United States Department of Interior

Geological Survey

CONTOUR: a modification of G. I. Evenden's general purpose contouring program

by

Richard H. Godson and Michael W. Webring

Open File Report 82-797

1982

Preface

The basic contouring subroutines of this program were developed by Gerald Evenden of the U.S. Geological Survey and implemented on a DEC-10 computer. The authors of this report converted the original code in order to make the program operational on a Honeywell Multics 68/80 computer and at the same time added several new features.

All subroutines are written in Multics Fortran IV except for two small PL/I subroutines. One of the subroutines left justifies alphanumeric data and the other compares two character strings.

The Honeywell Multics 68/80 is a 36-bit word machine and since there are several octal masking constants in the program, these constants will need to be changed in conversion to another computer of different word size.

Table of Contents

Abstract

A contouring program written for the DEC-10 computer (Evenden, 1975) has been modified and enhanced to operate on a Honeywell Multics 68/80 computer. The program uses a device independent plotting system (Wahl, 1977) so that output can be directed to any of several plotting devices by simply specifying one input variable.

Introduction

CONTOUR is a general purpose program for contouring two dimensional data that are given in a standard grid format described in appendix A. This format allows data to be ordered in several different ways: different x- and y-grid spacings, varying distances between grid positions in either the x- or y-direction and data arranged to form quadrilaterals whose interior angles do not exceed 180 degrees. The program recognizes grid values in excess of 1.0E38 as uncontourable, and thus irregularly shaped data sets may be contoured.

The following options are provided by the program: labelling of axes, printing of titles, drawing of border lines, plotting latitude and longitude tick marks, plotting positions of randomly spaced input data, drawing of state boundaries, dropping of contour lines in areas of high gradient, variable contour line widths, and plotting of L and H symbols in low and high closure areas. Details of these options are explained under Program Usage.

Description

The program was written to handle large gridded data sets commonly encountered in geophysical data processing. The grid has an origin at the lower left corner of an area and is ordered in rows, with row one at the bottom. A row consists of a sequence of column values that are normally

ordered from a westerly to easterly direction.

The data set is segmented for contouring into row tiers, the number of which is defined by the formula: (10000 - number of columns)/(1 + 1.25 * number of columns). The present implementation allows contouring of an infinite number of rows containing up to 2100 columns. The number of columns can be increased by changing two variables and recompiling the program.

The method of contouring is to trace each contour level through a tier and label the primary contours whenever possible. Since each row tier is contoured independently, excessive contour labelling is sometimes prevalent when many row tiers are involved in a data set.

Linear interpolation is used to obtain the coordinates of each contour line as it passes through the sides of grid cells. This produces angular contour lines in some cases and the amount of angularity is dependent on the grid interval of the data set and the scale at which the contours are to be plotted. An option is provided to fit a two dimensional cubic spline under tension (Cline, 1974) to the data and thereby create smooth contours. This option however increases the number of coordinate points by a factor of 4 and increases execution time by a factor of 2. See the variable sigma' under Program Usage for a detailed description of this option.

Program Usage

Instructions to the contour program are obtained from an ASCII segment created by the user on disk prior to running the program. This segment is read by Fortran unit 9 in namelist format. Attachment and detachment of the segment are accomplished internally in the program. The segment must contain the characters "&parms" followed by any number of the namelist variables explained below and ending with the character "&".

The gridded data file that is to be contoured must be attached to file10 which can be done externally or by specifying a varible in the command segment. If the random data points that were used to produce the grid are to be plotted, then a file giving positions of this data must be attached to file11 either externally or by specifying a variable in the command segment.

Following are explanations of the namelist variables, grouped under usage subheadings:

Contour levels

File attachments

ifile - the file name of the input gridded data set that is to be contoured. It must be enclosed in quotes with a maximum length of 56 characters. Default is blanks, which means that the input data set must be attached externally to file10.

ifile2 - the file name of the random data points that were used to produce the gridded data set. It must be enclosed in quotes with a maximum length of 56 characters. Default is blanks which means that the file must be attached externally to file11 if the position of random data points are to be plotted.

Contour levels

acval - a list of specific contour values to be used in contouring. Any number of values up to a maximum of 1000 may be specified and they must be in ascending sequence.

ncval - the number of values specified under acval above. Default is zero which means that a constant contour interval will be obtained from the variable dcval.

dcval - the contour interval. This value must be greater than zero if acval is not used. Default is zero.

Either acval and ncval or dcval must always be specified.

cmin - the lower limit of the contour values when dcval is used. If cmin is less than the minimum value of the data, then the minimum data value is used to determine the first contour value. Default is zero

cmax - the upper limit of the contour values when dcval is used. If cmax is greater than the maximum value of the data, then the maximum data value is used to determine the last contour value. Default is zero.

If both cmin and cmax are zero, then the range of contouring is determined from the data.

nsec - the interval of contour levels which are considered primary. This parameter is used when dcval is specified. Default is one, which means all contour levels are considered primary. Only primary contours are labelled.

fmtc - the format to be used for labelling the contour values (either F or E format). The parameter must be enclosed in quotes, e.g., "(F5.0)". A maximum number of 16 characters can be used. Default is blanks.

nchar - the maximum number of characters to be used from the above format when labelling the contours. For example, with nchar equal to 4 and

fmtc equal to "(F5.0)", four digits would be plotted without a trailing period. Contour labelling is performed for all specified contour levels (acval) and for all primary incremental levels (nsec). Default is zero which means no labelling.

size - the height in inches of labelled contour values. Default is .08.

idashs: determines line thicknesses and also which lines are to be dashed.

idashs = ± (N + ITH * 8) where:

N = 0 = all contours plotted as solid lines

N = -1 to -6 = secondary contour lines are plotted as dashed lines

N = 1 to 6 = primary contour lines are plotted as dashed lines

ITH = the additional thickness of the primary contours; e.g. ITH=1 produces primaries twice as wide, ITH=2 produces primaries three times as wide, etc.

Idashs value	Plotted line
0	
1	
2	
3	
4	
5	
6	

gradi - the maximum gradient in contour lines per inch before the secondary contours in a grid cell are not plotted. Default is 30.

Contour smoothing

nsig - a variable that determines whether a spline under tension will be applied to the contour line coordinates to create smoother contours. The use of this variable will create four times the number of original contour line coordinates and increase execution time by a factor of two. Default is zero for no splining. Any other number will cause splining to be performed.

sigma - a spline under tension factor that is used when 'nsig' is nonzero. The default is 5, which creates smooth contours for most data sets. A value of .001 approaches a pure two-dimensional cubic spline while a value of 50 causes very nearly linear interpolation. In some areas of steep gradient contours will cross when using the default value of this option. In these cases the value should be increased, and generally a value of 10 will be sufficient. In areas of low gradient the smoothness of the contours can be increased by decreasing the default value.

An alternate approach to smoothing using the above two variables is to use a smaller grid interval in the input data set.

Axis labelling

neat - a value that determines whether a rectangular border line will be drawn around the contour plot. The default is zero, meaning a border line will be drawn. Any other number will suppress the border.

title2 - 56 characters of identification to be plotted below the 56 characters obtained from the header record of the input gridded data file. This parameter must be enclosed in quotes, e.g., "scale 1:250,000". Default is blanks..

title3 - 56 characters of identification to be plotted below title2. This parameter must also be enclosed in quotes and the default is blanks.

sizel - the height in inches of the title lines. Default is zero, which means no title lines will be plotted.

adelx - the interval in data units along the x axis where a tick will be made. Default is zero which means that only the minimum and maximum values will be posted.

lintx - the interval of x axis ticks to be labelled. Default is 1.

fmtx - the format to be used when labelling the x axis (either F or E format). This parameter must be enclosed in quotes, e.g., "(F9.2)", and is limited to 16 characters. Default is blanks.

ncharx - the maximum number of characters to be used from the above format when labelling the x axis. Default is zero for no labelling.

sizex - the height in inches of the labelled x axis values. Default is .08.

adely - the interval in data units along the y axis where a tick will be made. Default is zero which means only the minimum and maximum values will be posted.

linty - the interval of y axis ticks to be labelled. Default is 1.

fmty - the format to be used when labelling the y axis (either F or E format). This parameter must be enclosed in quotes, e.g., "(F8.3)", and is limited to 16 characters. Default is blanks.

nchary - the maximum number of characters to be used from the above format when labelling the y axis. Default is zero for no labelling.

sizey - the height in inches of the labelled y axis value. Default is .08.

pllx - offset in inches of the x axis from the lower-left corner of the plot edge. Default is zero.

plly - offset in inches of the y axis from the lower-left corner of the plot edge. Default is zero.

The above two variables are used to provide space for axis labelling and title information.

Plotter type and scaling

iplotr - a number which determines which plotter device is to be used.

0 = Calcomp 7900

1 = Tektronix 4010 (default)

2 = Hewlett-Packard 7202

3 = Hewlett-Packard 7203

4 = Tektronix 4014 low resolution

5 = Tectronix 4014 high resolution

6 = General vector output, e.g., electrostatic printer/plotter

xscale - the x axis data units per inch. For example, if the grid interval is in inches then a value such as 250000 is used. Default is zero.

yscale - the y axis data units per inch. Default is zero.

If both xscale and yscale are xero then they will both be set to the same value that will allow plotting on the chosen plot device or 10 by 8 inches, whichever is smaller. If one scale factor is zero and the other not, then the nonzero scale factor will be assigned to the zero scale factor.

mscale - scaling factor that can be used in lieu of xscale and yscale. It is given in map scale units such as 62500, 250000, etc. Default is zero.

unit - the units of the grid interval in the input gridded data set to be contoured. 0 = inches, 1 = meters and 2 = kilometers. Default is 0. This parameter must be specified if mscale' is used.

Latitude/longitude tick marks

The following variables are to be used if plotting of latitude/longitude tick marks are desired. They are to be used only when the gridded data set coordinates have been calculated from a projection program where the central meridian, base latitude and type of projection is known. The longitude units can be either positive or negative. The unit' parameter explained above must be specified to use this option of the program.

basalt - a three unit array containing the base latitude in degrees, minutes and seconds. Default is 999,0,0.

cm - a three unit array containing the central meridian in degrees, minutes and seconds. Default is 999,0,0.

iproj - a number referring to the type of projection to be used.

1 = American polyconic

2 = universal transverse mercator (UTM)

3 = mercator

4 = lambert (standard parallels of 33^{o} and 45^{o})

5 = albers equal area for the conterminous U.S. (standard parallels of 29.5^{o} amd 45.5^{o})

6 = albers equal area for Alaska (standard parallels of 55^{o} and 65^{o})

7 = albers equal area for Hawaii (standard parallels of 8^{o} and 18^{o})

Default is 999 for no projection.

latm - a three unit array containing the minimum latitude in degrees, minutes and seconds. Default is 0,0,0.

latx - a three unit array containing the maximum latitude in degrees, minutes and seconds. Default is 0,0,0.

longm - a three unit array containing the left longitude in degrees, minutes and seconds. Default is 0,0,0.

longx - a three unit array containing the right longitude in degrees, minutes and second. Default is 0,0,0.

For longitudes east of Greenwich set longx less than longm.

tint - the interval of tick marks in minutes. Default is 15.

itpost - the tick mark interval where labelling will be performed. Default is 1.

sizep - the height in inches of the tick mark labels. Default is set at .20 inches.

Plotting of state boundaries

ibound - a variable that determines whether state boundaries are to be plotted. Default is 999, which means no state boundaries will be plotted. If plotting of boundaries are desired then set ibound to a number from 0 to 6 inclusive. The number selected will determine the type of line desired for the boundaries. See the "idashs" parameter under the heading Contour levels for the line types.

If boundaries are desired, then most of the variables explained under latitude/longitude tick marks must also be specified. These variables are baslat, cm, iproj, latm, latx, longm, longx and unit.

Plotting of positions of random data points

ispost - a variable that determines whether random data positions will be plotted and if so, what type of input data is to be read and posted. Default is zero which means no plotting of positions will be done.

1 = plot in vector form. The input records will contain the following information: x-position, y-position, magnitude, inclination and declination. The maximum number of records is 200. The inclinations are plotted next to a circle, which marks the xy position. The declination is indicated by the direction of an arrow and the magnitude by the length of the arrow.

2 = position plot with the records in an x, y and z format, where z is the value of the field at position x,y.

Negative values are used to indicate .post files, which are file outputs from the gravity reduction program bouguer. Post files are binary records containing the following information: 8 characters of station identification, x-coordinate, y-coordinate, and six z-values (free-air anomaly, complete bouguer anomaly one, complete bouguer anomaly two, elevation in feet, terrain correction and observed gravity).

-1 to -6 will print one of the z values next to the station symbol

-11 to -16 will print the station identification next to the station symbol

-21 to -26 will print the station identification and one of the z-values next to the station symbol

ifmtv - format of the input file; used when ispost is equal to equal to 1 or 2. A maximum of 56 characters is allowed and they must be enclosed in quotes, e.g., "(2F10.3,3F6.1)". Default is blanks which means the input data will be in binary form.

fmtv - the format to be used when labelling the z values (either E or F format). A maximum of 16 characters can be used and they must be enclosed in quotes, e.g., "(F6.1)". Default is "(F7.2)".

ncharv - the maximum number of characters to be used from the above format when plotting the z value. Default is zero for no plotting of z value.

vmin - used when ispost=1. It is the minimum vector length in inches. Default is zero.

vmax used when ispost=1. It is the maximum vector length in inches. Default is 1.

The vectors are logarithmically scaled from vmin to vmax.

szpost - the station symbol size in inches. Default is .08.

The following three variables are used to plot distinct symbols for different station identifications enountered in a post file.

nid - the number of different station symbols to be used; maximum is 19. Default is zero which means all stations will be plotted with one symbol.

ich - a list of symbol numbers to be used. These numbers refer to the desired characters as shown in appendix B (taken from the plot system documentation (Wahl, 1977). A maximum of 20 can be specified. Default numbers are 2, which is a diamond.

chid - a list of character strings of no more than four characters each that are used to match the first characters of identification on a .post file. A maximum of 20 strings can be specified. Default is blanks.

For example, let nid = 3, ich = 3,1,2,11 and chid = "abc", "tc01", "H". Station abc001 will be plotted with symbol 3, station tc01A will be plotted with symbol 1 and station Hur002 will be plotted with symbol 2. The nid + 1 number, in this case 11, will be used to plot all those stations that did not match the specified chid character strings. This number can be omitted and the default number 2 will be used.

To bypass contour plotting and just plot the random input locations, use the above variables and let dcval and ncval default to

zeros.

Low and high closures

lowhi - a parameter to plot the symbol "L" in low closures and the symbol "H" in high closures on contour plots. To implement this option set "lowhi" equal to a number of grid intervals to use as a search radius. Usually a value of 4 is adequate. Default is 0.

This option will not work if lowhi is greater than the number of columns in a grid or the grid intervals are not equal in either the x- or y-direction.

Program Execution

The data file to be contoured can be attached externally using the switchname file10, e.g., "io attach file10 vfile_ test.grd," or internally using the namelist parameter "ifile". To run the program type the word "contour." The program will then type the statement "enter command filename." The user then types in the name of his command segment and program execution continues.

If offline plotting devices are used the plotting system will ask questions about the tape and file number to be used or about disk file names. See Examples of Program Execution for the specific questions that are asked. If tape is used one should issue the Multics system command: "assign-resources tape_drive" (ar tape_drive) before running the program to be sure that a tape drive is available.

