A General Purpose Contouring System

By

Gerald Ian Evenden

Open-file report 75-317
1975

This report is preliminary and has not been edited or reviewed for conformity with U.S. Geological Survey standards.
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>1</td>
</tr>
<tr>
<td>DISCLAIMER</td>
<td>2</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>3</td>
</tr>
<tr>
<td>USER DESCRIPTION OF PROGRAM CONTUR</td>
<td>5</td>
</tr>
<tr>
<td>USER DESCRIPTION OF PROGRAM SPLN2D</td>
<td>15</td>
</tr>
<tr>
<td>USER DESCRIPTION OF PROGRAM SPLN2X</td>
<td>22</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>24</td>
</tr>
<tr>
<td>APPENDIX A. - STANDARD FILE</td>
<td>25</td>
</tr>
<tr>
<td>B. - LISTING OF PROGRAM CONTUR</td>
<td>27</td>
</tr>
<tr>
<td>C. - LISTING OF PROGRAM SPLN2D</td>
<td>61</td>
</tr>
<tr>
<td>D. - LISTING OF PROGRAM SPLN2X</td>
<td>83</td>
</tr>
<tr>
<td>E. - BASIC GRAPHICS SYSTEM</td>
<td>93</td>
</tr>
<tr>
<td>INDEX</td>
<td>107</td>
</tr>
</tbody>
</table>
Abstract

Three DecSystem-10 FORTRAN IV programs provide a general purpose system for contouring two-dimensional data. The system can provide both quick or final, publication quality contour maps on either interactive or offline plotting devices. Complete user documentation, with examples, and program listings are presented.
DISCLAIMER.

Although these programs have been subjected to many tests and considerable usage a warranty on accuracy or proper functioning is neither implied nor expressed.
INTRODUCTION.

This system of three FORTRAN programs provides a simple yet effective means of contouring grids of two dimensional data. The graphical phase of contouring is performed by program CONTUR which determines the contour path by linear interpolation and provides for contour labeling, axis labeling and dropping of contours in high gradient regions.

Lack of contour smoothness and accuracy of linear interpolation of data with considerable high frequency content may require a more finely interpolated grid. Program SPLN2D provides this interpolation service with bicubic spline methods. A third program, SPLN2X, is also provided to determine bicubic spline coefficients for data grids larger than can be handled by SPLN2D.

All input and output data grids and spline coefficient files must be in a standard form (see Appendix A). All programs also provide for missing data in the grid provided such data is flagged with the computer system maximum floating point number.

The following sections contain complete user documentation required to execute the contouring system. It is assumed that the prospective user is familiar with the facilities of the Decsystem-10 computer and simple FORMAT constructs and NAMELIST data entry of the FORTRAN language. Any user should not expect instant success on first using the programs because of the many details involved. It is suggested that familiarization of the routines be made by plotting on an interactive device such as the Tektronix 4010 and using only the required options.

These programs can be transported to other computer systems with a moderate amount of revision by an experienced programmer. Each program (listings in Appendixes B through D) are written in Decsystem-10 FORTRAN IV. Since attention was given to both timing and memory performance all three programs contain non-standard FORTRAN constructs in file manipulation and multiple entry subroutines. Program CONTUR also contains additional non-standard constructs involving bit manipulation of flag arrays by "logical masking" statements.

Several routines in programs SPLN2D and SPLN2X are based on Anderson's (1971) algorithms. The only basic changes were to provide for simple monotonic independent variables and reduction of work files and I/O operations in SPLN2X.

The listings contain all code except basic graphics routines and GMPRD (general matrix product) subroutine. Appendix E contains user documentation of the graphics system employed and GMPRD is commonly available in
SCIENTIFIC SUBROUTINE PACKAGES. NOTE THAT SUBROUTINES COMMON TO BOTH SPLN2D AND SPLN2X ARE LISTED ONLY UNDER SPLN2D.
CONTOURING SYSTEM

USER DESCRIPTION OF CONTUR

General.

CONTUR is a general purpose routine for contouring two dimensional data. The routine employs the contour tracing method where a contour is followed through the grid. This allows contour labeling and minimizes pen up-down motions for mechanical plotters. However, it requires more computer memory and time than methods completely contouring one grid cell at a time.

Coordinates of the contour line are determined by linear interpolation of the contour through the grid cell "walls". This will often produce angular contours for data grids containing significant high frequency components. Although such angularity does not matter for preliminary plots, smoothed, publication quality maps can be obtained by previous interpolation with program SPLN2D.

CONTUR can also contour quadrilateral grids where individual cell interior angles do not exceed 180 degrees. The input of such a grid is signalled to the program by the NZ factor of the standard file being equal to 3. Each element of the input grid must then contain X, Y and Z values. The spline interpolation routines, however, cannot handle this type of data.
Execution of CONTUR.

Prior to running CONTUR the logical device "PLOT" must be assigned to a device appropriate to the plotter employed. This can be done by:

ASSIGN TTY PLOT

for the Tektronix 4010 or Hewlett-Packard 7202A and 7203A, or by:

MOUNT MTA: PLOT/REELID: PLOTAP/VID: 7-TRK/WE

for the Gerber 622.

For testing the plot command file:

ASSIGN NULL PLOT

Can be employed.

CONTUR is executed by:

RUN CONTUR[333, 724]

After the program is loaded an asterisk is typed on the terminal. The user responds with the name and extension of a CONTUR command file (if the extension is omitted, ".DAT" is assumed). After the current command file is executed the routine will request another command file. This sequence can continue indefinitely. To exit from the program the user can enter "EXIT" after the asterisk is typed.
CONTUR COMMAND FILE.

THE COMMAND FILE IS AN ASCII FILE CONTAINING INFORMATION TO CONTROL FORMATTING OF THE CONTOUR PLOT, TITLING DATA AND NAME OF THE STANDARD TWO-DIMENSIONAL FILE TO BE CONTOURED. BASICALLY, THE FILE CONSISTS OF ONE OR MORE "COMMAND SETS". EACH COMMAND SET CONSISTS OF 4 FIXED FORMAT LINES FOLLOWED BY A " &PARMS" NAMELIST SECTION. THE NAMELIST SECTION MAY BE CONTINUED ON SEVERAL LINES (SEE FORTRAN MANUAL FOR DETAILS OF NAMELIST DATA ENTRY). EXCEPT FOR THE FIRST LINE (INPUT DATA FILE NAME) THE PROGRAM CONTROL FUNCTIONS ARE NOT ALTERED BETWEEN COMMAND SETS UNLESS:

A. NON-BLANK FIELDS ARE ENCOUNTERED IN CONTROL SET LINES 2, 3 OR 4 AND
B. THE KEYWORD IDENTIFIER IS ENTERED IN THE NAMELIST SECTION.

A CONSEQUENCE OF THIS FEATURE IS THAT THE INITIAL COMMAND SET CREATES DEFAULT VALUES FOR THE REMAINDER OF THE JOB RUN OR UNTIL THEY ARE EXPLICITLY ALTERED AGAIN.
Details of command set.

Line 1.

Characters 1 through 10 contain file name and extension of a standard format two dimensional file to be contoured. If the extension is absent ".DAT" is assumed. If the field is blank all parameters of the remainder of the command set are entered but execution of the grid input and contouring is bypassed.

Line 2.

Second title line to be plotted in the margin. The first title line is always obtained from the standard format file. Default: blanks.

Line 3.

Third title line. Default: blanks.

Line 4.

Formats to be used for contour and neatline annotation. Note that format data consists of not only these fields but also "NCHAR" and "SIZE" type variables in the &PARMS section. Improper mixture can brew strange results. Characters 1 through 20 contain format for contour label. Default: blanks. Characters 21 through 40 contain x-axis label format. Default: blanks. Characters 41 through 60 contain y-axis label format. Default: blanks.

Line 5.

NameList section.

Characters 2 through 8 must contain: &PARMS. Characters 9 on may contain nameList items as well a continuation lines. Section must be terminated by &:c. Remember: character 1 on each line is ignored. Except for the variables ACVAL, CMIN, CMAX and IDASHS all variables are expected to be greater than or equal to zero. If they are less than zero they will be assigned a zero value.

Contour levels.

Contours are determined by inputting either an array of "specified contour" levels or "incremental contour" levels. For the specified contour method:
CONTOURING SYSTEM. PRGE 9

USER DESCRIPTION OF CONTUR

NCVAL = NUMBER OF ELEMENTS IN ACVAL LIST.

FOR THE INCREMENTAL CONTOUR LEVEL METHOD:

NCVAL = 0 (DEFAULT VALUE).

DCVAL = DIFFERENTIAL CONTOUR LEVEL. MUST BE GREATER
THAN ZERO. DEFAULT = 0 THUS CREATING AN ERROR
CONDITION IF ALL CONTOUR SPECIFICATIONS ARE
IGNORED.

CMIN = LOWER LIMIT OF CONTOUR LEVELS.

CMAX = UPPER LIMIT OF CONTOUR LEVELS. IF
CMIN=CMAX=0, (DEFAULT) THEN RANGE OF CONTOURING
DETERMINED BY DATA.

MSEC = PRIMARY CONTOUR LEVEL INTERVAL. IF LESS THAN
OR EQUAL TO ONE THEN ALL LEVELS ARE CONSIDERED
PRIMARY. DEFAULT = 1.

GRADI = MAXIMUM GRADIENT, IN CONTOURS PER INCH;
BEFORE SECONDARY CONTOURS IN GRID CELL ARE NOT
PLOTTED. DEFAULT = 30.

IDASHS = SECONDARY-PRIMARY CONTOUR LINE DASHING.
= 0 ALL CONTOURS PLOTTED AS SOLID LINES.
= -1 SECONDARY CONTOURS PLOTTED AS DASHED LINES.
= 1 PRIMARY CONTOURS PLOTTED AS DASHED LINES.

CONTOUR LABELING IS PERFORMED FOR ALL SPECIFIED
LEVELS AND ALL PRIMARY INCREMENTAL LEVELS IF NCHAR
GREATER THAN ZERO. THE FOLLOWING PARAMETERS AFFECT THE
FORM OF THE CONTOUR LABELING.

NCHAR = NUMBER OF CHARACTERS IN FORMAT OUTPUT FIELD.
USUALLY EQUAL TO THE FIELD WIDTH PART OF THE FORMAT
STATEMENT IN FIELD 1 OF LINE 4. DEFAULT = 0 (NO
LABELING).

SIZE = HEIGHT, IN INCHES, OF LABELING CHARACTERS.
DEFAULT = 0.06.

PLOTTER SELECTION.

IPLTR = PLOTTER TYPE CODE.
= 0 GERBER 622, DEFAULT.
= 1 Tektaronic 4010
= 2 Hewlett-Packard 7202A
= 3 Hewlett-Packard 7203A
ALL OTHER VALUES WILL CAUSE ERROR CONDITIONS.
GRID SCALING.

XSCALE = X-axis data units per inch. Default = 0.

YSCALE = Y-axis data units per inch. Default = 0.
If both scaling factors are zero then both axis scales will be set equal to the same value which will allow plotting on the lesser of plot device board size or 10 by 8 inches. If one scale factor is not zero and the other is zero the zero factor will be assigned the non-zero value.

GENERAL PLOT ANNOTATION DETAILS.

SIZEL = HEIGHT OF LABEL LINE CHARACTERS, IN INCHES. Default = 0.07. If = zero then title lines not plotted.

NOTE: THE FOLLOWING X OR Y ([X/Y]) SUFFIXES REFER TO THE RESPECTIVE X, Y AXIS LABELING AND POSITION PARAMETERS.

NCHAR[X/Y] = NUMBER OF CHARACTERS IN RESULTING AXIS LABELING FORMAT FIELD. NORMALLY EQUAL TO "W" FIELD OF RESPECTIVE LINE 4 FORMAT FIELDS. Default = 0 (IF NO LABELING).

SIZE[X/Y] = HEIGHT, IN INCHES, OF AXIS LABELING CHARACTERS. Default = 0.

ADEL[X/Y] = INTERVAL OF AXIS TICKS. Default = 0. If equal to zero, then only minimum and maximum values posted.

LINT[X/Y] = PRIMARY INTERVAL OF POSTING. Default = 1. If less than 2 then every interval labelled.

PLL[X/Y] = POSITION OF LOWER-LEFT CORNER OF GRID AREA RELATIVE TO PLOT EDGE, IN INCHES. If equal to zero, then respective position determined automatically. Principally available for special purposes.
CONTOURING SYSTEM
USER DESCRIPTION OF CONTUR

EXAMPLE OF CONTUR COMMAND FILE.

THE FOLLOWING ASCII COMMAND FILE NAMED SAMPLE.CNT
CREATED BY EITHER TECO OR $DECK PERFORMED THE PLOTS ON THE
FOLLOWING PAGES. THE STANDARD TWO-DIMENSIONAL FILE LVGRAY
WAS CREATED BY UTILTY FROM PUNCH CARDS OF HAND DIGITIZED
DATA. THE RESULTS OF EACH COMMAND SET ARE SHOWN IN FIGS.
1, 2 AND 3.

LVGRAY
PLOT SAMPLE 1

&PARMS
  DCVAL=100., IPLTR=1,
&END
LVGRAY
PLOT SAMPLE 2
MORE COMPLEX CALL
<F6.0>
&PARMS
  NSEC=5, NCHAR=5, SIZE=.07,
&END
LVGRAY
PLOT SAMPLE 3

(F6.0)  (F8.0)  (F8.0)
&PARMS
  NCHARX=7, NCHARY=7, SIZEX=.08, SIZEY=.08,
  ADELX=1000., ADELY=1000., LINTX=10, LINTY=10,
&END

CONTUR W3S EXECUTED AS:

.RUN CONTUR[333,724]
*SAMPLE.CNT

<<TK4010X/Y:6.9/5.0-INCCH I/O:3/3[LN I/O:3871/3846
<<  DEV:NULL[PLT000.DAT][BLKS:19

<<TK4010X/Y:6.9/5.0-INCCH I/O:12/12[LN I/O:2642/2617
<<  DEV:NULL[PLT000.DAT][BLKS:15

<<TK4010X/Y:6.9/5.0-INCCH I/O:192/192[LN I/O:2402/2402
<<  DEV:NULL[PLT000.DAT][BLKS:18
*EXIT
STOP

END OF EXECUTION
CPU TIME: 25.18 ELAPSED TIME: 1:17.12
EXIT
Figure 1. Resulting contour map of command set 1.
Figure 2. Resulting contour map of command set 2.
Figure 3. Resulting contour map of command set 3.
SPLN2D PROVIDES TWO FUNCTIONS: 1) DETERMINE SPLINE COEFFICIENTS OF A TWO DIMENSIONAL GRID AND 2) TWO DIMENSIONAL INTERPOLATION. SPLINE POLYNOMIAL PROPERTIES AND ALGORITHMS FOR THEIR DETERMINATION ARE DESCRIBED BY ANDERSON (1971) AND HIS SUBROUTINES, WITH MINOR MODIFICATIONS, ARE EMPLOYED IN THIS PROGRAM. IN THE INTERPOLATION FUNCTION OF SPLN2D TWO BASIC MODES ARE PROVIDED: 1) SIMPLE INTEGRAL SUBDIVISION OF THE ORIGINAL GRID AND 2) DETERMINATION OF A NEW MESH AT EQUALLY SPACED COORDINATES.

THE SINGULAR DISADVANTAGE OF THE SPLINE METHOD IS MEMORY REQUIREMENTS. SPLN2D, AS CURRENTLY COMPILED, IS RESTRICTED TO DETERMINING SPLINE COEFFICIENTS OF GRIDS OF $I$ COLUMNS AND $J$ ROWS THAT MEET THE FOLLOWING RESTRICTION:

$$4 + I + 17 + I + J + 6 + \text{MAXIMUM OF } (I,J) + 12 \leq 20000$$

WHERE $I2$ IS THE NUMBER OF INTERPOLATED OUTPUT COLUMNS ($I2=0$ IF SPLINE POLYNOMIALS ARE OUTPUT). THIS REPRESENTS THE EQUIVALENT OF A $67 \times 67$ GRID LIMIT WHEN $I2=4+I$. IF PRE-DETERMINED POLYNOMIAL COEFFICIENTS ARE INPUT, THE INTERPOLATION PHASE REQUIRES THAT:

$$33 + I + 2 + I2 \leq 20000.$$ 

INPUT, COEFFICIENT AND INTERPOLATED FILES ARE ALL STANDARD FORMAT FILES. THE INPUT FILE MUST HAVE $NZ=1$ (SEE APPENDIX A) AND, OBVIOUSLY, THE INTERPOLATED FILE WILL ALSO HAVE $NZ=1$ WITH THE PGM VARIABLE SET TO "SPLINE-INT". THE COEFFICIENT FILE WILL HAVE $NZ=16$ AND PGM="SPLNCOEFF**". ALSO, THE COEFFICIENT FILE WILL HAVE $NCOL$ BY $NROW$ DATA EVEN THOUGH $NCOL-1$ BY $NROW-1$ COEFFICIENT SETS ARE EMPLOYED (FIRST COEFFICIENT OF $NCOL$ OR $NROW$ COORDINATES ARE FLAGGED) AND THE $Y$ VALUE OF EACH ROW RECORD CONTAINS THE ACTUAL VALUE REGARDLESS OF $DY$ SETTING.

OPTIONALLY, THE DIRECTIONAL DERIVATIVE AND SECOND VERTICAL DERIVATIVE CAN BE GENERATED AS THE INTERPOLATED OUTPUT. THESE ARE RESPECTIVELY DEFINED AS:

$$|\Delta u| = \left( \left( \frac{\partial u}{\partial x} \right)^2 + \left( \frac{\partial u}{\partial y} \right)^2 \right)^{1/2}$$

AND

$$\frac{\partial^2 u}{\partial x^2} = \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

WHERE $U$ IS THE BICUBIC SPLINE COEFFICIENTS. NOTE THAT FOR SECOND DERIVATIVE OUTPUT LINEAR SEGMENTS WILL RESULT. TO OBTAIN SMOOTH SECOND DERIVATIVE PlOTS RE-SPLINE AND INTERPOLATE WITH A SECOND PASS.
Execution of SPLN2D.

SPLN2D is executed by:
RUN SPLN2D(333,724)

The program responds by requesting:
ENTER INPUT FILE NAME.EXT:

If spline coefficients need to be determined the user enters the appropriate file name. If "EXIT" is entered the routine stops execution. Note that if a "extension" is omitted, "DAT" extension is assumed for all file name requests. If spline coefficients are already determined enter a blank line. The program then responds; in both cases, with

ENTER COEFFICIENT FILE NAME.EXT:

If the input file name is blank, then a name must be entered. If the input file name is not blank and it is desired to save the spline coefficients, then a name is entered. When coefficients are saved, the program starts over and asks for a new input file.

When coefficients are input or not saved the program proceeds to the interpolation phase and makes the following request for interpolated output file name:

ENTER INTERPOLATED FILE NAME.EXT:

If no file name is entered, the program starts over. After the name is entered a request for interpolation factors is asked for by:

ENTER PARAMETERS:

The user responds with:

$&P ARMS$ selected namelist parameter values &

After the output file is generated the routine starts over.

Interpolation parameters.

For simple subdivision of grid cells:
IDEIX=; IDELY= integral fractional parts that the respective X and Y axis of each grid cell are to be subdivided. One or both must be greater than zero. If one parameter is less than or equal to zero it is assigned the value of the other parameter. This method is most frequently used for "smoothing" of grids for contouring.
For generation of an equally spaced grid from spline coefficients:

- **XDEL**, **YDEL**\(=\) respective \(X\) and \(Y\) interval of grid spacing. If one parameter is less than or equal to zero it is assigned the value of the other parameter.

