICOUNT)  = DELTG(I)
      SDG(ICOUNT) = SDELG(I)
      PSI(ICOUNT) = PSI1
  100 CONTINUE
C
      NUM = ICOUNT
C
      RETURN
      END
C*********************************************************************
      SUBROUTINE SPIN(Q,N,MM,DET,IDEXP)
C-------------------------------------------------------------------
C  FUNCTION  :
C     SUBROUTINE SPIN IS A MATRIX INVERSION ROUTINE FOR SYMMETRIC
C     POSITIVE-DEFINITE MATRICES.  THE MATRIX INVERTED IS THE UPPER
C     N BY N PORTION OF THE MATRIX Q WHICH IS DIMENSIONED MM BY MM
C     IN THE CALLING ROUTINE.
C  AUTHOR    :   R.R. STEEVES
C  HISTORY   :   SEPTEMBER 1979 - VERSION  1.0
C-------------------------------------------------------------------
C
```

```
C  INPUT:
C       Q - THE MATRIX DIMENSIONED MM BY MM WHICH CONTAINS THE
C           MATRIX TO BE INVERTED.
C
C       N - THE DIMENSION OF THE ACTUAL PART( UPPER LEFT CORNER)
C           OF Q WHICH IS TO BE INVERTED. ( N MAY BE EQUAL BUT MUST
C           NOT BE LARGER THAN MM) .
C       MM- DIMENSIONED SIZE OF Q IN THE CALLING ROUTINE.
C
C
C  OUTPUT:
C
C       Q - THE UPPER LEFT N BY N PORTION CONTAINS THE INVERSE OF
C           THE INPUT UPPER LEFT N BY N PORTION.
C
C       DET - THE NON-EXPONENT PORTION OF THE DETERMINANT OF THE
C             INPUT N BY N (UPPER LEFT PORTION OF Q) MATRIX. SEE
C             IDEXP  BELOW.
C
C       IDEXP - THE EXPONENT (OF 10) PART OF THE DETERMINANT DESCRIBED
C               UNDER DET ABOVE. THUS THE DETERMINANT IS RETURNED IN
C               TWO PARTS CORRESPONDING TO
C                     DETERMINANT = DET * 10 ** IDEXP .
C               THIS IS DONE TO AVOID UNDER OR OVERFLOW IN THE
C               COMPUTATION OF THE DETERMINANT.  TO PRINT THE DETERM-
C               INANT THE USER SHOULD PRINT BOTH NUMBERS AS FOLLOWS;
C               (FOR EXAMPLE)
C                  PRINT 10,DET,IDEXP
C               10 FORMAT(' ','DETERMINANT= ',F7.4,'D',I4)
C
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      REAL*8 Q(MM,MM),DET,SUM,DSQRT,DABS,RPART,APART
      DET=0.D0
      DO 4 J=1,N
      DO 4 I=1,J
      IF(I.EQ.1) GO TO 2                                                  20
      M=I-1
      SUM=0.0D0
      DO 1 K=1,M
1     SUM=SUM+Q(K,I)*Q(K,J)
      Q(I,J)=Q(I,J)-SUM
2     IF(I.EQ.J) GO TO 3
      Q(I,J)=Q(I,J)/Q(I,I)
      GO TO 4
3     CONTINUE
      IF(Q(I,I).LE.0.D0) Q(I,I)=1.D-12
      DET=DET+DLOG10(Q(I,I))
      Q(I,I)=DSQRT(Q(I,I))
4     CONTINUE
      IDEXP=DET
      RPART=DET-IDEXP
      APART=DABS(RPART)
      IF(APART.LT.1.D-20) DET=1.D0
      IF(APART.LT.1.D-20) GO TO 10
      DET=10**RPART
   10 CONTINUE
      DO 7 J=1,N
      DO 7 I=1,J
      IF(I.LT.J) GO TO 5
      Q(J,J)=1.0D0/Q(J,J)
      GO TO 7
```

```
5       SUM=0.0D0
        M=J-1
        DO 6 K=I,M
6       SUM=SUM-Q(I,K)*Q(K,J)
        Q(I,J)=SUM/Q(J,J)
7       CONTINUE
        DO 9 J=1,N
        DO 9 I=1,J
        SUM=0.0D0
        DO 8 K=J,N
8       SUM=SUM+Q(I,K)*Q(J,K)
        Q(I,J)=SUM
        IF(I.EQ.J) GO TO 9
        Q(J,I)=SUM
9       CONTINUE
        RETURN
        END
//GO.FT01F001 DD DSN=A.M66774.WENZEL.D180D90.TEST.FEB,
//             DISP=SHR,LABEL=(,,,IN)
//GO.SYSIN DD *
GRID
1.0
52.0 260.0 67.0 290.0
57.0 270.0 62.0 280.0 10.0 10.0
2
   56.0000   300.0000
   56.0000   300.1667
//
```

2.7 Programs for the Computation of the "Truncation" Kernel

```
FTN77,I,,,E
$FILES(0,,,)
      PROGRAM COEFF ()
C
C     Compute numerical values for "truncation" kernel in
C     steps of .05 deg for the interval  0 < psi < 15 deg
C
C     Externals: TCPAL, LGNDR, DOPRO
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(    801)
      DIMENSION Q(    801)
      DIMENSION NAMD(8)
      DATA LP      /    800    /
      DATA NUL     /0 /
      DATA IONF    /1 /
      DATA LUI     /1 /           ! logical unit of terminal
      DATA LUD     /11/           ! logical unit of output file
      RHO = 45.D0/DATAN(1.D0)
C

      WRITE(LUI,1000)
 1000 FORMAT(/," ENTER DEGREE OF SPHEROIDAL KERNEL ",/,
     *          " IF L IS THE DEGREE OF THE SPHEROIDAL FIELD,",/,
     *          " THEN ENTER (L+1)",/)
      READ(LUI,*) L0
C
      WRITE(LUI,1010)
 1010 FORMAT(/," ENTER STOKES TRUNCATION RADIUS PSI0 [deg]",/)
      READ(LUI,*) PSI0
C
      WRITE(LUI,1020)
 1020 FORMAT(/," ENTER NAME OF OUTPUT FILE (8A2)",/)
      READ(LUI,1) (NAMD(I),I=1,8)
    1 FORMAT(8A2)
C
      OPEN (LUD, FILE=NAMD,STATUS='UN')
      ARGQ = COS(PSI0/RHO)
      IF(L0.LT.2) L0 = 2
C
      CALL TCPAL (ARGQ, P, LP)
C
C     LOOP FOR COEFFICIENTS OF SERIES EXPANSION
C     """""""""""""""""""""""""""""""""""""""""
      DO 320 IDEG = 1,L0-1
```

```
      Q(   EG) = 0.D0
  320 CONTINUE
C
      DL0 = DBLE(L0)
      Q(L0)    = (DL0-1.D0)*DL0/2.D0*P(L0+1)
      L01 = L0+1
      Q(L01) = DL0*(DL0+1.D0)/2.D0*P(L01+1)
     #          -(DL0-1.D0)/2.D0*(2.D0*DL0+1.D0)*ARGQ*P(L01)
C
      DO 330 IDEG = L0+1,LP-1
      I1 = IDEG+1
      DN = DBLE(IDEG)
      Q(I1) = (DN*((DN-2.D0)*P(I1-1)+(DN+1.D0)*P(I1+1)) -
     $         (DN-1.D0)*(2.D0*DN+1.D0)*ARGQ*P(I1))/2.D0
  330 CONTINUE
C
C     LOOP FOR DIFFERENT ARGUMENTS
C     ''''''''''''''''''''''''''''
      DO 340 IARG = 1,150
      ARG1 = DBLE(IARG)/10.D0 - .0500D0
      ARG  = COS(ARG1/RHO)
      CALL LGNDR (ARG, P, LP)
C
C     SUM SERIES
C     ''''''''''
      CALL DOPRD (SUM, Q, IONE, P, IONE, LP)
      I=LP
      WRITE(LUD,3400) ARG1, SUM
      WRITE(LUT,3400) ARG1, SUM
 3400 FORMAT(1X,F10.4,1X,G20.10)
  340 CONTINUE
C
 2000 STOP
      END
```