If the plot is general vector output for electrostatic plotters (plotter number 6), a question will be asked whether you want raster output or not. If the answer is "yes" then the vectors are sorted and converted to rasters for direct plotting by a Varian 22 inch plotter. This is used for very large plots with many contours as many minicomputers can not handle the vector-raster conversion of large data sets.

When the program is completed a Multics ready message is typed except when plotting on a Tektronix terminal. This omission on the Tektronix is to prevent printing over the plot on this type of device before a copy can be made. To return to command level in this case enter a line feed.

One plot is made with each execution of the program. To contour additional data sets in one process the user can attach to a different data file and run the program again. Several files can be created on tape this way by being sure to increase the file number by one each time the plotting system asks for the file number.

The command file can be tested by setting iplotr=6 and running the program. When the plotting system asks for tape or null, type the word "null." The plotting system will still ask you if you want raster output and for tape number and file number. One just answers with a "no" and two zeros and the output will be directed to a Multics file discard_ and the program will execute.

Error Messages

There are several program error messages that can be generated; they are all preceded by a "%." One of the most common messages is "% end of file while processing header of input file." This usually means that one forgot to attach file10 to the data set to be contoured. Another message that is not hard to generate is "% odd eof on command segment." This results by entering the wrong command segment name or forgetting to end the namelist with a "&."

When program error messages are generated all files are closed and the command segment is detached, except for those files that are attached externally. If one quits out of the program after reading the command segment, or a system error is encountered, then only the command segment is closed and detached. All other files will still be open. It will be necessary to close file10 and file11 (if used) before running the program again.

External Subroutines

The following subroutines in the device independent plotting system are called by the program:

pltset

scale

char

xaxis

yaxis

line

neat1

endpt

The PLI subroutines are as follows:

leftj

match

Command Segment Examples

1. The following example will contour a data set on a Tektronix 4010 using a contour interval of 20. No titles, axis labelling or contour labelling will be done. Automatic scaling will be used.

```
&parms dcval=20,&
```

2. The following example will contour a data set for the Varian electrostatic printer/plotter with contour labelling, axis labelling and plotting of titles.

```
&parms title2="scale 1:250,000", title3="May, 1977", fmtc="(F5.0)",
fmtx="(F6.0)", fmty="(E9.2)", nchar=5, ncharx=6, nchary=9, dcval=2,
nsec=5, size1=.1, adelx=900, adely=900, iplotr=6, pllx=2, plly=2,
xscale=250000,&
```

3. This example will contour a data set on a Calcomp plotter with contour labelling, contour smoothing, latitude/longitude tick marks, no border line, posting of random data locations and state boundaries.

```
&parms dcval=5, fmtc="(F5.0)", nchar=4, nsec=2, iplotr=0, idashs=-1,
xscale=6.35, unit=2, iproj=2, cm=96, baslat=0, latm=30, latx=35,30,
longm=109,30, longx=114, nsig=1, gradi=100, neat=1, tint=30,
itpost=2, ibound=5, ispost=-2, fmtv="(F5.0)", ncharv=4,&
```

Examples of Program Execution

1. The following example shows a command file and two runs of program contour. The lines starting with the characters "<<" are questions from the plot system software.

```
    &parms dcval=2,iplotr=6,
    fmtc="(F4.0)",nchar=3,nsec=1,gradi=100,
    pllx=2,plly=2,size1=.15,
    baslat=0,0,0,cm=71,30,0,iproj=1,tint=5,unit=2,
    latm=42,15,latx=42,45,longm=71,15,longx=71,45,
    xscale-1.5875,
    title2="scale = 1:62500",&
io   attach file10 vfile_   boston.faa
contour
enter command filename: conter.cmd
<<Pltsys: General output device type-(tape, disk, or null):  tape
<<Pltsys: Do you want raster output?  no
<<Pltsys: Plot tape:   rhg001
<<Pltsys: Plot file number:    1
Mounting volume rhg001 with a write ring.
rhg001 mounted on tape_05.
plot size (inches): x=  3.76  y=  3.41
<<Pltsys: Done with the tape in this process?  no
STOP
io detach file10
io attach file10 vfile_   boston.ba
contour
enter command filenam: conter.cmd
<<Pltsys: Plot file number:   2
```

```
plot size (inches): x =  3.76  y = 3.41
<<Pltsys: Done with the tape in this process?  yes
STOP
```

2. The following example shows a contour run with disk output.

```
io attach file10 vfile_  boston.faa
contour
enter command filename: conter.cmd
<<Pltsys: General output device type-(tape, disk or null): disk
<<Pltsys: Do you want raster output?  no
<<Pltsys: Enter file name: faa.plot
plot size (inches): x =  3.76  y =  3.41
STOP
```

3. The following example shows a contour run using the raster output option.

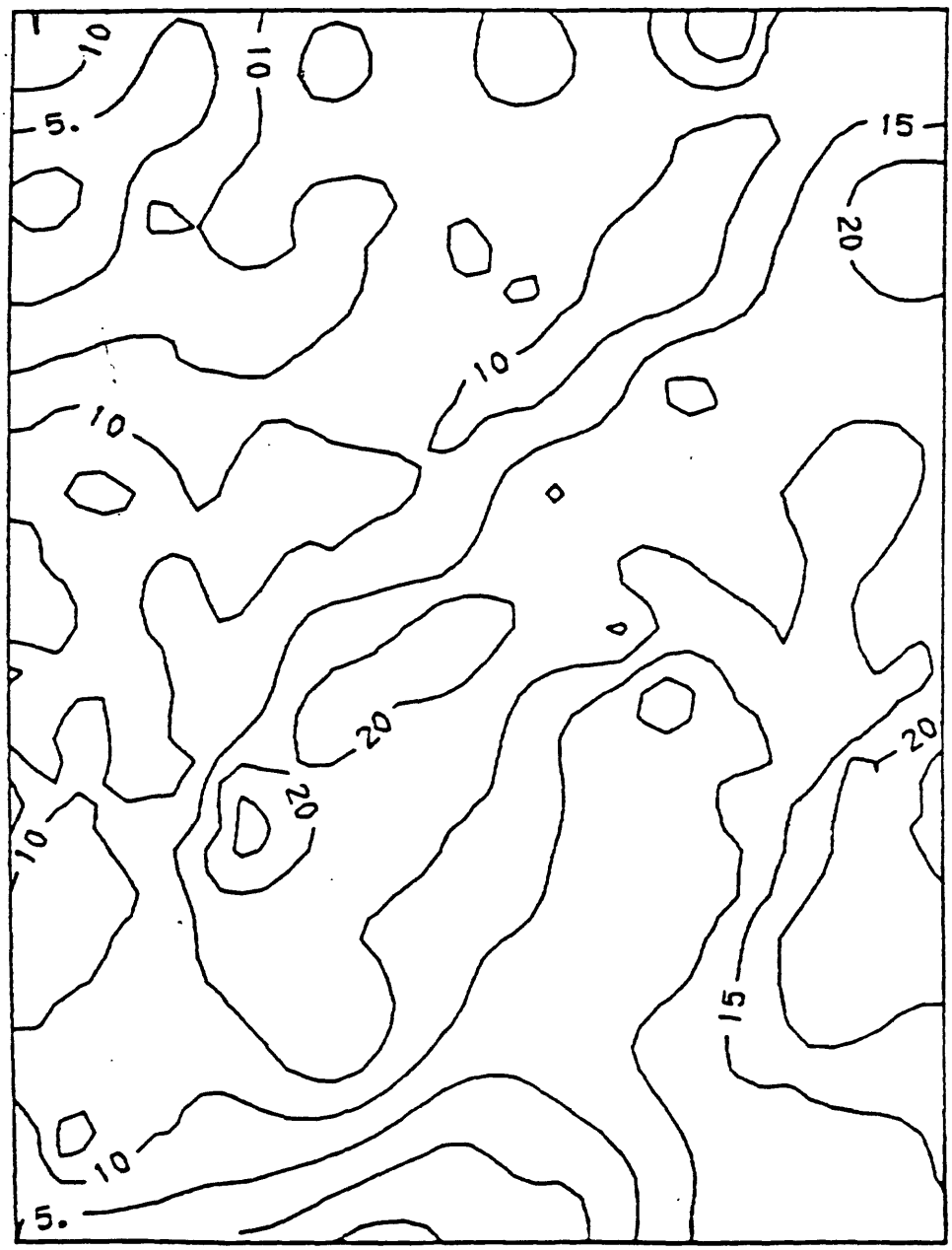
```
io attach file10 vfile_   >pp>Datap>Dataproc>rq>midwest.grd
enter command filename: >udd>Dataproc>RGodson>conter.cmd
<<Pltsys: General output device type-(tape, disk, or null):  tape
<<Pltsys: Do you want raster output?  yes
<<Pltsys: Plot tape:  kohn01
<<Pltsys: Plot file number:    1
Mounting volume kohn01 with a write ring.
kohn01 mounted on tape_04.
<<Pltsys: Reverse axes?   yes
plot size (inches): x =  3.76  y = 3.41
Sorting...
 24696 records sorted.
File 1 completed.
Sorting...
  9597 records sorted.
```

```
File 2 completed.

<<Pltsys: Done with the tape in this process?  yes

STOP
```

Plot Example 1

The plot below was produced using this command segment:

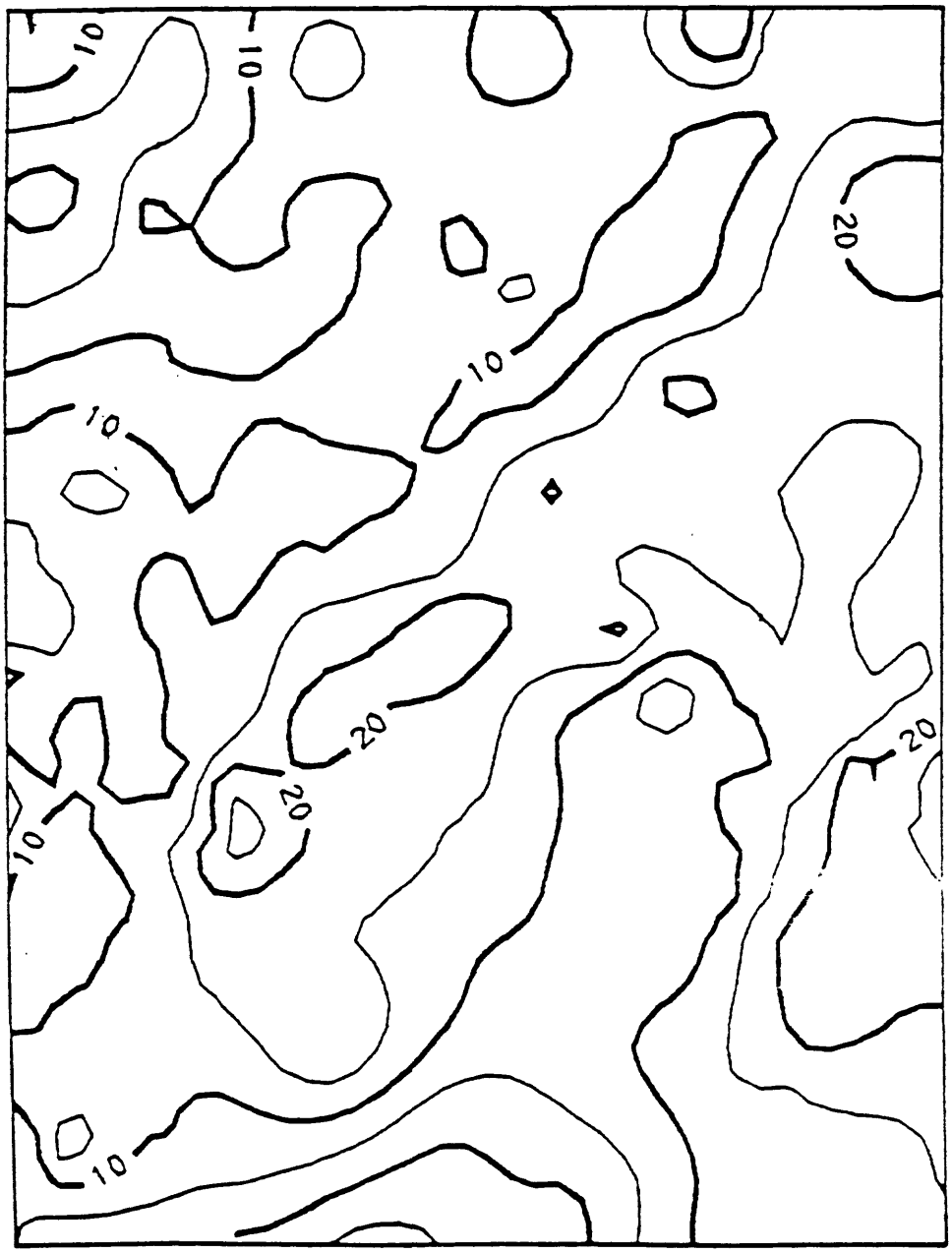
```
&parms nsec=1,fmtc="(F3.0)",pllx=1.,ifile="b.faa",
dcval=5,nchar=2,&
```



Plot Example 2

The plot below was produced using this command segment:

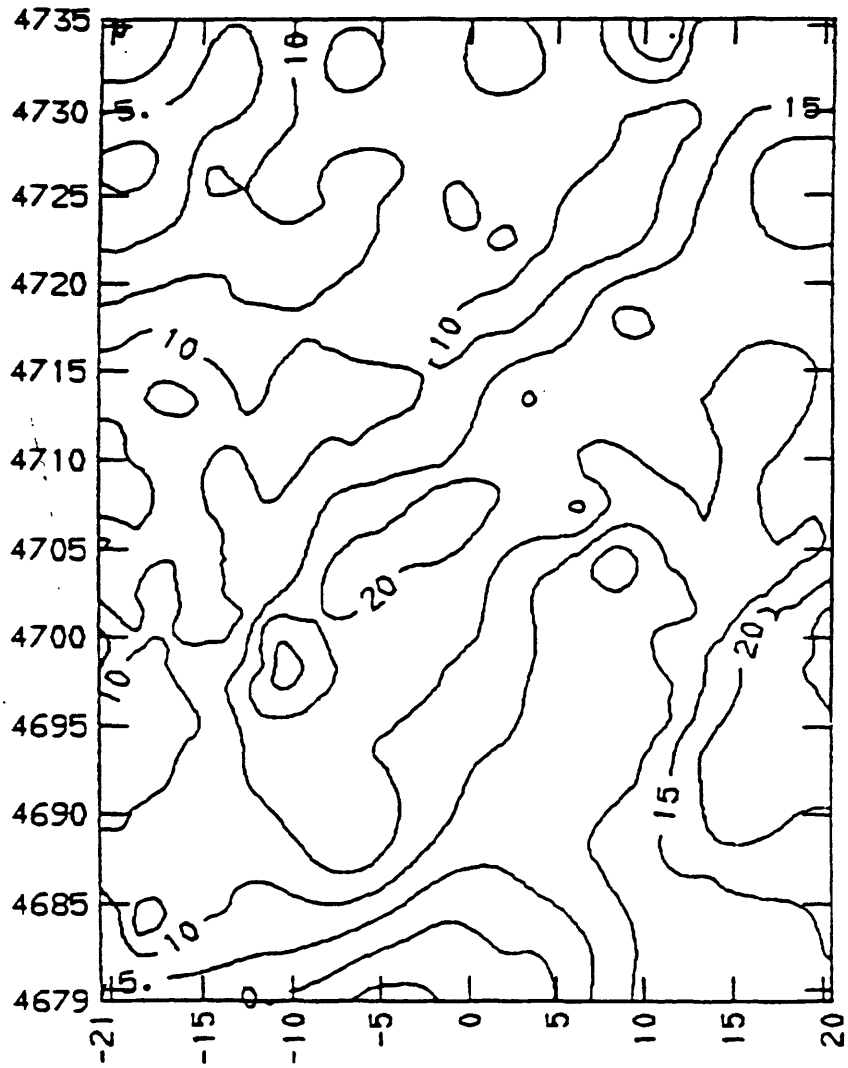
```
&parms nsec=2,fmtc="(f3.0)",plix=1.,ifile="b.faa",dcval=5,
nchar=2,idashs=8,&
```