- **XOFF**, **YOFF**\(=\) respective \(X\) and \(Y\) offsets algebraically added to the \(XO\), \(YO\) parameters of the input data or coefficient file (see Appendix A).

  *Default: \(XOFF=YOFF=0\).*

- **NCOLO**, **NRWD**\(=\) number of output columns and rows of new interpolated grid.

If any portion of the new grid coordinates lay outside the old grid system automatic flagging of empty cells occurs (i.e., no extrapolation). It is possible to generate an output grid completely flagged if poor values are selected.

For obtaining derivative output in either of the above modes:

- **IDER.LT.0** for directional-first derivative
- **IDER.GT.0** for second vertical derivative
- **IDER.EQ.0** (default) for normal interpolation.
CONTOURING SYSTEM.
USER DESCRIPTION OF SPLN2D

EXAMPLES.

Figure 4 shows a typical contour map of an "unsmoothed" grid. To obtain a smoother plot (underlined are user entries):

```plaintext
.RUN SPLN2D[333,724]
ENTER INPUT FILE NAME.EXT:LV
ENTER COEFFICIENT FILE NAME.EXT:
ENTER INTERPOLATED FILE NAME.EXT:
Enter PARAMETERS:
&&PARMS IDELX=4;&
ENTER INPUT FILE NAME.EXT:EXIT
STOP
END OF EXECUTION
CPU TIME: 3.40 ELAPSED TIME: 39.25
EXIT
```

The recomputed result is shown in Figure 5. Typically, subdividing grid blocks by 4 produces an adequate contour map. Occasionally, a 6 to 8 subdivision factor is needed for extremely rough data sets.

For the second example the same data set is input: the coefficients saved and an equal grid sub-area is generated (note: original data has 25 columns by 17 rows at 2000 unit intervals).

```plaintext
.RUN SPLN2D[333,724]
ENTER INPUT FILE NAME.EXT:LV
ENTER COEFFICIENT FILE NAME.EXT:LVCOEF
ENTER INPUT FILE NAME.EXT:
ENTER COEFFICIENT FILE NAME.EXT:LVCOEF
ENTER INTERPOLATED FILE NAME.EXT:
Enter PARAMETERS:
&&PARMS XDEL=500,XOFF=32710,NCOLO=20,
        YDEL=-500,NROWD=20;&
ENTER INPUT FILE NAME.EXT:EXIT
STOP
END OF EXECUTION
CPU TIME: 3.24 ELAPSED TIME: 1:18.12
EXIT
```

Figure 6 shows a contour plot of the interpolated grid.
Figure 4. Contour map of original unsmoothed grid.
Figure 5. Contour map of smoothed grid.
Figure 6. Contour map of equal spaced sub-area.
SPLN2X IS A ROUTINE TO DETERMINE SPLINE COEFFICIENTS FOR LARGE TWO DIMENSIONAL RECTANGULAR GRIDS OF UP TO 500 ROWS BY 500 COLUMNS. THIS IS AN IMPROVED ALGORITHM BASED UPON THE ORIGINAL PROGRAM BIGGRID (ANDERSON, 1971) WHICH OVERLAYS INTERMEDIATE RESULTS ON DISK. OPERATION OF THE PROGRAM IS NEARLY IDENTICAL TO SPLN2X EXCEPT INTERPOLATION IS NOT PROVIDED. IF INTERPOLATION IS DESIRED, SPLN2D CAN BE EXECUTED WITH SPLINE COEFFICIENTS GENERATED BY THIS PROGRAM AS INPUT.

Execution of SPLN2X

SPLN2X is executed by:

RUN SPLN2X[333,724]

The program responds by requesting:

ENTER INPUT FILE NAME.EXT:

A non-blank name must be entered. If the field is blank a re-request is made. To exit from SPLN2X enter "EXIT". If the input file appears to be acceptable, a request for the name of the spline coefficient file is made:

ENTER COEFFICIENT FILE NAME.EXT:

After the coefficient file has been written the program restarts and requests a new input file.

Because of the simplicity of execution and similarities to SPLN2D, no examples are presented.
REFERENCES.

ANDERSON, W. L., 1971, APPLICATION OF BICUBIC SPLINE FUNCTIONS TO TWO-DIMENSIONAL GRIDDED DATA; NATIONAL TECHNICAL INFORMATION SERVICE REPORT PB-203579, 87p.
APPENDIX A.

STANDARD FILE.

The established one or two dimensional file facilitates data linkage between commonly used programs. It consists of two basic parts: 1) a header record providing control parameters and, optionally, a following record of column coordinates and 2) the profile or grid data. Recording mode is FORTRAN 'BINARY'.

A. Header record (23 words long)
- ID: 70 ASCII characters of identification (14 words).
- PGM: 10 ASCII characters of creation program identification (2 words).
- NCOL: number of columns of data (integer; 1 word).
- NROW: number of rows of data (integer; 1 word). For one dimensional data: set to 1.
- NZ: number of words per data element (integer; 1 word). For single precision: single valued data use 1; double precision or complex use 2; double precision complex use 4; .... etc.
- XD: position of first column of data (real; 1 word).
- DX: equal spacing interval of columns (real; 1 word). If equal to zero, then coordinate for each column is in the data record; otherwise the following record consists of data.
- YD: position of first row (real; 1 word).
- DY: equal spacing interval of rows (real; 1 word). If equal to zero, the the coordinate for each row is the first word of each data record row.

B. Column coordinate record; present only if DX of header record equal to zero. Record consists of NCOL real words specifying the coordinates of each data column. Most programs require the coordinates to be in monotonic order.

C. Data record. Each data record contains one row of real data items. The total record length is NCOL times NZ plus 1 words. The first word contains the row coordinate if DY zero. When DY not zero, the first word is ignored. When specified, the row coordinates should be in monotonic sequence. Many programs accept incomplete data where the missing data is flagged by the maximum floating point number (octal 377777777777).
In general, I/O for this standard file can be stated in FORTRAN as:

```fortran
DIMENSION G(IZ,IX,IY),ID(I4),PGM(2),X(IX),Y(IY)
::
READ or WRITE (..) ID,PGM,NCOL,NROW,NZ,XO,DX,YO,DY
IF (DX.EQ.0.) READ or WRITE (..) (X(I),I=1,NCOL)
IF (DY.NE.0.) GO TO 15
DO 10 J=1,NROW
  READ or WRITE (..) Y(J),((G(K,I,J),K=1,NZ),I=1,NCOL)
  GO TO 20
15  DO 19 J=1,NROW
19  READ or WRITE (..) DUM,((G(K,I,J),K=1,NZ),I=1,NCOL)
20  CONTINUE
::
```

The above is a simplistic example and general usage is not warranted nor suggested. It is only provided as a guide.
APPENDIX B

LISTING OF PROGRAM CONTUR

PROGRAM CONTUR
C GENERAL CONTOURING PROGRAM
C DEVELOPED AND CODED BY..
C GERALD IAN EVENDEN
C U. S. GEOLOGICAL SURVEY
C DENVER FEDERAL CENTER
C DENVER, COLORADO 80225
C
DIMENSION WORK(10000)
C
C VERSION IV-B
C
NOTE.. THIS PROGRAM CONTAINS SEVERAL NON-ANSI FORTRAN
STATEMENTS. ALTHOUGH SEVERAL OF THESE CONSTRUCTS
MAY BE AVAILABLE ON OTHER COMPUTER SYSTEMS CARE MUST
BE EXERCISED IN THE TRANSPORTATION OF THIS PROGRAM.
C
A WARRANTEE ON PROPER FUNCTIONING OF THIS
PROGRAM IS NEITHER IMPLIED NOR EXPRESSED.
C
DOUBLE PRECISION TITLE1,TITLE2,TITLE3,FMTX,FMTY
NAMELIST /PARMS/
1 NCVRL,CMIN,CMAX,NSEC,DCVAL,XSCALE,YSCALE
2 ACVAL,GRADI,iplotR,NCHAR,SIZE,IDASHS
3 SIZEL,NCHARX,NCHARY,SIZEX,SIZEY,ADELX,ADELY,
4 PLLX,PLL,LY,LINTX,LINTY
COMMON /SEXAY/
1 TITLE1(7),TITLE2(8),TITLE3(8),FMTX(2),FMTY(2),
2 XX(2),YY(2),XP(4),YP(4),IPL0TR,SIZEL,NCHARX,NCHARY,
3 SIZEX,SIZEY,ADELX,ADELY,PLLX,PLL,LY,LINTX,LINTY,
4 XSCALE,YSCALE,XXSCAL,YSCAL
COMMON /CONCOM/ NCOL,NROW,BMIN,BMAX,
1 GRAD,GFL6,ILJ,IEJ,ILJ4,IEJ4,PRIME,ILJ4,CONT,
2 FLTMX,LMULT(0/3),IDASHS,LINET
C
LOGICAL PRIME
C
COMMON/CONTRC/ CMIN,CMAX,DCVAL,NCVAL,NSEC,GRADI
DOUBLE PRECISION CONNAME,FNAME,FMTC(2),BLANK
DIMENSION DUMDUM(6)
EQUIVALENCE (DUMDUM(1),CONNAME),(DUMDUM(4),FNAME)
LOGICAL NDMORE
DIMENSION ACVAL(1000)
EQUIVALENCE (ACVAL(1),WORK(1))
DATA NWORK/10000/
DATA DUMDUM/6*0./
DATA BLANK/
DATA ICMD,IFILE/5,10/

C GET COMMAND FILE NAME.

10 TYPE 20
20 FORMAT('<',*,$)
   ACCEPT 30,CONAME
30 FORMAT(A10)
   IF (CONAME.EQ. 'EXIT') STOP
   OPEN (UNIT=ICMD,MODE='ASCII',ACCESS='SEQIN',
        1 FILE=CONAME,DEVICE='DSK')
   NWKRES=NWORK

C GET CONTROL
C INPUT DATA FILE NAME AND FORMATS.
40 CALL HEADS(FNAME,TITLE2,TITLE3,FMTC,FMTX,FMTY,ICMD,I)
   IF (I.NE.0) GO TO 240

C GET NUMERIC CONTROL VALUES.
   READ(ICMD,PARMS,END=220)
   IF (NSEC.LT.0) NSEC=0
   IF (GRADI.LT.0.) GRADI=0.
   IF (NCHAR.LT.0) NCHAR=0
   IF (SIZE.LT.0.) SIZE=0.

C START CHECK-OUT.
C CONTOUR VALUES...
   IF (NCVRL.NE.0) GO TO 70

C INCREMENTAL CONTOURS MODE.
   NCVAL=0
   IF (DCVRL.NE.0) GO TO 60
   IF (FNAME.EQ.BLANK) GO TO 40
   TYPE 50
50 FORMAT('< %DCVRL LESS THAN OR EQUAL TO ZERO'>
   GO TO 40
60 IXAD=1
   GO TO 110

C SPECIFIED CONTOURS MODE.
70 IF (NCVRL.EQ.1) GO TO 100

C CHECK ASCENDENCY.
   DO 90 I=2,NCVRL
      IF (WORK(I),GT,WORK(I-1)) GO TO 90
      TYPE 80
80 FORMAT('< %NON-ASCENDING SPECIFIED CONTOURS'>
   GO TO 40
90 CONTINUE
   IXAD=NCVRL+1

C ALL THAT CAN BE DONE WITHOUT LOOKING AT DATA FILE.
C OPEN FILE AND PRELIM CHECK.
110 NWKRES=NWKRES-NCVAL
   IF (FNAME.EQ.BLANK) GO TO 40
   CALL OPENCK(IFILE,FNAME,XO,YO,DELX,DELY,
          1 WORK(IFAD),NWKRES,IQUAD)
   IF (NCOL.EQ.0) GO TO 40
C
C SLICE UP CORE
   IF (IQUAD.EQ.0) GO TO 150
   NRQMT=4+(NWKRES-NCOL)/(5*NCOL+4)
   IF (NRQMT.GE.2) GO TO 140
C
120 TYPE 130
130 FORMAT(’INSUFFICIENT MEMORY’)
   GO TO 210
C
140 IDAD=IXAD+NCOL
   IYAD=NWORK-NRQMT+1
   IFAD=IDAD+NRQMT+NCOL
   GO TO 160
C
150 NRQMT=4+NWKRES/(13*NCOL)
   IF (NRQMT.LT.2) GO TO 120
   K=NRQMT+NCOL
   IDAD=IXAD+K
   IYAD=NWORK-K+1
   IFAD=IDAD+K
C
C OK SO FAR, SCALE AND ANNOTATE
160 CALL SEXUAL(I)
   IF (I.NE.0) GO TO 210
   IF (IQUAD.EQ.0) GO TO 200
   K=IXAD+NCOL-1
   IF (DELX.NE.0.) GO TO 180
   DO 170 I=IXAD,K
170 WORK(I)=WORK(I)*XXSCA
   GO TO 200
180 XO=XO*XXSCA
   DELX=DELX*XXSCA
   DO 190 I=IXAD,K
   WORK(I)=XO
190 XO=XO+DELX
C
200 NPASS=(NRQMT+NRQMT-3)/(NRQMT-1)
   JRES=1+NRQMT-NRQMT-(NPASS-2)*NRQMT-1
C
   CALL SETCON(FMTC,SIZE,NCHAR)
   CALL CONTR(WORK(IFAD),WORK(IFAD),WORK(IFAD),
          1 WORK(IFAD),WORK(NRQMT,NCOL,XXSCAL,YSCAL,GFL6,
          2 YO*YSCAL,DELX*YSCAL,IQUAD,NPASS,JRES,IFILE)
C
C CLOSE PLOT
   CALL ENDPRT(0)
C
C CLOSE DATA FILE
210 CLOSE (UNIT=IFILE)
    GO TO 40
C
C COMMAND FILE END
220 TYPE 230
230 FORMAT ("%DOD EOF ON COMMAND FILE")
240 CLOSE (UNIT=ICMD)
    GO TO 10
    END
SUBROUTINE HEADS(FNAME,T1,T2,FMTA,FMTB,FMTC,
1  IFILE,IER)

C SET FIRST 4 LINES OF CONTROL DATA.

C IMPLICIT DOUBLE PRECISION (A-H,O-Z)

DIMENSION T1(8), T2(8), FMTA(2), FMTB(2), FMTC(2), STRL(8)

READ(IFILE,10,END=20) FNAME
10 FORMAT(8A10)
READ(IFILE,10,END=20) STRL
CALL UPDATE(T1,STRL,8)
READ(IFILE,10,END=20) STRL
CALL UPDATE(T2,STRL,8)
READ(IFILE,10,END=20) STRL
CALL UPDATE(FMTA,STRL,2)
CALL UPDATE(FMTB,STRL(3),2)
CALL UPDATE(FMTC,STRL(5),2)
IER=0
GO TO 30
20 IER=1
30 RETURN
END
SUBROUTINE UPDATE(A,B,N)
C
C UPDATE A IF B NON-BLANK
C
DOUBLE PRECISION A(1), B(1), BLANK
DATA BLANK/0.0/
DO 10 I=1,N
  IF (B(I), NE. BLANK) GO TO 20
10 CONTINUE
  GO TO 40
20 DO 30 I=1,N
30 A(I)=B(I)
40 RETURN
END
SUBROUTINE OPENCK(IFILE, FNAME, XD, YD, DELX, DELY, 
  1 XDATA, NWORK, IQUAD)

C C INITIALIZE GRID INPUT FILE.
C
DOUBLE PRECISION FNAME, DUMMY
DIMENSION XDATA(I)
DOUBLE PRECISION TITLE1, TITLE2, TITLE3, FMTX, FMTY
COMMON /SEDXY/ 
  1 TITLE1(3), TITLE2(3), TITLE3(3), FMTX(2), FMTY(2), 
  2 XX(2), YY(2), XP(2), YP(2), IPLTR, SIZEL, NCHARX, NCHARY, 
  3 SIZEX, SIZEY, ADELX, ADELY, PLLX, PLLY, LINTX, LINTY, 
  4 XSCL, YSCL, XXSCRL, YYSCRL
COMMON /CONCOM/ NCOL, NROW, BMIN, BMAX,
  1 GRAD, FGT, IJE, IJS, IJE1, IJS1, IJE2, IJ4, CONT,
  2 FLTMAX, LMULT(0/3), IDASH, LINET

C NCOL=0
IPASS=0
10 OPEN<UNIT=IFILE, DEVICE='DSK', MODE='BINARY'>,
  1 ACCESS='SEQIN', FILE=FNAME>
  IF (IPASS.EQ.0) GO TO 20
  READ(IFILE)
  IF (DELY.EQ.0.) READ(IFILE)
  RETURN

C
20 IPASS=1
  READ(IFILE, END=90) TITLE1, DUMMY, NCOL, NROW,
  1 IQUAD, XD, DELX, YD, DELY
  IF (NCOL.LT.3. OR. NROW.LT.3) GO TO 110
  IF (IQUAD.EQ.3. AND. DELX.EQ.0. .AND. DELY.EQ.0.)
  1 GO TO 80
  IF (IQUAD.NE.1) GO TO 130
  IF (DELY.NE.0.) GO TO 30
  IF (NCOL.GT.NWORK) GO TO 150
  READ<IFILE, END=90> (XDATA(I), I=1, NCOL)
  XX(1)=XDATA(1)
  XX(2)=XDATA(NCOL)
  GO TO 40
30 XX(1)=XD
  XX(2)=XD+(NCOL-1)*DELX
  40 IF (DELY.NE.0.) GO TO 50
  CALL YREAD<IFILE, YY, NROW>
  IF (NROW.LT.3) GO TO 110
  GO TO 60
50 YY(1)=YD
  YY(2)=YD+(NROW-1)*DELY
  60 IF (DELX.NE.0. .AND. DELY.NE.0.) RETURN
70 CLOSE<UNIT=IFILE>
  GO TO 10
80 I=NCOL*3
  IF (I.GT.NWORK) GO TO 150
  READ<IFILE>
  CALL XYREAD<IFILE, XX, YY, XDATA, I>
IF (NROW.LT.3) GO TO 110
   GO TO 70
90 TYPE 100
100 FORMAT(' %END OF FILE WHILE PROCESSING HEADER/')
   GO TO 170
110 TYPE 120
120 FORMAT(' %NO. ROWS OR COLUMNS LESS THAN 3')
   GO TO 170
130 TYPE 140
140 FORMAT(' %NO. Z ARGUMENTS GREATER THAN 1')
   GO TO 170
150 TYPE 160
160 FORMAT(' %CORE EXCEEDED FOR X OR ROW VALUES')
C
170 CLOSE(UNIT=IFILE)
   NCOL=0
   RETURN
END
SUBROUTINE XYREAD(IFILE, XR, YR, WORK, NCOL3)

C SCANS QUADRILATERAL GRID FOR X-Y RANGE

DIMENSION XR(2), YR(2), WORK(NCOL3)
COMMON /CONCOM/ NCOL, NROW, BMIN, BMAX,
1 GRAD, GFLG, IJS, IJE, IJSI, IJEI, PRIME, I4, CONT,
2 FLTMAX, LMULT(0/3), IDASHS, LINET

