ALGORITHM 531
Contour Plotting [J6]

WILLIAM V. SNYDER
Jet Propulsion Laboratory

Key Words and Phrases: contour plotting
CR Categories: 8 2
Language Fortran

DESCRIPTION

Given a two-dimensional array of samples of a surface, contour values, and a subroutine to draw lines, the subroutine GCONTR determines sequences of points in the plane which may be used to draw contours through equal values of the surface.

A contour plotting algorithm may be constructed by following contours from some starting point until they either close or intersect a boundary, or by examining each cell of the grid in turn and drawing all contours found inside the cell. The advantages of contour following are that contour labeling is relatively easy and that the pen of an incremental plotter does not move about as much without writing anything. The advantages of methods which draw all contours found inside a cell are that less auxiliary storage is needed and that each cell can be completely processed before going on to the next, thereby allowing generation of contours over a much larger array than can be accommodated in main memory at one time.

GCONTR is of the type which follows contours. On a representative problem, a program using GCONTR generated about 11,000 plotter commands with about 57,000 commands generated by a program using a cellular method. For this problem, grid lines, user identification, and a table of contour values required about 36,000 additional commands.

Received 17 June 1977 and 8 November 1977.

This work was sponsored by the National Aeronautics and Space Administration under Contract NAS 7-100; this paper presents the results of one phase of research carried out at the Jet Propulsion Laboratory The work was carried out using a Univac 1108 computer, CalComp model 565 drum plotters, a Calcomp model 1675 Com, and Tektronix 4000 series interactive graphics terminals.
Author's address: Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91103.

Since the data used by contour plotting programs are presented at discrete points and since the contour values are not necessarily equal to the dependent variable values at the nodal points of the mesh, some method must be used to estimate the point at which a contour value intersects the edge of a cell. If one uses nonlinear interpolation, there is the possibility of multiple intersections of a contour with an edge and the possibility of closed contours which intersect no edges; if one wants to follow the interpolated surface faithfully, drawing the contour entails finding zeros of a nonlinear bivariate function. Linear interpolation has the drawback that extrema occur only at nodal points. GCONTR uses linear interpolation. If this does not produce sufficiently smooth, accurate contours, we recommend computing more function values or interpolatory subtabulation to a finer mesh. If the required storage is larger than desired, the independent variable plane is easily divided into separately processed segments. GCONTR does not provide automatic subtabulation because different methods work better for different problems. Since GCONTR uses a contour following method, any implementation of automatic subtabulation would necessarily have a higher cost than a reasonable user implementation.

Once the points of intersection of a contour with edges have been determined, one must still decide how to draw the contour line between such intersections. We recommend that straight lines be used. Most other curves can cause balloons and crossing of contours of different values.

We now discuss the features which distinguish GCONTR from the methods of Cottafava and Le Moli [1] and Crane [2]. GCONTR and the algorithms described in [1] and [2] are similar in that all are contour following methods, the data are assumed to be presented on the nodes of a topologically rectangular grid, and linear interpolation is used along the edges of a cell.

GCONTR and algorithm [2] begin following a contour as soon as an edge of a cell of the coordinate grid is found which the contour line intersects. Edges intersected by the contour are flagged in an integer array in [2] and by marks in a bitmap in GCONTR. In algorithm [1] a preliminary detection of all intersections of edges and contours is done and then the contours are drawn. The details of the bookkeeping are not described in [1].

Algorithms [1] and [2] start the contour at the first edge found while searching along rows or columns of the data array. GCONTR reduces pen-up travel by searching along a rectangular spiral starting at the current pen position. All three algorithms draw all contours which intersect boundaries before drawing any contours which do not.

Algorithm [1] treats the case of a contour intersecting all four edges of a cell in a way which depends on the fixed order of examining the edges. If a cell is approached from different directions, different contours will be drawn. Algorithm [2] forces the contours to cross inside the cell. When several different contours intersect all four edges, the resulting pattern looks like an asterisk. GCONTR examines the linear interpolate along the "top" and "bottom" edges of the cell. If the interpolate on the top edge is less than the interpolate on the bottom edge, the contour line is drawn top-to-left and bottom-to-right. Otherwise, the contour line is drawn top-to-right and bottom-to-left. This methods selects the same contours if the axes are exchanged. To show this, note that the use of linear

interpolation along the edges of the unit cell implies that there is some estimated function value F_x which occurs at the same abscissa on the top and bottom edges, and some estimated function value F_y which occurs at the same ordinate on the "left" and "right" edges. It is easy to show that $F_x = F_y$. Therefore, the contour of value F_x is the only contour which crosses inside the cell, and it divides the cell into four rectangular regions. In two of these regions the esimated function value is less than F_x. In the other two, the estimated function value is greater than F_x.