```
FTN77,I,Y,E
$FMA /ARRY1/
      PROGRAM FPHI0 ()
C
C     Compute optimal truncation radius for the "truncation"
C     kernel for given degree of the spheroidal field and given
C     Stokes integration radius
C
C     Externals:   TCPAL, LGNDR, DOPRO
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(   801)
      DIMENSION Q(   801)
      COMMON /ARRY1 / P, Q
      DATA LP      /    800    /
      DATA LUI     /1/
      DATA EPS     /1.D-12/
      DATA SUMOLD /-10D0/
      DATA ISIOLD /1     /
      DATA IONE   /1     /
      RHO = 45.D0/DATAN(1.D0)
C
      WRITE(LUI,1000)
 1000 FORMAT(/," ENTER DEGREE OF SPHEROIDAL KERNEL ",/,
     *         ," IF L IS THE DEGREE OF THE SPHEROIDAL FIELD,",/,
     *         ," THEN ENTER (L+1)",/)
      READ(LUI,*) L0
      IF(L0.LT.2) L0 = 2
C
      WRITE(LUI,1010)
 1010 FORMAT(/,"ENTER STOKES TRUNCATION RADIUS PSI0 (DEG)",/)
      READ(LUI,*) PSI0
      ARGQ = COS(PSI0/RHO)
C
C     STARTING VALUE FOR PHI0
C
      PHI0 = PSI0
      DELT = PSI0/3.D0
C
      CALL TCPAL (ARGQ, P, LP)
C
C     LOOP FOR COEFFICIENTS OF SERIES EXPANSION
C     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      DO 320 IDEG = 1,L0-1
```

```
      Q(IDEG) = 0.D0
  320 CONTINUE
      DL0 = DBLE(L0)
      Q(L0)    =  (DL0-1.D0)*DL0/2.D0*P(L0+1)
      L01 = L0+1
      Q(L01) = DL0*(DL0+1.D0)/2.D0*P(L01+1)
     #          -(DL0-1.D0)/2.D0*(2.D0*DL0+1.D0)*ARGQ*P(L01)
      DO 330 IDEG = L0+1,LP-1
      I1 = IDEG+1
      DN = DBLE(IDEG)
      Q(I1) = (DN*((DN-2.D0)*P(I1-1)+(DN+1.D0)*P(I1+1)) -
     $          (DN-1.D0)*(2.D0*DN+1.D0)*ARGQ*P(I1))/2.D0
  330 CONTINUE
C
C     LOOP FOR DIFFFRENT ARGUMENTS
C     """""""""""""""""""""""""""""
 4000 CONTINUE
      ARG  = COS(PHI0/RHO)
      CALL LGNDR (ARG, P, LP)
C
C     SUM SERIES
C     """""""""""
      CALL DOPRO (SUM, Q, IONE, P, IONE, LP)
C
C     DECIDE HOW TO ITERATE
C
      IF(SUM.LT.0.D0)      GO TO 4500
      DIF = SUM - SUMOLD
      IF(ABS(DIF).LT.EPS) GO TO 5000
      ISINEW = NINT(SIGN(1.D0,DIF))
      IF(ISINEW.EQ.1) GO TO 4500
      ISIOLD = ISINEW
      DELT = -DELT/2.D0
 4500 CONTINUE
      PHI0 = PHI0 + DELT
      SUMOLD = SUM
      GO TO 4000
 5000 CONTINUE
      WRITE(LUI,3400) PHI0
 3400 FORMAT(//," OPTIMAL TRUNCATION RADIUS [DEG] :",F12.7)
      STOP
      END
```

```
FTN77,I,Y,E
$EMA /ARRY2/
      SUBROUTINE TCPAI (ARG, Q, MAX)
C
C     Compute Truncation Coefficients using M. Paul's algorithm
C     (see Bulletin Geodesique 1973, pp. 413-425)
C
C     Externals:  LGNDR
C
C     !!!!!!!!!!!   Maximum Degree is   800 !!!!!!!!
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Q (1)
      DIMENSION U(   803)              ! MAX(MAX)+3
      DIMENSION V(   803)              ! MAX(MAX)+3
      DIMENSION P(   802)              ! MAX(MAX)+2
      DIMENSION R(   801)              ! MAX(MAX)+1
      COMMON / ARRY2/ U,V,P,R
C
      IF(MAX.GT.   800) GO TO 2000
      ARG2 = ARG*ARG                        ! COSINE SQUARE
C
C     Legendre polynomials
C     ''''''''''''''''''''
      MAXI = MAX+1
      CALL LGNDR (ARG, P, MAXI)
C
C     Compute U and V Arrays
C     ''''''''''''''''''''''
      U(1) = 0.D0
      V(1) = 0.D0
      U(2) = 0.D0
      DSQ  = 0.D0
      ARG1 = 0.D0
      IF(ARG.GT..9999999999) GO TO 100
      ARG1 = DSQRT((1.D0-ARG)/2.D0)
      DSQ  = DSQRT(2.D0-2.D0*ARG)
      U(1) = LOG(1.D0 + 2.D0/DSQ)
      V(1) = U(1)
      U(2) = LOG(2.D0/(1.D0-ARG+DSQ))
  100 CONTINUE
      V(2) = ARG * V(1) + DSQ - 1.D0
      U(3) = ARG * U(2) - DSQ - ARG
      V(3) = (3.D0*ARG*V(2) - V(1) + DSQ)/2.D0
```

```
      DO 200 N=3,MAX+2
      N1 = N+1
      U(N1) = ((2*N-3)*ARG*U(N)-(N-2)*U(N-1)-DSQ
     $         +(P(N1-3)-P(N1-1))/(2*N-3))/(N-1)
      V(N1) = ((2*N-1)*ARG*V(N)-(N-1)*V(N-1)+DSQ)/N
  200 CONTINUE
C
C     Compute the R Array
C     ~~~~~~~~~~~~~~~~~~~
      R(1) = ARG + 1.D0
      DO 300 N=1,MAX
      N1 = N+1
      PN1 = P(N1)
      PNN = P(N)
      R(N1) = ((2*N-1)*R(N)+ARG*(PN1*PN1+PNN*PNN)
     $         -2.D0*PN1*PNN)/(2*N+1)
  300 CONTINUE
C
C     Truncation Coefficients
C     ~~~~~~~~~~~~~~~~~~~~~~~
      Q(1) = 0.D0
      Q(2) = 0.D0
      IF(ARG1.LT.1.D-10) GO TO 350
      Q(1) = -4*ARG1+5*ARG1**2+6*ARG1**3-7*ARG1**4
     $       +6*(ARG1**2-ARG1**4)*LOG(ARG1+ARG1**2)
      THR  = DBLE(3)
      Q(2) = -2*ARG1+4*ARG1**2+28*ARG1**3/THR-14*ARG1**4-8*ARG1**5
     $       +32*ARG1**6/THR+(6*ARG1**2-12*ARG1**4+8*ARG1**6)
     $       *LOG(ARG1+ARG1**2) -2*LOG(1.D0+ARG1)
  350 CONTINUE
C
      U1 = U(2)
      U2 = U(3)+1.D0
      U3 = U(4)+.5D0*ARG
C
      DO 400 N=2,MAX
      N1 = N+1
      AN = DBLE(N)
      UA1 = U(N1) +1.D0/(AN-1.D0)
      UA2 = U(N+2)+1.D0/AN+ARG/(AN-1.D0)
      UA3 = U(N+3)+1.D0/(AN+1.D0)+ARG/AN+(3.D0*ARG2-1.D0)/2.D0/(AN-1.D0)
      VA1 = V(N)  -1.D0/AN-ARG/(AN+1.D0)-(3.D0*ARG2-1.D0)/2.D0/(AN+2.D0)
      VA2 = V(N1) -1.D0/(AN+1.D0)-ARG/(AN+2.D0)
      VA3 = V(N+2)-1.D0/(AN+2.D0)
C
```