C INITIALIZE
NROW=0
READ(IFILE, END=70) DUMMY, WORK
XMIN=FLTMAX
YMIN=FLTMAX
XMAX=-FLTMAX
YMAX=XMAX

C SCAN LOOP
10 NROW=NROW+1
   IF (WORK(I+2).EQ.GFLG) GO TO 50
   DO 50 I=1, NCOL3, 3
   IF (WORK(I).LE.XMAX) GO TO 20
   XMAX=WORK(I)
   GO TO 30
20 IF (WORK(I).GE.XMIN) GO TO 30
   XMIN=WORK(I)
30 IF (WORK(I+1).GE.YMIN) GO TO 40
   YMIN=WORK(I+1)
   GO TO 50
40 IF (WORK(I+1).LE.YMAX) GO TO 50
   YMAX=WORK(I+1)
50 CONTINUE

READ(IFILE, END=60) DUMMY, WORK
GO TO 10

60 XR(1)=XMIN
XR(2)=XMAX
YR(1)=YMIN
YR(2)=YMAX

C 70 RETURN
END
SUBROUTINE YREAD(IFILE,YR,NROW)
C
C SCANS SPECIFIED ROW GRID
   DIMENSION YR(2)
C
   NROW=0
   READ(IFILE,END=20) YR(1)
C
   10 NROW=NROW+1
   READ(IFILE,END=20) A
   YR(2)=A
   GO TO 10
20 RETURN
   END
SUBROUTINE COORDA(X, Y)
DIMENSION X(0:0), Y(0:0)

C
C ESTABLISH LOCAL X AND Y ADDRESSES
RETURN
ENTRY COORD(I, J, XV, YV)
XV=X(I)
YV=Y(J)
RETURN
END
SUBROUTINE COORDA(X, Y, NCOL)

C QUADRILATERAL COORDINATES
C DIMENSION X(0:0), Y(0:0)
C RETURN

C ENTRY COORDA(I, J, XV, YV)
K=J*NCOL + I
XV=X(K)
YV=Y(K)
RETURN
END
SUBROUTINE SEXUAL(IER)
C
C GENERAL SETUP OF CONTOUR NICITIES.
C
DOUBLE PRECISION TITLE1,TITLE2,TITLE3,FMTX,FMTY
COMMON /SEXY/
1 TITLE1(7),TITLE2(8),TITLE3(8),FMTX(2),FMTY(2),
2 XX(2),YY(2),XP(4),YP(4),IPLDTR,SIZEL,NCHARX,NCHARY,
3 SIZEX,SIZEY,ADELX,ADELY,PLLX,PLLY,LINTX,LINTY,
4 XSCALE,YSSCALE,XXSCL,YYSCL
C
IF (SIZEX.LT.0.) SIZEX=0.
IF (SIZEY.LT.0.) SIZEY=0.
IF (NCHARX.LT.0) NCHARX=0.
IF (NCHARY.LT.0) NCHARY=0.
IF (ADELX.LT.0.) ADELX=0.
IF (ADELY.LT.0.) ADELY=0.
IF (LINTX.LT.0) LINTX=0.
IF (LINTY.LT.0) LINTY=0.
IF (SIZEL.LT.0.) SIZEL=0.
SIZELB=1.5*SIZEL
C
C MARGIN REQUIREMENT SETUP.
 IF (SIZEX.GT.0.) AND. NCHARX.GT.0) GO TD 10
 NCHARX=0.
10 IF (SIZEY.GT.0.) AND. NCHARY.GT.0) GO TD 20
 NCHARY=0.
20 IF (PLLY.LE.0) ) GO TD 30
 XP(3)=PLLY
 GO TD 40
30 XP(3)=(NCHARY+0.6)*SIZEY
40 IF (PLLY.LE.0.) GO TD 50
 YP(3)=PLLY
 GO TD 60
50 YP(3)=(NCHARY+0.6)*SIZEX+4.0*SIZELB
C
60 DX=XX(2)-XX(1)
 DY=YY(2)-YY(1)
 CALL PLTSET(IPLDTR,XP(4),YP(4),0)
C
C CHECK SCALING
 IF (XSCALE.LE.0) ) GO TD 70
 IF (YSSCALE.LE.0) ) YSCALE=XSCALE
 GO TD 80
70 IF (YSSCALE.LE.0) ) GO TD 90
 XSCALE=YSSCALE
80 XXSCAL=SIGN(1./XSCALE,DX)
 YYSCL=SIGN(1./YSSCALE,DY)
C
C FIXED SCALING
 XP(1)=ABS(DX*XXSCL)
 YP(1)=ABS(DY*YYSCL)
CONTOURING SYSTEM.

Appendix B. Listing of CONUR.

\[
\begin{align*}
XP(4) &= XP(1) + XP(3) + 0.501 * SIZEX \\
YP(4) &= YP(1) + YP(3) + 0.501 * SIZEY \\
& \text{GO TO 120}
\end{align*}
\]

C

C RELATIVE SCALING. EG. XXSCAL=YYSCAL=0.
\[
\begin{align*}
90 & \quad XP(4) = \text{AMIN1}(XP(4), 10.) \\
& \quad YP(4) = \text{AMIN1}(YP(4), 8.) \\
& \quad XP(1) = XP(4) - XP(3) - 0.501 * SIZEX \\
& \quad YP(1) = YP(4) - YP(3) - 0.501 * SIZEY \\
& \quad \text{IF} \ (XP(1) \ \text{GT.} \ 0. \ \text{AND.} \ YP(1) \ \text{GT.} \ 0.) \ \text{GO TO 110}
\end{align*}
\]

110 GO TO 140

100 FORMAT(\' %MARGIN REQUIRES ALL PLOT AREA\')

100 FORMAT(\' %MARGIN REQUIRES ALL PLOT AREA\')

110 FORMAT(\' %MARGIN REQUIRES ALL PLOT AREA\')

C

C INITIRL LABELEING SCALE CALL

120 CALL SCALE(XX, YY, XP, YP, 4, IER)

130 CALL SCALE(XX, YY, XP, YP, 4, IER)

140 RETURN

END
SUBROUTINE CONTR (X, Y, Z, F, C, NROW, NCOL, XXSCAL, YYSCAL,)
  1 GFLG, YD, DEIY, IOUAD, NPASS, JRES, IFILE)

C BASIC INPUT AND TIER CONTROL
C
DIMENSION X(1), Y(1), Z(1), F(1), C(1)

C EXTERNAL COORD, COORD

C SETUP MODE CONTROL
IF (IOUAD.EQ.3) GO TO 10
CALL COORDA (X, Y)
ITYPE = -1
IF (DEIY.EQ.0) ITYPE = 0
GO TO 20
10 CALL COORDA (X, Y, NCOL)
ITYPE = 1

C REMAINING INITIALIZATION
20 KLRDW = (NROW-1) * NCOL
LAST = 0
JE = NROW

C TIER LOOP
DO 190 IPASS = 1, NPASS
IF (IPASS.EQ.NPASS) JE = JRES
IF (IPASS.EQ.1) GO TO 70

C MOVE DOWN LAST ROW
JS = 2
JJ = NCOL
DO 30 I = 1, NCOL
30 Z(I) = Z(KROW + I)
IF (ITYPE) 40, 40, 50
40 Y(I) = Y(NROW)
GO TO 80
50 DO 60 I = 1, NCOL
X(I) = X(KROW + I)
60 Y(I) = Y(KROW + I)
GO TO 80

C FIRST ROW
70 JS = 1
JJ = 0

C GET DATA
80 DO 140 J = JS, JE
IF (ITYPE) 90, 100, 110
90 READ (IFILE, END = 130) DUM, (Z(JJ + I), I = 1, NCOL)
  Y(J) = YD
  YD = YD + DEIY
GO TO 140
100 READ (IFILE, END = 130) Y(J), (Z(JJ + I), I = 1, NCOL)
  Y(J) = Y(J) * YYSCAL
CONTOURING SYSTEM.

APPENDIX B. Listing of CONTUR.

```
GO TO 140
110 READ(IFILE,END=130) DUM,(X(JJ+I),Y(JJ+I),Z(JJ+I),
  1 I=1,NCOL)
   DO 120 I=1,NCOL
     KK=JJ+I
     IF (Z(KK).EQ.GFLG) GO TO 120
     X(KK)=X(KK)*XXSCAL
     Y(KK)=Y(KK)*YYSCAL
 120 CONTINUE
     GO TO 140
130 JE=J-1
   IF (JE.LT.2) RETURN
     LAST=1
   GO TO 150
140 JJ=JJ+NCOL
C
C CALL CONTUR
150 IF (ITYPE) 160,160,170
160 CALL CONTUR(Z,F,C,COORD,JE)
     GO TO 180
170 CALL CONTUR(Z,F,C,COORD,JJ)
180 IF (LAST.NE.0) RETURN
190 CONTINUE
     RETURN
END
```
SUBROUTINE CONTR(GRID, FLAGS, ACVAL, COORD, NROWF)

C BASIC CONTOURING CONTROL SUBROUTINE.

C DIMENSION GRID(1), FLAGS(1), ACVAL(1), COORD(1)
COMMON /CONTR/ CMIN, CMAX, DCVAL, NCVL, NSEC, GRADI
COMMON /CONCOM/ NCOL, NROW, BMIN, BMAX,
1 GRAD, GFLG, IJS, IJE, IJSI, IJIE, PRIME, IJ4, CONT,
2 FLTMAX, LMULT(0/3), IDASHS, LINET

C LOGICAL PRIME
C NROW=NROWF
C SET GRADIENT.
IF (NSEC.LE.0.OR.NCVL.GT.0) GO TO 10
GRAD=(GRADI+DCVAL)**2
C SET FLAGS, ETC..
10 CALL SETUP(GRID, FLAGS, COORD)
IF (BMIN.GE.BMAX) GO TO 150
IF (NCVL.EQ.0) GO TO 50
C FIND LOWER LIMIT.
DO 20 I=1, NCVL
IF (ACVAL(I).LE.BMIN) GO TO 20
II=I
GO TO 30
20 CONTINUE
C CONTOURING LOOP
30 PRIME=.TRUE.
DO 40 I=II, NCVL
CONT=ACVAL(I)
IF (CONT.GE.BMAX) GO TO 150
CALL SETLAB
40 CALL SCAN(GRID, FLAGS, COORD)
GO TO 150
C EXECUTION FOR DELTA CONTOUR LEVELS.
50 IF (CMIN.EQ.0 .AND. CMAX.EQ.0) GO TO 60
ICONT=AMAX1(BMIN, CMIN)/DCVAL
AMX=AMIN1(BMAX, CMAX)
GO TO 70
60 ICONT=BMIN/DCVAL
AMX=BMAX
C CONTOURING LOOP.
70 CONT=ICONT+DCVAL
IF (CONT.GE.AMX) GO TO 150
PRIME=NSEC.GT.0 .AND. MOD(ICONT, NSEC).EQ.0
IF (PRIME) GO TO 100
IF (IDASHS) 80,140,90
CONTOURING SYSTEM.
APPENDIX B. LISTING OF CONTUR.

80 LINET=1
    GO TO 140
90 LINET=0
    GO TO 140
100 IF (IDASHS) 110,130,120
110 LINET=0
    GO TO 130
120 LINET=1
130 CALL SETLAB
140 CALL SCAN(GRID,FLAGS,COORD)
    ICONT=ICONT+1
    GO TO 70

C
C
C DONE WITH BLOCK
150 RETURN
END
SUBROUTINE SETUP (GRID, FLAGS, COORD)
C
C INITIALIZE FLAG ARRAY.
C
DIMENSION GRID(0:0), FLAGS(0:0)
INTEGER FLAGS
C
COMMON /CONCOM/ NCOL, NROW, BMIN, BMAX,
1 GRAD, GFLG, IJS, IJE, IJSI, IJEI, PRIME, IJ4, CONT,
2 FLTMAX, LMUL<(0/3), IDASHS, LINET
C
LOGICAL PRIME
C
C INITIALIZATION OF ALL PARAMETERS (ARITHMETIC AND LOGICAL)
C UNIQUE TO ROW BLOCK BEING CONToured
C
C
C SYSTEM OF FLAGS
C
BIT DEFINITION (FOR BIT ON)
C
0 EDGE-SIDE 4 POSSIBLE CUT
C
1 EDGE-SIDE 3 POSSIBLE CUT
C
2 EDGE-SIDE 2 POSSIBLE CUT
C
3 EDGE-SIDE 1 POSSIBLE CUT
C
4 EDGE 1 CUT NOT MADE AND NOT CHECKED
C
5 INTERIOR SCAN (EDGE 1)
C
6 GRADIENT (DROP SECONDARY CONTOURS)
C
7 BLOCK GOOD (CONTourABLE)
C
INTEGER FG
LOGICAL CCC, COLLRC, ROWLG, BIT, KK, COLLG
C
C START OF OPERATIONS
C
NROW1=NROW-1
NCOL1=NCOL-1
I=0
I1J=1
I1J=I+1
I1J1=NCOL
I1J1=I1J1+1
I1J1=-NCOL
IJS=(NCOL+NROW+3)/4 -1
DO 10 I=0, IJS
10 FLAGS(I)=0
IJS="37777777777
IJE=0
C
C PRESET BMIN, BMAX
C
BMIN=FLTMAX
BMAX=-BMIN
C
C START OF SETUP SCANNING
C
DO 130 J=0, NROW1
C
ROWLG=.FALSE.
COLLR=.FALSE.
IF (GRAD.EQ.0 .OR. J.EQ.MROW1) GO TO 20
CALL COORD(0,J,XLL,YLL)
CALL COORD(0,J+1,XUL,YUL)
20 CONTINUE
DO 120 I=0,NCOL1
IF (J.EQ.0) GO TO 30
COLLG=BIT(FLAGS,IJH1,1)
GO TO 40
30 COLLG=.FALSE.
40 IF (I.EQ.NCOL1) GO TO 70
CCC=COLLG
IF (J.EQ.MROW1) GO TO 70
C
C DETERMINE IF MESH BLOCK FLAGGED
C
IF (GFLG NE 0 .AND.
1 (GRID(IJ).EQ.GFLG .OR. GRID(I1J).EQ.GFLG .OR.
2 GRID(I1J).EQ.GFLG .OR. GRID(I1J).EQ.GFLG)) GO TO 70
C
C MESH BLOCK CONTURABLE
C
FG=1
IJS=MINT(IJS,IJ)
IJE=MAX0(IJE,IJ)
C
C CHECK AND SET LEFT EDGE (2)
C
IF (ROWLG) GO TO 50
IF (GRID(IJ1).GT.GRID(IJ)) FG=FG+32
ROWLG=.TRUE.
C
C CHECK AND SET LOWER EDGE (1)
C
50 L=0
IF (.NOT.COLLG) L=L+4
IF (GFLG) L=L+4
FG=FG+L
C
C CHECK AND SET GRADIENT
C
IF (GRAD.EQ.0.) GO TO 60
CALL COORD(I+1,J,XLR,YLR)
CALL COORD(I+1,J+1,XUR,YUR)
IF ((GRID(IJ)-GRID(I1J))**2.GT.
1 (XLL-XLR)**2+(YLL-YLR)**2)**GRAD
2 .OR. (GRID(IJ)-GRID(I1J))**2.GT.
3 (XLL-XUL)**2+(YLL-YUL)**2)**GRAD
4 .OR. (GRID(I1J)-GRID(I1J1))**2.GT.
5 (XUL-XUR)**2+(YUR-YUL)**2)**GRAD
6 .OR. (GRID(I1J)-GRID(I1J1))**2.GT.
7 (XUR-XLR)**2+(YLR-YUR)**2)**GRAD
8 .OR. (GRID(IJ)-GRID(I1J1))**2.GT.
9    ((XUL-XLR)**2+(YUL-YLR)**2)*GRAD
1    .OR. (GRID(IJ)-GRID(IJ1))**2.GT.
2    ((XUL-XLR)**2+(YUL-YLR)**2)*GRAD) FG=FG+2
     XLL=XLR
     YLL=YL
     XUL=XUR
     YUL=YUR
C DETERMINE BMIN,BMAX
C
60 BMIB=AMIN1(BMIN,GRID(IJ))
    BMIBX=AMAXI(BMAX,GRID(IJ))
    L=IJ/4
    FLG(FLG(L)=FLG(L).OR.(FG+LMULT(IJ,AND.3))
    GO TO 110
C MESH BLOCK NON-CONTOURABLE
C
C CHECK AND SET SIDE 3 AND 4 FLAGS
C
70 KK=.FALSE.
    IF (.NOT.ROWLG) GO TO 80
    IF (GRID(IJ).LT.GRID(IJ1)) CALL SET(FLG,IJ-1,128)
    KK=.TRUE.
80 IF (.NOT.COLLG) GO TO 90
    IF (GRID(IJ).LT.GRID(IJ1)) CALL SET(FLG,ijn1,64)
    KK=.TRUE.
90 IF (.NOT.(COLLCR.OR.KK)) GO TO 100
    BMIB=AMIN1(BMIN,GRID(IJ))
    BMIBX=AMAXI(BMAX,GRID(IJ))
100 ROWLG=.FALSE.
    FG=0
C END OF MESH
C
110 IJ=IJ1
    IJ1=IJ1+1
    IJ1=IJ1+1
    COLLCR=CCC
    IJN1=IJN1+1
120 CONTINUE
C 130 CONTINUE
C
    IJS=IJS/4
    IJE=IJE/4
    IJS1=(IJS+NCOL)/4
    IJE1=IJE
    RETURN
C END
SUBROUTINE SCAN(GRID,FLAGS,COORD)
C
C SCANS GRID FOR UNDRAFTED CONTOUR LEVEL.
C
DIMENSION GRID(0:0),FLAGS(0:0),COORD(1)
INTEGER FLRGS

CDMCDN, CDNCDM, NCDL, NRDW, BMIN, BMAX,
1 GRID, GFLG, IJS, IJE, IJSI, IJEI, PRIME, IJ4, CONT,
2 FLTMAX, LMULT(0:3), IDASHS, LINET

LOGICAL PRIME
C
C SET INTERIOR FLAGS.
IF (IJSI.LT.0) GO TO 20
DO 10 IJ=IJSI, IJEI
10 FLRGS(IJ)=FLRGS(IJ)+1
C
C SETUP FOR EDGE SCAN
C
20 IF (IJS.LT.0) GO TO 180
ASSIGN 40 TO ISWA
ASSIGN 160 TO ISWB
EDGE=.TRUE.
IJ4=IJS
IJB=IJE
IJSNEW=-1
MASK="360360360360"
MASKB="360"
MASKD="20"
MASKF=1
30 IJ4B=IJ4+4+3

C
C BASIC WORD SCAN
C
DO 170 IJ=IJ4, IJB
LFLAG=FLAGS(IJ)
IF ((LFLAG.AND.MASK).EQ.0) GO TO ISWB,(160,150)
IF (IJSNEW.LT.0) IJSNEW=IJ
IJENEW=IJ
IJ4=IJ4B
IJ4N=IJ4+NCOL
C
C SUB WORD SCAN
C
DO 140 K=1,4
IF ((LFLAG.AND.MASKB).EQ.0) GO TO 130
GO TO ISWA,(40,110)
C
C RIGHT EDGE
C
40 IF ((LFLAG.AND."2000").EQ.0) GO TO 60
IF (CONT.LE.GRID(IJ4+1)) GO TO 50
CALL RESET(FLGS, IJ4, "200")
GO TO 80
50 IF (CONT.LE.GRID(IJ4N+1)) GO TO 60
CALL TRACE(GRID, FLGS, COORD, 4, MASKF)
GO TO 80

C TOP EDGE
C
60 IF ((LFLG.AND."100") EQ 0) GO TO 80
IF (CONT.LE.GRID(IJ4N+1)) GO TO 70
CALL RESET(FLGS, IJ4, "100")
GO TO 100
70 IF (CONT.LE.GRID(IJ4N)) GO TO 80
CALL TRACE(GRID, FLGS, COORD, 3, MASKF)
GO TO 100