Plot Example 3

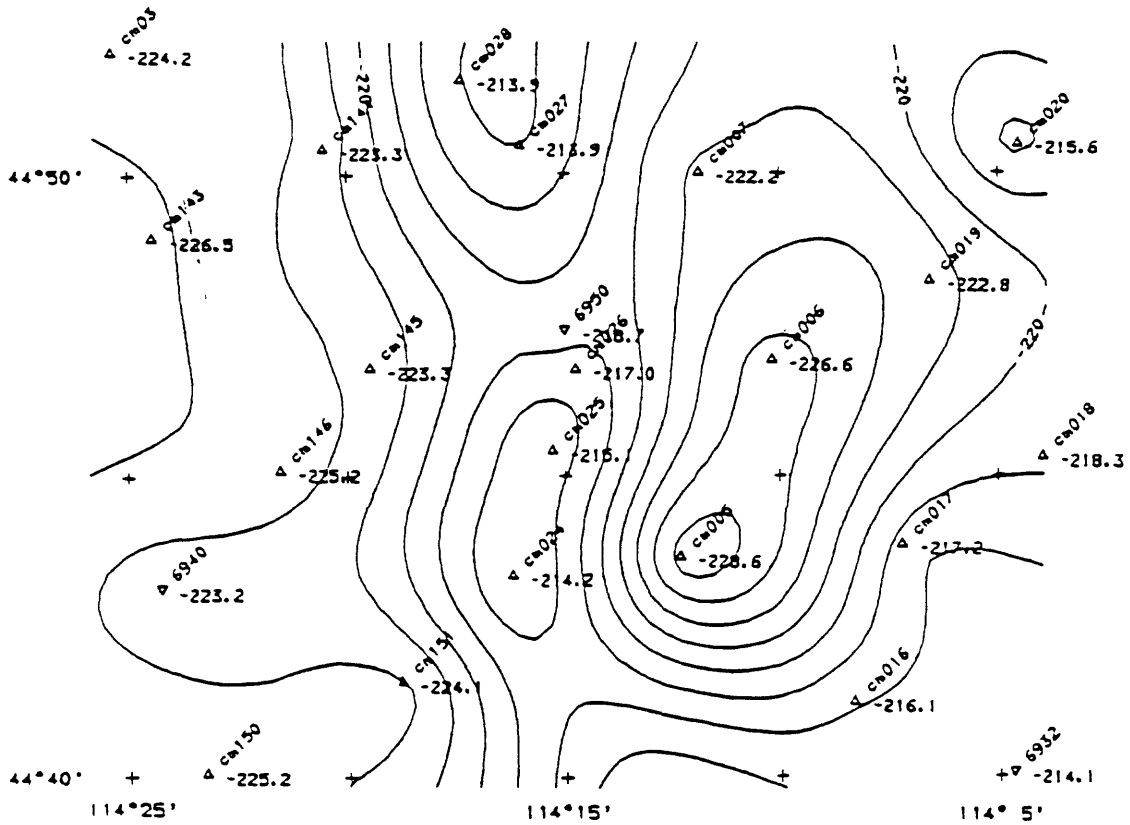
The plot below was produced using this command segment:

```
&parms nsec=1,fmtc="(f3.0)",pllx=1,plly=1,ifile="b.faa",dcval=5,
nchar=2,nsig=1,sigma=1,
adelx=5,adely=5,ncharx=3,nchary=4,fmtx="(f4.0)",fmty="(f5.0)",&
```



The plot below was produced using this command segment:

```
&parms
ifile="test.grd",ifile2="test.bxyz"
dcval=2.
nsec=5
fmtc="(f5.0)",nchar=4
size1=.1,iplotr=1,xscale=0.,yscale=0.
size=.06
neat=1,p11x=1.,p11y=1.
baslat=44,0,0,cm=115,0,0,iproj=2,sizep=.07,unit=2
tint=5.,itpost=2
latm=43,45,0,latx=45,0,0
longm=113,45,0,longx=116,15,0
ispost=-22,fmtv="(f10.1)",ncharv=6
nid=6,chid="cm","lv","ch","cp","st","p",ich=11,6,3,3,4,4,12
szpost=.06
$
```



station posting example

References

Cline, A. K., 1974, Scalar- and planar-values curve fitting using splines under tension: Communications of ACM, v. 17, no. 4, p. 218-223.

Evenden, Gerald Ian, 1975, A general purpose contouring system: U.S. Geol. Survey Open-File Report 75-317, 107 p.

Wahl, Ronald R., 1977, Users guide for the basic plot system on Multics: U.S. Geological Survey, unpublished documentation.

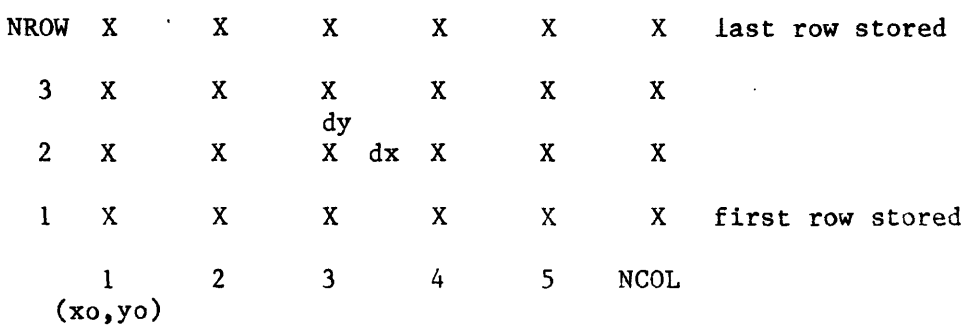
Appendix A

Standard Grid

The grid can be any one of many types: (a) a rectangular grid with equal spacing in the x- and y-directions, (b) a rectangular grid with constant but unequal x- and y-intervals, (c) a rectangular grid with varying distances between grid positions in either the x- or y-direction or both and d) a quadrilateral grid which consists of connecting quadrilaterals whose interior angles do not exceed 180 degrees.

The file of the gridded data consists of two basic parts: (1) a header record and optionally, a following record that contains the x-coordinates for each column and (2) a series of data records, each containing the column values for one row.

The following diagram shows the relationship of the grid elements in the usual case where dx and dy are positive.



A. Header record (23 words long)

id: 56 ASCII characters of identification (14 words).

pgm: 8 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).

nz: number of words per data element (integer, 1 word). For single precision use 1, double precision or complex use 2, double precision

complex use 4. For quadrilateral grids this value is 3.

xo: position of first column of data (real, 1 word).

dx: equal spacing interval of columns (real, 1 word).

If equal to zero, then coordinates for each column are in the following data record; otherwise the following record consists of data for row one.

yo: position of first row (real, 1 word).

dy: equal spacing interval of rows (real, 1 word). If equal to zero, 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 is equal to zero. Record consists of ncol real words specifying the coordinates of each data column in monotonic order. If nz=3 then this record is present but the values are meaningless.

C. Data record. Each data record contains one row of real data items. The total record length is ncol times (nz plus 1) words. For quadrilateral data the sequence of data values is x, y, and z. The first word contains the row coordinate if dy of the header record is zero, else the value is a dummy. Again, the row coordinates should be in monotonic sequence, if specified.

In general, i/o for this standard file can be stated in fortran as:

```
      dimension g(iz,ix,iy),id(14),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
10    read or write (..) y(j),((g(k,i,j),k=1,nz),i=1,ncol)
      go to 25
15    do 20 j=1,nrow
```

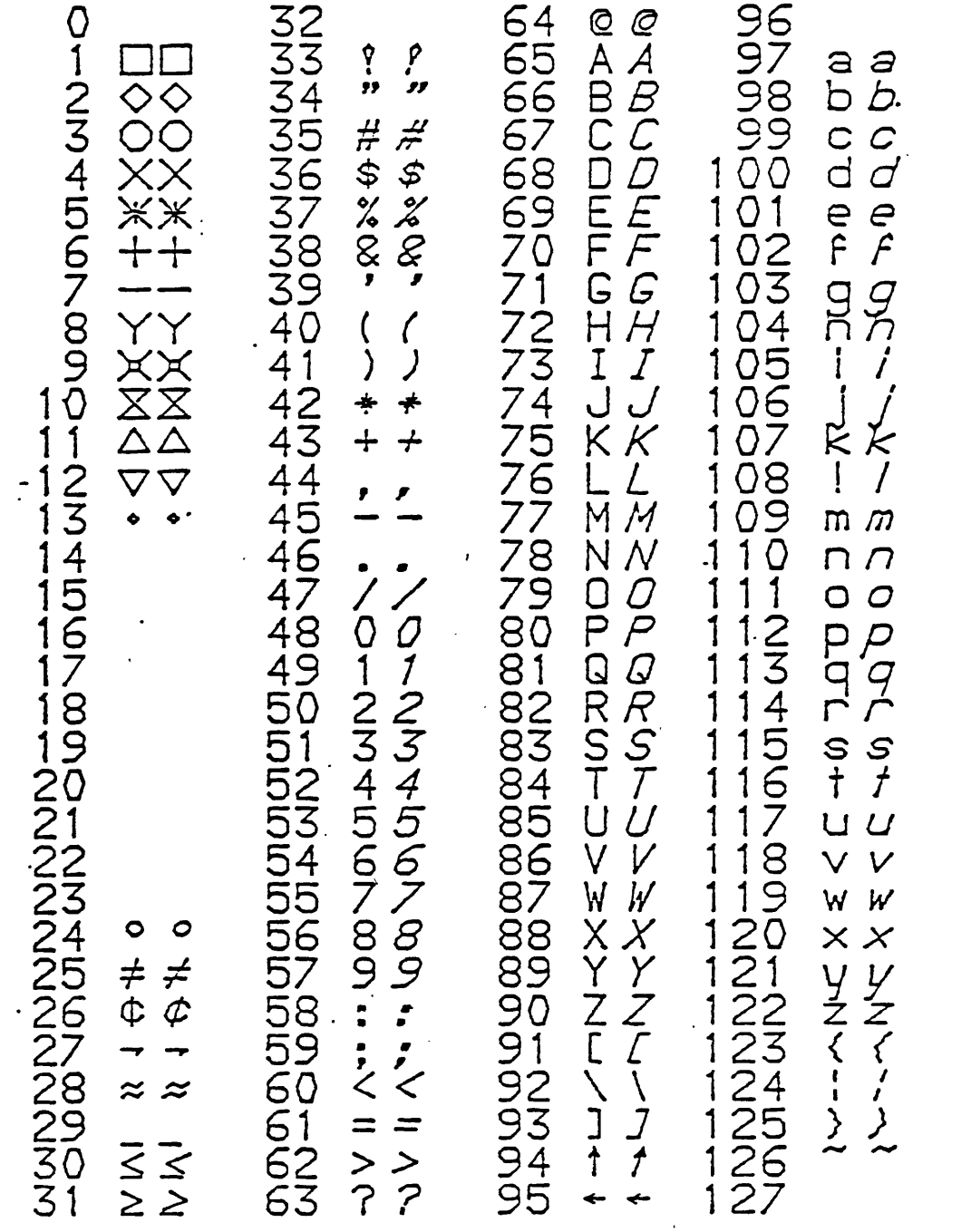
```
20    read or write (..) dum,((g(k,i,j),k=1,nz),i=1,ncol)


25    continue
```

In the usual case where dx and dy are constant and nz=1, the code simplifies to:

```
      dimension g(ix,iy),id(14),pgm(2)

      read or write(..) id,pgm,ncol,nrow,nz,xo,dx,yo,dy
      do 10 j=1,nrow


10    read or write (..) dum,(g(i,j),i=1,ncol)
```