```
      F1   = (2.D0*(2.D0*AN+1.D0)/AN/(AN+1.D0)*(U1-U3)
     $        -(AN+2.D0)*(UA1-UA3)-(AN-1.D0)*(VA3-VA1))*P(N1)
      F2   = (3.D0*U2-(AN+2.D0)*UA2+(AN-1.D0)*VA2)*(P(N+2)-P(N))
      F1   = AN*(AN+1.D0)/(2.D0*AN+1.D0)/(AN-1.D0)/(AN+2.D0)*(F1+F2)
      F2   = -(2.D0*AN*AN+2.D0*AN+1.D0)/(AN-1.D0)/((2.D0*AN+1.D0)**2)
     $       *P(N1)*(P(N+2)-P(N))
      Q(N1) = F1 + F2 + (2.D0*AN+1.D0)/(AN-1.D0)*R(N1)
  400 CONTINUE
      GO TO 1000
C
 2000 CONTINUE
C
C     Error message to terminal (logical unit 1)
C
      WRITE(1,2001) MAX
 2001 FORMAT(1X,"/TCPAL:",/,1X,
     *       "DIMENSIONING OF ARRAYS INSUFFICIENT. REQUESTED: ",I5)
 1000 CONTINUE
      RETURN
      END
```

```
FTN77,I,Y, E
      SUBROUTINE LGNDR (ARG, P, MAX)
C
C     Compute Legendre Polynomials using
C     recurrence relations
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION P(1)
C
C     Set Initial Values
C     ''''''''''''''''''
      P(1) = 1.D0
      P(2) = ARG
C
C     Compute Polynomials up to Degree MAX
C     ''''''''''''''''''''''''''''''''''''
      DO 100 L=2,MAX
      R = DBLE(L)
      P(L+1)  = (2*R-1)/R*P(L)*ARG - (R-1)/R*P(L-1)
  100 CONTINUE
      RETURN
      END
```

```
FTN7X,I,Y,E
      SUBROUTINE DOPRO(SCALAR,V1,INC1,V2,INC2,N)
C
C     Scalar product of vectors V1 and V2
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION V1(1), V2(1)
C
      SCALAR = 0.D0
      DO 100 I=1,N
      I1 = 1+(I-1)*INC1
      I2 = 1+(I-1)*INC2
      SCALAR = SCALAR + V1(I1)*V2(I2)
  100 CONTINUE
      RETURN
      END
```

Screen copy of sample run of program COEFF

(Input part only)

```
CI.10> COEFF

 ENTER DEGREE OF SPHEROIDAL KERNEL
 IF L IS THE DEGREE OF THE SPHEROIDAL FIELD,
 THEN ENTER (L+1)

51

 ENTER STOKES TRUNCATION RADIUS PSI0 [deg]

0.5

 ENTER NAME OF OUTPUT FILE (8A2)

C005::AK
```

Program COEFF output file generated in the sample run

```
 .0500  -.6352274155
 .1500  -.4097598817
 .2500  -.5937662741
 .3500  -.6137125445
 .4500  -.4524700464
 .5500  -.6441680044
 .6500  -.6917835843
 .7500  -.5284451491
 .8500  -.4251896481
 .9500  -.3313381102
1.0500  -.2443315798
1.1500  -.1614584497
1.2500  -.7575544421E-01
1.3500   .4778034195E-02
1.4500   .8404162629E-01
1.5500   .1587637783
1.6500   .2329758779
1.7500   .3047420898
1.8500   .3771662187
1.9500   .4341187881
2.0500   .4934514996
2.1500   .5492422862
2.2500   .5995406127
2.3500   .6435180810
2.4500   .6919575938
2.5500   .7152043621
2.6500   .7421347971
2.7500   .7741002284
2.8500   .7738343357
2.9500   .7795058960
3.0500   .7783813495
3.1500   .7695641770
3.2500   .7528895127
3.3500   .7229113778
3.4500   .6986867250
3.5500   .6612888052
3.6500   .6166735104
3.7500   .5654561607
3.8500   .5085454175
3.9500   .4462541153
4.0500   .3785255918
4.1500   .3058184113
4.2500   .2291804363
4.3500   .1494506225
4.4500   .6992268735E-01
```

4.5500	-.18033438988E-01
4.6500	-.1044654731
4.7500	-.1912571330
4.8500	-.2777496342
4.9500	-.3635641567
5.0500	-.4479736399
5.1500	-.5298717084
5.2500	-.6083620558
5.3500	-.6829937026
5.4500	-.7532956455
5.5500	-.8184113231
5.6500	-.8774259193
5.7500	-.9298301907
5.8500	-.9753753156
5.9500	-1.013594754
6.0500	-1.043790369
6.1500	-1.065488597
6.2500	-1.078613089
6.3500	-1.083115426
6.4500	-1.078692487
6.5500	-1.065047236
6.6500	-1.042248551
6.7500	-1.010405796
6.8500	-.9702847334
6.9500	-.9212982295
7.0500	-.8638689826
7.1500	-.7985584843
7.2500	-.7259604469
7.3500	-.6464699281
7.4500	-.5604939099
7.5500	-.4687374248
7.6500	-.3720911270
7.7500	-.2713108308
7.8500	-.1670083707
7.9500	-.59946743548E-01
8.0500	.48861431588E-01
8.1500	.1584154920
8.2500	.2679209662
8.3500	.3766156779
8.4500	.4835356234
8.5500	.5874155891
8.6500	.6879613338
8.7500	.7838572795
8.8500	.8745188462
8.9500	.9590131212

9.0500	1.036482913
9.1500	1.106310893
9.2500	1.167949728
9.3500	1.220822789
9.4500	1.264210980
9.5500	1.297683930
9.6500	1.321002555
9.7500	1.333923120
9.8500	1.336127343
9.9500	1.327412821
10.0500	1.307829494
10.1500	1.277544221
10.2500	1.236663857
10.3500	1.185306730
10.4500	1.123797852
10.5500	1.052644003
10.6500	.9724392194
10.7500	.8835987989
10.8500	.7867201746
10.9500	.6825952207
11.0500	.5721043771
11.1500	.4560556798
11.2500	.3352425102
11.3500	.2104093140
11.4500	.8324056586E-01
11.5500	-.4581631002E-01
11.6500	-.1756136369
11.7500	-.3051452753
11.8500	-.4333522720
11.9500	-.5590389096
12.0500	-.6812184774
12.1500	-.7989640587
12.2500	-.9112636029
12.3500	-1.017132214
12.4500	-1.115571286
12.5500	-1.205824105
12.6500	-1.287152208
12.7500	-1.358770802
12.8500	-1.419951800
12.9500	-1.470152115
13.0500	-1.508966476
13.1500	-1.535994463
13.2500	-1.550848511
13.3500	-1.553292873
13.4500	-1.543268461

13.5500	-1.520874195
13.6500	-1.486095705
13.7500	-1.439072752
13.8500	-1.380128547
13.9500	-1.309730205
14.0500	-1.228363963
14.1500	-1.136536020
14.2500	-1.034891437
14.3500	-.9242473290
14.4500	-.8054798776
14.5500	-.6794439544
14.6500	-.5470459799
14.7500	-.4093364062
14.8500	-.2674579027
14.9500	-.1225239673

Screen copy of sample run of program FPHI0