C LEFT EDGE
C
80 IF ((LFLG.AND."40") EQ 0) GO TO 100
IF (CONT.LE.GRID(IJ4N)) GO TO 90
CALL RESET(FLGS, IJ4, "40")
GO TO 130
90 IF (CONT.LE.GRID(IJ4)) GO TO 100
CALL TRACE(GRID, FLGS, COORD, 2, MASKF)
GO TO 130

C BOTTOM EDGE AND INTERIOR SCAN
C
100 IF ((LFLG.AND."20") EQ 0) GO TO 130
110 IF (CONT.LE.GRID(IJ4)) GO TO 120
CALL RESET(FLGS, IJ4, MASKF)
GO TO 130
120 IF (CONT.GT.GRID(IJ4+1))
   1 CALL TRACE(GRID, FLGS, COORD, 1, MASKF)
C
130 LFLG=LFLG/512
   IJ4=IJ4-1
   IJ4N=IJ4N-1
140 CONTINUE
C
GO TO 160
C
INTERIOR RANGE SET
C
150 IF ((LFLG.AND."004004004004") EQ 0) GO TO 160
IF (IJSNEW.LT.0) IJSNEW=IJ
   IJNEW=IJ
160 IJ4B=IJ4B+4
C
170 CONTINUE
C
FIRST TIME?
C
   IF (.NOT.EDGE) GO TO 190
C UPDATE EDGE RANGE
C
IJS=IJSNEW
IJE=IJENEW
180 IF (IJSI.LT.0) GO TO 200
C
C SETUP FOR INTERIOR SCAN
C
EDGE=.FALSE.
ASSIGN 110 TO ISWA
ASSIGN 150 TO ISWB
IJSNEW=-1
IJA=IJSI
IJB=IJJEI
MASK="010010010010"
MASKB="010"
MASKD="4"
MASKF=MASKB
GO TO 30
C
C UPDATE INTERIOR BOUNDARY INDEXES
C
190 IJSI=IJSNEW
    IJEI=IJENEW
C
C END OF SCAN
C
C
200 RETURN
C
END
SUBROUTINE TRACE(GRID,FLAGS,COORD,ISIDE,MASK)

C FOLLOWS CONTOUR THROUGH GRID UNTIL
C EDGE OR CLOSURE FOUND.
C
DIMENSION GRID(0:0),FLAGS(0:0)
INTEGER FLRGS
COMMON /CDNDMx NCOLi-NRDW* BM, BMflX?
GRRD . GFLG* IJS* IJE
PRIME, IJ4, CONT,
FLTMAX, LMULT(0:3), IDASHS, LINET

C LOGICAL PRIME
C
C TRACE AND PLOT CONTOUR THROUGH GRID
C
C GRID INDEXING
C SIDE 3
C I01 + + I11
C SIDE 2 SIDE 4
C I00 + + I10
C SIDE 1
C
C IN-- ENTRRNCE SIDE
C HIGH-EQUAL POINT(INN) ALWAYS ON LEFT WHEN LOOKING ALONG
C CONTOUR LINE
C IOL-- POINT OPPOSITE ENTRRNCE SIDE ON LEFT
C IOR-- POINT OPPOSITE ENTRRNCE SIDE ON RIGHT
C
DIMENSION X(100), Y(100)
C
C SET UP TRACING START
C
LOGICAL BIT, POST
POST=PRIME
I=MOD(IJ4, NCOL)
J=IJ4/NCOL
NPTS=1
IC=0
I00=IJ4
I10=I00+1
I01=I00+NCOL
I11=I01+1
GO TO 10, 20, 30, 40, 50

10 CALL RESET FLAGS, I00, "10"
FRACT=CONT-GRID(I00))/(GRID(I10)-GRID(I00))
CALL COORD(I, J, XA, YA)
CALL COORD(I+1, J, XB, YB)
IOL=I01
IOR=I11
IN=1
GO TO 50
CONTOURING SYSTEM.
APPENDIX B. LISTING OF CONTUR.

20 FRACT=〈CONT-GRID〈100〉/〈GRID〈101〉-GRID〈100〉〉
CALL COORD〈I, J, XA, YA〉
CALL COORD〈I, J+1, XB, YB〉
IDL=I11
IDR=I10
IN=2
GO TO 50

30 FRACT=〈CONT-GRID〈101〉/〈GRID〈111〉-GRID〈101〉〉
CALL COORD〈I, J+1, XA, YA〉
CALL COORD〈I+1, J+1, XB, YB〉
IDL=I11
IDR=I10
IN=3
GO TO 50

40 FRACT=〈CONT-GRID〈110〉/〈GRID〈111〉-GRID〈110〉〉
CALL COORD〈I+1, J, XA, YA〉
CALL COORD〈I+1, J+1, XB, YB〉
IDL=I10
IDR=I00
IN=4

C
C START TRACE LOOP
C

50 X〈NPTS〉=XA+〈XB-XA〉*FRACT
Y〈NPTS〉=YA+〈YB-YA〉*FRACT
IF (〈.NOT.PRIME.AND.F.〈〈PRMS〉, 100, 2〉〉 GO TO 80
NPTS=NPTS+1
IF 〈NPTS.LE.100〉 GO TO 90
IF 〈POST〉 GO TO 60
CALL LINE〈X, Y, 100, IC, LINET〉
POST=PRIME
GO TO 70

60 CALL LABEL〈X, Y, 100, IC, POST〉
70 NPTS=1
IC=1
GO TO 90

80 IF 〈NPTS.EQ.1〉 GO TO 90
CALL LINE〈X, Y, NPTS, IC, LINET〉
IC=0
NPTS=1

C
C DETERMINE EXIT SIDE
C

90 K=IN+2
IF 〈GRID〈IDR〉.LT.CONT〉 K=K-1
IF 〈GRID〈IDL〉.LT.CONT〉 K=K-2
IF 〈K-IN〉 110, 100, 120

C
C SADDLE DECISION (Daghoff)
C

100 IF 〈〈GRID〈100〉+GRID〈110〉+GRID〈101〉+GRID〈111〉〉
1 *0.25.LT.CONT〉 GO TO 120
K=IN+2
GO TO 120
Contouring System.

Appendix B. Listing of Contur.

110 K=IN
C
C COMPUTE SIDE BRANCH
C
120 GO TO (140, 150, 160, 130, 140, 150), K
C
C EXIT BOTTOM -- SIDE 1
C
130 FRACT=(CONT-GRID(I10))/(GRID(I11)-GRID(I10))
   CALL COORD(I, J, XA, YA)
   CALL COORD(I+1, J, XB, YB)
   IF (J .EQ. 0) GO TO 170
   I01=I10
   I00=I10-NCOL
   IF (.NOT.BIT(FLAGS, I00, 1)) GO TO 170
   IN=3
   I11=I10
   I10=I10-NCOL
   I01=I11
   IOR=I00
   J=J-1
   GO TO 50
C
C EXIT LEFT -- SIDE 2
C
140 FRACT=(CONT-GRID(I00))/(GRID(I01)-GRID(I10))
   CALL COORD(I, J, XA, YA)
   CALL COORD(I+1, J, XB, YB)
   IF (I .EQ. 0) GO TO 170
   I10=I01
   I00=I00-1
   IF (.NOT.BIT(FLAGS, I00, 1)) GO TO 170
   IN=4
   I11=I01
   I01=I01-1
   I01=I00
   IOR=I01
   I=I-1
   GO TO 50
C
C EXIT TOP -- SIDE 3
C
150 FRACT=(CONT-GRID(I01))/(GRID(I11)-GRID(I01))
   CALL COORD(I, J+1, XA, YA)
   CALL COORD(I+1, J+1, XB, YB)
   IF (.NOT.BIT(FLAGS, I01, "MASK")) GO TO 170
   CALL RESET(FLG$/I01,"10"
   IN=1
   I00=I01
   I10=I11
   I01=I01+NCOL
   I11=I11+NCOL
   I01=I01
   IOR=I11
J=J+1  
GO TO 50  
C  
C EXIT RIGHT -- SIDE 4  
C  
160 FRACT=(GRID(I11)-GRID(I10))/GRID(I10)  
CALL COORD(I+1,J,XA,YA)  
CALL COORD(I+1,J+1,XB,YB)  
IF (.NOT.BIT(FLAGS,110,1)) GO TO 170  
IN=2  
I00=I10  
I01=I11  
I10=I10+1  
I11=I11+1  
I0L=I11  
I0R=I10  
I=I+1  
GO TO 50  
C  
C END OF TRACE LOOP  
C  
170 X(NPTS)=XA+(XB-XA)*FRACT  
Y(NPTS)=YA+(YB-YA)*FRACT  
IF (NPTS.LE.1.AND.IC.EQ.0) GO TO 190  
C  
C DUMP RESIDUAL X-Y TO PLOTTER  
C  
IF (POST) GO TO 180  
CALL LINE(X,Y,NPTS,IC,LINET)  
GO TO 190  
180 CALL LABEL(X,Y,NPTS,IC,POST)  
190 RETURN  
C  
C END OF TRACING  
C  
END
SUBROUTINE LABEL(X,Y,NPTS,ICC,POST)
C
C LABEL SCAN CONTOUR LINE ARRAY LOOKING FOR
C A STRAIGHT SEGMENT.
C IF FOUND, THE CONTOUR IS LABELLED.
C
DIMENSION X(1),Y(1),FMT(1)
LOGICAL POST
COMMON /CONCOM/ NCOL,NROW,BMIN,BMAX,
1 GRAB,GFLG,IIJ,IIJE,IJS,IJEI,PRIME,II4,CONT,
2 FLTMAX,LMULT(0/3),IDASHS,LINET
C
LOGICAL PRIME
DOUBLE PRECISION CHARST(3)
LOGICAL NOLAB
EQUIVALENCE (F,SUM),(ISA,YA)
IF (NOLAB) GO TO 50
C
C SETUP AND SCAN
C
IS=2
ISA=3
SUM=0.0
IN1=2
IE=NPTS-1
IF (IE.LT.3) GO TO 50
XB=X(IS)
YB=Y(IS)
DO 40 I=ISA,IE
SUM=SUM+SQRT((X(I)-X(IN1))**2+(Y(I)-Y(IN1))**2)
10 DIST=(X(I)-XB)**2+(Y(I)-YB)**2
IF (DIST.LT.WDIST) GO TO 30
DIST=SQRT(DIST)
IF ((SUM/DIST).GT.1.02) GO TO 20
IE=I
GO TO 60
20 IS=IS+1
SUM=SUM-SQRT((X(IS)-XB)**2+(Y(IS)-YB)**2)
XB=X(IS)
YB=Y(IS)
IF (IS.LT.I) GO TO 10
30 IN1=I
40 CONTINUE
C
C CAN'T FIND SPOT, CONTINUE LINE WITHOUT LABELING.
50 CALL LINE(X,Y,NPTS,ICC,LINET)
   GO TO 70
C
C PLOTTABLE LOCATION.
60 CALL LINE(X,Y,IS,ICC,LINET)
   XA=X(IE)-X(IN1)
   YA=Y(IE)-Y(IN1)
   F=1.-DIST/WDIST/SQRT(XA*XA+YA*YA)
   XA=F*XA+X(IN1)
CONTOURING SYSTEM.

APPENDIX B. LISTING OF CONTUR.

```
YA=F*YA + Y(IN1)
PHI=ATAN2(YA-YB,XA-XB)
IF (ABS(PHI).GT.1.5707963)
  PHI=PHI-SIGN(3.1415927,PHI)
CALL CHAR(0.5*(XA+XB),0.5*(YA+YB),CHARST,NCHF,
  1 2,SIZE,PHI,CCOR,0.0)
CALL LINE(XA,YA,1,0,LINET)
CALL LINE(X(IE),Y(IE),NPTS-IE+1,1,LINET)
POST=.FALSE.

C 70 RETURN
C
C ENTRY TO SET LABELING CONSTANTS.
C MUST BE CALLED BEFORE SETLAB.
ENTRY SETCON(FMT,SIZ,NCHAR)
  NCHAR=NCHAR
  SIZE=SIZ
  NOLAB=.NOT.(NCHAR.GT.0.AND.SIZE.GT.0.)
RETURN
C
C ENTRY TO ESTABLISH LABELING CHARACTER STRING.
C MUST BE CALLED BEFORE LABEL.
ENTRY SETLAB
  IF (NOLAB) GO TO 80
  ENCODE(30,FMT,CHARST) CONT
  NCHF=NCHAR
  CALL LFJUST(CHARST,NCHF)
  WDIST=(NCHF+1)*SIZE
  WDIST2=WDIST*WDIST
  CCOR=-.5*WDIST+SIZE
80 RETURN
END
```
LOGICAL FUNCTION BIT(LF,I,MASK)  

C RETURNS TRUE VALUE IF ANY BIT(S) OF LF ARRAY MATCH MASK BITS.

C

DIMENSION LF(0:0)
COMMON /CONCOM/ NCOL,NROW,BMIN,BMAX,
1 GRAD,GFLG,IJS,IJE,IJSI,IJEI,PRIME,IIJ,CONT,
2 FLTMX,LMULT(0:3),IDASHS,LINET

C

LOGICAL PRIME
BIT=(LF(I/4)/LMULT(I,AND,3)).AND.MASK).NE.0
RETURN
END
SUBROUTINE SET(LF,I,MASK)

C SETS BITS OF LF TO SETTING OF MASK BITS.

C DIMENSION LF(0:0)
COMMON /CONCOM/ NCOL,NROW,BMIN,BMAX,
1 GRAD,GPQG,IJS,IJE,IJSI,IJEI,PRIME,IJ4,CONT,
2 FLTMAX,LMULT(0:3),IDASHS,LINET

C LOGICAL PRIME
K=I/4
LF(K)=LF(K).OR.(MASK*LMULT(I.AND.3))
RETURN
END
SUBROUTINE RESET(LF, I, MASK)
C
C RESETS BITS OF LF CORRESPONDING
C TO BITS IN MASK.
C
DIMENSION LF(0:0)
COMMON /CONCOM/ NCOL, NROW, EMIN, EMAX,
1 GRAD, GFLG, IJS, IJE, IJSI, IJEI, PRIME, IJ4, CONT,
2 FLTMAX, LMULT(0:0), IDASHS, LINET
C
LOGICAL PRIME
K=I/4
LF(K)=LF(K).AND.(MASK*LMULT(I,.AND.3))
1 .XOR."777777777777"
RETURN
END
CONTOURING SYSTEM.
APPENDIX B. LISTING OF CONTUR.

BLOCK DATA BLKCON
DOUBLE PRECISION TITLE1,TITLE2,TITLE3,FMTX,FMTY
COMMON /SEXY/
1 TITLE1(7),TITLE2(8),TITLE3(8),FMTX(2),FMTY(2),
2 XX(2),YY(2),XP(4),YP(4),IPLDTR,SIZEL,NCHAX,NCHARY,
3 SIZEX,SIZEY,ADELX,ADELY,PLLX,PLLX,LINTX,LINTY,
4 XSCL,YSCL,XXSCL,YYSCAL

COMMON /CONTRC/ CMIN,CMAK,DCVAL,NCVAL,NSEC,GRADI
COMMON /CONC/ VRL,NCVRL,NSEC,SIZEL,0.0
1 GRAD,GFLG,IJS,IJE,IJSL,IJEI,PRIME,IJ4,CONT,
2 FLTMX,LMULT(0/3),IDASHS,LINET

LOGICAL PRIME
DATA LMUL/134217728,262144,512,1/
DATA CMIN,CMAK,DCVAL,NCVAL,NSEC,SIZEL/3*0.,2*0,0.07/
DATA GRADI,GFLG,FLTMX/30.*2**37777777777777777777/
DATA TITLE2,TITLE3/16*'/
DATA IPLDTR,NCHAX,NCHARY,LINTX,LINTY/5*0/
DATA SIZEX,SIZEY,ADELX,ADELY,PLLX,PLLX,
1 XSCL,YSCL/8*0.7/
DATA XP(2),XP(2)/2*0.7/
END
LISTING OF PROGRAM SPLM2D

TWO DIMENSIONAL SPLINE INTERPOLATION SYSTEM

CODED BY
GERALD IAN EVENDEN
U. S. GEOLOGICAL SURVEY
DENVER FEDERAL CENTER
DENVER, COLORADO 80225

DIMENSION WORK(20000)

WHEN COMPILED ON THE DEC SYSTEM-10 WITH THE FORTRAN-10
COMPILER THE PROGRAM REQUIRES APPROXIMATELY
12 + NWORK/1000 K-WORDS OF MEMORY.

TO COMPILE A VERSION TO HANDLE LARGER GRIDS, MERELY
CHANGE THE VALUE OF NWORK AND THE DIMENSION OF WORK.

NOTE... THIS PROGRAM CONTAINS SEVERAL NON-ANSI FORTRAN
STATEMENTS. ALTHOUGH SEVERAL OF THE CONSTRUCTS MAY BE
AVAILABLE ON OTHER COMPUTER SYSTEMS, CARE MUST BE TAKEN
IN THE TRANSPORTATION OF THIS PROGRAM.

NAMELIST /PARMS/ IDELX, IDELY, XOFF, YOFF, XDEL, YDEL
1, NCOLD, NROW0, IDER
DOUBLE PRECISION TITLE, PGMID
COMMON TITLE(7), PGMID, NCOL, NROW, NZ, X0, IDELX
1, Y0, IDEL, IYP, IUP, IIP, ISP, IWP, IRP, ID, IOUT
2, IDVAL, NFX, NFY, NROWD, NCOL1, NROW1, IDER
EQUIVALENCE (NFX, IDELX), (NFY, IDELY)

DOUBLE PRECISION NAME, BLANK
DIMENSION INAME(3)
EQUIVALENCE (INAME(1), NAME)

DATA NWORK/20000/
DATA INAME(3), BLANK/0,
DATA ITTY/1/

IDVAL=337777777777
ID=10
IDUT=11
IDER=0

OPEN (UNIT=ITY, DEVICE='TTY', MODE='ASCII')

10 WRITE(ITY, 20)
20 FORMAT(' ENTER INPUT FILE NAME,EXT: ',$)
    READ(ITY,30) NAME
30 FORMAT(A10)
    IF (NAME.NE.BLANK) GO TO 60

C
C GET FROM SAVE AREA
    WRITE(ITY, 280)
    READ(ITY, 30) NAME
    IF (NAME.NE.BLANK) GO TO 50
    WRITE(ITY, 40)
40 FORMAT('''NO ORIGINAL OR SAVE FILE GIVEN''')
    GO TO 490
50 ISP=-1
    GO TO 70

C
C GET NEW INPUT
60 IF (NAME.EQ.'EXIT ') GO TO 490
    ISP=0
70 OPEN(UNIT=10,DEVICE='DSK',ACCESS='SEQIN',
       MODE='BINARY',FILE=NAME)
    READ(IO,END=470) TITLE,PGMID,NCOL,NROW,NZ,
       DELX,DELY
    IF (ISP.GE.0) GO TO 80
    IF (PGMID.EQ.'SPLCOEFF' .AND. NZ.EQ.16) GO TO 110
    GO TO 90
80 IF (NZ.EQ.1) GO TO 110
90 WRITE(ITY,100)
100 FORMAT('''INVALID INPUT FILE''')
    GO TO 490

C
C ALLOCATE MEMORY
110 IYP=NCOL+1
    IF (ISP.LT.0) GO TO 140
    IUP=IYP+NROW
    NXY=NCOL*NROW
    IPP=IUP+NXY
    IPP=IPP+NXY
    ISP=IP+NXY
    IWP=ISP+NXY
    IF (IWP+6*MAX0(NCOL,NROW).LE.NWORK) GO TO 150
120 WRITE(ITY,130)
130 FORMAT('''INPUT TOO LARGE''')
    GO TO 490
140 IUP=IYP+2
    NXY=16*NCOL
    IWP=IUP+NXY
    IRP=IWP+NXY
    GO TO 160

C
150 IRP=IWP+NCOL+16
160 IF (IRP.GE.NWORK) GO TO 120
    IF (DELX.NE.0.) GO TO 170
    READ(IO,END=470) (WORK(I), I=1, NCOL)
    GO TO 190
CONTOURING SYSTEM.