Appendix B

Plot of Plot System Character Set



```
c
c  general contouring program
c
c  developed and coded by..
c    u. s. geological survey
c    gerald ian evenden
c    denver, colorado  80225
c
c  modified for the honeywell/multics computer
c   by r.h. godson - april,1977
c
      automatic work(10000)
c
c  grids of up to 2100 columns can be contoured with nwork=10000.
c  in practice, 30 to 40 rows per tier is prefered so that 200
c  to 250 columns is a desirable limit.  the formula for rows
c  per tier (rpt) is ......
c      rpt=(nwork-ncols)/(1+1.25*ncols).
c  if larger grids are contoured, recompile with nwork and
c  dimension of work increased.
c
c  note.. this program contains several non-ansi fortran
c    statements.  although several of these constructs
c    may be available on other computer systems care must be
c    execised in the transportation of this program..
c
      character*56 title1,title2,title3,ifmtv
      character*16 fmtc,fmtx,fmty,fmtv
      real latm,latx,lonam,lonax
      common/sexy/title1,title2,title3,fmtx,fmty,
     1 xx(2),yy(2),xp1(4),yp1(4),iplotr,sizel,ncharx,nchary,
     2 sizex,sizey,adelx,adely,pllx,plly,lintx,linty,
     3 xscale,yscale,xxscal,yyscal,mscale
      common /vector/ispost,vmax,vmin,fmtv,ncharv,ifmtv,
     1 ich(20),szpost,lowhi,chid(20),nid
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gflg,iis,iie,iisi,iiei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
      common/llp/latm(3),latx(3),lonam(3),longx(3),cm(3),baslat(3),
     1 iproj,xxx(2),yyy(2),sizep,unit,ip,neat,tint,itpost,ibound
      common/contrc/ cmin,cmax,dcval,ncval,nsec,gradi,nchar,size
       character*50 coname
      logical nomore,prime
      dimension acval(1000),idim(1)
      equivalence (acval(1),work(1))
      data nwork/10000/,idim(1)/0/
c
      data icard/5/,iprint/6/,icmd/9/,igrid/10/,ista/11/
c
c  get command file name and attach
c
   10 write(iprint,20)
   20 format(" enter command filename :"$)
      read(icard,30)coname
   30 format(v)
      open(icmd,file=coname,form="formatted",mode="in")
c
c  get control data
c
      call name(acval,*240)
```

```
c
c  general contouring program
c
c  developed and coded by..
c    u. s. geological survey
c    gerald ian evenden
c    denver, colorado  80225
c
c  modified for the honeywell/multics computer
c   by r.h. dodson - april,1977
c
      automatic work(10000)
c
c  grids of up to 2100 columns can be contoured with nwork=10000.
c  in practice, 30 to 40 rows per tier is prefered so that 200
c  to 250 columns is a desirable limit.  the formula for rows
c  per tier (rpt) is ......
c       rpt=(nwork-ncols)/(1+1.25*ncols).
c  if larger grids are contoured, recompile with nwork and
c  dimension of work increased.
c
c  note.. this program contains several non-ansi fortran
c    statements.  although several of these constructs
c    may be available on other computer systems care must be
c    excised in the transportation of this program..
c
      character*56 title1,title2,title3,ifmtv
      character*16 fmtc,fmtx,fmty,fmtv
      real latm,latx,lonam,lonax
      common/sexy/title1,title2,title3,fmtx,fmty,
     1 xx(2),yy(2),xp1(4),yp1(4),iplotr,sizel,ncharx,nchary,
     2 sizex,sizey,adelx,adely,pllx,plly,lintx,linty,
     3 xscale,yscale,xxscal,yyscal,mscale
      common /vector/ispost,vmax,vmin,fmtv,ncharv,ifmtv,
     1 ich(20),szpost,lowhi,chid(20),nid
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gflg,iis,iie,iisi,ijei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
      common/llp/latm(3),latx(3),lonam(3),lonax(3),cm(3),baslat(3),
     1 iproj,xxx(2),yyy(2),sizep,unit,ip,neat,tint,itpost,ibound
      common/contrc/ cmin,cmax,dcval,ncval,nsec,gradi,nchar,size
       character*50 coname
      logical nomore,prime
      dimension acval(1000),idim(1)
      equivalence (acval(1),work(1))
      data nwork/10000/,idim(1)/0/
c
      data icard/5/,iprint/6/,icmd/9/,igrid/10/,ista/11/
c
c  get command file name and attach
c
   10 write(iprint,20)
   20 format(" enter command filename :"$)
      read(icard,30)coname
   30 format(v)
      open(icmd,file=coname,form="formatted",mode="in")
c
c  get control data
c
      call name(acval,$240)
```

```
      nwkres=nwork
c
      if (nsec.lt.0) nsec=0
      if (gradi.lt.0.) gradi=0.
      if (nchar.lt.0) nchar=0
      if (size.lt.0.) size=0.
      if (nsig.ne.0) nsig=4
      if (sigma.gt.0.) sigma=-sigma
c
c  start check-out.
      if(mscale.le.0) go to 40
c  convert map scale to data units/inch
      if(unit.gt.2.5 .or. unit.lt.-.5) stop " invalid 'unit' parm"
      unitc=1.0
      if(unit.gt.0.5) unitc=39.370079
      if(unit.gt.1.5) unitc=39370.079
      xscale=float(mscale)/unitc
      yscale=xscale
c
c  contour values...
c
   40 if (ncval.gt.0.) go to 70
c
c  incrimental contours mode.
c
      if (dcval.ge.0.) go to 60
      write(iprint,50)
   50 format(" %dcval less than zero")
      stop
   60 ixad=1
      go to 110
c
c  specified contours mode.
c
   70 if (ncval.eq.1) go to 100
c
c  check ascendency.
c
      do 90 i=2,ncval
      if (work(i).gt.work(i-1)) go to 90
      write(iprint,80)
   80 format(" %non-ascending specified contours")
      stop
   90 continue
  100 ixad=ncval+1
c
c  all that can be done without looking at data file.
c
c  open file and prelim check.
c
  110 nwkres=nwkres-ncval
      call openck(igrid,xo,yo,delx,dely,
     1 work(ixad),nwkres,iquad)
      if (ncol.eq.0) go to 260
      oxo=xo
      oyo=yo
      odx=delx
      ody=dely
      nco=ncol
      nro=nrow
```

```
c
c     slice up core
c
      if(iquad.eq.3) go to 140
      nrowt=4*(nwkres-ncol)/(5*ncol+4)
      if(nrowt.ge.2) go to 130
c
      write(iprint,120)
  120 format(" %insufficient memory")
      go to 230
c
  130 idad=ixad+ncol
      iyad=nwork-nrowt+1
      ifad=idad+nrowt*ncol
      go to 150
c
  140 nrowt=4*nwkres/(13*ncol)
      if(nrowt.lt.2) go to 110
      k=nrowt*ncol
      idad=ixad+k
      iyad=nwork-k+1
      ifad=idad+k
c
c     ok so far, scale and annotate
c
  150 call sexual(i)
      if(i.ne.0) go to 230
      if(iquad.eq.3) go to 190
      k=ixad+ncol-1
      if(delx.ne.0) go to 170
      do 160 i=ixad,k
  160 work(i)=work(i)*xxscal
      go to 190
  170 xo=xo*xxscal
      delx=delx*xxscal
      do 180 i=ixad,k
      work(i)=xo
  180 xo=xo+delx
c
  190 npass=(nrow+nrowt-3)/(nrowt-1)
      jres=1+nrow-nrowt-(npass-2)*(nrowt-1)
c
      call setcon(size,nchar)
c
c     branch around contour trace
c
      if(dcval.eq.0.) go to 200
      call conttr(work(ixad),work(iyad),work(idad),
     1 work(ifad),work,nrowt,ncol,xxscal,yyscal,qflg,
     2 yo*yyscal,dely*yyscal,iquad,npass,jres,igrid,
     3 ixad,iyad)
  200 xx(1)=oxo
      yy(1)=oyo
      xx(2)=(nco-1)*odx+oxo
      yy(2)=(nro-1)*ody+oyo
      call scale(xx,yy,xo1,yo1,3,ier)
      if(ispost.eq.0) go to 210
c
c  post stations
c
```

```
      call vector(size)
      close(ista)
  210 if(lowhi.eq.0) go to 220
c
c  plot high and low symbols
c
      nc2=nco+2*lowhi
      iw=2*lowhi+1
      rewind igrid
      if(nc2*(iw+4)+iw .gt. nwork) go to 220
      call hilow(work,work(nc2+1),work(2*nc2+1),work(3*nc2+1),
     1 work(4*nc2+1),work(nc2*(iw+4)+1),nc2,iw,size)
  220 continue
c
c  close plot
c
      call endpt(idim)
c
c  close data file
c
  230 close(igrid)
      go to 270
c
c  command file end
c
  240 write(iprint,250)
  250 format(" %odd eof on command segment")
  260 continue
  270 stop
      end
c***************
      subroutine name(acval,*)
c
c     namelist input and common block initialization
c
      common/sexy/title1,title2,title3,fmtx,fmty,
     1 xx(2),yy(2),xp1(4),yp1(4),iplotr,sizel,ncharx,nchary,
     2 sizex,sizey,adelx,adely,pllx,plly,lintx,linty,
     3 xscale,yscale,xxscal,yyscal,mscale
      common /vector/ispost,vmax,vmin,fmtv,ncharv,ifmtv,
     1 ich(20),szpost,lowhi,chid(20),nid
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gflg,ijs,ije,iisi,ijei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
      common/llp/latm(3),latx(3),lonam(3),longx(3),cm(3),baslat(3),
     1 iproj,xxx(2),yyy(2),sizep,unit,ip,neat,tint,itpost,ibound
      common/labcom/charst,fmtc,idum,dum1,dum2,dum3,dum4
c
      logical prime
c
      common/contrc/ cmin,cmax,dcval,ncval,nsec,gradi,nchar,size
      dimension acval(1000)
      character*56 title1,title2,title3,ifile,ifile2,iblank,ifmtv
      character*16 fmtc,fmtx,fmty,iblank1,fmtv
      integer charst(6)
      real latm,latx,lonam,longx
      namelist /parms/ ncval,cmin,cmax,nsec,dcval,xscale,yscale,mscale,
     1 gradi,iplotr,nchar,size,idashs,acval,
     2 sizel,ncharx,nchary,sizex,sizey,adelx,adely,
     3 pllx,plly,lintx,linty,sigma,nsig,latm,latx,lonam,longx,
```

```
     4 cm,baslat,iproj,sizep,unit,angle,neat,tint,itpost,ibound,
     5 title2,title3,fmtc,fmtx,fmty,ispost,vmax,vmin,fmtv,ncharv,ifmtv,
     6 ich,szpost,lowhi,chid,nid,ifile,ifile2
      data icmd/9/,igrid/10/,ista/11/,iblank/" "/,iblank1/" "/
      lmult(1)=134217728
      lmult(2)=262144
      lmult(3)=512
      lmult(4)=1
      ibound=999
      lowhi=0
      ispost=0
      do 10 i=1,20
      ich(i)=2
   10 chid(i)=" "
      nid=0
      szpost=.08
      vmin=0.
      vmax=1.
      ncharv=0
      fmtv="(f7.2)"
      ifmtv=" "
      cmin=0.
      cmax=0.
      dcval=0.
      ncval=0
      nchar=0
      nsec=1
      size1=0.
      gradi=30.
      gflg=1.0e38
      fltmax=1.0e38
      iplotr=1
      idashs=0
      linet=0
      ncharx=0
      nchary=0
      lintx=1
      linty=1
      size=.08
      sizex=.08
      sizey=.08
      adelx=0.
      adely=0.
      bllx=0.
      blly=0.
      xscale=0.
      yscale=0.
      mscale=0
      xp1(2)=0.
      yp1(2)=0.
      sigma=-5.
      nsig=0
      sizep=0.08
      iproj=999
      unit=999.
      angle=0.
      neat=0
      tint=15.
      itpost=1
      cm(1)=999.
```

```
      cm(2)=0.
      cm(3)=0.
      baslat(1)=999.
      baslat(2)=0.
      baslat(3)=0.
      do 20 i=1,3
      latm(i)=0.
      latx(i)=0.
      longm(i)=0.
      longx(i)=0.
   20 continue
      title2=iblank
      title3=iblank
      fmtc=iblank1
      fmtx=iblank1
      fmty=iblank1
      ifile=" "
      ifile2=" "
      read(icmd,parms,end=30)
      close (icmd)
      if(ifile.ne." ") open(igrid,file=ifile,mode="in")
      continue
      if(ifile2.ne." " .and. ifmtv.eq." ") open(ista,file=ifile2,
     1 mode="in")
      if(ifile2.ne." " .and. ifmtv.ne." ")
     1 open(ista,file=ifile2,form="formatted",mode="in")
      ip=iplotr
      return
   30 close(icmd)
      return 1
      end
c***************
      subroutine openck(igrid,xo,yo,delx,dely,
     1 xdata,nwork,iquad)
c
c  initialize grid input file.
c
      common/sexy/title1,title2,title3,fmtx,fmty,
     1 xx(2),yy(2),xp1(4),yp1(4),iplotr,sizel,ncharx,nchary,
     2 sizex,sizey,adelx,adely,pllx,plly,lintx,linty,
     3 xscale,yscale,xxscal,yyscal,mscale
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,qfla,iis,iie,iisi,iiei,prime,ii4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
      character*56 title1,title2,title3
      character*16 fmtx,fmty
      dimension xdata(1),dummy(2)
      data iprint/6/
c
      read(igrid,end=70) title1,dummy,ncol,nrow,iquad,xo,delx,yo,dely
      if (ncol.lt.3.or.nrow.lt.3) go to 90
      if(iquad.eq.3.and.delx.eq.0..and.dely.eq.0.) go to 60
      if (ncol.gt.nwork) go to 130
      if (iquad.ne.1) go to 110
      if (delx.ne.0) go to 10
      read(igrid,end=70) (xdata(i),i=1,ncol)
      xx(1)=xdata(1)
      xx(2)=xdata(ncol)
      go to 20
   10 xx(1)=xo
```

```
      xx(2)=xo+(ncol-1)*delx
   20 if(delv.ne.0) go to 30
      call yread(igrid,yy,nrow)
      if(nrow.lt.3) go to 90
      go to 40
   30 yy(1)=yo
      yy(2)=yo+(nrow-1)*dely
   40 if(delx.ne.0..and.dely.ne.0.) return
   50 rewind igrid
      read(igrid)
      if(delx.eq.0.) read(igrid)
      return
   60 i=ncol*3
      if(i.gt.nwork) go to 130
      read(igrid)
      call xyread(igrid,xx,yy,xdata,i)
      if(nrow.lt.3) go to 90
      go to 50
   70 write(iprint,80)
   80 format(" %end of file while processing header of input file")
      go to 150
   90 write(iprint,100)
  100 format(" %no. rows or columns less than 3")
      go to 150
  110 write(iprint,120)
  120 format(" %nz on header of standard file is greater than 1")
      go to 150