```
CI.10> FPHI0

 ENTER DEGREE OF SPHEROIDAL KERNEL
 IF L IS THE DEGREE OF THE SPHEROIDAL FIELD,
 THEN ENTER (L+1)

21

ENTER STOKES TRUNCATION RADIUS PSI0 [DEG]

3.5


 OPTIMAL TRUNCATION RADIUS [DEG] :  12.2107468
CI.10>
```

ADDENDUM

FUNCTION SEVALD
(FOR USE WITH PROGRAM ALTINT)

```
      DOUBLE PRECISION FUNCTION SEVALD(N,U,X,Y,B,C,D)
      INTEGER N
      REAL*8 U,X(N),Y(N),B(N),C(N),D(N)
      REAL*8 DX
      INTEGER I,J,K
      DATA I/1/
      IF (I.GE.N) I=1
      IF (U.LT.X(I)) GO TO 10
      IF (U.LE.X(I+1)) GO TO 30
   10 I=1
      J=N+1
   20 K=(I+J)/2
      IF (U.LT.X(K)) J=K
      IF (U.GE.X(K)) I=K
      IF (J.GT.I+1) GO TO 20
   30 DX=U-X(I)
      SEVALD=Y(I)+DX*(B(I)+DX*(C(I)+DX*D(I)))
      RETURN
      END
```

ERRATA

TWO CORRECTIONS TO
PROGRAM SGSINT
(ENTIRE PROGRAM INCLUDED
WITH CORRECTIONS SHOWN
ON PAGES 100 AND 101)

```
//SGSINT   JOB NOTIFY=6461
/*SERVICE -4
/*JOBPARM T=15,L=99,R=5120,PRINT=ALL
// EXEC FORTVCLG,RC=5120K,RL=2048K,RG=5120K
//FORT.SYSIN DD *
C-------------------------------------------------------------------------
C*************************************************************************
C                                                                        *
C  PROGRAM NAME :    SGSINTT                                             *
C  FUNCTION     :  PARTIAL STOKES INTEGRATION OF SPARSE POINT            *
C                  GRAVITY ANOMALIES .                                   *
C COMPILER      :  VS FORTRAN VERSION 2                                  *
C AUTHOR        :  JOHN MANTHA                                           *
C HISTORY       :  JULY   26, 1986 - VERSION 1.0                         *
C REFERENCE     :  U.N.B.  TECHNICAL  REPORT  #                          *V
C*************************************************************************
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C  PARAMETERS:   NAME   I/O   DESCRIPTION
C
C                NP       I    NUMBER OF COMPUTATION POINTS
C                PSIO     I    RADIUS OF SPHERICAL CAP
C                ALATSW   I    LAT OF SOUTHWEST CORNER OF BLOCK A
C                ALONSW   I    LONG OF SOUTHWEST CORNER OF BLOCK A
C                ALATNE   I    LAT OF NORTHEAST CORNER OF BLOCK A
C                ALONNE   I    LONG OF NORTHEAST CORNER OF BLOCK A
C                PHISW    I }  THE SOUTH WEST AND NORTH EAST CORNER
C                ALASW    I }  COORDINATES OF THE COMPUTATIONAL
C                PHINW    I }  AREA.
C                ALASE    I }
C                XNTPHI   I    THE GRID INTERVAL IN LAT. DIRECTION
C                              IN MINUTES.
C                XNTALA   I    THE GRID INTERVAL IN LON. DIRECTION
C                              IN MINUTES.
C                PHI      I    CURRENT ARRAYS OF LAT. OF CENTER POINT
C                ALA      I    CURRENT ARRAYS OF LONG. OF CENTER POINT
C                PHIO     D    LAT OF CENTER POINT
C                PHIO1    D    LAT OF NEXT CENTER POINT
C                ALAO     D    LONG OF CENTER POINT
C                DLATSB   D    LAT OF SOUTHERLY BOUND. OF BLOCK B
C                DLONWB   D    LONG OF WESTERLY BOUND. OF BLOCK B
C                DLATNB   D    LAT OF NORTHERLY BOUND. OF BLOCK B
C                DLONEB   D    LONG OF WESTERLY BOUND. OF BLOCK B
C                SLATS    D    LAT OF SOUTHERLY BOUND. OF STRIP
C                SLATN    D    LAT OF NORTHERLY BOUND. OF STRIP
C                SLONE    D    LONG OF EASTERLY BOUND. OF COMP. CAP
C                SLONW    D    LONG OF WESTERLY BOUND. OF COMP. CAP
C                              VALUES STORED IN STRIP
C                SIDG     D    ARRAY OF GRAVITY ANOMALIES
C                SSDG     D    CORRES. STAND. DEV.
C                SLAT     D    CORRES. LATITUDES
C                SLON     D    CORRES. LONGITUDES
C                              VALUES USED IN SUBROUTINE INTGR
C                DELTG    I    ARRAY OF GRAVITY ANOMALIES  (MGAL)
C                SDELG    I    CORRESP. STAND. DEV.        (MGAL)
C                PHIS     I    CORRESP. LATITUDES          (RAD)
C                ALAS     I    CORRESP. LONGITUDES         (RAD)
C                DG       O    ARRAY OF GRAV.ANOM. WITHIN PSIO (MGAL)
C                SDG      O    ARRAY OF CORRESP. STAND. DEV. (MGAL)
C                PSI      O    ARRAY OF CORRESP. SPHERICAL DISTANCES (RAD)
C                NMAX     I    DIMENSION OF THESE ARRAYS AS SPECIFIED
C                              IN CALLING PROGRAM
```

look for changes on pages: 3, 4.

```
C                NTOTAL  I     ACTUAL DIMENSION OF DELTG,SDELG,PHI,ALA
C                DN      O     COMPUTED GEOID UNDULATION (M)
C                SDN     O     CORRESPONDING STAND. DEV. (M)
C
C NOTE:  ALL VALUES ARE ENTERED INTO THE PROGRAM AS DECIMAL DEGREES
C
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      CHARACTER * 5 MODE
      DIMENSION PHI(1891),ALA(1891)
      DIMENSION ALAT(50000),ALON(50000),ADG(50000),ASDG(50000)
      DIMENSION SLAT(50000),SLON(50000),SIDG(50000),SSDG(50000)
      DIMENSION PHIS(10000),ALAS(10000),DELTG(10000),SDELTG(10000)
      DIMENSION DG(10000),SDG(10000),PSI(10000)
         NMAX=10000
C
C NOTE: THE DIMENSION VALUES OF ARRAYS FOR PHI AND ALA CORRESPOND TO
C       THE NUMBER OF COMPUTATION POINTS(NP).
C
C-----------------------------------------------------------------------
C     INITIALIZATIONS OF CONSTANTS
C
      PI=DARCOS(-1.DO)
      DR=PI/180.DO
      NPB=2000
C-----------------------------------------------------------------------
C
C #1) READ IN THE MODE OF OPERATION.POINT MODE OR GRID MODE.
C
       READ(5,1000) MODE
 1000  FORMAT (A5)
C
C #2) READ IN CAPSIZE
C
      READ (5,*)PSIO
C
C #3) READ IN SOUTWEST AND NORTHEAST CORNER OF BLOCK A
C
      READ(5,*)ALATSW,ALONSW,ALATNE,ALONNE
C
C #4) READ IN THE SOUTHWEST AND NORTHEAST CORNER OF GRIDDED BLOCK
C     ALONG WITH THE DESIRED INTERVAL.
C
       READ(5,*)PHISW,ALASW,PHINW,ALASE,XNTPHI,XNTALA
C
C #5) READ IN THE NUMBER OF POINTS
C
      READ(5,*) NP
C
C #5) READ IN THE COORDINATES OF THE CENTER POINTS
C
      DO 11 I=1,NP
        READ(5,*)PHI(I),ALA(I)
 11    CONTINUE
C
         IF(MODE.EQ.'GRID')THEN
C
C-----------------------------------------------------------------------
C                        GRID MODE
```