APPENDIX C. LISTING OF SPLN2D

170 F=0.
   DO 180 I=1,NCOL
      WORK(I)=X(I)+F*DELX
   180 F=F+1.

C INPUT GRID

190 IF (ISP.LT.0) GO TO 290
   J=IUP-1
   K=IYP
   DO 220 I=1,NROW
      READ(IO,END=200) WORK(K),(WORK(J+L),L=1,NCOL)
      GO TO 210
   200 NROW=I-1
      GO TO 230
   210 J=J+NCOL
      K=K+1
   220 CONTINUE

C

230 IF (NCOL.GT.2.AND.NROW.GT.2) GO TO 250
   WRITE(IO,240)
   240 FORMAT ('%NCOL OR NROW LESS THAN 3')
   GO TO 490

C GENERATE Y VALUES, IF NECESSARY

250 IF (DELY.EQ.0) GO TO 270
   F=0.
   J=IYP+NROW-1
   DO 260 I=IYP+1
      WORK(I)=Y(I)+F*DELY
   260 F=F+1.

C EVERYTHING LOADED, GENERATE COEFFICIENT ARRAYS

270 CLOSE(UNIT=IO)
   CALL SPLN2A(WORK(IUP),NCOL,NROW,DELX,DELY,
   1 WORK,WORK(IYP),IVAL,WORK(IIP),WORK(IQP),
   2 WORK(ISP),WORK(IWP))

C SAVE GOODIES.

280 WRITE(IO,280)
   280 FORMAT (' ENTER COEFFICIENT FILE NAME. EXT:','$')
   READ(IO,30) NAME
   IF (NAME.EQ.BLANK) GO TO 290
   OPEN(UNIT=IO,DEVICE='DSK',ACCESS='SEQOUT',
   1 MODE='BINARY',FILE=NAME)
   CALL PUTCOF(WORK,WORK(IYP),WORK(IWP),NCOL)
   CLOSE(UNIT=IO)
   GO TO 10

C GET OUTPUT FILE NAME

290 WRITE(IO,300)
   300 FORMAT (' ENTER INTERPOLATED FILE NAME. EXT:','$')
   READ(IO,30) NAME
   IF (NAME.EQ.BLANK) GO TO 10
   OPEN(UNIT=IDOUT,DEVICE='DSK',ACCESS='SEQOUT',
   1 MODE='BINARY',FILE=NAME)
   CALL SPLN2A(WORK(IUP),NCOL,NROW,DELX,DELY,
   1 WORK,WORK(IYP),IVAL,WORK(IIP),WORK(IQP),
   2 WORK(ISP),WORK(IWP))
   CALL PUTCOF(WORK,WORK(IYP),WORK(IWP),NCOL)
   GO TO 10
CONTOURING SYSTEM.

APPENDIX C. LISTING OF SPLN2D

1 MODE='BINARY', FILE=NAME>
PGMID='SPLINE-INT'
IDELX=0
IDELY=0
XOFF=0.
YOFF=0.
XDEL=0.
YDEL=0.
WRITE(ITYY,310)
310 FORMAT(' ENTER PARAMETERS:
READ(ITYY,F/AMS,END=450)
IF (XDEL.NE.0.) GO TO 360
IF (YDEL.NE.0.) GO TO 370
IF (IDELX.GT.0) GO TO 340
IF (IDELY.GT.0) GO TO 330
WRITE(ITYY,320)
320 FORMAT(' INVALID SUBDIVISION PARAMETERS'
GO TO 490
C
330 IDELX=IDELY
GO TO 350
340 IF (IDELY.LE.0) IDELY=IDELX
350 NCOLO=(NCOL-1)*IDELX+1
NROWD=(NROW-1)*IDELY+1
DELY=DELY/IDELX
IDELY=IDELY/IDELY
GO TO 400
C
C EQUAL GRID
360 IF (YDEL.NE.0) GO TO 380
YDEL=XDEL
GO TO 380
370 IF (XDEL.EQ.0) XDEL=YDEL
380 DELX=XDEL
DELY=YDEL
XO=XO+XOFF
YO=YO+YOFF
IDELX=0
IF (NCOLO.GT.0.AND.NROWD.GT.0) GO TO 400
WRITE(ITYY,390)
390 FORMAT(' NCOLO OR NROWD LESS THAN 1'
GO TO 440
C
C CHECK WORK OVERFLOW
400 IF (APP+NCOLO-1.LT.NWORK) GO TO 420
WRITE(ITYY,410)
410 FORMAT(' NO MEMORY FOR NEW GRID GENERATION'
GO TO 440
C
C WRITE HEADER
420 NZ=1
WRITE(IOUT) TITLE,PGMID,NCOLO,NROWD,NZ,XO,DELX,YO,DELY
NCOL1=NCOL-1
NROW1=NROW-1

CONTOURING SYSTEM.
APPENDIX C. LISTING OF SPLM2D

IF (IDELX.NE.0) GO TO 430
CALL DELGR(WORK,WORK(IYP),WORK(IRP),WORK(IWP),
1 WORK(IUP),NCOLO)
GO TO 440
430 CALL FINEGR(WORK,WORK(IYP),WORK(IRP),WORK(IWP)
1 WORK(IUP),NCOLO)
C
C DONE
440 CLOSE(UNIT=IOUT)
CLOSE(UNIT=IO)
GO TO 10
C
C
450 WRITE(ITY,460)
460 FORMAT(' %EOF ON NAMELIST READ')
GO TO 490
470 WRITE(ITY,480)
480 FORMAT(' %EOF ON HEADER OR COLUMN SPACING')
C
490 CLOSE(UNIT=ITY)
CLOSE(UNIT=IO)
CLOSE(UNIT=IOUT)
STOP
END
CONTOURING SYSTEM.
APPENDIX C. LISTING OF SPLN2D

SUBROUTINE FINEGR(X, Y, R, C, W, NCOLO)

C SUBDIVIDE EACH ORIGINAL GRID BLOCK INTO
C EQUAL SECTIONS.
DIMENSION W(1), X(1), Y(1), R(NCOLO), C(16, 1), B(4), DR(4)
C
DOUBLE PRECISION TITLE, PGMID
COMMON TITLE(7), PGMID, NCOL, NROW, NZ, XD, DELX,
1 YD, DELY, IYP, IUP, IPP, ISP, IWP, IRP, ID, IDUT
2, DVAL, NF, NY, NROW, NCOL, NROW1, NCOLO, IDR
C
IF (IDER) 10, 20, 30
10 ILS = 2
20 ILS = 1
30 ILS = 3
40 YFRAC = 1, FLOAT(NFY)
XFRAC = 1, FLOAT(NFX)
NFX1 = NFX - 1
NFY1 = NFY + 1
IF (DELX .NE. 0.) GO TO 70
C
C GENERATE AND WRITE X ARRAY, IF NECESSARY
MD = 2
R(1) = X(1)
DO 60 I = 1, NCOLO
XDEL = (X(I) + 1 - X(I)) * XFRAC
XY = XDEL
DO 50 J = 1, NFX
R(MD) = X(I) + XY
MO = MO + 1
XY = XY + XDEL
50 CONTINUE
60 CONTINUE
WRITE(IDUT) R
C
C SET R TO DVALS
70 DO 80 I = 1, NCOLO
80 R(I) = DVAL
C
C ROW LOOP
DO 320 JJ = 1, NROW1
JJ = JJ
IF (ISP .LT. 0) GO TO 90
CALL SPLN2B(C, JJ)
GO TO 100
90 CALL GETSPL(C, JJJ, Y, W, NCOLO)
100 YVA = Y(JJJ)
YDEL = (Y(JJJ + 1) - YVA) * YFRAC
YV = 0.
C
C ROW SUBDIVISION
DO 310 J = 1, NFY1
MO=2
MOE=1

C COLUMN LOOP
DO 300 I=1,NCOL1
IF (J.EQ.1) GO TO 260
IF (C(I).EQ.DVAL) GO TO 270
110 XDEL=(X(I+1)-X(I))#XFRAC
IF (IDER) 120,160,140
120 DO 130 L=1,4
130 DL(L)=-C(L+4,I)+YV#(2.0C(L+8,I)+3.0YV+C(L+12,I))
GO TO 160
140 DO 150 L=1,4
150 DL(L)=2.0C(L+8,I)+6.0YV+C(L+12,I)
160 DO 170 L=ILS,4
170 B(L)=(YV+C(L+12,I)+C(L+8,I))#YV+C(L+4,I)+YV+C(L,I)
IF (IDER) 180,200,190
180 IF (R(MOE).EQ.DVAL)
1 R(MOE)=-SORT(B(2)+2.0DR(1))
GO TO 210
190 IF (R(MOE).EQ.DVAL)
1 R(MOE)=-(2.0B(3)+DR(1))
GO TO 210
200 IF (R(MOE).EQ.DVAL) R(MOE)=B(1)
210 MOE=MO+NX
XY=0.

C FILL WITH VALUES
DO 250 L=MO,MOE
XY=XY+XDEL
IF (IDER) 230,220,240
220 RL(L)=(B(4)+XY+B(3)+XY+B(2)+XY+B(1))
GO TO 250
230 DRX=B(2)+XY+B(2)+XY+B(3)+XY+B(4))
DRY=DR(1)+XY+DR(2)+XY+DR(3)+XY+DR(4)
RL(L)=SORT(DRX+DRX+DRY+DRY)
GO TO 250
240 RL=-B(2)+DR(1)+XY+(6.0B(4)+DR(2)+XY+(3)
1 DR(3)+XY+DR(4))
250 CONTINUE
GO TO 290

C FIRST ROW CHECK
260 IF (R(MOE).EQ.DVAL).AND.C(I).NE.DVAL) GO TO 110
MOE=MO+NX
GO TO 290

C INSERT DVALS
270 MOE=MO+NX
DO 280 L=MO,MOE
280 RL=DVAL
290 MOE=MOE+1
300 CONTINUE
IF (J.EQ.NFY1) GO TO 310
WRITE (IDOUT) YVA, R
IF (J.NE.NFY1) R(1)=DVAL
YV=YV+YDEL
YVA=YVA+YDEL
310 CONTINUE
320 CONTINUE
WRITE (IDOUT) YVA, R
RETURN
END
SUBROUTINE DEG3R(X,Y,R,C,W,NCOLO)
C
C SUBDIVIDE GRID INTO EQUALLY SPACED GRID.
C
DIMENSION W(1),X(1),Y(1),R(NCOLO),C(16,1),B(4),DR(4)
C
DOUBLE PRECISION TITLE,PGMID
COMMON TITLE(7),PGMID,NCOLO,NROW,NZ,XO,DELX,
1 YO,DELY,IYP,IP,IPP,ISP,IWP,IRP,ID,IDOUT
2_NVAL,NFX,NFY,NROW0,NCOL1,NROW1,IDER
C
IF (IDR) 10,20,30
10 ILS=2
   GO TO 40
20 ILS=1
   GO TO 40
30 ILS=3
40 JJ=1
   JJJ=1
   IF (ISP.LT.0) CALL GETSPL(C,JJJ,Y,W,NCOL)
   FY=0.
   C
   ROW LOOP
   DO 280 J=1,NROWD
   YVA=YD+FY*DELY
   C
   CHECK Y RANGE
   IF (DELY.LT.0.) GO TO 60
   IF (YVA.LT.Y(JJJ)) GO TO 250
50 IF (YVA.LE.Y(JJJ+1)) GO TO 80
   IF (JJ.GE.NROW1) GO TO 250
     JJ=JJ+1
     JJJ=JJ
   IF (ISP.LT.0) CALL GETSPL(C,JJJ,Y,W,NCOL)
   GO TO 50
60 IF (YVA.GT.Y(JJJ)) GO TO 250
70 IF (YVA.GE.Y(JJJ+1)) GO TO 80
   IF (JJ.GE.NROW1) GO TO 250
     JJ=JJ+1
     JJJ=JJ
   IF (ISP.LT.0) CALL GETSPL(C,JJJ,Y,W,NCOL)
   GO TO 70
   C
   SET COEFFICIENTS FOR ROW
80 YV=YVA-Y(JJJ)
   IF (ISP.GE.0) CALL SPLN2B(C,JJ)
     IL=0
     II=1
     FX=0.
   C
   COLUMN LOOP
   DO 240 I=1,NCOLO
   XVA=XO+FX*DELX
   C
C CHECK RANGE
  IF (DELX.LT.0.) GO TO 100
  IF (XVA.LT.X(I)) GO TO 220
  90 IF (XVA.LE.X(I+1)) GO TO 120
  IF (XVR.LE.X(I+1)) GO TO 220
  II=II+1
  GO TO 90
  100 IF (XVA.GT.X(I)) GO TO 220
  110 IF (XVR.GE.X(I+1)) GO TO 120
  IF (XVR.GE.X(I+1)) GO TO 220
  II=II+1
  GO TO 110

C CHECK EMPTY CELL
  120 IF (C(1,II).EQ.DVRL) GO TO 220

C GENERATE B IF NEW CELL
  IF (II.EQ.ID) GO TO 180
  DO 130 L=1,4
  130 B(L)=C(L+4,II)+YV+C(L+8,II)+YV+
    1 C(L+4,II)+YV+C(L,II)
  IL=II
  IF (IDER) 140,180,160
  140 DO 150 L=1,4
  150 DR(L)=C(L+4,II)+YV+(2.0 C(L+8,II)+3.0 YV+C(L+12,II))
  GO TO 180
  160 DO 170 L=1,4
  170 DR(L)=2.0 C(L+8,II)+6.0 YV+C(L+12,II)

C GENERATE R VALUE
  180 XV=XVA-X(I)
  IF (IDER) 200,190,210
  190 R(I)=(B(4)+XV+B(3)*XY+B(2))*XY+B(1)
  GO TO 230
  200 DRX=B(2)+XV+B(3)+XV*B(4)
  DRY=DR(1)+XY+DR(2)+XY+DR(3)+XY+DR(4))
  R(I)=SORT(DRX+DRX+DRY)
  GO TO 230
  210 R(I)=(B(3)+DR(1)+XY+6.0 B(4)+DR(2)+XY+B(4))
    1 DR(3)+XY+DR(4))
  GO TO 230

C FILL WITH DVRL
  220 R(I)=DVRL
  230 FX=FX+1.
  240 CONTINUE

C GENERATE DVRL ROW
  250 DO 260 I=1,NCOLD
  260 R(I)=DVRL
  270 WRITE(IDOUT) YVA,R
  FY=FY+1.
280 CONTINUE
C
C ALL DONE
   RETURN
   END
SUBROUTINE SPLN2A(U,NX,NY,DX,DY,X,Y,DVAL,P,Q,S,W)
C      BY W.L. ANDERSON, U.S. GEOLOGICAL SURVEY,
C      DENVER, COLORADO
C----GENERALIZED TWO-DIM. BICUBIC SPLINE COEFFICIENT
C DETERMINATION WHERE FIRST ENTRY (CALL SPLN2A( ... ))
C COMPUTES ARRAYS P,Q,S NEEDED BY SECOND ENTRY
C (CALL SPLN2B( ... ))--WHICH IN TURN
C COMPUTES THE BICUBIC SPLINE COEFFICIENTS FOR ANY
C SELECTED GRID ROW.
C
C===============================================================
C----FIRST ENTRY (SPLN2A) PARAMETERS:
C  U      = INPUT GRID ARRAY DIMENSIONED U(NX,NY)--ALSO
C            SEE DVAL PARM.
C  NX     = NO. COLUMNS IN ARRAYS U,P,Q,S (NX.GT.2)
C  NY     = NO. ROWS IN ARRAYS U,P,Q,S (NY.GT.2)
C  DX     = EQUAL X-INTERVAL OPTION WHEN DX.GT.0.
C            (USE DUMMY ARRAY X)
C  = 0.0 WHEN UNEQUAL X-VARIABLE ARRAY X IS
C            SUPPLIED (SEE BELOW)
C  DY     = EQUAL Y-INTERVAL OPTION WHEN DY.GT.0.
C            (USE DUMMY ARRAY Y)
C  = 0.0 WHEN UNEQUAL Y-VARIABLE ARRAY Y IS
C            SUPPLIED (SEE BELOW)
C  X      = X-VARIABLE ARRAY DIMENSIONED X(NX)--USED ONLY
C            IF DX=0.0
C            ARRAY X MUST BE IN ASCENDING ORDER WHEN DX=0.0
C  Y      = Y-VARIABLE ARRAY DIMENSIONED Y(NY)--USED ONLY
C            IF DY=0.0
C            ARRAY Y MUST BE IN ASCENDING ORDER WHEN DY=0.0
C  DVAL   = FLAG FOR MISSING DATA--WHEN ANY
C            U(IX,IY),P,IY).Q,IY),S,IY) GE DVAL.
C            E.G. USE DVAL=1.0E38
C  P,Q,S  = INTERMEDIATE PARTIAL DERIV. ARRAYS DIMN'D
C            P(NX,NY),Q(NX,NY),S(NX,NY)--OUTPUT FROM ENTRY
C            SPLN2A AND INPUT TO ENTRY SPLN2B.
C  W      = WORK VECTOR DIMENSIONED W(6*MAX0(NX,NY))
C
C----SUBROUTINE "SPLN1" AND "DVALS" CALLED BY SPLN2A.
C  (NOTE: SPLN2A MUST BE CALLED BEFORE ANY CALLS TO SPLN2B)
C
C===============================================================
C----SECOND ENTRY (SPLN2B) PARAMETERS:
C  ENTRY SPLN2B(U,NX,NY,DX,DY,X,Y,DVAL,P,Q,S,C,IY)
C  U,NX,NY,DX,DY,X,Y,DVAL,P,Q,S -- SAME DEFINITIONS AS
C  FOR SPLN2A --
C  C            = OUTPUT BICUBIC COEFF ARRAY DIMENSIONED
C(16,NX)
C  CORRESPONDS TO SELECTED RECTANGLE ROW IY.
C  NOTE: C(I,IX)=DVAL IF ANY CORNER OF RECTANGLE
C  HAS U=DVAL.
C -------- C(I,IX) IS GIVEN FOR
C  ROW IY WHERE I=1,16 FOR EACH IX=1,NX-1
C  ALSO C(I,NX) IS USED AS WORKING STORAGE, I=1,16.
CONTOURING SYSTEM

APPENDIX C. LISTING OF SPLN2D

C IY = SELECTED ROW INDEX TO COMPUTED C ARRAY C
C 1.6E.IY.LE.NY-1).
C
C--SUBROUTINE "GMPRD" CALLED BY SPLN2D
C (GMPRD IS AN IBM SSP-ROUTINE).
C--ALSO CALLED ARE SUBROUTINES "SETRX" AND "SETRY"
C (AS REQUIRED)
C
C=================================================================
C--NOTE: NO CHECKS ARE MADE ON CALLING PARAMETERS.
C THEREFORE THE USER SHOULD INSURE ALL PARAMETERS MEET
C SPECIFICATIONS AS DESCRIBED ABOVE.
C
C DIMENSION U(NX, NY), P(NX, NY), Q(NX, NY), S(NX, NY),
1 RX(16), AY(16), AA(16),
2 X(1), Y(1), W(1), D(2), C(16, NX)
DATA AX, AY/1.0, 4*0., 1., 10*0.,
1 1.0, 4*0., 1., 10*0./
KD=0
D(1)=0.0
D(2)=0.0
M=MAX0(NX, NY)
M1=1+M
M2=M+M1
M3=M+M2
M4=M+M3
M5=M+M4
DO 20 J=1, NY
DO 10 I=1, NX
10 W(I)=U(I, J)
 CALL DVALS(W, NY, DVRL)
 CALL SPLIN(W, Y, W, W(1), W(2), W(M2), W(M3), KD,
 1 D, W(M4), W(M5))
20 CONTINUE
DO 50 I=1, NX
DO 30 J=1, NY
30 W(J)=U(I, J)
 CALL DVALS(W, NY, DVRL)
 CALL SPLIN(W, Y, W, W(1), W(2), W(M2), W(M3), KD,
 1 D, W(M4), W(M5))
 DO 40 L=1, NY
40 Q(I,L)=W(M+L)
50 CONTINUE
 CALL SPLIN(W, Y, W, W(1), W(2), W(M2), W(M3), KD,
 1 D, W(M4), W(M5))
 CALL SPLIN(W, Y, W, W(1), W(2), W(M2), W(M3), KD,
 1 D, W(M4), W(M5))
 KD=1
DO 80 I=1, NX
DO 60 J=1, NY
60 W(J)=P(I, J)
 D(1)=S(I, 1)
 D(2)=S(I, NY)
 CALL SPLIN(W, Y, W, W(1), W(2), W(M3), KD,
CONTOURING SYSTEM.

