U. S. DEPARTMENT OF THE INTERIOR

U.S. GEOLOGICAL SURVEY

Cartographic Production for the Louisiana Barrier Island Erosion Study: 4. Processing Contours for Color Fill

by Dorothy Hopkins and Jeffrey H. List

Open-File Report #91-305

Cartographic Production for the Louisiana Barrier Island Erosion Study: 4. Processing Contours for Color Fill

by Dorothy Hopkins and Jeffrey H. List

Introduction

This report is the fourth in a series of US Geological Survey open-file reports detailing the cartographic production methods developed for the Louisiana Barrier Island Erosion Study, Atlas of Seafloor Change from 1887 to 1989 (I-2150-B, List et al., 1991). Because of the data complexity and an accelerated production schedule, these maps were produced entirely through digital means. The open-file reports in this series describe the techniques developed and/or adopted by the cartographic production group at the USGS's Center for Coastal Geology and Regional Marine Studies in St. Petersburg, Florida. While these methods were found to be efficient and useful for this project, they are certainly not the only possible methods, nor necessarily even the best methods available.

This report describes the techniques used to process contour lines for color filled bathymetric and bathymetric change maps with several types of software: Interactive Surface Modeling (ISM version 6.93B) and ARC/INFO (version 5.0.1). The contour lines were generated from an ISM grid (a two dimensional matrix of

elevation values used as a surface model), and were edited in ARC/INFO to create polygons, and then the polygons were processed on the Scitex R-280 to create negatives for printing. These features are shown as color filled polygons on the bathymetric and bathymetric change maps of Atlas I-2150-B.

Creating ARC/INFO Coverages from ISM Contour Files

To create contour output files, the ISM grid must be in its final form, with all grid node editing completed. Contour output, listed on the contour map options menu in ISM, should be selected to create an ascii contour output file (Attachment 1) at the same time a new plot file is created. We noticed that the lines created from the contour output file, before any processing was done, were not as smooth as the contour lines shown on the plots we made at the scale 1:100k. This is due to an unusual generalizing characteristic of the ISM software; the number of data points for each contour line is affected by the scale used in making this final plot file. For example, the contour output file of a map plotted at 1:50k will have many more data points for each contour line than the output file of a map plotted at 1:100k. Although the largest scale maps for our atlas are 1:100k, the scale we used was 1:37k. This scale was determined to be small enough to ensure that the number of data points in the contour output file is manageable, while it is large enough that the contours will be as smooth in the contour output file as hey are when shown on 1:100k plots generated directly by ISM.

To ensure that every contour near the grid boundaries is congruent with the contours on adjacent grids, extra bathymetric

data is included along the edges when creating each grid. To eliminate this overlapping data in the contour output files, the CONWIN.FOR program (Attachment 2) is used to establish minimum and maximum x and y values for each grid area. Once they are reduced to the desired areas, the contour output files can be translated to the ARC/INFO format using a FORTRAN program written for us by Greg Green (USGS Office of Mineral Resources, Denver, Colorado). This program splits the contour output file into two separate files. One file will be used to generate the linework; it contains x and y values and a line ID number for each line segment (Attachment 3). The other file will be used to add attributes to the line segments; it contains the line ID numbers and z, or elevation, values (Attachment 4). In ARC/INFO attributes are necessary to identify arcs for selection in interactive editing (ARCEDIT) and in graphic displays (ARCPLOT).

In ARC, the GENERATE command with the LINE option is used to create an ARC/INFO coverage from each of the contour output x and y files. The structure of the new coverages will be like a directory containing the following files: ARC (line data), ARX, BND (boundary data), LAB (label or point data), LOG (a record of the coverage), TIC (reference points), TXT, and TXX. The next step is to BUILD each coverage, with the LINE option, to create an Arc Attribute Table (AAT). The AAT is an INFO reference file which stores values, such as Length, for each line feature, and it will be updated during BUILD and CLEAN operations. After the AAT is created, a new file should be DEFINEd in INFO containing 2 attributes: COVER-ID, and ATTB. Using the ADD FROM command in INFO, values for these items will be added from the attribute

file containing the ID numbers and the ISM contour output z values. Then using the JOINITEM command, the z values now loaded into the ATTB field, will be added to the AAT of the coverage. When all of the contour coverages have been created and elevation values have been added, the individual coverages can be joined together to make one larger coverage for the entire study area. To join the coverages automatically, use the MAPJOIN command with NET ALL options in ARC. Or to join the coverages interactively, in ARC, COPY one of the coverages and name it the main or total coverage, then in ARCEDIT using the GET command, contour arcs from the adjacent coverages can be added (this step will not be quick). After all the arcs are added to the total coverage, attributes such as MAJOR1 and MINOR1 for bathymetric coverages and CODE and DEPTH for bathymetric change coverages, can be added to the AAT using the ADDITEM command in ARC. The values for these new attributes should be calculated in ARCEDIT as follows: for bathymetric contours MINOR1 = ATTB and MAJOR1 = 24, for bathymetric change contours CODE = 2 and DEPTH = ATTB. Then using the GET command, arcs from the data limit boundary coverage (Processing Land Features, Open-File Report #91-212, Hopkins and List, 1991) should be added and these arcs should be assigned exclusive values such as MAJOR1 and MINOR1 = -9999, or CODE = 9.

Processing Arcs and Labeling ARC/INFO Coverages

In ARC, use the CLEAN command, with a tolerance of 1m as a minimal FUZZY and DANGLE distance, to update the AAT and create intersections where arcs are crossing, such as at the data limit

boundary. The FUZZY distance refers to the minimum distance allowed between arcs before snapping them together; the DANGLE distance refers to the minimum length of an arc that is not snapped to other arcs at both ends. The CLEAN command will also recognize arcs forming polygons and create a Polygon Attribute Table (PAT) for the coverage. The PAT is an INFO reference file which stores values, such as code and depth, for all arcs forming polygons and their associated label points.