```
C                      **********
C
C     THE PROGRAM WILL USE THE GRID MODE AND CALCUALTE THE COORDINATES
C     OF THE GRID INTERSECTION LINES THROUGH THE CALL TO THE SUBROUTINE
C     AREA.
C
C   THE AREA:
C           SUBROUTINE GENERATES THE COMPUTATION POINTS FROM MESH (AREA)
C           GIVEN THE BOUNDARIES OF THE GRID AND THE INTERVAL ALONG EACH
C           DIRECTION.
C
C  INPUT:
C  *******PHISW:LATITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C         ALASW:LONGITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C         PHINW:LATITUDE OF THE NORTH WEST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C         ALASE:LONGITUDE OF THE SOUTH EAST CORNER OF GRID TO BE COMPUTED
C                 IN DEGREES.
C        XNTPHI:INTERVAL ALONG THE LATITUDINAL DIRECTION IN MINUTES
C        XNTALA:INTERVAL ALONG THE LONGITUDINAL DIRECTION IN MINUTES
C
C    OUTPUT:
C    ******** PHI:ARRAY OF LATITUDES OF COMPUTATION POINTS IN DEGREES.
C             ALA:ARRAY OF LONGITUDES OF COMPUTATION POINTS IN DEGREES.
C               K:NUMBER OF EVALUATION POINTS IN THE AREA.
C              IP:NUMBER OF COMPUTAION POINTS IN THE LATITUDINAL
C                 DIRECTION.
C              IL:NUMBER OF COMPUTATION POINTS IN THE LONGITUDINAL
C                 DIRECTION.
C
C           THE MAXIMUM NUMBER OF POINTS THAT CAN BE STORED
C           IN THE ARRAYS PHI AND ALA ARE 10,000.
C-----------------------------------------------------------------------
C
      CALL AREA(PHISW,ALASW,PHINW,ALASE,XNTPHI,XNTALA,PHI,ALA,K)
       NP=K
       ENDIF
C
C #6) READ IN THE GRAVITY VALUES FOR BLOCK (A) FROM DISK
C
            I=0
 333     READ(1,111,END=3333)DLAT,DLON,DDG,DSDG,DH,DSH
            IF(DLAT.LT.ALATSW)GO TO 333
            IF(DLON.LT.ALONSW.OR.DLON.GT.ALONNE)GO TO 333
            IF(DLAT.GT.ALATNE)GO TO 3333
              ALAT(I)=DLAT
              ALON(I)=DLON
              ADG(I)=DDG
             ASDG(I)=DSDG
                 I=I+1
              ICOUNT=I
              GO TO 333
3333  CONTINUE
111     FORMAT(2F10.2,4F10.4),
C
C FORM BOUNDARIES OF INNERBLOCK B,THE DISTANCE BEING PSIO FROM
C       BOUNDARIES OF BLOCK A.
C
C DETEMINE THE CHANGE IN LATITUDE DUE TO CONVERGENCE.
```

this is not the right format. This should be the format of the gravity anomaly reduced files in Labrador Sea and Hudson Bay areas, namely:

(2F10.4, 2F8.2, 2F7.1)

```
C
                  RPSIO=PSIO+DR
                  RPSION=(RPSIO/DCOS(ALATNE+DR))+0.2D-3
                  RPSIOS=RPSIOS/DR
                  RPSION=RPSION/DR
C
C FORMING OF BOUNDARIES OF BLOCK B
C
                BLONWB=ALONSW+RPSION
                BLONEB=ALONNE-RPSION
                BLATSB=ALATSW+PSIO
                BLATNB=ALATNE-PSIO
C
C  DO LOOP BEGINS TO TEST LOCATION OF CENTER POINTS.
C
          DO 13 JJ=1,NP
                PHIO=PHI(JJ)
                ALAO=ALA(JJ)
C
C  TEST TO SEE IF CENTER POINTS LIE OUTSIDE BLOCK B.
C
         IF(PHIO.LT.BLATSB.OR.PHIO.GT.BLATNB)GO TO 555
         IF(ALAO.LT.BLONWB.OR.ALAO.GT.BLONEB)GO TO 555
         IF(JJ.EQ.1)GO TO 777
         PHIO1=PHI(JJ-1)
         CHECK=DABS(PHIO-PHOI1)
         IF(CHECK.LT.0.D-6)GO TO 155
 777     CONTINUE
C
C DETERMINE THE BOUNDARIES OF THE STRIP TO BE LOADED FROM BLOCK B.
C
                   SLATS=PHIO-(PSIO+0.02)
                   SLATN=PHIO+(PSIO+0.02)
C
          DO 99 BER=1,NPB
                  SLAT(BER)=0
                  SLON(BER)=0
                  SIDG(BER)=0
                  SSDG(BER)=0
99        CONTINUE
C
                       NPB=0
         DO 15 JJK=1,ICOUNT
             IF(ALAT(JJK).LT.SLATS)GO TO 15
             IF(ALAT(JJK).GT.SLATN)GO TO 155
C
                       NPB=NPB+1
C
                  SLAT(NPB)=ALAT(JJK)
                  SLON(NPB)=ALON(JJK)
                  SIDG(NPB)=ADG(JJK)
                  SSDG(NPB)=ASDG(JJK)
C
 15     CONTINUE
 155    CONTINUE
C
C DETERMINE THE BOUNDARIES IN THE LONGITUDINAL DIRECTION TO
C FORM THE CAP BOUNDARIES.
C
                  BPSIO=RPSIO/DCOS(PHIO+DR)
                  BPSIO=(BPSIO+0.2D-3)/DR
```

this should be 1.D-6 not ~~0.D-6~~

This "error" does not have an effect to the computed results however.