  130 write(iprint,140)
  140 format(" %core exceeded for x values")
c
  150 close (igrid)
      ncol=0
      return
      end
c***************
      subroutine yread(igrid,yr,nrow)
c
c     scans specified row grid
c
      dimension yr(2)
      nrow=0
      read(igrid,end=20) yr(1)
   10 nrow=nrow+1
      read(igrid,end=20) a
      yr(2)=a
      go to 10
   20 return
      end
c***************
      subroutine xyread(igrid,xr,yr,work,ncol3)
c
c     scans quadrilateral grid for x-y range
c
      dimension xr(2),yr(2),work(ncol3)
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,qflg,iis,iie,iisi,iiei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
c
c     initialize
c
```

```
      nrow=0
      read(iarid,end=70) dummy,work
      xmin=fltmax
      xmax=-fltmax
      ymin=fltmax
      ymax=-fltmax
c
c scan loop
c
   10 nrow=nrow+1
      if(work(i+2).gt.aflg) 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
c
      read(iarid,end=60) dummy,work
      go to 10
c
   60 xr(1)=xmin
      xr(2)=xmax
      yr(1)=ymin
      yr(2)=ymax
c
   70 return
      end
c***************
      subroutine conttr(x,y,z,f,c,nrow,ncol,xxscal,yyscal,
     1 aflg,yo,dely,iquad,npass,ires,iarid,ixad,iyad)
c
c     basic input and tier control
c
      dimension x(1),y(1),z(1),f(1),c(1)
c
c     setup mode control
c
      if(iquad.eq.3) go to 10
      call coorda(ixad,iyad,ncol,iquad)
      itype=-1
      if(dely.eq.0) itype=0
      go to 20
   10 call coorda(ixad,iyad,ncol,iquad)
      itype=1
c
c     remaining initialization
c
   20 klrow=(nrow-1)*ncol
      last=0
      ie=nrow
c
c     tier loop
c
```

```
      do 170 ipass=1,npass
      if(ipass.eq.npass) je=jres
      if(ipass.eq.1) go to 70
c
c move down last row
c
      js=2
      jj=ncol
      do 30 i=1,ncol
   30 z(i)=z(klrow+i)
      if(itype) 40,40,50
   40 v(1)=y(nrow)
      go to 80
   50 do 60 i=1,ncol
      x(i)=x(klrow+i)
   60 v(i)=y(klrow+i)
      go to 80
c
c     first row
c
   70 js=1
      ij=0
c
c     get data
c
   80 do 140 j=js,je
      if(itype) 90,100,110
   90 read(iarid,end=130) dum,(z(jj+i),i=1,ncol)
      y(j)=yo
      yo=yo+dely
      go to 140
  100 read(iarid,end=130) y(j),(z(jj+i),i=1,ncol)
      y(j)=y(j)*vyscal
      go to 140
  110 read(iarid,end=130) dum,(x(jj+i),y(jj+i),z(jj+i),i=1,ncol)
      do 120 i=1,ncol
      kk=jj+i
      if(z(kk).gt.aflg) go to 120
      x(kk)=x(kk)*xxscal
      y(kk)=y(kk)*vyscal
  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 contr
c
  150 call contr(z,f,c,c,je)
  160 if(last.ne.0) return
  170 continue
      return
      end
c***************
      subroutine contr(arid,flags,work,acval,nrowf)
c
c  basic contouring control subroutine.
c
```

```
      dimension grid(1),flags(1),work(1),acval(1)
      integer flags
      common/contrc/ cmin,cmax,dcval,ncval,nsec,gradi,nchar,size
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gfla,iis,iie,iisi,iiei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,siama,nsig
c
      logical prime
c
      nrow=nrowf
c
c  set gradient.
c
      if (nsec.le.0.or.ncval.gt.0) go to 10
      grad=(gradi*dcval)**2
c
c  set flags, etc..
c
   10 call setup(grid,flags,work)
      if (bmin.ge.bmax) go to 150
      if (ncval.eq.0) go to 50
c
c  find lower limit.
c
      do 20 i=1,ncval
      if (acval(i).le.bmin) go to 20
      ii=i
      go to 30
   20 continue
c
c  contouring loop
c
   30 prime=.true.
      do 40 i=ii,ncval
      cont=acval(i)
      if (cont.ge.bmax) go to 150
      call setlab
   40 call scan(grid,flags,work)
      go to 150
c
c  execution for delta contour levels.
c
   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
c  contouring loop.
c
   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
c  decode primary contour line thickness
   80 linet=iabs(mod(idashs,8))
      go to 140
   90 linet=0
```

```
      go to 140
  100 if (idashs) 110,130,120
  110 linet=iabs(idashs/8)*8
      go to 130
  120 linet=idashs
  130 call setlab
  140 call scan(grid,flags,work)
      icont=icont+1
      go to 70
c
c
c  done with block
  150 return
      end
c***************
      subroutine setup(grid,flags,work)
c
c  initialize flag array.
c
      dimension grid(1),flags(1),work(1)
      integer flags
c
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gfla,iis,iie,ijsi,ijei,prime,ij4,cont,
     2 fltmax,lmult(4),idasns,linet,sigma,nsig
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,colla
      data idval/o377777777777/
c
c start of operations
c
      nrow1=nrow-1
      ncol1=ncol-1
      ij=1
      i1j=2
      ij1=ncol+1
      i1j1=ij1+1
      ijn1=-ncol
      ijs=(ncol*nrow+3)/4
      do 10 i=1,ijs
   10 flags(i)=0
      ijs=idval
```

```
      ije=0
c
c oreset bmin,bmax
c
      bmin=fltmax
      bmax=-bmin
c
c start of setup scanning
c
      do 150 j=0,nrow1
      rowlg=.false.
      collrc=.false.
      if (grad.eq.0.or.j.eq.nrow1) go to 20
      call coord(work,0,j,xll,yll)
      call coord(work,0,j+1,xul,yul)
   20 continue
      do 140 i=0,ncol1
      if (j.eq.0) go to 30
      collg=bit(flags,ijn1,1)
      go to 40
   30 collg=.false.
   40 if (i.eq.ncol1) go to 90
      ccc=collg
      if (j.eq.nrow1) go to 90
c
c determine if mesh block flagged
c
      if (gflg.ne.0.and.
     1 (grid(ij).gt.gflg.or.grid(i1j).gt.gflg.or.
     2  grid(ij1).gt.gflg.or.grid(i1j1).gt.gflg)) go to 90
c
c mesh block contourable
c
      fg=1
      ijs=min0(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 (grid(ij).gt.grid(i1j)) l=4
      if (.not.collg) l=l*4
      fg=fg+l
c
c check and set gradient
c
      if (grad.eq.0.) go to 80
      call coord(work,i+1,j,xlr,ylr)
      call coord(work,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(ij1))**2.gt.
     3       ((xll-xul)**2+(yll-yul)**2)*grad) go to 60
      if((grid(ij1)-grid(i1j1))**2.gt.
```

```
     1       ((xul-xur)**2+(yul-yur)**2)*arad
     2 .or.(arid(ij1)-arid(i1j1))**2.gt.
     3       ((xur-xlr)**2+(yur-ylr)**2)*arad) go to 60
      if((arid(ij)-arid(i1j1))**2.lt.
     1       ((xll-xur)**2+(yll-yur)**2)*arad
     2 .and.(arid(i1j)-arid(ij1))**2.lt.
     3       ((xul-xlr)**2+(yul-ylr)**2)*arad) go to 70
   60 fg=fg+2
   70 xll=xlr
      yll=ylr
      xul=xur
      yul=yur
c
c determine bmin,bmax
c
   80 bmin=amin1(bmin,arid(ij))
      bmax=amax1(bmax,arid(ij))
      call set(flags,ij-1,fg)
      go to 130
c
c mesh block non-contourable
c
c check and set side 3 and 4 flags
c
   90 kk=.false.
      if (.not.rowlg) go to 100
      if (arid(ij).gt.arid(ij1)) call set(flags,ij-2,128)
      kk=.true.
  100 if (.not.collg) go to 110
      if (arid(ij).lt.arid(i1j)) call set(flags,ijn1,64)
      kk=.true.
  110 if (.not.(collrc.or.kk)) go to 120
      bmin=amin1(bmin,arid(ij))
      bmax=amax1(bmax,arid(ij))
  120 rowlg=.false.
      fg=0
c
c end of mesh
c
  130 ij=i1j
      i1j=i1j+1
      ij1=i1i1
      i1j1=i1j1+1
      collrc=ccc
      ijn1=ijn1+1
  140 continue
c
  150 continue
c
      ijs=(ijs-1)/4
      ije=(ije-1)/4
      ijsi=(ijs+ncol)/4
      ijei=ije
      return
c
      end
c***************
      subroutine scan(arid,flags,work)
c
c  scans arid for undrafted contour level.
```

```
c
      dimension grid(1),flags(1),work(1)
      integer flags
c
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gflg,ijs,ije,ijsi,ijei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
      logical prime,mask1,mask2,mask3,mask4,mask5,mask6,
     1 mask7,mask8,mask9,mask10,mask11,izero
      logical mask,maskb,maskd,maskf,jtemp,mtemp,lflag,ktemp,edge
      equivalence(mask1,mask1),(mask2,mask2),(mask3,mask3),
     1 (mask4,mask4),(mask5,mask5),(mask6,mask6),(mask7,mask7),
     2 (mask8,mask8),(mask9,mask9),(mask10,mask10),(mask11,mask11)
      equivalence(mask,mask),(maskb,maskb),(maskd,maskd),(maskf,maskf)
      equivalence(itemp,jtemp),(ltemp,mtemp),(lflag,mflag),
     1 (ktemp,kktemp)
      data mask1/o4004004004/,mask2/o360360360360/,mask3/o360/,
     1 mask4/o20/
      data mask5/o0100100100100/,mask6/o010/,mask7/o4/,
     1 mask8/o200/,mask9/o100/,mask10/o40/,mask11/o0040040040040/
      data izero/.false./
c
c
c  set interior flags.
c
      if (ijsi.lt.0) go to 20
      do 10 ij=ijsi+1,ijei+1
      itemp=flags(ij)
      mtemp=jtemp.and.mask1
      ltemp=ltemp*2
      ktemp=jtemp.or.mtemp
   10 flags(ij)=kktemp
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.
      ija=ijs
      ijb=ije
      ijsnew=-1
      mask=mask2
      maskb=mask3
      maskd=mask4
      maskf=1
   30 ij4b=ija*4+3
c
c  basic word scan
c
      do 170 ij=ija,ijb
      mflag=flags(ij+1)
      if ((lflag.and.mask).eq.izero) 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 ((iflag.and.maskb).eq.izero) go to 130
      go to iswa,(40,110)
c
c  right edge
c
   40 if ((iflag.and.mask8).eq.izero) go to 60
      if (cont.le.grid(ij4+2)) go to 50
      call reset(flags,ij4,nask8)
      go to 80
   50 if (cont.le.grid(ij4n+2)) go to 60
      call trace(grid,flags,work,4,naskf)
      go to 80
c
c  top edge
c
   60 if ((iflag.and.mask9).eq.izero) go to 80
      if (cont.le.grid(ij4n+2)) go to 70
      call reset(flags,ij4,nask9)
      go to 100
   70 if (cont.le.grid(ij4n+1)) go to 80
      call trace(grid,flags,work,3,naskf)
      go to 100
c
c  left edge
c
   80 if ((iflag.and.mask10).eq.izero) go to 100
      if (cont.le.grid(ij4n+1)) go to 90
      call reset(flags,ij4,nask10)
      go to 130
   90 if (cont.le.grid(ij4+1)) go to 100
      call trace(grid,flags,work,2,naskf)
      go to 130
c
c  bottom edge and interior scan
c
  100 if ((iflag.and.mask4).eq.izero) go to 130
  110 if (cont.le.grid(ij4+1)) go to 120
      call reset(flags,ij4,naskd)
      go to 130
  120 if (cont.gt.grid(ij4+2))
     1 call trace(grid,flags,work,1,naskf)
c
  130 mflag=mflag/512
      ij4=ij4-1
      ij4n=ij4n-1
  140 continue
c
      go to 160
c
c  interior range set
c
  150 if ((iflag.and.mask11).eq.izero) go to 160
      if (ijsnew.lt.0) ijsnew=ij
      ijenew=ij
  160 ij4b=ij4b+4
c
  170 continue
c
c  first time?
```

```
c
      if (.not.edge) go to 190
c
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=ijei
      nask=nask5
      naskb=nask6
      naskd=nask7
      naskf=naskb
      go to 30
c
c  update interior boundary indicies
c
  190 ijsi=ijsnew
      ijei=ijenew
c
c  end of scan
c
  200 return
c
      end
c***************
      subroutine trace(grid,flags,work,iside,mask)
c
c  follows contour through grid until
c  edge or closure found.
c
      dimension grid(1),flags(1),work(1)
      integer flags
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gflg,ijs,ije,ijsi,ijei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
c
      logical prime,bit,post
c
c trace and plot contour through grid
c
c
c grid indexing
c                    side 3
c          i01 +             + i11
c
c      side 2                    side 4
c
c          i00 +             + i10
c                    side 1
c
c in-- entrance side
```

```
c high=equal point(inn) always on left when looking along
c      contour line
c iol-- point opposite entrance side on left
c ior-- point opposite entrance side on right
c
       dimension x(101),y(101)
       data mask6/o10/
c
c set up tracing start
c
       post=prime
       i=mod(ij4,ncol)
       j=ij4/ncol
       npts=1
       ic=0
       i00=ij4+1
       i10=i00+1
       i01=i00+ncol
       i11=i01+1
       go to (10,20,30,40),iside
    10 call reset(flags,i00-1,mask6)
       fract=(cont-grid(i00))/(grid(i10)-grid(i00))
       call coord(work,i,j,xa,ya)
       call coord(work,i+1,j,xb,yb)
       iol=i01
       ior=i11
       in=1
       go to 50
    20 fract=(cont-grid(i00))/(grid(i01)-grid(i00))
       call coord(work,i,j,xa,ya)
       call coord(work,i,j+1,xb,yb)
       iol=i11
       ior=i10
       in=2
       go to 50
    30 fract=(cont-grid(i01))/(grid(i11)-grid(i01))
       call coord(work,i,j+1,xa,ya)
       call coord(work,i+1,j+1,xb,yb)
       iol=i10
       ior=i00
       in=3
       go to 50
    40 fract=(cont-grid(i10))/(grid(i11)-grid(i10))
       call coord(work,i+1,j,xa,ya)
       call coord(work,i+1,j+1,xb,yb)
       iol=i00
       ior=i01
       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.bit(flags,i00-1,2)) go to 60
       npts=npts+1
       if (npts.le.100) go to 70
       call conplot(x,y,npts,ic,post,1)
       go to 70