Algorithms [1] and [2] consider all data in the array. GCONTR provides a means whereby the user may designate excluded elements of the array. The edges connecting excluded elements to elements not excluded are not examined. This feature has been found to be very useful in practice, as data are frequently not presented on a complete rectangular grid. When using methods which do not have this feature, one frequently finds extraneous contours drawn between the regions containing data and the regions containing no data.

The output method is not described in [1]. The output from algorithm [2] is an ordered set of points in an array which define the contour lines to be drawn. GCONTR calls a line drawing subroutine provided by the user. If the user wishes to take the risks noted above associated with using other than straight lines for drawing the contours, points may be stored in the line drawing subroutine, and a method such as is described in [3] may be used to generate smooth contours. Transformations may be applied to the output of algorithm [2] and to the points provided by GCONTR to the line drawing subroutine for such purposes as coordinate system conversion. Note that neither algorithm [2] nor GCONTR require the grid lines to be straight or uniformly spaced.

REFERENCES

1 COTTAFAVA, G., AND LE MOLI, G. Automatic contour map. *Comm. ACM 12,* 7 (July 1969), 386–391
2. CRANE, C M. Contour plotting for functions specified at nodal points of an irregular mesh based on an arbitrary two parameter co-ordinate system. *Comptr. J. 15* (1972), 382–384.
3. MCCONALOGUE, D J An automatic french-curve procedure for use with an incremental plotter. *Comptr J. 14* (1971), 207–209.

ALGORITHM

[Summary information and part of the listing is printed here. The complete listing is available from the ACM Algorithms Distribution Service (see inside back cover for order form), or may be found in "Collected Algorithms from ACM."]