```
                    SLONE=ALAO+BPSIO
                    SLONW=ALAO-BPSIO
C
                      NTOTAL=0
            DO 16 KJ=1,NPB
              IF(SLON(KJ).GT.SLONE.OR.SLON(KJ).LT.SLONW)GO TO 16
                      NTOTAL=NTOTAL+1
               PHIS(NTOTAL)=SLAT(KJ)*DR
               ALAS(NTOTAL)=SLON(KJ)*DR
              DELTG(NTOTAL)=SIDG(KJ)
             SDELTG(NTOTAL)=SSDG(KJ)
 16        CONTINUE
C
C CHANGE VALUES TO RADIANS IN ORDER TO INPUT INTO
C SUBROUTINE INTGR.
C
               PHIO=PHIO*DR
               ALAO=ALAO*DR
C
              CALL INTGR (RPSIO,PHIO,ALAO,DELTG,SDELTG,PHIS,ALAS,
     *                     DG,SDG,PSI,NMAX,NTOTAL,NUM,DN,SDN)
C
                     PHIO=PHIO/DR
                     ALAO=ALAO/DR
                  SDN1=0.DO
C
         WRITE(6,1002)PHIO,ALAO,DN,SDN1
         WRITE(6,1002)PHIO,ALAO,DN,SDN1
 1002     FORMAT(' ',F10.4,1X,F10.4,1X,F10.3,1X,F8.3)
C
           GO TO 13
 555       CONTINUE
           WRITE(6,*)'POINT LIES OUTSIDE OF COMPUTATION AREA'
 13        CONTINUE
             STOP
             END
C*********************************************************************
       SUBROUTINE AREA(PHISW,ALASW,PHINW,ALASE,XNTPHI,XNTALA,PHI,ALA,K)
       IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION PHI(1891),ALA(1891)
C---------------------------------------------------------------------
C      FUNCTION :     GENERATES THE COMPUTATION POINTS
C                     GIVEN THE BOUNDARIES OF THE GRID AND
C                     AND THE INTERVAL ALONG EACH DIRECTION.
C      AUTHOR     :   HASSAN FASHIR
C      HISTORY    :   FEBRUARY 16, 1985 - VERSION  1.0
C---------------------------------------------------------------------
C INPUT:
C ******
C       PHISW   : LATITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C       ALASW   : LONGITUDE OF THE SOUTH WEST CORNER OF GRID TO BE COMPUTED
C       PHINW   : LATITUDE OF THE NORTH WEST CORNER OF GRID TO BE COMPUTED
C       ALASE   : LONGITUDE OF THE SOUTH EAST CORNER OF GRID TO BE COMPUTED
C       XNTPHI  : INTERVAL ALONG THE LATITUDINAL DIRECTION IN MINUTES
C       XNTALA  : INTERVAL ALONG THE LONGITUDINAL DIRECTION IN MINUTES
C
C OUTPUT:
C ********
C       PHI     : ARRAY OF LATITUDES OF COMPUTATION POINTS.
C       ALA     : ARRAY OF LONGITUDES OF COMPUTATION POINTS.
C       K       : NUMBER OF EVALUATION POINTS IN THE AREA
```

```
C       IP      : NUMBER OF COMPUTAION POINTS IN THE LATITUDINAL DIR.
C       IL      : NUMBER OF COMPUTATION POINTS IN THE LONGITUDINAL DIR.
C
C          THE MAXIMUM NUMBER OF POINTS THAT CAN BE STORED
C          IN THE ARRAYS PHI AND ALA IS 10,000.
C-----------------------------------------------------------------------
      XMIN=DFLOAT(60)
      XMINE=(ALASE-ALASW)*XMIN
      IL=(XMINE/XNTALA)+1
      XMINW=(PHINW-PHISW)*XMIN
      IP=(XMINW/XNTPHI)+1
      K=IL*IP
       DO 10 I=1,IP
        DO 11 J=1,IL
         PHI((I-1)*IL+J)=PHISW+DFLOAT(I-1)*XNTPHI/XMIN
         ALA((I-1)*IL+J)=ALASW+DFLOAT(J-1)*XNTALA/XMIN
 11     CONTINUE
 10    CONTINUE
       RETURN
       END
C***********************************************************************
      SUBROUTINE INTGR (PSIO,PHIO,ALAO,DELTG,SDELG,PHI,ALA,
     *                  DG,SDG,PSI,NMAX,NTOTAL,NUM,DN,SDN)
C-----------------------------------------------------------------------
C     FUNCTION   :   ANALYTICAL STOKES INTEGRATION IN A SPHERICAL CAP
C     AUTHOR     :   ALFRED KLEUSBERG
C     HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C-----------------------------------------------------------------------
C  PARAMETERS:     NAME    I/O    DESCRIPTION
C  ***********
C                  PSIO     I     RADIUS OF SPHERICAL CAP (RAD)
C                  PHIO     I     LATITUDE  OF CENTRE POINT    (RAD)
C                  ALAO     I     LONGITUDE OF CENTRE POINT    (RAD)
C                  DELTG    I     ARRAY OF GRAVITY ANOMALIES   (MGAL)
C                  SDELG    I     CORRESP. STAND. DEV.         (MGAL)
C                  PHI      I     CORRESP. LATITUDES           (RAD)
C                  ALA      I     CORRESP. LONGITUDES          (RAD)
C                  DG       O     ARRAY OF GRAV.ANOM. WITHIN PSIO (MGAL)
C                  SDG      O     ARRAY OF CORRESP. STAND. DEV. (MGAL)
C                  PSI      O     ARRAY OF CORRESP. SPHER. DISTANCES (RAD)
C                  NMAX     I     DIMENSION OF THESE ARRAYS AS SPECIFIED
C                                  IN CALLING PROGRAM
C                  NTOTAL   I     ACTUAL DIMENSION OF DELTG,SDELG,PHI,ALA
C                  NUM      O     ACTUAL DIMENSION OF DG,SDG,PSI
C                  DN       O     COMPUTED GEOID UNDULATION (M)
C                  SDN      O     CORRESPONDING STAND. DEV. (M)
C
C
C     EXTERNALS:      CUFIT, STINT
C     **********
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION DELTG(NMAX), SDELG(NMAX), PHI(NMAX), ALA(NMAX)
      DIMENSION DG(NMAX),    SDG(NMAX),   PSI(NMAX)
      DIMENSION COEFF(10), SCOEFF(10)
      MAX = 10
C
C     SELECT GRAVITY ANOMALIES WITHIN A SPHERICAL DISTANCE OF PSIO
C     ***********************************************************
      CALL SELCT (PSIO,PHIO,ALAO,DELTG,SDELG,PHI,ALA,NMAX,NTOTAL,
```

```
     *             DG,SDG,PSI,NUM,PSIMIN)
C
              IF(NUM.GT.1.AND.PSIMIN.LT.(PSIO/3.DO))GO TO 1101
                DN=-999.9
                SDN=-999.9
               GO TO 2101
 1101 CONTINUE
C
C      POLYNOMIAL FIT FOR GRAVITY ANOMALIES
C      """""""""""""""""""""""""""""""""""
       CALL CUFIT (PSIO,DG,SDG,PSI,NMAX,NUM,COEFF,SCOEFF,MAX,MCOEFF)
C
C      ANALYTICAL STOKES INTEGRATION
C      """""""""""""""""""""""""""""
       CALL STINT (PSIO,COEFF,SCOEFF,MAX,MCOEFF,DN,SDN)
C
 2101    CONTINUE
       RETURN
       END
C*******************************************************************************
       SUBROUTINE CUFIT (PSIO,DG,SDG,PSI,NMAX,NUM,
      *                  COEFF,SCOEF,MAX,MCOEFF)
C-------------------------------------------------------------------------------
C      FUNCTION   :   POLYNOMIAL FIT TO GRAVITY ANOMALIES
C      AUTHOR     :   ALFRED KLEUSBERG
C      HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C-------------------------------------------------------------------------------
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C   PARAMETERS:    NAME    I/O     DESCRIPTION
C   """""""""""
C                  PSIO      I     MAXIMUM SPHERICAL DISTANCE (RAD)
C                  DG        I     ARRAY OF GRAVITY ANOMALIES (MGAL)
C                  SDG       I     ARRAY OF CORRESP. STAND. DEV. (MGAL)
C                  PSI       I     ARRAY OF CORRESP. SPHER. DISTANCES (RAD)
C                  NMAX      I     DIMENSION OF THESE ARRAYS AS SPECIFIED
C                  NUM       I     ACTUAL DIMENSION OF THESE ARRAYS
C                  COEFF     O     POLYNOMIAL COFFICIENT ARRAY (MGAL)
C                  SCOEF     O     CORRESP. STAND. DEV.        (MGAL)
C                  MAX       O     DIMENSION OF THESE ARRAYS AS SPECIFIED
C                  MCOEFF    O     ACTUAL DIMENSION OF THESE ARRAYS
C
C   EXTERNALS:     SPIN
C   """""""""""
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
       IMPLICIT REAL*8 (A-H,O-Z)
C
       DIMENSION DG(NMAX), SDG(NMAX), PSI(NMAX)
       DIMENSION COEFF(MAX), SCOEF(MAX)
C
       DIMENSION A(10), ATA(10,10), ATL(10)
       DIMENSION ASIG(10), ASIGA(10,10), SC(10,10)
       ZERO = 0.DO
       MAXI = 10
C
       MCOEFF = MAX
       IF(NUM.LE.MCOEFF) MCOEFF = NUM-1
C
  500 CONTINUE
C
C      INITIALIZE ARRAYS
```