    60 if(npts.eq.1) go to 70
       call conplot(x,y,npts,ic,post,2)
```

```
c
c determine exit side
c
   70 k=in+2
      if (grid(ior).lt.cont) k=k-1
      if (grid(iol).lt.cont) k=k-2
      if (k-in) 90,80,100
c
c saddle decision (dayhoff)
c
   80 if ((grid(i00)+grid(i10)+grid(i01)+grid(i11))*0.25.lt.cont)
     1 go to 100
      k=in+2
      go to 100
   90 k=in
c
c     compute side branch
c
  100 go to (120,130,140,110,120,130),k
c
c exit bottom -- side 1
c
  110 fract=(cont-grid(i00))/(grid(i10)-grid(i00))
      call coord(work,i,j,xa,ya)
      call coord(work,i+1,j,xb,yb)
      if (j.eq.0) go to 150
      i01=i00
      i00=i00-ncol
      if (.not.bit(flags,i00-1,1)) go to 150
      in=3
      i11=i10
      i10=i10-ncol
      iol=i10
      ior=i00
      j=j-1
      go to 50
c
c exit left -- side 2
c
  120 fract=(cont-grid(i00))/(grid(i01)-grid(i00))
      call coord(work,i,j,xa,ya)
      call coord(work,i,j+1,xb,yb)
      if (i.eq.0) go to 150
      i10=i00
      i00=i00-1
      if (.not.bit(flags,i00-1,1)) go to 150
      in=4
      i11=i01
      i01=i01-1
      iol=i00
      ior=i01
      i=i-1
      go to 50
c
c exit top -- side 3
c
  130 fract=(cont-grid(i01))/(grid(i11)-grid(i01))
      call coord(work,i,j+1,xa,ya)
      call coord(work,i+1,j+1,xb,yb)
      if (.not.bit(flags,i01-1,mask)) go to 150
```

```
      call reset(flags,i01-1,mask6)
      in=1
      i00=i01
      i10=i11
      i01=i01+ncol
      i11=i11+ncol
      iol=i01
      ior=i11
      j=i+1
      go to 50
c
c exit right -- side 4
c
  140 fract=(cont-grid(i10))/(grid(i11)-grid(i10))
      call coord(work,i+1,i,xa,ya)
      call coord(work,i+1,j+1,xb,yb)
      if (.not.bit(flags,i10-1,1)) go to 150
      in=2
      i00=i10
      i01=i11
      i10=i10+1
      i11=i11+1
      iol=i11
      ior=i10
      i=i+1
      go to 50
c
c end of trace loop
c
  150 x(npts)=xa+(xb-xa)*fract
      y(npts)=ya+(yb-ya)*fract
      if (npts.le.1.and.ic.eq.0) go to 160
      call conplot(x,y,npts,ic,post,3)
  160 return
c
c end of tracing
c
      end
c***************
      logical function bit(lf,i,mask)
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,qflg,iis,iie,iisi,ijei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
      dimension lf(1)
      integer three
      logical ii,jj,kk,ll,mm,izero
      equivalence (i1,ii),(three,jj),(mask1,kk),(l,ll),(mm,mmm)
      data three/3/,izero/.false./
      i1=i
      mask1=mask
      mm=ii.and.jj
      index=mmm+1
      j=i/4+1
      l=lf(j)/lmult(index)
      bit=(ll.and.kk).ne.izero
      return
      end
c***************
      subroutine set(lf,i,mask)
      common /concom/ ncol,nrow,bmin,bmax,
```

```
     1 grad,gflg,iis,iie,iisi,iiei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,siama,nsig
      dimension lf(1)
      integer three
      logical ii,jj,kk,ll,mm
      equivalence (i1,ii),(three,jj),(k,kk),(l,ll),(mm,mmm)
      data three/3/
      i1=i
      j=i/4+1
      l=lf(j)
      mm=ii.and.jj
      index=mmm+1
      k=mask*lmult(index)
      ii=ll.or.kk
      lf(j)=i1
      return
      end
c***************
      subroutine reset(lf,i,mask)
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,gflg,iis,iie,iisi,iiei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,siama,nsig
      dimension lf(1)
      integer three,ones
      logical ii,jj,kk,ll,mm
      equivalence (i1,ii),(three,jj),(k,kk),(l,ll),(mm,mmm)
      data three/3/,ones/o777777777777/
      i1=i
      mm=ii.and.jj
      index=mmm+1
      k=mask*lmult(index)
      call xor(k,ones,mmm)
      j=i/4+1
      l=lf(j)
      ii=ll.and.mm
      lf(j)=i1
      return
      end
c***************
      subroutine xor(a,b,c)
c  exclusive or.  upon transport, check
c  for proper function
      logical a,b,c,ab
      equivalence(ab,iab)
      ab=a.and.b
      iab=-1-iab
      c=(a.or.b).and.ab
      return
      end
c***************
      subroutine coorda(ixad,ivad,ncol,iquad)
      dimension x(1)
      k=ixad
      l=ivad
      ncoll=ncol
      isw=iquad
      return
      entry coord(x,i,j,xv,yv)
      go to (10,10,20),isw
   10 xv=x(k+i)
```

```
      yv=x(l+j)
      return
   20 ll=j*ncoll+i
      xv=x(k+ll)
      yv=x(l+ll)
      return
      end
c***************
      subroutine conplot(x,y,npts,ic,post,isw)
      common/concom/ncol,nrow,bmin,bmax,grad,gflg,ijs,ije,
     1 ijsi,ijei,prime,ij4,cont,fltmax,lmult(4),idashs,linet,
     2 sigma,nsig
      dimension x(101),y(101),xx(400),yy(400),xp(100),yp(100),temp(200)
      logical post,prime
c
      if(npts.le.1) go to 100
      if(isw.eq.1) npts=100
      if(nsig.eq.0) go to 150
c
c     check for duplicate points
c
      mmt=0
      do 10 mt=2,npts
      if(x(mt).eq.x(mt-1).and.y(mt).eq.y(mt-1)) go to 20
   10 continue
      go to 40
   20 do 30 lt=mt,npts
      if(x(lt).eq.x(lt-1).and.y(lt).eq.y(lt-1)) mmt=mmt+1
      llt=lt-mmt
      x(llt)=x(lt)
      y(llt)=y(lt)
   30 continue
      npts=npts-mmt
      if(npts.le.1) return
c
c     spline points
c
   40 call kurv1(npts,x,y,slp1,slpn,xp,yp,temp,s,sigma)
      nnt=(npts-1)*nsig
      t1=1./nnt
      t2=-t1
      nnt=nnt+1
      do 50 int=1,nnt
      t1=t1+t2
      call kurv2(t1,xs,ys,npts,x,y,xp,yp,s,sigma)
      xx(int)=xs
      yy(int)=ys
   50 continue
      go to (60,90,110),isw
   60 if (post) go to 70
      call line(xx,yy,nnt,ic,linet)
      post=prime
      go to 80
   70 call label(xx,yy,nnt,ic,post)
   80 npts=1
      ic=1
      return
   90 call line(xx,yy,nnt,ic,linet)
      ic=0
      npts=1
```

```
      return
  100 xx(1)=x(1)
      yy(1)=y(1)
      nnt=npts
  110 if(post) go to 120
      call line(xx,yy,nnt,ic,linet)
      return
  120 call label(xx,yy,nnt,ic,post)
      return
c
c     no splining
c
  130 go to(140,170,180),isw
  140 if(post) go to 150
      call line(x,y,npts,ic,linet)
      post=prime
      go to 160
  150 call label(x,y,npts,ic,post)
  160 npts=1
      ic=1
      return
  170 call line(x,y,npts,ic,linet)
      npts=1
      ic=0
      return
  180 if(post) go to 190
      call line(x,y,npts,ic,linet)
      return
  190 call label(x,y,npts,ic,post)
      return
      end
c***************
      subroutine label(x,y,npts,icc,post)
c
c  label scans contour line array looking for
c  a straight segment.
c  if found,  the contour is labelled.
c
      common/labcom/charst,fmtc,nchf,size,wdist,wdist2,ccor
      dimension x(1),y(1),xxa(1),yya(1)
      common /concom/ ncol,nrow,bmin,bmax,
     1 grad,aflg,ijs,ije,ijsi,ijei,prime,ij4,cont,
     2 fltmax,lmult(4),idashs,linet,sigma,nsig
c
      character*16 fmtc
      integer charst(6)
      logical nolab,post,prime
      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.wdist2) 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 fine spot, continue line without labelling.
   50 call line(x,y,npts,icc,linet)
c
      go to 70
c
c  plottable location.
c
   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)
      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.)
      xxa(1)=xa
      yya(1)=ya
      call line(xxa(1),yya(1),1,0,linet)
      call line(x(ie),y(ie),npts-ie+1,1,linet)
      post=.false.
c
   70 return
c
c  entry to set labelling constants.
c  must be called before setlab.
c
      entry setcon(siz,ncha)
      nchar=ncha
      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.
c
      entry setlab
      if (nolab) go to 80
      encode(charst,fmtc) cont
      nchf=nchar
      call leftj(charst,nchf)
      wdist=(nchf+1)*size
      wdist2=wdist*wdist
```

```
      ccor=-.5*wdist+size
   80 return
      end
c***************
      subroutine sexual(ier)
c
c  general setup of contour niceties.
c
      dimension it1(14),it2(14),it3(14)
      character*56 title1,title2,title3
      integer fmtx1(4),fmty1(4)
      character*16 fmtx,fmty
      real latm,latx,lonam,lonax
      dimension idim(1)
      common/sexy/title1,title2,title3,fmtx,fmty,
     1 xx(2),yy(2),xp1(4),yp1(4),iplotr,sizel,ncharx,nchary,
     2 sizex,sizey,adelx,adely,pllx,plly,lintx,linty,
     3 xscale,yscale,xxscal,yyscal,mscale
      common/llp/latm(3),latx(3),lonam(3),lonax(3),cm(3),baslat(3),
     1 iproj,xxx(2),yyy(2),sizep,unit,ip,neat,tint,itpost,ibound
      equivalence (title1,it1(1)),(title2,it2(1)),(title3,it3(1)),
     1 (fmtx,fmtx1(1)),(fmty,fmty1(1))
      data iprint/6/,idim(1)/0/
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.
      if (sizep.lt.0.) sizep=0.
      sizelb=1.5*sizel
c
c  margin requirement setup.
c
      if (sizex.gt.0..and.ncharx.gt.0) go to 10
      ncharx=0
      sizex=0.
   10 if (sizey.gt.0..and.nchary.gt.0) go to 20
      nchary=0
      sizey=0.
   20 if (pllx.le.0.) go to 30
      xp1(3)=pllx
      go to 40
   30 xp1(3)=(ncharx+0.6)*sizex
   40 if (plly.le.0.) go to 50
      yp1(3)=plly
      go to 60
   50 yp1(3)=(nchary+0.6)*sizey+4.0*sizelb
c
   60 dx=xx(2)-xx(1)
      dy=yy(2)-yy(1)
      call pltset(iplotr,xp1(4),yp1(4),idim)
c
c  check scaling
c
      if (xscale.le.0.) go to 70
```

```
      if (yscale.le.0.) yscale=xscale
      go to 80
   70 if (yscale.le.0.) go to 90
      xscale=yscale
   80 xxscal=sign(1./xscale,dx)
      yyscal=sign(1./yscale,dy)
c
c  fixed scaling.
c
      xp1(1)=abs(dx*xxscal)
      yp1(1)=abs(dy*yyscal)
      xp1(4)=xp1(1)+xp1(3)+sizex+13.*sizep
      yp1(4)=yp1(1)+yp1(3)+sizey+3.*sizep
      go to 120
c
c  relative scaling, eg. xxscal=yyscal=0.
c
   90 xp1(4)=amin1(xp1(4),10.)
      yp1(4)=amin1(yp1(4),8.)
      xp1(1)=xp1(4)-xp1(3)-sizex-6.*sizep
      yp1(1)=yp1(4)-yp1(3)-sizey-3.*sizep
      if (xp1(1).gt.0..and.yp1(1).gt.0.) go to 110
      write(iprint,100)
  100 format(" %margin requires all plot area")
      ier=2
      go to 180
  110 xxscal=xp1(1)/dx
      yyscal=yp1(1)/dy
      xy=amin1(abs(xxscal),abs(yyscal))
      xxscal=sign(xy,xxscal)
      yyscal=sign(xxscal,yyscal)
      xp1(1)=abs(dx*xxscal)
      yp1(1)=abs(dy*yyscal)
c
c  initial labelling scale call
c
  120 call scale(xx,yy,xp1,yp1,4,ier)
      if (ier.ne.0) go to 180
      if(iplotr.ne.1 .and. iplotr.ne.4) write(iprint,130) xp1(4),yp1(4)
  130 format(/," plot size (inches): x=",f7.2,"  y=",f7.2)
c
c  plot labels.
c
      if (sizel.le.0.) go to 140
      call char(sizelb,sizelb,it3,5b,3,sizel,0.,0.,0.)
      yt=sizelb+sizelb
      call char(sizelb,yt,it2,5b,3,sizel,0.,0.,0.)
      yt=yt+sizelb
      call char(sizelb,yt,it1,5b,3,sizel,0.,0.,0.)
c
c  plot axis.
c
  140 if(adelx.eq.0.) go to 150
      call xaxis(xx,yy,xp1,adelx,lintx,sizex,fmtx1,ncharx)
  150 if(adely.eq.0.) go to 160
      call yaxis(yy,xx,yp1,adely,linty,sizey,fmty1,ncharv)
  160 if(neat.eq.0) call neat1
      xxx(1)=xx(1)
      xxx(2)=xx(2)
      yyy(1)=yy(1)
```

```
      yyv(2)=yv(2)
      if(iproj.eq.999 .or. tint.eq.0) go to 170
c
c     plot lat-long
c
      call llpost
  170 if(ibound .ne. 999) call state
c
c  rescale plotter for grid.
c
      xx(1)=xx(1)*xxscal
      xx(2)=xx(2)*xxscal
      yy(1)=yy(1)*yyscal
      yy(2)=yy(2)*yyscal
      call scale(xx,yy,xp1,yp1,3,ier)
  180 return
      end
c***************
      subroutine kurv1(n,x,y,slp1,slpn,xp,yp,temp,s,sigma)
c
c     spline under tension routine
c
      dimension x(n),y(n),xp(n),yp(n),temp(n)
      degrad=1.745329e-2
      nm1=n-1
      np1=n+1
      delx1=x(2)-x(1)
      dely1=y(2)-y(1)
      dels1=sqrt(delx1*delx1+dely1*dely1)
      dx1=delx1/dels1
      dy1=dely1/dels1
c     determine slopes if necessary
      if(sigma.lt.0.) go to 70
      slpp1=slp1*degrad
      slppn=slpn*degrad
c     set up right hand sides of tridiagonal system for xp and yp
   10 xp(1)=dx1-cos(slpp1)
      yp(1)=dy1-sin(slpp1)
      temp(1)=dels1
      s=dels1
      if(n.eq.2) go to 30
      do 20 i=2,nm1
      delx2=x(i+1)-x(i)
      dely2=y(i+1)-y(i)
      dels2=sqrt(delx2*delx2+dely2*dely2)
      dx2=delx2/dels2
      dy2=dely2/dels2
      xp(i)=dx2-dx1
      yp(i)=dy2-dy1
      temp(i)=dels2
      delx1=delx2
      dely1=dely2
      dels1=dels2
      dx1=dx2
      dy1=dy2
c     acculumulate polygonal arclength
      s=s+dels1
   20 continue
   30 xp(n)=cos(slppn)-dx1
      yp(n)=sin(slppn)-dy1
```

```
c       denormalize tension factor
        sigmap=abs(sigma)*float(n-1)/s
c       preform forward elimination on tridiagonal system
        dels=sigmap*temp(1)
        exps=exp(dels)
        sinhs=.5*(exps-1./exps)