APPENDIX C. LISTING OF SPLN2D

1 DO, W(M4), W(M5)
   DO 70 L=1, NY
70 $ (I, L)= W (M+L)
80 CONTINUE
RETURN
C$ Entry SPLN2B $$$$$$
ENTRY SPLN2B (C, IY)
   IF (KD.EQ.0) GO TO 90
C-- KD=1 (AFTER CALL SPLN2A) TO INITIIZE SPLN2B:
   KD=0
   NX1=NX-1
   IF (DX.NE.0.0) CALL SETAX (AX, DX)
   IF (DY.NE.0.0) CALL SETAY (AY, DY)
90 IY1=IY+1
   IF (DY.EQ.0.0) CALL SETAY (AY, Y(IY1)-Y(IY))
   DO 110 I=1, NX1
      I1=I+1
      IF (U(I, IY).GE.DYAL.OR.U(I1, IY).GE.DYAL.OR.U(I, IY1)
         .GE.DYAL.OR.U(I1, IY1).GE.DYAL) GO TO 100
      IF (DX.EQ.0.0) CALL SETAX (AX, X(I1)-X(I))
      AA(1)=U(I, IY)
      AA(2)=P(I, IY)
      AA(3)=U(I1, IY)
      AA(4)=P(I1, IY)
      AA(5)=Q(I, IY)
      AA(6)=S(I, IY)
      AA(7)=Q(I1, IY)
      AA(8)=S(I1, IY)
      AA(9)=U(I, IY1)
      AA(10)=P(I, IY1)
      AA(11)=U(I1, IY1)
      AA(12)=P(I1, IY1)
      AA(13)=Q(I, IY1)
      AA(14)=S(I, IY1)
      AA(15)=Q(I1, IY1)
      AA(16)=S(I1, IY1)
      CALL GMPRD (AX, AA, C(1, NX), 4, 4, 4)
      CALL GMPRD (C(1, NX), AY, C(1, I), 4, 4, 4)
   GO TO 110
100 C(I, I)=DYAL
110 CONTINUE
RETURN
END
SUBROUTINE DVALS(Y,LIM,DVAL)
C--PRESETS ROW<OR COLUMN> VECTOR CONTAINING ANY DVAL'S
   DIMENSION Y(1)
   IF(Y(1).GE.DVAL) GO TO 30
10 DO 20 I=2,LIM
   IF(Y(I).LT.DVAL) GO TO 20
   Y(I)=Y(I-1)
   IF(I.EQ.LIM) GO TO 20
   IF(Y(I+1).LT.DVAL) Y(I)=Y(I+1)
20 CONTINUE
RETURN
30 DO 40 I=2,LIM
   IF(Y(I).GE.DVAL) GO TO 40
   Y(I)=Y(I)
   GO TO 10
40 CONTINUE
RETURN
END
SUBROUTINE SETAX(AX, DD)
C--SET ROWS 2,3 OF ARRAY AX(16) GIVEN SPACING DD
DIMENSION AX(16)
DX1 = 1.0/DD
DX2 = DX1 + DX1
DX3 = DX2 + DX1
AX(3) = -3.0 + DX2
AX(4) = 2.0 + DX3
AX(7) = -2.0 + DX1
AX(8) = DX2
AX(11) = -AX(3)
AX(12) = -AX(4)
AX(15) = -DX1
AX(16) = DX2
RETURN
END
SUBROUTINE SETAY(AY, DD)
C--SET COLUMNS 2,3 OF ARRAY AY(16) GIVEN SPACING DD
DIMENSION AY(1)
D Y1 = 1.0/DD
DY2 = DY1 * DY1
DY3 = DY2 * DY1
AY(9) = -3.0 * DY2
AY(10) = -2.0 * DY1
AY(11) = -AY(9)
AY(12) = -DY1
AY(13) = 2.0 * DY3
AY(14) = DY2
AY(15) = -AY(13)
AY(16) = DY2
RETURN
END
SUBROUTINE SPLIN1(M,N,X,Y,A,B,C,T,D,P,S)
C---ONE DIMENSIONAL CUBIC SPLINE INTERPOLATION
C
C BY W.L.ANDERSON, U.S. GEOLOGICAL SURVEY,
C DENVER, COLORADO
C
C PARMS--- M= NUMBER OF DATA POINTS .GT. 2
C H= EQUAL INTERVAL OPTION WHEN H.GT.0.
C <USE DUMMY X HERE>,
C UNEQUAL INTERVALS IF H=0.
C <X REQUIRED STORAGE>
C X= INDEPENDENT VARIABLE <DIM .GE. M>.
C Y= DEPENDENT VARIABLE <DIM .GE. M>.
C A,B,C=COEFFICIENT ARRAYS <EACH DIM .GE. M>
C RESULTS ARE RETURNED IN 1ST (M-1) ELEMENTS
C OF A,B,&C.
C ALSO USED AS WORK ARRAYS DURING EXECUTION.
C T= TYPE OF BOUNDARY CONDITION SUPPLIED IN D
C ARRAY.
C USE T=1 IF 1ST DERIVATIVES GIVEN AT END
C POINTS, OR T=0 IF 2ND DERIVATIVES GIVEN
C AT END POINTS.
C D= BOUNDARY ARRAY <DIM 2> AT POINT 1 AND M
C RESPECTIVELY.
C P,S= WORK ARRAYS <EACH DIM=M>.
C--ERROR RETURN WITH M=-<ABS(M)> IF ANY PARAM OUT OF RANGE.
C THE RESULTING CUBIC SPLINE IS OF THE FORM:
C Y=Y(I)+A(I)*X-X(I)+B(I)*(X-X(I))^2+C(I)*(X-X(I))^3
C FOR I=1,2,...,M-1
C
REAL X(1),Y(1),A(1),B(1),C(1),D(2),P(1),S(1),MUL
INTEGER T
IF(T.LT.0.OR.T.GT.1.OR.M.LT.3) GO TO 190
M=M-1
IF(T.EQ.0) GO TO 130
C--1ST DERIVATIVE BOUNDARIES GIVEN
NE=M-1
IF(H) 10,80,10
C--EQUAL SPACING H .GT. 0. AND T=1
10 HH=3.0/H
DO 20 I=1,NE
B(I)=4.0
C(I)=1.0
A(I)=1.0
20 P(I)=HH*(Y(I+2)-Y(I))
P(I)=P(I)-D(I)
P(NE)=P(NE)-D(2)
C--SOLUTION OF TRIDIAGONAL MATRIX EQ. OF ORDER NE
30 FA=1./B(1)
C(1)=C(1)*FA
P(1)=P(1)*FA
DO 40 I=2,NE
MUL=1.0/(B(I)-A(I)*C(I-1))
C(I)=MUL*C(I)
40 CONTINUE
CONTOURING SYSTEM.

APPENDIX C. LISTING OF SPLN2D

40 P(I)=MUL*(P(I)-A(I)*P(I-1))
C--OBTAIN SPLINE COEFFICIENTS
    A(NE+T)=P(NE)
    I=NE-1
50 A(I+T)=P(I)-C(I)*A(I+T+1)
    I=I-1
    IF(I. GE. 1) GO TO 50
    IF(T.EQ. 0) GO TO 60
    A(I)=D(I)
    A(M)=D(2)
60 IF(H.EQ. 0.) GO TO 110
    HH=1.0/H
    MUL=HH*(Y(I+1)-Y(I))
    B(I)=HH*(3.0*MUL-A(I+1)+2.0*A(I))
    70 C(I)=HH+HH*(-2.0*MUL+A(I+1)+A(I))
    RETURN
C--UNEQUAL SPACING H=0.. AND T=1
80 DO 90 I=1,N
90 S(I+1)=X(I+1)-X(I)
    DO 100 I=1,NE
    B(I)=2.0*(S(I+1)+S(I+2))
    C(I)=S(I+1)
    A(I)=S(I+2)
100 P(I)=3.0*(S(I+1)*S(I+2)+Y(I+2)-Y(I+1))+S(I+2)*2*
    Y(I+1)-Y(I))/S(I+1)*S(I+2))
    P(I)=P(I)-S(I)*D(I)
    P(NE)=P(NE)-S(N)*D(2)
    GO TO 30
110 DO 120 I=1,N
    HH=1.0/S(I+1)
    MUL=(Y(I+1)-Y(I))*HH*2
    B(I)=3.0*MUL-A(I+1)+2.0*A(I)*HH
120 C(I)=(-2.0*MUL+A(I+1)+A(I))*HH*HH
    RETURN
C--2ND DERIVATIVE BOUNDARIES GIVEN
130 NE=NE+1
    IF(H) 140,160,140
C--EQUAL SPACING H .GT. 0 AND T=0
140 HH=3.0/H
    DO 150 I=2,N
    B(I)=4.0
    C(I)=1.0
    A(I)=1.0
150 P(I)=HH*(Y(I+1)-Y(I-1))
    B(I)=2.0
    B(NE)=2.0
    C(I)=1.0
    C(NE)=1.0
    A(NE)=1.0
    P(I)=HH*(Y(2)-Y(1))-0.5*H*D(I)
    P(NE)=HH*(Y(M)-Y(N))+0.5*H*D(2)
    GO TO 30
C--UNEQUAL SPACING H=0 AND T=0
CONTOURING SYSTEM.

APPENDIX C. LISTING OF SPLNE2D

160 DO 170 I=1,N
170 $X(I+1)$=$X(I+1)$-$X(I)$
    N1=N-1
    DO 180 I=1,N1
    B(I+1)=2.0*$(X(I+1)+X(I+2))$
    C(I+1)$=$I+1$)
    A(I+1)$=$I+2$)
180 P(I+1)=$3.0*(X(I+1)**2+(Y(I+2)-Y(I+1))**2$)
    1 $(Y(I+1)-Y(I))$/$(X(I+1)+X(I+2))$
    B(1)=2.0
    B(NE)=2.0
    C(1)=1.0
    C(NE)=1.0
    A(NE)=1.0
    P(1)=$3.0*(Y(2)-Y(1))$/S(2) - 0.5*S(2)*D(I)
    P(NE)=$3.0*(Y(M)-Y(M'))$/S(M) + 0.5*S(M)*D(2)
    GO TO 30

190 M=-1ABS(M)
RETURN
END
C ROUTINE TO SAVE SPLINE COEFFICIENTS

SUBROUTINE PUTCDF(X,Y,C,NC)
DIMENSION C(16,NC),X(1),Y(1)
DOUBLE PRECISION TITLE,PGMID
COMMON TITLE(7),PGMID,NCOL,NROW,NZ,XD,DELX,
1 YD,DELY,IYP,IPP,IRP,ISP,IRP,ID,IDOUT
2 ,DVAL,NFX,NFY,NROWD,NCOL1,NROW1,IDER
DOUBLE PRECISION PGMID2
DATA NZA,PGMID2/16,'SPLCOEFF**'/

WRITE(ID) TITLE,PGMID2,NCOL,NROW,NZA,XD,DELX,YD,DELY
IF (DELX.EQ.0.) WRITE(ID) X
DO 40 I=1,NROW
IF (I.EQ.NROW) GO TO 10
CALL SPLN2I(C,I)
C(I,NCOL)=DVAL
GO TO 30
10 DO 20 J=1,NCOL
20 C(I,J)=DVAL
30 WRITE(ID) Y(I),C
40 CONTINUE
RETURN
END
C ROUTINE TO RETRIEVE SPLINE COEFFICIENTS
C
SUBROUTINE GETSPL(C,J,Y,WORK,NC)
DIMENSION C(16,NC),WORK(16,NC),Y(1)
DOUBLE PRECISION TITLE,PGMID
COMMON TITLE(7),PGMID,NCOL,NROW,NZ,XO,DELY,
1 YD,DELY,YP,UP,IFF,ISP,IP,IRP,ID,IDOUT
2 NCOL,NFX,NFY,NROW1,NCOL1,NROW1,IDER
C
IF (J.GT.1) GO TO 20
READ (ID,END=50) Y(1),C
10 READ (ID,END=50) Y(2),WORK
   J=1
   RETURN
20 DO 40 I=1,NCOL
   DO 30 J=1,16
30   C(J,I)=WORK(J,I)
40 CONTINUE
   Y(1)=Y(2)
   GO TO 10
C
50 WRITE (ITTY,60)
60 FORMAT ('%PREMATURE EOF ON COEFFICIENT FILE')
   STOP
   END
APPENDIX D.

LISTING OF PROGRAM SPLN2X

C SPLINE COEFFICIENT PROGRAM
C
C CODED BY
C   GERALD IAN EVENDEN
C   U. S. GEOLOGICAL SURVEY
C   DENVER FEDERAL CENTER
C   DENVER, COLORADO 80225
C
C COEFFICIENTS FOR GRID UP TO 500 ROWS AND COLUMNS
C CAN BE DETERMINED BY THIS PROGRAM. IF LARGER GRIDS
C CHANGE THE DIMENSIONS IN SUBROUTINE SPLN2X.
C
C FOR THE DEC-SYSTEM 10 MEMORY REQUIRED IS APPROXIMATELY...
C 10 + 26*K/1000 K-WORDS
C WHERE K IS THE DIMENSION OF THE SPLN2X ARRAYS.
C
C NOTE... THIS PROGRAM CONTAINS SEVERAL NON-ANSI FORTRAN
C STATEMENTS. ALTHOUGH SEVERAL OF THE CONSTRUCTS ARE
C AVAILABLE ON OTHER COMPUTER SYSTEMS, CARE MUST BE
C EXERCISED IN THE TRANSPORTATION OF THIS ROUTINE.
C
DOUBLE PRECISION NAME,FILES,PGM
COMMON TITLE(14),PGM,NCOL,NROW,NZ,
1 XD,DX,YD,DY,
2 FILES(5),NAMES(3),IVAL
DIMENSION HEAD(23)
EQUIVALENCE (HEAD(1),TITLE(1)),(NAME,NAMES(1))

C DOUBLE PRECISION BLANK
DATA BLANK//
C
10 TYPE 20
20 FORMAT('ENTER INPUT FILE NAME.EXT:/',$)
ACCEPT 30,FILES(1)
30 FORMAT(A10)
   NAME=FILES(1)
   IF (NAME.EQ.BLANK) GO TO 10
   IF (NAME.EQ.'EXIT') STOP
   OPEN(UNIT=10,FILE=NAME,DEVICE='DSK',
      1 ACCESS='SEQIN',MODE='BINARY')
   READ(10,END=80,ERR=80) HEAD
   IF (NCOL.LE.500.AND.NROW.LE.500) GO TO 60
   TYPE 40
40 FORMAT('NCOL OR NROW GREATER THAN 500')
50 CLOSE(UNIT=10)
   GO TO 10

C
60 IF (NZ.EQ.1) GO TO 100
   TYPE 70

70 FORMAT('< %NZ NOT EQUAL TO 1')
       GO TO 50
C
80 TYPE 90
90 FORMAT('< %INPUT EOF OR ERROR ON HEADER')
       GO TO 50
C
C INPUT LOOKS OK, PROCEED
100 TYPE 110
110 FORMAT('< ENTER COEFFICIENT FILE NAME, EXT:',$)
       ACCEPT 30,FILES(5)
       NAME=FILES(5)
       IF (NAME.EQ.BLANK) 60 TO 100
       CALL SPLN2X
       GO TO 10
       END
BLKSPX
DOUBLE PRECISION NAME, FILES, PGM
COMMON TITLE(14), PGM, NCOL, NROW, NZ,
  XD, DX, YO, DY,
2 FILES(5), NAMES(3), DVAL
DIMENSION HEAD(23)
EQUIVALENCE (HEAD(1), TITLE(1)), (NAME, NAMES(1))
! DATA FILES/000, 'LPS991.TMP', 'LPS992.TMP',
  'LPS993.TMP', 000/!
DATA DVAL/"377777777777777/
END
CONTOURING SYSTEM.

Appendix D. Listing of SPLN2X

C DISK STORAGE OVERLAY SPLINE COEFFICIENT ROUTINE.
C BASED ON EARLIER ANDERSON ROUTINE IN PROGRAM BIGGRID.
C THIS ROUTINE, HOWEVER, REQUIRES 3 (VS. 8) SCRATCH
C FILES AND 8*NROW+4*NCOL LESS READS AND WRITES.
C NOTE...IF DIMENSIONS ARE CHANGED, ENSURE ARRAYS VA AND CU
C HAVE ONE MORE WORD THAN BASIC SIZE. ONLY THE VALUES 500
C AND 501 SHOULD BE CHANGED.

SUBROUTINE SPLN2X
DOUBLE PRECISION NAME,FILES,PGM
1 XD,DX,YD,DY,
2 FILES<5>,NAMES<3>,DVRL
DIMENSION HEAD<23>
EQUIVRLENCE <HEAD(1),TITLE(1)>, <NAME,NAMES(1)>
C
DIMENSION V(500),VA(501),A(500),B(500),C(500),
1 W1(500),W2(500),DJ1(500),DJY(500),D(2),
2 X(500),Y(500),CD(16,500)
C
EQUIVRLENCE <V(1),VA(2)>
C
DIMENSION CU(501,2),CP(500,2),CQ(500,2),CS(500,2)
1 ,WC(500)
EQUIVRLENCE <VA(1),CU(1,1)>, <A(1),CU(1,2)>,
1 <B(1),CP(1,1)>, <C(1),CP(1,2)>,
2 <DJ1(1),CQ(1,1)>, <DJY(1),CQ(1,2)>,
3 <W1(1),CS(1,1)>, <W2(1),CS(1,2)>,
4 <CD(1,1),WC(1)>
NCOL1=NCOL+1
D(1)=0.
D(2)=0.
C
IF EQUI-X GENERATE X ARRAY OR INPUT.
IF (DX.NE.0.) GO TO 10
CALL GETV(10,2,X,NCOL)
GO TO 30
C
10 F=0.
DO 20 I=1,NCOL
X(I)=XD+F*DX
20 F=F+1.
C
GENERATE P ARRAY
30 DO 40 I=1,NROW
CALL GETV(10,2,VA,NCOL1)
C
SAVE Y VALUES
Y(I)=VA(I)
CALL DVRLS(Y,NCOL,DVRL)
CONTOURING SYSTEM.