```
C     ********************
      DO 100 I=1,MAXI
      A(I) = ZERO
      ASIG(I) = ZERO
      ATL(I) = ZERO
      DO 100 J=1,MAXI
      ATA(I,J) = ZERO
       IF(I.EQ.J)ATA(I,J)=1.0D-12
      ASIGA(I,J) = ZERO
  100 CONTINUE
C
      DO 150 I=1,NUM
      OBS = DG(I)
      SIG = SDG(I)
C
C     NORMALIZE ARGUMENTS
C     ***********************
      PS  = PSI(I)/PSIO
      IF(PS.LT.1.0D-4) PS = ZERO
C
C     DESIGN MATRIX ROW
C     *********************
      A(1) = 1.DO
      ASIG(1) = SIG
      DO 200 K=2,MCOEFF
      A(K)= PS*A(K-1)
      ASIG(K) = A(K)*SIG
  200 CONTINUE
C
C     NORMAL EQUATION MATRIX AND RIGHT HAND SIDE
C     **************************************************
      DO 300 K=1,MCOEFF
      ATL(K)  = ATL(K) + A(K)*OBS
      DO 300 J=1,MCOEFF
      ATA(J,K) = ATA(J,K) + A(K)*A(J)
      ASIGA(J,K) = ASIGA(J,K) + ASIG(K)*ASIG(J)
  300 CONTINUE
C
  150 CONTINUE
C
C     INVERT NORMAL EQUATION MATRIXENTS
C     *************************************
      CALL SPIN(ATA,MCOEFF,MAXI,DET,IDEXP)
C
C     SEE IF INVERSION REGULAR
C     ************************
      IF(IDEXP.GT.-3) GO TO 1000
C
C     ILL CONDITIONED NORMAL EQUATION MATRIX
C     **************************************************
      MCOEFF = MCOEFF - 1
      GO TO 500
C
C     SOLVE FOR POLYNOMIAL COEFFICIENTS
C     *********************************
 1000 CONTINUE
      DO 600 I=1,MCOEFF
      COEFF(I) = ZERO
      DO 600 J=1,MCOEFF
      COEFF(I) = COEFF(I) + ATA(I,J)*ATL(J)
  600 CONTINUE
```

```
C
C     STANDARD DEVIATIONS OF COEFFICIENTS
C     ###################################
      DO 700 I=1,MCOEFF
      DO 700 J=1,MCOEFF
      SC(I,J) = ZERO
      DO 700 K=1,MCOEFF
      SC(I,J) = SC(I,J) + ATA(I,K)*ASIGA(K,J)
  700 CONTINUE
      DO 800 I=1,MCOEFF
      DO 800 J=1,MCOEFF
      ASIGA(I,J) = ZERO
      DO 800 K=1,MCOEFF
      ASIGA(I,J) = ASIGA(I,J) + SC(I,K)*ATA(K,J)
  800 CONTINUE
      DO 900 I=1,MCOEFF
      SCOEF(I) = DSQRT(ASIGA(I,I))
  900 CONTINUE
C
      RETURN
      END
C*********************************************************************
      SUBROUTINE STINT (PSIO,COEFF,SCOEF,MAX,MCOEFF,DN,SDN)
C---------------------------------------------------------------------
C     FUNCTION   :   ANALYTICAL STOKES INTEGRATION
C     AUTHOR     :   ALFRED KLEUSBERG
C     HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C---------------------------------------------------------------------
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C PARAMETERS:    NAME    I/O    DESCRIPTION
C ############
C                PSIO     I     RADIUS OF INTEGRATION (RAD)
C                COEFF    I     NORMALIZED POLYN. COEFF. FOR
C                                GRAVITY ANOMALIES
C                SCOEF    I     CORRESP. STANDARD DEVIATIONS
C                MAX      I     DIMENSIN OF THESE ARRAYS AS SPECIFIED
C                MCOEFF   I     ACTUAL DIMENSION OF THESE ARRAYS
C                DN       O     GEOID UNDULATION
C                SDN      O     CORRESP. STAND. DEV.
C
C EXTERNALS:     NONE
C ###########
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION COEFF(MAX), SCOEF(MAX)
      DIMENSION CSTOK(7)
C
C     DEFINE POL.COFF. FOR STOKES FUNCTION*SIN(PSI)
C     (NORMALIZED FOR THE INTERVAL PSI<2.5 DEG
C     ########################################
            CSTOK(1)=1.0D0
            CSTOK(2)=.390334D0
            CSTOK(3)=-0.961691D0
            CSTOK(4)=2.34308D0
            CSTOK(5)=-3.37295D0
            CSTOK(6)=2.48583D0
            CSTOK(7)=-0.726341D0
      TWODEG = 2.5D0/45.D0*DATAN(1.D0)
      ISEVEN = 7
```

```
      R = 6.370D+6
      GAMMA = 981.D+3
C
C     DENORMALIZE STOKES COEFFICIENTS
C     """""""""""""""""""""""""""""""
      DO 100 I=2,ISEVEN
      CSTOK(I) = CSTOK(I)/TWODEG**(I-1)
  100 CONTINUE
C
C     DENORMALIZE COEFFICIENT FOR GRAVITY ANOMALIES
C     """""""""""""""""""""""""""""""""""""""""""""
      DO 200 I=2,MCOEFF
      COEFF(I) = COEFF(I)/PSIO**(I-1)
      SCOEF(I) = SCOEF(I)/PSIO**(I-1)
  200 CONTINUE
C
C     INTEGRATE
C     """""""""
      DN  = 0.DO
      SDN = 0.DO
      DO 300 IS = 1,ISEVEN
      DO 300 IC = 1,MCOEFF
      ISC = IS+IC-1
      A   = CSTOK(IS)/ISC*PSIO**ISC
      DN  = DN  + A*COEFF(IC)
      SDN = SDN + (A*SCOEF(IC))**2
  300 CONTINUE
      DN  = DN  * R/GAMMA
      SDN = DSQRT(SDN) * R/GAMMA
C
      RETURN
      END
C*******************************************************************************
      DOUBLE PRECISION FUNCTION SPHER (PHI1,ALA1,PHI2,ALA2)
C-------------------------------------------------------------------------------
C     FUNCTION   :   COMPUTE SPHERICAL DISTANCE FROM SPHERICAL
C                    COORDINATES OF TWO POINTS.
C     AUTHOR     :   ALFRED KLEUSBERG
C     HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C-------------------------------------------------------------------------------
C
C  PARAMETERS:      NAME       I/O      DESCRIPTION
C  """""""""""
C                   PHI1        I       LATITUDE  OF POINT #1 (RAD)
C                   ALA1        I       LONGITUDE OF POINT #1 (RAD)
C                   PHI2        I       LATITUDE  OF POINT #2 (RAD)
C                   ALA2        I       LONGITUDE OF POINT #2 (RAD)
C
C  EXTERNALS:       NONE
C  """""""""""
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
C
      CP =DSIN(PHI1)*DSIN(PHI2)+DCOS(PHI1)*DCOS(PHI2)*DCOS(ALA2-ALA1)
      IF(CP.GT.1.DO) CP = 1.DO
      SPHER = DARCOS(CP)
C
      RETURN
      END
C*******************************************************************************
      SUBROUTINE SELCT (PSIO,PHIO,ALAO,DELTG,SDELG,PHI,ALA,
```