        sinhin=1./(temp(1)*sinhs)
        diag1=sinhin*(dels*.5*(exps+1./exps)-sinhs)
        diagin=1./diag1
        xp(1)=diagin*xp(1)
        yp(1)=diagin*yp(1)
        spdiag=sinhin*(sinhs-dels)
        temp(1)=diagin*spdiag
        if(n.eq.2) go to 50
        do 40 i=2,nm1
        dels=sigmap*temp(i)
        exps=exp(dels)
        sinhs=.5*(exps-1./exps)
        sinhin=1./(temp(i)*sinhs)
        diag2=sinhin*(dels*(.5*(exps+1./exps))-sinhs)
        diagin=1./(diag1+diag2-spdiag*temp(i-1))
        xp(i)=diagin*(xp(i)-spdiag*xp(i-1))
        yp(i)=diagin*(yp(i)-spdiag*yp(i-1))
        spdiag=sinhin*(sinhs-dels)
        temp(i)=diagin*spdiag
        diag1=diag2
   40   continue
   50   diagin=1./(diag1-spdiag*temp(nm1))
        xp(n)=diagin*(xp(n)-spdiag*xp(nm1))
        yp(n)=diagin*(yp(n)-spdiag*yp(nm1))
c       preform back substitution
        do 60 i=2,n
        ibak=np1-i
        xp(ibak)=xp(ibak)-temp(ibak)*xp(ibak+1)
        yp(ibak)=yp(ibak)-temp(ibak)*yp(ibak+1)
   60   continue
        return
   70   if(n.eq.2) go to 80
c       if no slopes are given, use second order interpolation on
c       input data for slopes at endpoints
        dels2=sqrt((x(3)-x(2))**2+(y(3)-y(2))**2)
        dels12=dels1+dels2
        c1=-(dels12+dels1)/dels12/dels1
        c2=dels12/dels1/dels2
        c3=-dels1/dels12/dels2
        sx=c1*x(1)+c2*x(2)+c3*x(3)
        sy=c1*y(1)+c2*y(2)+c3*y(3)
        slpp1=atan2(sy,sx)
        delnm1=sqrt((x(n-2)-x(nm1))**2+(y(n-2)-y(nm1))**2)
        deln=sqrt((x(nm1)-x(n))**2+(y(nm1)-y(n))**2)
        delnn=delnm1+deln
        c1=(delnn+deln)/delnn/deln
        c2=-delnn/deln/delnm1
        c3=deln/delnn/delnm1
        sx=c3*x(n-2)+c2*x(nm1)+c1*x(n)
        sy=c3*y(n-2)+c2*y(nm1)+c1*y(n)
        slpn=atan2(sy,sx)
        go to 10
c       if only two points and no slopes are given, use straight
c       line segment for curve
```

```
   80 xp(1)=0.
      xp(2)=0.
      vp(1)=0.
      vp(2)=0.
      s=dels1
      return
      end
c**************
      subroutine kurv2(t,xs,ys,n,x,y,xp,yp,s,sigma)
c
c     spline under tension routine
c
      dimension x(n),y(n),xp(n),yp(n)
c     denormalize sigma
      sigmap=abs(sigma)*float(n-1)/s
c     stretch unit interval into arclength distance
      tn=abs(t*s)
c     for negative t start search where previously terminated
c     otherwise start from beginning
      if(t.lt.0) go to 10
      i1=2
      sum=0.
   10 continue
c     determine into which segment tn is mapped
      do 30 i=i1,n
      delx=x(i)-x(i-1)
      dely=y(i)-y(i-1)
      dels=sqrt(delx*delx+dely*dely)
      if(sum+dels-tn) 20,40,40
   20 sum=sum+dels
   30 continue
c     if abs(t) is greater than 1., return terminal point
c     on curve
      xs=x(n)
      ys=y(n)
      return
c     set up and perform interpolation
   40 del1=tn-sum
      del2=dels-del1
      exps1=exp(sigmap*del1)
      sinhd1=.5*(exps1-1./exps1)
      exps=exp(sigmap*del2)
      sinhd2=.5*(exps-1./exps)
      exps=exps1*exps
      sinhs=.5*(exps-1./exps)
      xs=(xp(i)*sinhd1+xp(i-1)*sinhd2)/sinhs+
     1 ((x(i)-xp(i))*del1+(x(i-1)-xp(i-1))*del2)/dels
      ys=(yp(i)*sinhd1+yp(i-1)*sinhd2)/sinhs+
     1 ((y(i)-yp(i))*del1+(y(i-1)-yp(i-1))*del2)/dels
      i1=i
      return
      end
c***************
      subroutine state
c
c     plot world data bank 2 u.s.boundaries on contour map
c
      common/llp/latm(3),latx(3),lonam(3),longx(3),cm(3),baslat(3),
     1 iproj,xxx(2),yyy(2),sizep,unit,ip,neat,tint,itpost,ibound
      dimension xlat(500),xlon(500),xx(500),yy(500)
```

```
      integer rank
      real latm,latx,lonam,lonax,llat,llong
      data in/14/,in1/15/
      dmstod(a,b,c)=a+sign(b,a)*.01666667+sign(c,a)*.0002777R
      pbs=dmstod(baslat(1),baslat(2),baslat(3))
      xlatm=dmstod(latm(1),latm(2),latm(3))
      xlatx=dmstod(latx(1),latx(2),latx(3))+.0001
      d=-1.
      pcm=sign(dmstod(cm(1),cm(2),cm(3)),d)
      xlonam=sign(dmstod(lonam(1),lonam(2),lonam(3)),d)
      xlonax=sign(dmstod(lonax(1),lonax(2),lonax(3)),d)
      if(abs(xlatx).le.abs(xlatm).or.abs(xlongx).le.abs(xlonam))
     1 return
      open(in,attach="vfile_ >online>Reg>Pams>wdb2.table")
      open(in1,attach="vfile_ >online>Reg>Pgms>wdb2.dat",
     1 access="direct")
      if(iproj.eq.999) go to 60
      sfac=39.370079
      if(unit.eq.1.) sfac=1.
      if(unit.eq.2.) sfac=.001
      if(iproj.gt.4) call setalb(iproj)
c
c     call with baslat & cm  to determine y(pbs) at baslat
c
      xxxx=pbs
      call project(xxxx,pcm,yy,pbs,pcm,sfac,iproj)
c
c     plot boundaries on projected map
c
   10 read(in,end=110) lineid,rank,np,lbeg,llat,llong,ulat,ulong
      if(llat.gt.xlatx) go to 110
      if(ulat.lt.xlatm) go to 10
      if(llong.gt.xlonam) go to 10
      if(ulong.lt.xlonax) go to 10
      nrec=np*.002+1
      if(mod(np,500).eq.0) nrec=nrec-1
      irec=lbeg-1
      icon=0
      do 30 i=1,nrec
      l=i*500
      if(l.gt.np) go to 40
      irec=irec+1
      read(in1'irec) (xlat(j),xlon(j),j=1,500)
      do 20 k=1,500
      call project(xlat(k),xlon(k),xx(k),yyyy,pcm,sfac,iproj)
      yy(k)=yyyy-pbs
   20 continue
      call line(xx,yy,500,icon,ibound)
   30 continue
      go to 10
   40 nr=np-(l-500)
      irec=irec+1
      read(in1'irec) (xlat(i),xlon(i),i=1,nr)
      do 50 k=1,nr
      call project(xlat(k),xlon(k),xx(k),yyyy,pcm,sfac,iproj)
      yy(k)=yyyy-pbs
   50 continue
      call line(xx,yy,nr,icon,ibound)
      go to 10
c
```

```
c     plot boundaries on unprojected map where the coordinates
c     are in lat/lon seconds
c
   60 read(in,end=110) lineid,rank,np,lbeg,llat,llong,ulat,ulong
      if(llat.gt.xlatx) go to 110
      if(ulat.lt.xlatm) go to 60
      if(llong.gt.xlonam) go to 60
      if(ulong.lt.xlonax) go to 60
      nrec=np*.002+1
      if(mod(np,500).eq.0) nrec=nrec-1
      irec=lbeg-1
      icon=0
      do 80 i=1,nrec
      l=i*500
      if(l.gt.np) go to 90
      irec=irec+1
      read(in1'irec) (xlat(j),xlon(j),j=1,500)
      do 70 k=1,500
      xx(k)=sign(xlon(k),xxx(2))*3600.
      yy(k)=xlat(k)*3600.
   70 continue
      call line(xx,yy,500,icon,ibound)
   80 continue
      go to 60
   90 nr=np-(l-500)
      irec=irec+1
      read(in1'irec) (xlat(i),xlon(i),i=1,nr)
      do 100 k=1,nr
      xx(k)=sign(xlon(k),xxx(2))*3600.
      yy(k)=xlat(k)*3600.
  100 continue
      call line(xx,yy,nr,icon,ibound)
      go to 60
  110 close(in)
      close(in1)
      return
      end
c***************
      subroutine llpost
c
c     post lat-long tick marks on contour map
c     for longitude east set lonax < lonam
c
      common/llp/latm(3),latx(3),lonam(3),longx(3),cm(3),baslat(3),
     1 iproj,xxx(2),yyy(2),sizep,unit,iplotr,neat,tint,itpost,ib
      integer a,a1,ct(1)
      real latm,latx,lonam,lonax
      dimension a(3),zx(100),zy(100),ilod(100),
     1 ilom(100),ilad(100),ilam(100),ilas(100),ilos(100)
      logical nos,nom
      data conm/.0166667/,cons/.0002778/,a1/"-"/,ct/"+"/,num/100/
      dmstod(aa,b,c)=aa+sign(b,aa)*conm+sign(c,aa)*cons
c
      if(abs(cm(1)).gt.360..or.abs(baslat(1)).gt.90.) return
      nbs=dmstod(baslat(1),baslat(2),baslat(3))
      xlonam=dmstod(lonam(1),lonam(2),lonam(3))
      xlonax=dmstod(lonax(1),lonax(2),lonax(3))
      d=-1.
      if(abs(xlonax).lt.abs(xlonam)) d=1.
      ncm=sign(dmstod(cm(1),cm(2),cm(3)),d)
```

```
      xlonam=sign(xlonam,d)+1.e-5
      xlonax=sign(xlonax,d)
      xlatm=dmstod(latm(1),latm(2),latm(3))
      xlatx=dmstod(latx(1),latx(2),latx(3))+1.e-5
      if(abs(xlatx).le.abs(xlatm)) return
      tint=abs(tint)
      xinc=tint/60.
      id1=int(tint*60.)
      id2=sign(id1,d)
      sfac=39.370079
      if(unit.eq.1.) sfac=1.
      if(unit.eq.2.) sfac=.001
   10 if(iproj.gt.4) call setalb(iproj)
c
      call prjctl(obs,pcm,dum,vb,pcm,sfac,iproj)
c
c     generate lat-long tick values
c
      longx(1)=abs(longx(1))
      longx(2)=abs(longx(2))
      longx(3)=abs(longx(3))
      ilod(1)=longx(1)
      ilom(1)=longx(2)-amod(longx(2),tint)
      ilos(1)=longx(3)-amod(longx(3),float(id2))
      labl=ilod(1)*3600+ilom(1)*60+ilos(1)
      zx(1)=sign(dmstod(ilod(1),ilom(1),ilos(1)),d)
      do 20 i=2,num
      zx(i)=zx(i-1)+xinc
      labl=labl+id2
      ilod(i)=int(float(labl)*2.777778e-4)
      ilom(i)=int(float(labl-ilod(i)*3600)*1.6666667e-2)
      ilos(i)=labl-ilod(i)*3600-ilom(i)*60
      if(zx(i).gt.xlonam) go to 30
   20 continue
   30 nlon=i-1
      ilad(1)=latm(1)
      ilam(1)=latm(2)-amod(latm(2),tint)
      ilas(1)=latm(3)-amod(latm(3),float(id1))
      labl=ilad(1)*3600+ilam(1)*60+ilam(3)
      zy(1)=dmstod(ilad(1),ilam(1),ilas(1))
      do 40 i=2,num
      zy(i)=zy(i-1)+xinc
      labl=labl+id1
      ilad(i)=int(float(labl)*2.777778e-4)
      ilam(i)=int(float(labl-ilad(i)*3600)*1.6666667e-2)
      ilas(i)=labl-ilad(i)*3600-ilam(i)*60
      if(zy(i).gt.xlatx) go to 50
   40 continue
   50 nlat=i-1
      nos=.false.
      tmp=tint*itpost
      if(tmp-aint(tmp) .eq. 0.) nos=.true.
      nom=.false.
      if(amod(tint*float(itpost),60.).eq.0.) nom=.true.
      js=0
      je=nlat
      do 60 j=1,nlat
      call prjctl(zy(j),zx(1),dum,vdist,pcm,sfac,iproj)
      vdist=vdist-vb
      if(vdist.ge.vyy(1) .and. js.eq.0) js=j
```

```
      if(ydist.le.vyv(2)) je=j
   60 continue
      if(js.eq.0) js=1
      is=0
      ie=nlon
      do 70 i=1,nlon
      call prjctl(zy(js),zx(i),xdist,dum,pcm,sfac,iproj)
      if(xdist.ge.xxx(1) .and. is.eq.0) is=i
      if(xdist.le.xxx(2)) ie=i
   70 continue
      if(is.eq.0) is=1
c
c     plot longitude labels
c
      do 80 i=is,ie,itpost
      ilod(i)=iabs(ilod(i))
      call prjctl(zy(js),zx(i),xdist,dum,pcm,sfac,iproj)
      call latlab(1.,0,sizep,xdist,yvy(1),ilod(i),ilom(i),
     1 ilos(i),nos,nom)
      if(iplotr.eq.1 .or. iplotr.eq.4) go to 80
      call prjctl(zy(je),zx(i),xdist,dum,pcm,sfac,iproj)
      call latlab(-1.,0,sizep,xdist,vyv(2),ilod(i),ilom(i),
     1 ilos(i),nos,nom)
   80 continue
c
c     plot latitude labels
c
      do 90 j=js,je,itpost
      call prjctl(zy(j),zx(is),dum,ydist,pcm,sfac,iproj)
      ydist=ydist-yb
      call latlab(1.,1,sizep,xxx(1),ydist,ilad(j),ilam(j),
     1 ilas(j),nos,nom)
      if(iplotr.eq.1 .or. iplotr.eq.4) go to 90
      call prjctl(zy(j),zx(ie),dum,ydist,pcm,sfac,iproj)
      ydist=ydist-yb
      call latlab(-1.,1,sizep,xxx(2),ydist,ilad(j),ilam(j),
     1 ilas(j),nos,nom)
   90 continue
c
c     plot lat-long marks
c
      sz=sizep*1.5
      do 110 i=is,ie
  100 do 110 j=js,je
      call prjctl(zy(j),zx(i),x,y,pcm,sfac,iproj)
      y=y-yb
      call char(x,y,ct(1),1,2,sz,0.,0.,0.)
  110 continue
      return
      end
c***************
      subroutine latlab(side,lat,sz,x,y,jd,jm,js,nosec,nomin)
      external char(descriptors)
      character c3*3,c2*2,c1*1,cs*4
      equivalence (cs,ics)
      logical nosec,nomin
      data ics/o"42000000000"/
c cs is double quotes
      if(jm.ne.0 .or. js.ne.0) nomin=.false.
      if(js.ne.0) nosec=.false.
```

```
      hsz=sz*.5
      xof=3.*sz
      if(lat.eq.1) go to 10
      yof=-(2.*sz*side)
      xc=1.5*sz
      if(nosec) xc=0.
      if(nomin) xc=-1.5*sz
      go to 30
   10 yof=0.
      if(side.eq.-1.) go to 20
      xc=8.*sz
      if(nosec) xc=4.*sz
      if(nomin) xc=sz
      go to 30
   20 xc=3.*sz*side
   30 encode(c3,40) id
   40 format(i3)
      call char(x,y,c3,3,2,sz,0.,-(xof+xc),yof)
      c1="o"
      call char(x,y,c1,1,2,hsz,0.,-xc,hsz+yof)
      if(nomin) return
      encode(c2,50) im
   50 format(i2)
      call char(x,y,c2,2,2,sz,0.,sz-xc,yof)
      c1="'"
      call char(x,y,c1,1,2,sz,0.,xof-xc,yof)
      if(nosec) return
      encode(c2,50) is
      xof=xof+sz
      call char(x,y,c2,2,2,sz,0.,xof-xc,yof)
      c1=cs
      xof=xof+sz*2.
      call char(x,y,c1,1,2,sz,0.,xof-xc,yof)
      return
      end
c***************
      subroutine hilow(jx,zmax,jm,zmin,w,jref,nc2,iw,size1)
c
c  routine to plot h or l symbols at local maxima/minima on contours.
c  'local' is defined by a search radius in grid units, parameter 'iw'
c  is the window size ... 2*radius+1.
c  coded by Mike Webring 1/80.
c
      dimension w(nc2,iw),jx(nc2),jm(nc2),zmax(nc2),zmin(nc2),jref(iw)
      character id*56,p*8
      data low/76/,ihigh/72/,dv/o376777777777/
      read(10) id,p,nc,nr,nz,xo,dx,yo,dy
      if(nc.lt.iw .or. nz.gt.1 .or. dx.eq.0. .or.
     1  dy.eq.0.) return
c
      int=0