APPENDIX D. LISTING OF SPLN2X

CALL SPLINI(NCOL,DX,X,Y,A,B,C,0,D,W1,W2)
CALL PUTV(11,I,A,NCOL)
40 CONTINUE

CLOSE(UNIT=10)
CLOSE(UNIT=11)

C Y ARRAY NEEDS TO BE GENERATED?
   IF (DY.EQ.0.) GO TO 60

C YES.
   F=0.
   DO 50 I=1,NROW
       Y(I)=YD+F*DY
   50 F=F+1.

C GENERATE Q ARRAY
60 DO 80 I=1,NCOL
   DO 70 J=1,NROW
      CALL GETV(10,J,YA,NCOL)
   70 A(J)=Y(I)
   CLOSE(UNIT=10)
   CALL DVALS(A,NROW,DVAL)
   CALL SPLINI(NROW,DY,Y,A,YV,B,C,0,D,W1,W2)
   CALL PUTV(12,I,YV,NROW)
   DJY(I)=V(NROW)
   DJ1(I)=V(I)
80 CONTINUE

C GENERATE CROSS DERIVATIVES
   CALL SPLINI(NCOL,DX,X,DJ1,A,B,C,0,D,W1,W2)
   DO 90 I=1,NCOL
90 DJ1(I)=A(I)
   CALL SPLINI(NCOL,DX,X,DJY,A,B,C,0,D,W1,W2)
   DO 100 I=1,NCOL
100 DJY(I)=A(I)

C GENERATE REMAINDER OF S ARRAY
   DO 120 I=1,NCOL
   DO 110 J=1,NROW
      CALL GETV(11,J,W1,NCOL)
   110 V(J)=W1(I)
   CLOSE(UNIT=11)
   D(1)=DJ1(I)
   D(2)=DJY(I)
   CALL SPLINI(NROW,DY,Y,A,YV,B,C,1,D,W1,W2)
   CALL PUTV(13,I,A,NROW)
120 CONTINUE

NZ=16
PGM=`SPLCOEFF++`
NAME=FILES(5)
OPEN (UNIT=14, FILE=NAME, MODE='BINARY',  
1 ACCESS='SEQUOIT', DEVICE='DSK')  
WRITE(14) HEAD  
IF (WX.EQ.0.0) CALL PUTV(14, 2, X, NCOL)  
CALL SPL2AX(X, Y, CO, NCOL)

C  
C INITIAL LOAD  
CALL GETV(10, 1, CU(1, 1), NCOL1)  
CALL GETV(11, 1, CP(1, 1), NCOL)  
CLOSE(UNIT=12)  
CLOSE(UNIT=13)  
DO 130 J=1, NCOL  
CALL GETV(12, J, WC, NROW)  
C(J, 1) = WC(1)  
CALL GETV(13, J, WC, NROW)  
CS(J, 1) = WC(1)  
130 CONTINUE

C  
C GENERATE SPLINE COEFFICIENTS  
DO 150 I=2, NROW  
K2 = MOD(I-1, 2) + 1  
K1 = MOD(I, 2) + 1  
CALL GETV(10, I, CU(1, K2), NCOL1)  
CALL GETV(11, I, CP(1, K2), NCOL)  
CLOSE(UNIT=12)  
CLOSE(UNIT=13)  
DO 140 J=1, NCOL  
CALL GETV(12, J, WC, NROW)  
C(J, K2) = WC(1)  
CALL GETV(13, J, WC, NROW)  
CS(J, K2) = WC(1)  
140 CONTINUE  
CALL SPL2BX(CU(2, K1), CP(1, K1), CO(1, K1), CS(1, K1),  
1 CU(2, K2), CP(1, K2), CO(1, K2), CS(1, K2), I-1)  
150 CONTINUE

C  
C LAST ROW OF DVALS REQUIRED  
DO 160 I=1, NCOL  
160 CO(I, 1) = DVAL  
CALL PUTV(Y(NROW), CO, 16, NCOL)  
CLOSE(UNIT=10)  
CLOSE(UNIT=11, DISPOSE='DELETE')  
CLOSE(UNIT=12, DISPOSE='DELETE')  
CLOSE(UNIT=13, DISPOSE='DELETE')  
CLOSE(UNIT=14)  
RETURN

END
SUBROUTINE SPL2BX(X,Y,C,NX)
DOUBLE PRECISION NAME,PGM
DIMENSION U(1),P(1),Q(1),S(1),UI(1),PI(1),
1 QI(1),SI(1)
COMMON TITLE(14),PGM,NCOL,NROW,NZ,
1 XQ,DX,YQ,DY,
2 FILES(5),NAMES(3),DVAL
DIMENSION HEAD(23)
EQUIVALENCE (HEAD(1),TITLE(1)),(NAME,NAMES(1))
DIMENSION X(1),Y(1),C(16,NX)

C DIMENSION AX(16),AY(16),AA(16)
DATA AX,Ay/1.,4*0.,1.,10*0.,1.,4*0.,1.,10*0./
C
C ESTABLISH ONE TIME VALUES AND ADDRESSES.
C
NX1=NX-1
NX16=NX*16
IF(DX.NE.0.0) CALL SETAX(AX,DX)
IF(DY.NE.0.0) CALL SETAY(AY,DY)
RETURN
C
C GENERATE ROW OF SPLINE POLYNOMIAL
C COEFFICIENTS.
C
ENTRY SPL2BX(U,P,Q,S,UI,PI,SI,IY)
IYI=IY+1
IF(DY.EQ.0.0) CALL SETAY(AY,Y(IYI)-Y(IY))
DO 20 I=1,NX1
11=1+1
IF(U(I).EQ.DVAL.OR.U(11).EQ.DVAL.OR.
1 UI(I).EQ.DVAL.OR.UI(11).EQ.DVAL) GO TO 10
IF(DX.EQ.0.0) CALL SETAX(AX,X(I1)-X(I))
AA(1)=U(I)
AA(2)=P(I)
AA(3)=U(I1)
AA(4)=P(I1)
AA(5)=Q(I)
AA(6)=S(I)
AA(7)=O(I)
AA(8)=S(I1)
AA(9)=UI(I)
AA(10)=PI(I)
AA(11)=UI(I1)
AA(12)=PI(I1)
AA(13)=QI(I)
AA(14)=SI(I)
AA(15)=QI(I1)
AA(16)=SI(I1)
CALL GMPRD(AX,AA,C(1,NX),4,4,4)
CALL GMPRD(C(1,NX),AY,C(1,I),4,4,4)
GO TO 20
10 C(1,I)=DVAL
20 CONTINUE
C(1,NX)=DVAL
CALL PUTX(Y(IY)+C,NX16)
RETURN
END
SUBROUTINE GETV(IN,IREC,V,IDIM)
DIMENSION V(IDIM)
DOUBLE PRECISION NAME,FILES,PGM
COMMON TITLE(14),PGM,NCOL,NROW,NZ,
1 XO,DX,YD,DY,
2 FILES(5),NAMES(3),IVAL
DIMENSION HEAD(23)
EQUIVALENCE (HEAD(1),TITLE(1)),(NAME,NAMES(1))

GET ROUTINE FOR INPUTTING SPLINE DATA.

<table>
<thead>
<tr>
<th>IN</th>
<th>FILE NAME</th>
<th>DATA</th>
<th>RECORD SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>LP$991.TMP</td>
<td>P-ROWS</td>
<td>NCOL</td>
</tr>
<tr>
<td>12</td>
<td>LP$992.TMP</td>
<td>Q-COLS</td>
<td>NROW</td>
</tr>
<tr>
<td>13</td>
<td>LP$993.TMP</td>
<td>S-COLS</td>
<td>NROW</td>
</tr>
</tbody>
</table>

IF (IREC.NE.1) GO TO 10

FILE NEEDS TO BE REOPENED.
NAME=FILES(IN-9)
OPEN(UNIT=IN,FILE=NAME,MODE='BINARY',
1 ACCESS='SEQIN',DEVICE='DSK')
IF (IN.NE.10) GO TO 10
READ(IN,END=20,ERR=20)
IF (DX.EQ.0) READ (IN,ERR=20,END=20)

GET DATA
10 READ(IN,END=20,ERR=20) V
RETURN

ERROR CONDITION, LIGHT FUSE AND BOMB OUT.
20 TYPE 30
30 FORMAT(' \%I/O ERROR ON INPUT FILES')
STOP
END
SUBROUTINE PUTF(INOUT,IREC,V,IDIM)
DIMENSION V(IDIM)
DOUBLE PRECISION NAME,FILES,PGM,
COMMON TITLE(14),PGM,NCOL,NROW,NZ,
1 XD,YD,YO,DX
2 FILES(5),NAMES(3),DVAL
DIMENSION HEAD(23)
EQUIVALENCE (HEAD(1),TITLE(1)),(NAME,NAMES(1))

C OUTPUT ROUTINE FOR COEFFICIENT DATA.
IF (IREC.NE.1) GO TO 10

C FILE NEEDS TO BE REOPENED.
NAME=FILES(INOUT-9)
OPEN(UNIT=INOUT,FILE=NAME,MODE='BINARY',
1 ACCESS='SEQOUT',DEVICE='DSK')

C PUT DATA.
10 WRITE(INOUT) V
RETURN

C OUTPUT ROUTINE FOR POLYNOMIAL
C ROW.
C
ENTRY PUTF(Y,V,IDIM)
C
WRITE(14) Y,V
RETURN
END
APPENDIX E.

BASIC GRAPHICS SYSTEM.

The following user documentation is included to provide explanation of the graphics calls in program CONTUR. The graphics system is mostly in DEC System 10 assembly language and, as a consequence, is of little general interest.
INTRODUCTION.

Plotter software systems are provided to facilitate programmer usage of digital plotters in much the same manner as compiler language input-output statements facilitate usage of I/O devices. Such systems should attempt to achieve two, often mutually exclusive, goals: 1) simplicity and 2) flexibility. An additional feature should include the ability of the basic system to create machine instructions for more than one plotter and thus allow easy switching of plotting devices with minimal program recoding. This system provides for three plotters selected by one entry call and provides for all basic plotting operations with reasonable ease of usage.

This plotting system has four basic entries:

- **SCALE**: for plot initialization and scaling;
- **LINE**: for line drawing;
- **CHAR**: for character drawing;
- **ENDPLT**: for plot termination and diagnostics

and several supplementary service entries. The following description completely defines all entry parameters. A final section is included which contains a program example along with resultant plot. The reader should carefully read and understand the **SCALE** call description first as most fundamental definitions and conventions are described in this section.

Plot scaling.

The plotting system scaling call is required to initialize the software to the user and plotting machine requirements. In addition to opening plot output file and general system initialization a result of this call establishes two regions: 1) a "plot area" and 2) a "data area". The plot area defines the maximum limits of any plot operations. The plot area is, of course, limited to the physical dimensions of the plotting area of the machine employed. In plotting parameters referring to the plot area "plot units" are employed. Plot units have a default value of inches but can be redefined as centimetres with a call to PLTSET. The origin of the plot area coordinate system is always located at the lower-left hand corner of the plot area and is positive along the X or right hand axis and Y or vertical axis. Fig. 1 shows basic layout and general meaning of **SCALE** parameters.
Figure 1. SCALE geometric parameters.

The data area is always less than or equal to the plot area. Only data points whose X - Y values are in this area will be plotted. Data units are completely arbitrary and may be of any type: REAL, INTEGER, real or imaginary part of complex or double precision.

Scaling is performed by calling the entry SCALE:

CALL SCALE(DXP,DYP,XP,YP,NDPTS,ICODE).

When a successful SCALE call has been made the plotting system assumes a "scaled state". SCALE may be called again in a scaled state for redefining factors of the plot area. Termination of the scaled state and resultant completion of the plot is made by calling ENDFLT and a subsequent SCALE call will initialize a new plot. Plotting calls (i.e., LINE and CHAR) are only allowed when the system is in the scaled state.

SCALE parameter description.
DXP and DYP are two element arrays which define the data range. They may be of any standard FORTRAN IV type as defined in parameter XP and YP. DXP(1) and DYP(1) define the respective left and bottom limits of the data area in data units. DXP(2) and DYP(2) define the respective right and top limits of the data area in data units. For example, are most often the respective minimum and maximum X data values to be plotted. However, if a data axis is to be reversed (increasing to the left) the DXP(1) would be the maximum X and DXP(2) the minimum X. Of course, the same applies to DYP(1) and DYP(2). The only and obviou restriction is that DXP(1) .NE. DXP(2) and DYP(1) .NE. DYP(2).

XP and YP are type REAL arrays with NOPTS elements.

XP(1) and YP(1) define the respective X and Y size of the data area in plot units. Both must always be specified and greater than zero.

XP(2) and YP(2) is a switch to define the respective X and Y data type and whether a linear or logarithmic axis is to be plotted.

<table>
<thead>
<tr>
<th>XP(2), YP(2) VALUE</th>
<th>RESPECTIVE AXIS DATA TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;=1 or =8 or =9</td>
<td>REAL</td>
</tr>
<tr>
<td>=2 or 10</td>
<td>INTEGER</td>
</tr>
<tr>
<td>=3 or =11</td>
<td>REAL PART OF COMPLEX</td>
</tr>
<tr>
<td>=4 or =12</td>
<td>IMAGINARY PART OF COMPLEX</td>
</tr>
<tr>
<td>=5 or =13</td>
<td>DOUBLE PRECISION</td>
</tr>
</tbody>
</table>

When XP(2) or YP(2) are <8 then the respective axis are linear. When >= 8 then axis logarithmic. Note that XP(2) and YP(2) define the type for all data value references in LINE and CHAR calls. If NOPTS <2 then X and Y data values assumed to be REAL and linear.

XP(3) and YP(3) define the position of the lower-left corner of the data area on the plot area in plot units. If NOPTS<=2 then the lower-left edge of the data area is coincident with the lower-left edge of the plot area (XP(3)=YP(3)=0).

XP(4) and YP(4) define the size of the plot area in plot units. If NOPTS<=3 and this call made while not in the scaled state then the plot area will be computed by the following:

\[
XP(4) = XP(1) + XP(3)
\]
YP(4) = YP(1) + YP(3)

If XP(4) and YP(4) are specified then they must be greater than or equal to the values of the previous expression. XP(4) and YP(4) should not be specified for a SCALE call when the system is in the scaled state.

NOPTS is an integer value specifying the number of parameters in the XP-YP arrays. In all cases: 1 ≤ NOPTS ≤ 4 and when in the scaled state: NOPTS ≤ 3.

ICODE is an integer variable which is set to 0 upon return from SCALE if no fault could be found with the input parameters. If errors were detected a non-zero value is return and all subsequent plotting calls are disallowed except for another SCALE call. If appropriate user action is not taken and another SCALE call is made with the assumption of redefining the data area then unpredictable results can be expected.

Line plotting.

Plotting of lines is performed by calling the entry line:

CALL LINE(X,Y,N,ICON,IPN)

The LINE entry sequentially connects 2 or more X - Y data points with straight line vectors. LINE provides for continuation of lines generated by previous calls without concern for storing the last X - Y point of the previous call. Any part of the line inside the data area connecting two X - Y points is drafted regardless of one or both of the X - Y points being outside the data area. The drafted lines may also be in dashed form.

If smooth, curved lines are required the user is responsible for ensuring that the spacing of the points is adequate for the resulting straight line approximation.

Line parameter description.

X,Y coordinates in data units which define the sequence of line segments to be drawn. If N=1 then X-Y may be simple variables; else arrays of N elements.

N is an integer variable defining the number of X-Y points.

ICON is an integer variable which specifies whether the
LINE IS TO BE CONTINUED FROM THE LAST X-Y POINT OF A PREVIOUS CALL TO THE FIRST POINT OF THE CURRENT CALL. CONTINUATION ONLY APPLIES TO PREVIOUS CALL WITH IDENTICAL IPN PARAMETER VALUE. IF A PREVIOUS CALL WAS NEVER MADE THE LINE STARTS AT X(1) - Y(1). ICON=0 DO NOT CONTINUE; "NEW LINE". ICON.NE.0 CONTINUE LINE FROM PREVIOUS POINT.

IPN IS AN INTEGER VARIABLE WHICH SELECTS UP TO 8 PSEUDO PENS WHICH OPERATE INDEPENDENTLY OF EACH OTHER. NORMALLY, EACH PSEUDO PEN IS SET TO A UNIQUE MODE OF DRAFTING (SOLID, DASHED) FOR VISUAL IDENTIFICATION ON THE PLOT. THE DEFAULT MODE FOR EACH IPN IS:

<table>
<thead>
<tr>
<th>IPN VALUE</th>
<th>PLOTTING CHARACTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>_______________</td>
</tr>
<tr>
<td>1</td>
<td>- - - - - - - -</td>
</tr>
<tr>
<td>2</td>
<td>- - - - - - - -</td>
</tr>
<tr>
<td>3</td>
<td>- - - - - - - -</td>
</tr>
<tr>
<td>4</td>
<td>- - - - - - - -</td>
</tr>
<tr>
<td>5</td>
<td>- - - - - - - -</td>
</tr>
<tr>
<td>6</td>
<td>- - - - - - - -</td>
</tr>
<tr>
<td>7</td>
<td>- - - - - - - -</td>
</tr>
</tbody>
</table>

CHARACTER PLOTTING.

Plotting or posting of characters and symbols is performed by calling the entry CHAR:

CALL CHAR(X,Y,A,N,ICODE,SIZE,THETA,XOFF,YOFF).

The standard character set contains 101 gothic characters, including upper and lowercase alpha, numeric and punctuation and, optionally, the characters may be italicized. In addition, 13 special symbols usually used for point plotting are included. Note that special symbols cannot be italicized.

A call to CHAR may plot either a character or symbol at a number of X - Y locations or plot a string of characters or symbols starting at one X - Y location. The X - Y location, in the later case, may be either in plot units or data units.

CHAR PARAMETER DESCRIPTION.

X AND Y COORDINATES IN EITHER DATA OR PLOT UNITS WHICH DEFINE THE POSITION OF THE CENTER OF THE FIRST CHARACTER TO BE PLOTTED. NOTE THAT WHEN ICODE=0 OR 1 X AND Y ARE ARRAYS OF N ELEMENTS. WHEN ICODE<=2
X and Y are in data units and when ICODE=3 X and Y are in plot units.

A is a character or character string to be plotted. When ICODE=1 A contains one character. If the first character or byte of A is a binary zero then the character is selected from a right justified integer word. Otherwise the character is assumed to be left justified in A. When ICODE=1, A is a simple variable and the contained character is repeated for N X - Y points. When ICODE=0, A is an integer array of dimension N with a unique character plotted with each respective X(N) - Y(N) point. For ICODE=2 or 3, A is a character string and N is the number of characters in the string. X and Y are simple variables in this case.

N is an integer variable defining the number of points or characters to be plotted.