```
     *                    NMAX,NTOTAL,DG,SDG,PSI,NUM,PSIMIN)
C-----------------------------------------------------------------------
C      FUNCTION   :   SELECT GRAVITY ANOMALIES WHICH FALL WITHIN
C                     A SPECIFIED SPHERICAL DISTANCE.
C      AUTHOR     :   ALFRED KLEUSBERG
C      HISTORY    :   JUNE 06, 1986 - VERSION  1.0
C-----------------------------------------------------------------------
C
C PARAMETERS:      NAME     I/O      DESCRIPTION
C ************
C                  PSIO      I       SPHERICAL DISTANCE (RAD)
C                  PHIO      I       LATITUDE OF CENTRE POINT  (RAD)
C                  ALAO      I       LONGITUDE OF CENTRE POINT (RAD)
C                  DELTG     I       ARRAY OF GRAVITY ANOMALIES
C                  SDELG     I       ARRAY OF CORRESP. STAND. DEV.
C                  PHI       I       ARRAY OF CORRESP. LATITUDES  (RAD)
C                  ALA       I       ARRAY OF CORRESP. LONGITUDES (RAD)
C                  NMAX      I       DIMENSION OF THESE ARRAYS AS SPEC.
C                  NTOTAL    I       ACTUAL DIMENSION OF THESE ARRAYS
C                  DG        O       ARRAY OF SELECTED GRAVITY ANOMALIES
C                  SDG       O       ARRAY OF CORRESP. STAND. DEV.
C                  PSI       O       ARRAY OF CORRESP. SPHER. DISTANCES (RAD)
C                  NUM       O       NUMBER OF SELECTED GRAVITY ANOMALIES
C
C EXTERNALS:       SPHER
C ***********
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION DELTG(NMAX),  SDELG(NMAX),  PHI(NMAX), ALA(NMAX)
      DIMENSION DG(NMAX),     SDG(NMAX),    PSI(NMAX)
C
      ICOUNT = 0
       PSIMIN=PSIO
C
      DO 100 I=1,NTOTAL
      PSI1 = SPHER (PHIO,ALAO,PHI(I),ALA(I))
      IF(PSI1.GT.PSIO) GO TO 100
      IF(PSI1.LT.PSIMIN) PSIMIN=PSI1
      ICOUNT = ICOUNT+1
      DG(ICOUNT)  = DELTG(I)
      SDG(ICOUNT) = SDELG(I)
      PSI(ICOUNT) = PSI1
  100 CONTINUE
C
      NUM = ICOUNT
C
      RETURN
      END
C*********************************************************************
      SUBROUTINE SPIN(Q,N,MM,DET,IDEXP)
C-----------------------------------------------------------------------
C  FUNCTION  :
C      SUBROUTINE SPIN IS A MATRIX INVERSION ROUTINE FOR SYMMETRIC
C      POSITIVE-DEFINITE MATRICES.  THE MATRIX INVERTED IS THE UPPER
C      N BY N PORTION OF THE MATRIX Q WHICH IS DIMENSIONED MM BY MM
C      IN THE CALLING ROUTINE.
C  AUTHOR     :   R.R. STEEVES
C  HISTORY    :   SEPTEMBER 1979 - VERSION  1.0
C-----------------------------------------------------------------------
C
```

```
C  INPUT:
C       Q - THE MATRIX DIMENSIONED MM BY MM WHICH CONTAINS THE
C           MATRIX TO BE INVERTED.
C
C       N - THE DIMENSION OF THE ACTUAL PART( UPPER LEFT CORNER)
C           OF Q WHICH IS TO BE INVERTED. ( N MAY BE EQUAL BUT MUST
C           NOT BE LARGER THAN MM) .
C       MM- DIMENSIONED SIZE OF Q IN THE CALLING ROUTINE.
C
C
C  OUTPUT:
C
C       Q - THE UPPER LEFT N BY N PORTION CONTAINS THE INVERSE OF
C           THE INPUT UPPER LEFT N BY N PORTION.
C
C       DET - THE NON-EXPONENT PORTION OF THE DETERMINANT OF THE
C             INPUT N BY N (UPPER LEFT PORTION OF Q) MATRIX. SEE
C             IDEXP  BELOW.
C
C       IDEXP - THE EXPONENT (OF 10) PART OF THE DETERMINANT DESCRIBED
C               UNDER DET ABOVE. THUS THE DETERMINANT IS RETURNED IN
C               TWO PARTS CORRESPONDING TO
C                    DETERMINANT = DET * 10 ** IDEXP .
C               THIS IS DONE TO AVOID UNDER OR OVERFLOW IN THE
C               COMPUTATION OF THE DETERMINANT.  TO PRINT THE DETERM-
C               INANT THE USER SHOULD PRINT BOTH NUMBERS AS FOLLOWS;
C               (FOR EXAMPLE)
C                  PRINT 10,DET,IDEXP
C               10 FORMAT(' ','DETERMINANT= ',F7.4,'D',I4)
C
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      REAL*8 Q(MM,MM),DET,SUM,DSQRT,DABS,RPART,APART
      DET=0.D0
      DO 4 J=1,N
      DO 4 I=1,J
      IF(I.EQ.1) GO TO 2                                                  20
      M=I-1
      SUM=0.0D0
      DO 1 K=1,M
1     SUM=SUM+Q(K,I)*Q(K,J)
      Q(I,J)=Q(I,J)-SUM
2     IF(I.EQ.J) GO TO 3
      Q(I,J)=Q(I,J)/Q(I,I)
      GO TO 4
3     CONTINUE
      IF(Q(I,I).LE.0.D0) Q(I,I)=1.D-12
      DET=DET+DLOG10(Q(I,I))
      Q(I,I)=DSQRT(Q(I,I))
4     CONTINUE
      IDEXP=DET
      RPART=DET-IDEXP
      APART=DABS(RPART)
      IF(APART.LT.1.D-20) DET=1.D0
      IF(APART.LT.1.D-20) GO TO 10
      DET=10**RPART
   10 CONTINUE
      DO 7 J=1,N
      DO 7 I=1,J
      IF(I.LT.J) GO TO 5
      Q(J,J)=1.0D0/Q(J,J)
      GO TO 7
```

```
5       SUM=0.0D0
        M=J-1
        DO 6 K=I,M
6       SUM=SUM-Q(I,K)*Q(K,J)
        Q(I,J)=SUM/Q(J,J)
7       CONTINUE
        DO 9 J=1,N
        DO 9 I=1,J
        SUM=0.0D0
        DO 8 K=J,N
8       SUM=SUM+Q(I,K)*Q(J,K)
        Q(I,J)=SUM
        IF(I.EQ.J) GO TO 9
        Q(J,I)=SUM
9       CONTINUE
        RETURN
        END
//GO.FT01F001 DD DSN=A.M66774.WENZEL.D180D90.TEST.FEB,
//             DISP=SHR,LABEL=(,,,IN)
//GO.SYSIN DD *
GRID
1.0
52.0 260.0 67.0 290.0
57.0 270.0 62.0 280.0 10.0 10.0
2
   56.0000   300.0000
   56.0000   300.1667
//
```