      ilt=0
      ihw=iw/2
      ihw1=ihw+1
      jwt=0
      mid1=ihw1
      jw1=iw+1
      lmtx=nc2-ihw+1
      do 20 i=1,iw
      do 10 i=1,nc2
```

```
   10 w(i,j)=dv
   20 jref(j)=i
      ir=0
      do 30 j=inw1,iw
      call rowio(nc,w(inw1,j),-1,10,10,ie)
   30 ir=ir+1
      do 40 i=1,nc2
      zmax(i)=-1.e20
      zmin(i)=1.e20
      jx(i)=0
   40 jm(i)=0
      iptr=1
      inew=1
      istop=0
      v=vo
c
   50 do 80 i=inw1,nc2-ihw
      if(jx(i).gt.jwt) go to 70
      zmax(i)=-1.e20
      do 60 j=1,iw
      w2=w(i,j)
      if(w2.lt.zmax(i)) go to 60
      if(w2.eq.dv) go to 60
      zmax(i)=w2
      jx(i)=jref(j)
   60 continue
      go to 80
   70 w2=w(i,inew)
      if(w2.lt.zmax(i)) go to 80
      if(w2.eq.dv) go to 80
      zmax(i)=w2
      jx(i)=jref(inew)
   80 continue
c
      do 110 i=inw1,nc2-inw
      if(jm(i).gt.jwt) go to 100
      zmin(i)=1.e20
      do 90 j=1,iw
      w2=w(i,j)
      if(w2.gt.zmin(i)) go to 90
      zmin(i)=w2
      jm(i)=jref(j)
   90 continue
  100 w2=w(i,inew)
      if(w2.gt.zmin(i)) go to 110
      zmin(i)=w2
      jm(i)=jref(inew)
  110 continue
c
      m=inw1
  120 iwl=m+inw
      iwt=m-inw
      if(jx(m).ne.midl) go to 150
      do 130 i=iwt,iwl
      if(zmax(i).gt.zmax(m)) go to 180
  130 continue
      x=xo+float(m-inw1)*dx
      call char(x,v,inigh,1,1,size1,0.,0.,0.)
  140 m=m+inw1
      go to 190
```

```
  150 if(jm(m).ne.midl) go to 180
      do 160 i=iwt,iwl
      if(zmin(i).lt.zmin(m)) go to 180
  160 continue
      x=xo+float(m-ihw1)*dx
      call char(x,y,low,1,1,sizel,0.,0.,0.)
  170 m=m+ihw1
      go to 190
c
  180 m=m+1
  190 if(m.lt.lmtx) go to 120
c
      if(ir.eq.nr) go to 200
      call rowio(nc,w(ihw1,iptr),-1,10,10,ie)
      if(ie.ne.0) go to 230
      ir=ir+1
      go to 220
  200 if(istop.eq.ihw) go to 230
      do 210 i=1,nc2
  210 w(i,iptr)=dv
      istop=istop+1
c
  220 inew=iptr
      iref(iptr)=jwl
      iptr=iptr+1
      if(iptr.gt.iw) iptr=1
      jwt=jwt+1
      midl=midl+1
      jwl=jwl+1
      y=y+dy
      go to 50
  230 return
      end
c***************
      subroutine rowio(n,z,iop,idev,jdev,iend)
c
c  WHERE IOP<0 READ; IOP=0 WRITE; IOP>0 READ&WRITE
c
      dimension z(n)
      iend=0
      if(iop)10,20,10
   10 read(idev,end=40) xo,z
      if(iop)30,30,20
   20 write(jdev) xo,z
   30 return
   40 iend=1
      return
      end
c***************
      subroutine vector(sizel)
c
c     plots random data values and positions
c     in either vector or scalar form
c
      external char(descriptors),match(descriptors),
     1 leftj(descriptors)
      common /vector/iswt,sizev,vmin,fmtv,nchar,ifmtv,
     1 ich(20),szpost,lowhi,chid(20),nid
      automatic x(200),y(200),vi(200),vd(200),vm(200)
      dimension hz(6)
```

```
      character*4 chid
      character*56 ifmtv,blank
      character ivl*3,sta*24,fmtv*16,bid*8
      data arrow/o06l/,con/.017453292/,pi/3.14159265/,a/.9423/,
     1 blank/" "/,labl/8/
      ichr=1
      if(nchar.le.0) ichr=2
      if(nid.gt.19) nid=19
      if(nchar.gt.24) nchar=24
      ich=ich(1)
      if(iswt.gt.1 .or. iswt.lt.0) go to 80
      n=1
   10 if(ifmtv.ne.blank)
     1 read(11,ifmtv,end=20)x(n),y(n),vm(n),vi(n),vd(n)
      if(ifmtv.eq.blank)
     1 read(11,end=20) x(n),y(n),vm(n),vi(n),vd(n)
      n=n+1
      go to 10
   20 n=n-1
      vmax=-1.e10
      do 30 i=1,n
      if(vm(i).gt.vmax) vmax=vm(i)
   30 continue
      sca=sizev/alog(vmax)
      pima=pi-a
      pia=pi+a
      pima2=2.*pi-a
      do 70 i=1,n
      incl=int(vi(i))
      v=alog(vm(i))
      if(v.lt.0.) v=0.
      vlen=sca*v+vmin
      th=-vd(i)*con
      if(th.gt.a .and. th.lt.pima) go to 40
      if(th.gt.pia .and. th.lt.pima2) go to 40
      if(th.le.a) go to 50
      if(th.ge.pi .and. th.le.pia) go to 50
      xl=0.
      yl=sizel
      go to 60
   40 xl=sizel
      yl=0.
      go to 60
   50 xl=0.
      yl=-sizel
   60 encode(ivl,fmtv) incl
      call char(x(i),y(i),3,1,1,.05,0.,0.,0.)
      call char(x(i),y(i),arrow,1,1,vlen,th,0.,0.)
      call char(x(i),y(i),ivl,3,2,sizel,0.,xl,yl)
   70 continue
      go to 200

c
   80 hsz=-sizel*.5
      hszm=-hsz
      xoff=2.*sizel
      idbr=-1
      ibr=-1
      if(ifmtv.eq.blank) ibr=0
      if(iswt.gt.0) go to 90
      ibr=1
```

```
      if(iswt.lt.-10) idbr=0
      if(iswt.lt.-20) idbr=1
      iz=iabs(iswt)
      if(iz.gt.10) iz=iz-10
      if(iz.gt.10) iz=iz-10
   90 continue
      if(ibr) 100,110,120
  100 read(11,ifmtv,end=200) sx,sy,sz
      go to 130
  110 read(11,end=200)sx,sy,sz
      go to 130
  120 read(11,end=200) bid,sx,sy,bz
      sz=bz(iz)
  130 if(nid.eq.0) go to 140
      call match(nid,chid,bid,isvm)
      jch=ich(isvm)
  140 call char(sx,sy,jch,1,1,szpost,0.,0.,0.)
      go to(150,170),icbr
  150 if(idbr) 160,170,160
  160 encode(sta,fmtv) sz
      call char(sx,sy,sta,nchar,2,sizel,0.,xoff,hsz)
  170 if(idbr) 190,180,180
  180 sta=bid
      l=labl
      call leftj(sta,l)
      call char(sx,sy,sta,l,2,sizel,.785,xoff,0.)
  190 if(ibr)100,110,120
  200 return
      end
c***************
      subroutine prjctl(vlat,xlon,x,y,cm,sfac,iprojt)
c
c     call projection routines
c
      double precision yl,xl,x1,y1,degr
      data degr/.0174532925199433d0/
      yl=dble(vlat)*degr
      xl=dble(xlon-cm)*degr
      go to (10,20,30,40,50,50,50),iprojt
   10 call polv(yl,xl,x1,y1)
      go to 60
   20 call utmfwd(yl,xl,x1,y1,ier)
      go to 60
   30 call merctr(yl,xl,x1,y1)
      go to 60
   40 call lambrt(yl,xl,x1,y1)
      go to 60
   50 call albers(yl,xl,x1,y1)
   60 x=sngl(x1)*sfac
      y=sngl(y1)*sfac
      return
      end
c***************
      subroutine project(vlat,xlon,x,y,cm,sfac,iprojt)
c
c     call projection routines
c
      double precision yl,xl,x1,y1
      yl=dble(vlat)*.01745329d0
      xl=dble(abs(cm)-abs(xlon))*.01745329d0
```

```
      go to (10,20,30,40,50,50,50),iprojt
   10 call poly(y1,x1,x1,y1)
      go to 60
   20 call utmfwd(y1,x1,x1,y1)
      go to 60
   30 call merctr(y1,x1,x1,y1)
      go to 60
   40 call lambrt(y1,x1,x1,y1)
      go to 60
   50 call albers(y1,x1,x1,y1)
   60 x=sngl(x1)*sfac
      y=sngl(y1)*sfac
      return
      end
c***************
      subroutine poly(phi,dlamb,x,y)
      double precision phi,dlamb,xc,x,y,a,b,t,q,phi2,c1,c2
      a(xc)=
     1    6.378206402718907d 06 +xc*(
     2   -3.167517353503576d 06 +xc*(
     3    2.478805037574243d 05 +xc*(
     4   -3.530710396439220d 03 +xc*(
     5   -6.565371848240127d 02 +xc*(
     6    6.822539551727124d 01 +xc*(
     7   -2.888860980506611d 00 ))))))
      b(xc)=
     1    3.189103200618349d 06 +xc*(
     2   -2.115275857345996d 06 +xc*(
     3    4.144758431728325d 05 +xc*(
     4   -3.625295427368928d 04 +xc*(
     5    1.322429746943889d 03 +xc*(
     6    4.427163005616853d 01 +xc*(
     7   -8.630738701658902d 00 +xc*(
     8    4.279183401413811d-01 )))))))
      q(xc)=
     1    6.335034386662446d 06 +xc*(
     2    2.144094496140833d 04 +xc*(
     3   -4.182226973512467d 03 +xc*(
     4    3.609995316780635d 02 +xc*(
     5   -1.346978283534684d 01 ))))
      t(xc)=
     1    9.999999957157490d-01+xc*(
     2   -1.666665796975878d-01+xc*(
     3    8.333050613721043d-03+xc*(
     4   -1.980904608528695d-04+xc*
     5    2.605165638554101d-06 )))
      phi2=phi**2
      c1=dlamb*phi
      c2=(c1*t(phi2))**2
      x=dlamb*q(phi2)*t(c2)
      y=dlamb*c1    *b(phi2)*t(.25d0*c2)**2 + phi*q(phi2)
      return
      end
c***************
c-------------------------------------------------------------
c
c  SUBROUTINE UTMFWD
c
c  PURPOSE..
c    TO DETERMINE BASIC UTM COORDIATES FROM LATITUDE
```

```
c     AND LONGITUDE INPUT.  CLARK 1866 SPHEROID CONSTANTS.
c
c   USAGE..
c     CALL UTMFWD(PHI,DLAM,X,Y,IER)
c
c   PARAMETER DESCRIPTION..
c     PHI     - INPUT LATITUDE IN DOUBLE PRECISION RADIANS.
c               DABS(PHI).LT.1.3963 (80 DEGREES).
c     DLAM    - INPUT LONGITUDE IN DOUBLE PRECISION RADIANS.
c               DABS(DLAM).LT.0.06109 (3.5 DEGREES).
c               NOTE.. DLAM MEASURED FROM CENTRAL MERIDIAN.
c     X       - PROJECTED COORDINATE IN METERS FROM THE
c               CENTRAL MERIDIAN (DOUBLE PRECISION).
c     Y       - PROJECTED COORDINATE IN METERS FROM THE
c               EQUATOR (DOUBLE PRECISION).
c     IER     - ERROR RETURN CODE.
c               IER=0 SUCCESSFUL COMPLETION.
c               IER.NE.0 ERROR IN PHI OR DLAM INPUT.
c
c   REMARKS..
c     THIS ROUTINE IS ACCURATE TO 1 METER.  ABSOLUTE
c     ERROR LESS THAN .5 METERS.
c     NEAREST CENTRAL MERIDIAN MAY BE DETERMINED EXTERNALLY
c     BY..
c     CN=DSIGN(10800D0,LON)+AINT(SNGL(LON/21600D0))*21600D0
c       WHERE LON IS LONGITUDE IN DOUBLE PRECISION SECONDS.
c     FALSE EASTING (500000 METERS) AND NORTHING (0 FOR
c     NORTHERN LATITUDES, 10000000 METERS FOR SOUTHERN
c     LATITUDES) FOR TRUE UTM COORDINATES MUST BE COMPUTED
c     EXTERNALLY.
c
c   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED..
c     NONE
c
c   DEVELOPED AND CODED BY..
c     GERALD IAN EVENDEN
c     U. S. GEOLOGICAL SURVEY
c
c-----------------------------------------------------------------
c
      subroutine utmfwd(phi,dlam,x,y)
c
      double precision phi,dlam,x,y,d12,p
c
c     CHECK INPUT PRAMETERS..
c
      p=phi*phi
      d12=dlam*dlam
      if (p .le.1.94965360d0.and.d12.le.3.7319881d-3)
     1    go to 10
      ier=-1
      return
c
   10 ier=0
c
c     DETERMINE PROJECTED COORDINATES..
c
      y=phi*(
     1     6332500.47d0+p*(21431.67d0+p*(-4179.269d0+p*(
     2     359.981d0-p*13.267d0)))
```

```
     3 -dl2*(-3187827d0+p*(2114440d0+p*(-414363.5d0+p*(
     4      36344.6d0-p*1420.3d0)))
     5  -dl2*(1334935d0+p*(-2356027d0+p*(1371758d0-p*
     6      267852d0)))))
      x=dlam*(6375655.2d0+p*(-3166253.6d0+p*(247800.26d0+p*(
     1      -3569.65d0+p*(-617.35d0+p*50.89d0))))
     2  -dl2*(-1069479d0+p*(2661562d0+p*(-1785956d0+p*(
     3    485045d0-p*52022d0)))))
c
      return
c
      end
c***************
      subroutine merctr(lat,long,x,y)
      double precision lat,long,x,y,a,halfpi,b2da2,z,phi
      data a/6378206.d0/,halfpi/1.5707963267949d0/,
     1 b2da2/9.932315290818186d-1/
      x=a*long
c
c     compute z/2
c
      z=0.5d0*(halfpi-datan(b2da2*dsin(lat)/dcos(lat)))
      y=a*dlog(dcos(z)/dsin(z))
      return
      end
c***************
      subroutine lambrt(lat,long,x,y)
      double precision lat,long,x,y,z,k,l,r,b2da2,halfpi,theta
      data k/1.245265545827296d7/,l/6.304998344052458d-1/,
     1 b2da2/9.932315290818186d-1/,halfpi/1.5707963267949662d0/
c
c     compute z/2
c
      z=0.5d0*(halfpi-datan(b2da2*dsin(lat)/dcos(lat)))
c
c     compute r
c
      r=k*(dsin(z)/dcos(z))**l
c
c      compute coordinates
c
      theta=l*long
      x=r*dsin(theta)
      y=-r*dcos(theta)
      return
      end
c***************
      subroutine albers(ylat,xlon,x,y)
      common/albcom/n,rho1sq,sinbt1,twoc2n
      double precision ylat,xlon,x,y,n,rho1sq,sinbt1,twoc2n,
     1 nus,nals,nhaw,rho295,rho55,rho8,tcnus,tcnals,tcnhaw,
     2 a1,b,c1,d,e1,f1,g,h,theta,rho,sinbet,sinphi,s2,
     3 sbt295,sbt55,sbt8
      data a1/9.954804334645587d-1/,b/4.49202460774588d-3/,
     1 c1/2.736435989866449d-5/,d/1.763992166249299d-7/,
     2 e1/1.160814577272288d-9/,f1/7.714265487727804d-12/,
     3 g/5.154557173568170d-14/,h/3.455700205911349d-16/,
     4 nus/6.629034937887094d-1/,nals/8.627447947235633d-1/,
     5 nhaw/2.241096394314637d-1/,rho295/8.491969239674584d13/,
     6 rho55/1.806308673895081d13/,rho8/7.943986660586285d14/,
```

```
     7 tcnus/1.34647092189276 9d14/,tcnals/9.4094108484536 36d13/,
     8 tcnhaw/3.622298555079059d14/,sbt295/4.907351753179611d-1/,
     9 sbt55/8.17929054505878 68d-1/,sbt8/1.385562096187223d-1/
       sinphi=dsin(ylat)
       s2=sinphi*sinphi
       sinbet=sinphi*(a1+s2*(b+s2*(c1+s2*(d+s2*
     1 (e1+s2*(f1+s2*(g+s2*h))))))
       rho=dsqrt(rho1sq+twoc2n*(sinbt1-sinbet))
       theta=n*xlon
       x=rho*dsin(theta)
       y=-rho*dcos(theta)
       return
c
c      set up constants
c
       entry setalb(iproj)
       if(iproj-6)10,20,30
    10 n=nus
       rho1sq=rho295
       sinbt1=sbt295
       twoc2n=tcnus
       return
    20 n=nals
       rho1sq=rho55
       sinbt1=sbt55
       twoc2n=tcnals
       return
    30 n=nhaw
       rho1sq=rho8
       sinbt1=sbt8
       twoc2n=tcnhaw
       return
       end
```