NAME(n): indicates a Fortran module with n records
NAME^T(n): indicates "NAME" is included for testing purposes
Contents: FILLO(22), MARK1(19), IGET(17), GCONTR(291), TESTI^T(31), TEST2^T(74), DRAW2^T(33), DRAWI^T(32), CGRID^T(139)

```
      SUBROUTINE GCONTR(Z, NRZ, NX, NY, CV, NCV, ZMAX, BITMAP, DRAW)    GCO   10
C                                                                       GCO   20
C     THIS SUBROUTINE DRAWS A CONTOUR THROUGH EQUAL VALUES OF AN ARRAY. GCO   30
C                                                                       GCO   40
C     *****      FORMAL ARGUMENTS      ************************************GCO   50
C                                                                       GCO   60
```

```
C     Z IS THE ARRAY FOR WHICH CONTOURS ARE TO BE DRAWN.  THE ELEMENTS  GCO   70
C     OF Z ARE ASSUMED TO LIE UPON THE NODES OF A TOPOLOGICALLY         GCO   80
C     RECTANGULAR COORDINATE SYSTEM - E.G. CARTESIAN, POLAR (EXCEPT     GCO   90
C     THE ORIGIN), ETC.                                                 GCO  100
C                                                                       GCO  110
C     NRZ IS THE NUMBER OF ROWS DECLARED FOR Z IN THE CALLING PROGRAM.  GCO  120
C                                                                       GCO  130
C     NX IS THE LIMIT FOR THE FIRST SUBSCRIPT OF Z.                     GCO  140
C                                                                       GCO  150
C     NY IS THE LIMIT FOR THE SECOND SUBSCRIPT OF Z.                    GCO  160
C                                                                       GCO  170
C     CV ARE THE VALUES OF THE CONTOURS TO BE DRAWN.                    GCO  180
C                                                                       GCO  190
C     NCV IS THE NUMBER OF CONTOUR VALUES IN CV.                        GCO  200
C                                                                       GCO  210
C     ZMAX IS THE MAXIMUM VALUE OF Z FOR CONSIDERATION.  A VALUE OF     GCO  220
C     Z(I,J) GREATER THAN ZMAX IS A SIGNAL THAT THAT POINT AND THE      GCO  230
C     GRID LINE SEGMENTS RADIATING FROM THAT POINT TO IT'S NEIGHBORS    GCO  240
C     ARE TO BE EXCLUDED FROM CONTOURING.                               GCO  250
C                                                                       GCO  260
C     BITMAP IS A WORK AREA LARGE ENOUGH TO HOLD 2*NX*NY*NCV BITS.  IT  GCO  270
C     IS ACCESSED BY LOW-LEVEL ROUTINES, WHICH ARE DESCRIBED BELOW.     GCO  280
C     LET J BE THE NUMBER OF USEFUL BITS IN EACH WORD OF BITMAP,        GCO  290
C     AS DETERMINED BY THE USER MACHINE AND IMPLEMENTATION OF           GCO  300
C     THE BITMAP MANIPULATION SUBPROGRAMS DESCRIBED BELOW.  THEN        GCO  310
C     THE NUMBER OF WORDS REQUIRED FOR THE BITMAP IS THE FLOOR OF       GCO  320
C         (2*NX*NY*NCV+J-1)/J.                                          GCO  330
C                                                                       GCO  340
C     DRAW IS A USER-PROVIDED SUBROUTINE USED TO DRAW CONTOURS.         GCO  350
C     THE CALLING SEQUENCE FOR DRAW IS:                                 GCO  360
C                                                                       GCO  370
C         CALL DRAW (X,Y,IFLAG)                                         GCO  380
C         LET NX = INTEGER PART OF X, FX = FRACTIONAL PART OF X.        GCO  390
C         THEN X SHOULD BE INTERPRETED SUCH THAT INCREASES IN NX        GCO  400
C         CORRESPOND TO INCREASES IN THE FIRST SUBSCRIPT OF Z, AND      GCO  410
C         FX IS THE FRACTIONAL DISTANCE FROM THE ABSCISSA CORRESPONDING GCO  420
C         TO NX TO THE ABSCISSA CORRESPONDING TO NX+1,                  GCO  430
C         AND Y SHOULD BE INTERPRETED SIMILARLY FOR THE SECOND          GCO  440
C         SUBSCRIPT OF Z.                                               GCO  450
C         THE LOW-ORDER DIGIT OF IFLAG WILL HAVE ONE OF THE VALUES:     GCO  460
C             1 - CONTINUE A CONTOUR,                                   GCO  470
C             2 - START A CONTOUR AT A BOUNDARY,                        GCO  480
C             3 - START A CONTOUR NOT AT A BOUNDARY,                    GCO  490
C             4 - FINISH A CONTOUR AT A BOUNDARY,                       GCO  500
C             5 - FINISH A CLOSED CONTOUR (NOT AT A BOUNDARY).          GCO  510
C                 NOTE THAT REQUESTS 1, 4 AND 5 ARE FOR PEN-DOWN        GCO  520
C                 MOVES, AND THAT REQUESTS 2 AND 3 ARE FOR PEN-UP       GCO  530
C                 MOVES.                                                GCO  540
C             6 - SET X AND Y TO THE APPROXIMATE 'PEN' POSITION, USING  GCO  550
C                 THE NOTATION DISCUSSED ABOVE.  THIS CALL MAY BE       GCO  560
C                 IGNORED, THE RESULT BEING THAT THE 'PEN' POSITION     GCO  570
C                 IS TAKEN TO CORRESPOND TO Z(1,1).                     GCO  580
C         IFLAG/10 IS THE CONTOUR NUMBER.                               GCO  590
C                                                                       GCO  600
C     *****     EXTERNAL SUBPROGRAMS     ********************************GCO  610
C                                                                       GCO  620
C     DRAW IS THE USER-SUPPLIED LINE DRAWING SUBPROGRAM DESCRIBED ABOVE.GCO  630
C     DRAW MAY BE SENSITIVE TO THE HOST COMPUTER AND TO THE PLOT DEVICE.GCO  640
C     FILL0 IS USED TO FILL A BITMAP WITH ZEROES.  CALL FILL0 (BITMAP,N)GCO  650
C     FILLS THE FIRST N BITS OF BITMAP WITH ZEROES.                     GCO  660
C     MARK1 IS USED TO PLACE A 1 IN A SPECIFIC BIT OF THE BITMAP.       GCO  670
C     CALL MARK1 (BITMAP,N) PUTS A 1 IN THE NTH BIT OF THE BITMAP.      GCO  680
```

```
C     IGET IS USED TO DETERMINE THE SETTING OF A PARTICULAR BIT IN THE  GCO  690
C     BITMAP.  I=IGET(BITMAP,N) SETS I TO ZERO IF THE NTH BIT OF THE    GCO  700
C     BITMAP IS ZERO, AND SETS I TO ONE IF THE NTH BIT IS ONE.          GCO  710
C     FILL0, MARK1 AND IGET ARE MACHINE SENSITIVE.                      GCO  720
C                                                                       GCO  730
C     ************************************************************************GCO  740
C                                                                       GCO  750
      REAL Z(NRZ,1), CV(1)                                            GCO  760
      INTEGER BITMAP(1)                                               GCO  770
```