ICODE is an integer variable selecting the desired plotting definition of X, Y, A and italicizing. Basically, the two lowest bits of ICODE define the use and meaning of X, Y, A, and N as previously discussed. If 4 is added to ICODE the characters are italicized. The following is a summation of the effect of ICODE on each of the first four CHAR arguments:

<table>
<thead>
<tr>
<th>X and Y</th>
<th>A</th>
<th>N</th>
<th>ICODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA UNITS</td>
<td>ARRAY</td>
<td>DIMENSION</td>
<td>0 and 4</td>
</tr>
<tr>
<td>ARRAY</td>
<td>SIMPLE</td>
<td>VARIABLE</td>
<td>1 and 5</td>
</tr>
<tr>
<td>DATA UNITS</td>
<td>SIMPLE</td>
<td>STRING</td>
<td>2 and 6</td>
</tr>
<tr>
<td>ARRAY</td>
<td>CHARACTER</td>
<td>NUMBER</td>
<td>3 and 7</td>
</tr>
<tr>
<td>SIMPLE VAR</td>
<td>IN DATA UNITS</td>
<td>OF CHARACTERS</td>
<td></td>
</tr>
<tr>
<td>SIMPLE VAR</td>
<td>STRING</td>
<td>NUMBER</td>
<td></td>
</tr>
<tr>
<td>SIMPLE VAR</td>
<td>STRING</td>
<td>NUMBER</td>
<td></td>
</tr>
</tbody>
</table>

SIZE is a real variable defining the size of the characters to be plotted in plot units. Note that SIZE must be greater than zero.

THETA is a real variable defining the rotation, in radians, of the characters from the positive X axis about the X - Y point.

XOFF and YOFF are real variables which define a shift in plot units of the center of the first character...
from the specified $X - Y$ coordinates. Note that the shift is made prior to $\theta$ rotation.

Plot termination.

A call to ENDPLT is required when a plot is complete. This call performs the following function: 1) spills all output work buffer; 2) generates closure check ticks for incremental plotters; 3) prints plot diagnostics and 4) returns plotting software to an unscaled state. Any subsequent call to the plot system must be another SCALE call or a call to PLTIL or PLTSET. To execute ENDPLT:

CALL ENDPLT(0).

ENDPLT parameters.

The argument is not currently employed but is included for possible future expansion.

Plotter selection.

The graphics system entry PLTSET allows the user to select the plotter desired. If a call is not made to this routine the Gerber 622 is selected as a default. The plotter selected remains in effect until another PLTSET call is executed. In addition, the routine returns the physical size of the plot area and allows the user to select the type of plot units desired (inches or centimetres). To execute PLTSET:

CALL PLTSET(ICODE, XBOARD, YBOARD, INCM).

The call to PLTSET must be made while the plotting system is not in the scaled state.

PLTSET parameter description.

ICODE selects the plotting device.

ICODE=0 Gerber 622
ICODE=1 Tektronix 4010
ICODE=2 Hewlett-Packard 7202A
ICODE=3 Hewlett-Packard 7203A
Any other value will cause an error condition.

XBOARD and YBOARD are the returned $X$ and $Y$ axis size in plot units of the device selected. Principally employed in scaling the dimensions of the users plot to fit the device selected.
INCM SELECTS THE PLOT UNITS DESIRED.
INCM<=0 DOES NOT CHANGE PLOT UNITS
INCM=1 SELECTS INCHES AS PLOT UNITS
INCM>1 SELECTS CENTIMETRES AS PLOT UNITS.

NEAT LINE PLOTTING.

A RATHER TRIVIAL, BUT FREQUENTLY USED, ENTRY IS PROVIDED TO DRAFT A NEAT LINE AROUND THE DATA AREA. TO EXECUTE NEATLN:

CALL NEATLN

NEATLN PARAMETER DESCRIPTION.

No PARAMETERS.

AXIS LABELING.

FOR TYPE REAL DATA FOUR ROUTINES ARE PROVIDED FOR LINEAR OR LOGARITHMIC AXIS LABELING. THE CALLS FOR EACH ROUTINE ARE:

CALL XAXIS(DXP, DYP, XP, DEL, IP, SIZE, FMT, NFMT) FOR LINEAR X-AXIS LABELING,
CALL YAXIS(DYP, DXP, YP, DEL, IP, SIZE, FMT, NFMT) FOR LINEAR Y-AXIS LABELING,
CALL XAXISL(DXP, DYP, XP, SIZE, FMT, NFMT) FOR LOGARITHMIC X-AXIS LABELING, AND
CALL YAXISL(DYP, DXP, YP, SIZE, FMT, NFMT) FOR Y-AXIS LOGARITHMIC LABELING.

FOR ALL FOUR ENTRIES THE PARAMETERS HAVE IDENTICAL USAGE EXCEPT THAT DEL AND IP DO NOT APPLY TO THE LOGARITHMIC CALLS. AN IMPORTANT ITEM TO REMEMBER WHEN USING THE AXIS ENTRIES IS THAT THE PLOT AREA DEFINED BY SCALE SHOULD BE SUFFICIENTLY LARGER THAN THE DATA AREA TO ACCOMMODATE THE AXIS ANNOTATION.

PARAMETER DESCRIPTION.

DXP, DYP, XP AND YP ARE NORMALLY THE SAME PARAMETERS AS OCCURRING IN THE LAST SCALE CALL. THEY PROVIDE THE DATA RANGE AND PHYSICAL SIZE OF THE DATA AREA TO THE ROUTINES.

DEL IS THE INTERVAL (IN DATA UNITS) OF THE ANNOTATION TICK. IF DEL<=0 AND NFMT>0 THEN ONLY THE MAXIMA AND MINIMA OF THE DATA AREA ARE ANNOTATED.

IP IS THE INTERVAL BETWEEN EACH LABELING OF THE TICK MARKS. IF IP<=.1 THEN EACH TICK MARK LABELLED.
SIZE is the height (in plot units) of the characters used in annotating the axis.

FMT is an array containing an object time format character string. If NFMT is zero, FMT may be any dummy value.

NFMT is the number of resultant characters created by the "WRITE"ing of the FMT format. Restriction: NFMT.<=.20.

File control.

The user may change the logical device name of the output file by using the plotting system entry PLTFIL. Principal usage of this entry are for plots always requiring a specific plotter and creation of "flash" files. To execute PLTFIL:

CALL PLTFIL(DEV,FIL,EXT,NO).

PLTFIL may be called only while the plotting system is not in an scaled state.

PLTFIL parameter description.

DEV is from 1 to 6 ASCII characters to be used as a logical or physical device name of the plotting output file. Default logical output device name is PLOT.

FIL is from 1 to 3 ASCII characters to be employed as a directory device name prefix. If FIL=0 then "PLT" assumed.

EXT is from 1 to 3 ASCII characters to be employed as a directory device name extension. If EXT=0 then "DAT" assumed.

NO is file sequence number used as a 3 digit suffix to the file name. If NO<0 then sequence number not altered. The value is initially zero.

If output is to be a directory device (ie. DISK) the output file is generated with a default name of:

PLTxxx.DAT

where xxx (file sequence number) is initially zero and incremented by 1 for each new output file during the job. If the user is merely employing PLTFIL to force output to a single plotter dependent device like the TTY,
CALL PLTFIL('TTY',0,0,-1)

is adequate. Note that none of the plotter output devices is a directory device.

"Flash" plotting.

In production plotting, a portion of the plot may remain constant from one plot to another. It may be desirable, especially if the constant portion of the plot is complex, to create and save the plotter commands of this section and merely copy from this saved file to the actual plot file when needed. This process is defined here as "flashing".

Basically, the "flash" file is usually created as a separate job by extracting the constant portion of the plot generation code from the original program and, using PLTFIL, route the plotting output to disk. The original program is then modified by replacing the original constant plotting code with a call to the system copying routine FLASH. It should be noted that the flash file must be generated for the same plotter that is employed when the FLASH routine is called. To execute FLASH:

    CALL FLASH(DEV,FIL,EXT,NO).

FLASH parameter description.

The parameters have the same usage as described in the PLTFIL call.

Character string justification.

It is often desirable to determine the number of characters and remove leading and trailing blanks from an unknown character string before plotting. Typical usage would be programmatic input of string data for titling which will centered on the plot. The routine LFJUST is provided for this purpose and is executed as:

    CALL LFJUST(STRNG,N)

On entry STRNG is a string of N characters and upon return the characters in STRNG are left justified (no leading blanks) and N is the number of characters excluding trailing blanks.
Example Program.

This is a simple program to demonstrate several of the common character and line plotting methods and the scaling of the plot to an arbitrary object time selected plotter. The resultant plot is shown in Fig. 2.

```
DIMENSION XD(2), YD(2), XS(4), YS(4), XX(0/49), Y(0/49)
DATA XD, YD/0., 6.2831853, -1., 1./
DATA XS, YS/2*0., 1., 3*0., 1., 0./
C
C REQUEST PLOTTER NUMBER
  TYPE 10
  10 FORMAT(\' ENTER PLOTTER NO.\'=\',\$)
  ACCEPT 20, IPLTR
  20 FORMAT(I)
C
C SELECT PLOTTER AND GET BOARD SIZE
  CALL PLTSET(IPLTR, XBD, YBD, 0)
C
C RELATIVE SCALING
  XS(4) = AMIN1(6., XBD)
  YS(4) = AMIN1(6., YBD)
  XS(1) = XS(4) - 1.1
  YS(1) = YS(4) - 1.1
C
C SCALING CALL
  CALL SCALE(XD, YD, XS, YS, 4, ICODE)
C
C QUIT IF ERROR
  IF (ICODE.NE. 0) GO TO 40
C
C CREATE ARRAY OF SINE VALUES
  DO 30 I = 0, 49
    XX(I) = FLOAT(I)*.1282283
  30
C
C EXAMPLE OF CONTINUED LINE CALL
  CALL LINE(XX(I), COS(XX(I)), 1, 1, 1)
  30 Y(I) = SIN(XX(I))
C
C PLOT SINE CURVE ARRAY
  CALL LINE(XX, Y, 50, 0, 0)
C
C POST SINE CURVE POINTS
  CALL CHAR(XX, Y, 2, 50, 1., 0., 0., 0., 0.)
C
C LABEL AXIS
  CALL XAXIS(XD, YD, XS, 2.5, .09, \(F4.1)\', 4)
  CALL YAXIS(YD, XD, YS, 1., 10., .09, \(F4.1)\', 4)
  CALL NEATLN
  CALL CHAR((XD(1)+XD(2))*.5, YD(1), 'VALUE',
            1 5, 2., 12, 0., -.3, -.5)
  CALL CHAR((YD(1)+YD(2))*.5, 'AMPLITUDE',
            1 9, 2., 12, 1.5708, -.54, .5)
```
CALL CHAR(1.,1,‘PLOT EXAMPLE 1’,14,7,.15,
1 0.,0.,0.)

C
DONE PLOTTING
CALL ENDPLOT(0)
GO TO 60

C
40 TYPE 50
50 FORMAT(‘ CANNOT SCALE’)
60 STOP
END

PLOT EXAMPLE 1

Figure 2. Resultant plot of example program.
**Plot of plot system character set.**

<table>
<thead>
<tr>
<th>Character set</th>
<th>Code</th>
<th>Character set</th>
<th>Code</th>
<th>Character set</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>000 O O</td>
<td>040</td>
<td>100 O O</td>
<td>140</td>
<td></td>
<td></td>
</tr>
<tr>
<td>001 □ □</td>
<td>041</td>
<td>101 A A</td>
<td>141</td>
<td>a a</td>
<td></td>
</tr>
<tr>
<td>002 ◇ ◇</td>
<td>042</td>
<td>102 B B</td>
<td>142</td>
<td>b b</td>
<td></td>
</tr>
<tr>
<td>003  O O</td>
<td>043</td>
<td>103 C C</td>
<td>143</td>
<td>c c</td>
<td></td>
</tr>
<tr>
<td>004  X X</td>
<td>044</td>
<td>104 D D</td>
<td>144</td>
<td>d d</td>
<td></td>
</tr>
<tr>
<td>005  * *</td>
<td>045</td>
<td>105 E E</td>
<td>145</td>
<td>e e</td>
<td></td>
</tr>
<tr>
<td>006 + +</td>
<td>046</td>
<td>106 F F</td>
<td>146</td>
<td>f f</td>
<td></td>
</tr>
<tr>
<td>007 − −</td>
<td>047</td>
<td>107 G G</td>
<td>147</td>
<td>g g</td>
<td></td>
</tr>
<tr>
<td>010 Y Y</td>
<td>050</td>
<td>110 H H</td>
<td>150</td>
<td>h h</td>
<td></td>
</tr>
<tr>
<td>011  X X</td>
<td>051</td>
<td>111 I I</td>
<td>151</td>
<td>i i</td>
<td></td>
</tr>
<tr>
<td>012  X X</td>
<td>052</td>
<td>112 J J</td>
<td>152</td>
<td>j j</td>
<td></td>
</tr>
<tr>
<td>013  Δ Δ</td>
<td>053</td>
<td>113 K K</td>
<td>153</td>
<td>k k</td>
<td></td>
</tr>
<tr>
<td>014  \ \</td>
<td>054</td>
<td>114 L L</td>
<td>154</td>
<td>l l</td>
<td></td>
</tr>
<tr>
<td>015 * *</td>
<td>055</td>
<td>115 M M</td>
<td>155</td>
<td>m m</td>
<td></td>
</tr>
<tr>
<td>016</td>
<td>056</td>
<td>116 N N</td>
<td>156</td>
<td>n n</td>
<td></td>
</tr>
<tr>
<td>017</td>
<td>057</td>
<td>117 O O</td>
<td>157</td>
<td>o o</td>
<td></td>
</tr>
<tr>
<td>020</td>
<td>060</td>
<td>120 P P</td>
<td>160</td>
<td>p p</td>
<td></td>
</tr>
<tr>
<td>021</td>
<td>061</td>
<td>121 Q Q</td>
<td>161</td>
<td>q q</td>
<td></td>
</tr>
<tr>
<td>022</td>
<td>062</td>
<td>122 R R</td>
<td>162</td>
<td>r r</td>
<td></td>
</tr>
<tr>
<td>023</td>
<td>063</td>
<td>123 S S</td>
<td>163</td>
<td>s s</td>
<td></td>
</tr>
<tr>
<td>024</td>
<td>064</td>
<td>124 T T</td>
<td>164</td>
<td>t t</td>
<td></td>
</tr>
<tr>
<td>025</td>
<td>065</td>
<td>125 U U</td>
<td>165</td>
<td>u u</td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>066</td>
<td>126 V V</td>
<td>166</td>
<td>v v</td>
<td></td>
</tr>
<tr>
<td>027</td>
<td>067</td>
<td>127 W W</td>
<td>167</td>
<td>w w</td>
<td></td>
</tr>
<tr>
<td>030</td>
<td>070</td>
<td>130 X X</td>
<td>170</td>
<td>x x</td>
<td></td>
</tr>
<tr>
<td>031</td>
<td>071</td>
<td>131 Y Y</td>
<td>171</td>
<td>y y</td>
<td></td>
</tr>
<tr>
<td>032</td>
<td>072</td>
<td>132 Z Z</td>
<td>172</td>
<td>z z</td>
<td></td>
</tr>
<tr>
<td>033</td>
<td>073</td>
<td>133 \ \</td>
<td>173</td>
<td>\ \</td>
<td></td>
</tr>
<tr>
<td>034</td>
<td>074</td>
<td>134 \ \</td>
<td>174</td>
<td>\ \</td>
<td></td>
</tr>
<tr>
<td>035</td>
<td>075</td>
<td>135 J J</td>
<td>175</td>
<td>J J</td>
<td></td>
</tr>
<tr>
<td>036</td>
<td>076</td>
<td>136 \ \</td>
<td>176</td>
<td>\ \</td>
<td></td>
</tr>
<tr>
<td>037</td>
<td>077</td>
<td>137 \ \</td>
<td>177</td>
<td>\ \</td>
<td></td>
</tr>
</tbody>
</table>
INDEX

ABSTRACT ........................................ 1
ANDERSON, W. L. ............................. 3

CONTOUR BLOCK DATA ......................... 60
CONTOUR COMMAND FILE ...................... 7
CONTOUR EXAMPLES ........................... 11
CONTOUR USER DESCRIPTION ............... 5

DISCLAIMER ...................................... 2

ENTRY CHAR ...................................... 98
ENTRY COORD .................................... 37
ENTRY COORD0 .................................. 38
ENTRY ENDIPLT ................................ 100
ENTRY FLASH .................................... 103
ENTRY LFJUST ................................... 103
ENTRY LINE ..................................... 97
ENTRY NEATLN ................................... 101
ENTRY PLTSET ................................... 100
ENTRY PUTX ...................................... 92
ENTRY SCALE .................................... 95
ENTRY SETCOM ................................... 56
ENTRY SETLAB ................................... 56
ENTRY SPL2BX ................................... 89
ENTRY SPLN2B .................................. 74
ENTRY XAXIS ..................................... 101
ENTRY XAXISL ................................... 101
ENTRY YAXIS ..................................... 101
ENTRY YAXISL ................................... 101
EXAMPLES OF SPLN2D ......................... 18
EXECUTION OF CONTOUR ....................... 6
EXECUTION OF SPLN2D ......................... 16
EXECUTION OF SPLN2X ......................... 23

FUNCTION BIT .................................... 57

INTRODUCTION .................................... 3

PROGRAM CONTOUR ............................. 27
PROGRAM SPLN2D ............................... 61
PROGRAM SPLN2X ............................... 83

QUADRILATERAL GRIDS ......................... 5

REFERENCES .................................... 24

SPLN2D USER DESCRIPTION ................... 15, 22
SPLN2X BLOCK DATA ............................ 85
STANDARD FILE DESCRIPTION ............... 25
SUBROUTINE CONTR ............................. 43
SUBROUTINE CONTRR ............................ 41
INDEX (Cont'd)

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>COORDA</td>
<td>37</td>
</tr>
<tr>
<td>COORD0</td>
<td>38</td>
</tr>
<tr>
<td>DELGR</td>
<td>69</td>
</tr>
<tr>
<td>DVALS</td>
<td>75</td>
</tr>
<tr>
<td>FINEGR</td>
<td>66</td>
</tr>
<tr>
<td>GETSPL</td>
<td>82</td>
</tr>
<tr>
<td>GETV</td>
<td>91</td>
</tr>
<tr>
<td>GMFD</td>
<td>3</td>
</tr>
<tr>
<td>HEADS</td>
<td>31</td>
</tr>
<tr>
<td>LABEL</td>
<td>55</td>
</tr>
<tr>
<td>OPENCX</td>
<td>33</td>
</tr>
<tr>
<td>PUTCOF</td>
<td>81</td>
</tr>
<tr>
<td>PUTV</td>
<td>92</td>
</tr>
<tr>
<td>RESET</td>
<td>59</td>
</tr>
<tr>
<td>SCAN</td>
<td>48</td>
</tr>
<tr>
<td>SET</td>
<td>58</td>
</tr>
<tr>
<td>SETAX</td>
<td>76</td>
</tr>
<tr>
<td>SETAY</td>
<td>77</td>
</tr>
<tr>
<td>SETUP</td>
<td>45</td>
</tr>
<tr>
<td>SEXUAL</td>
<td>39</td>
</tr>
<tr>
<td>SPL2AX</td>
<td>89</td>
</tr>
<tr>
<td>SPLIM1</td>
<td>78</td>
</tr>
<tr>
<td>SPLN2A</td>
<td>72</td>
</tr>
<tr>
<td>SPLN2X</td>
<td>86</td>
</tr>
<tr>
<td>TRACE</td>
<td>51</td>
</tr>
<tr>
<td>UPDATE</td>
<td>32</td>
</tr>
<tr>
<td>XYREAD</td>
<td>35</td>
</tr>
<tr>
<td>YREAD</td>
<td>36</td>
</tr>
</tbody>
</table>

Transportability 3