After CLEANing, the coverage should be checked for errors in ARCEDIT; arcs which fall short of the data limit line should be EXTENDed, and nodes (a feature found at the beginning and end of every arc) which are unconnected or dangling can be snapped together with the MOVE command. Another type of error occurs along the edges of adjacent grids. The CONWIN.FOR program does not include in the output file data points which fall outside the defined x and y limits. This means that if a contour begins in one grid area and loops around into another grid's area then returns into the original grid, some points from this contour line will be excluded from the output file. Thus the last point inside the limit before it crosses into the next grid, and the very next point along the same line where it comes back to the original grid will be connected in the arc coverage. This extraneous connecting line must be SPLIT from the genuine contour line and DELETEd. When all the linework has been corrected, and all polygons are closed, use the BUILD command with the POLY option to update the PAT. The CREATELABELS command should now be used; this command will place a label point into all polygons which do not already have one (at this stage that is every

polygon). As soon as labels are created, attributes should be added, such as MAJOR and MINOR for bathymetric coverages and CODE and DEPTH for bathymetric change coverages, using the ADDITEM command.

Label Attributing

Two different methods are used for label attributing the bathymetric and the bathymetric change maps. The bathymetric method utilizes a FORTRAN program to add depth values to the labels by adding together the elevations of the contour arcs on both sides of the label point and dividing that sum by two. This method is quick and reasonably accurate for bathymetric coverages; a few errors commonly occur along the edges or in the corners of the coverage. However, for the bathymetric change maps, many miscalculated label values were found, so we devised a separate method for these coverages. The bathymetric change method involves back interpolating a file containing the x and y coordinates of each label point to extract depth values from the original ISM grid.

To calculate label attribute values for a bathymetric coverage, the coverage must first be transformed into a DLG file using the ARCDLG command. We then run another FORTRAN program written for us by Greg Green (USGS Office of Mineral Resources, Denver, Colorado) to create a DLG output file and 2 output data files containing arc depth (coverage.ACODE) and polygon depth (coverage.PCODE) values. Using the DLGARC OPTIONAL command the output DLG file is transformed into a coverage again. The coverage should then be CLEANed with the POLY option, and then

use the BUILD command with the LINE option. To place the arc and polygon values into the new coverage, the JOINITEM command is used, with coverage.ACODE added to the coverage.AAT, and coverage.PCODE added to the coverage.PAT. The coverage should be checked in ARCEDIT to ensure that no incorrect values have been place in corner polygons

To calculate label attribute values for a bathymetric change coverage, the UNGENERATE command is used with the POINT option to create a data file containing x and y data for every label point. These files are then translated to the ISM format (F12.2,F12.2). On a UNIX system, this conversion can be accomplished with the command:

```
awk '{printf("%12.2f%12.2f\n",$1,$2)}' input_file > output_file.
```

Once the file is in the ISM format, the coordinates are divided into separate files covering each individual grid area using the POLYWIN.FOR program (Attachment 5), and are then added to the ISM table of contents (TOC) as a Scattered Data type file. While in ISM, on the File Operations menu, a Field Operation is selected, then a Back Interpolation is selected. Using the original ISM grid, and the x and y data file, a new file is created with the original x and y data and z, or elevation, data from the original ISM grid. This file should be checked in ARCEDIT for null elevation values created during the Back Interpolation process. The new xyz file is then translated to the ARC/INFO format using the XYZ2ARC.C program (Attachment 6). This program reads in the three columns of data (x, y, and z) and writes out a generate file (ID number, x, and y) and an attribute file (ID number, and z) in the ARC/INFO format. In ARC, using

the GENERATE command with the POINT option, a label coverage is created from the generate file. In INFO, DEFINE a temporary file containing two attributes: COVER-ID, and DEPTH; then ADD FROM the attribute file to load the elevation values into the INFO file. In ARC, use the BUILD command to create a PAT for the new label coverage, and then, use the ADDITEM command to create an attribute field for DEPTH. Using the JOINITEM command the elevation values from the temporary INFO file can be added to the points coverage. All of the original label points should be DELETEd in ARCEDIT and the new labels should be added to the original linework coverage with the GET command. The coverage should then be checked at the edges and in the corners in ARCEDIT, to ensure that all polygons are labeled and that all labels have the proper value calculated for DEPTH.

Edit and Correction of ARC/INFO Coverages

To ensure that the coverage has been coded with the proper elevation values, all polygon labels must be edited very carefully, with special attention to the edges and corners, and near islands and channels. For our needs, the most efficient method of independent verification of these polygon values is the comparison of a solid color filled plot, generated from the ARC/INFO coverage on an electrostatic plotter, and a color image plot, generated from the original ISM grid file on an inkjet printer.

To create solid color filled plots of the coverage, a shadeset must be created to access the color symbols of the electrostatic plotter (symbols 1 to 999). This shadeset must have the TYPE

option defined as HARDWARE, and the PATTERN option defined as negative numbers for all electrostatic solid color fill symbols available (-1 to -999). The shadeset must be created in INFO, and then it should be stored in the Symbols directory, under Arcexe50, so that it may be accessed from any ARC directory on the system.

Once the shadeset is completed, the plot file can be created. In ARCPLOT, polygons are recognized as a feature type and they can be selected in groups with the RESELECT command by the DEPTH values. For example, all bathymetric polygons are selected with DEPTH values LESS THAN 0 AND GREATER THAN -2, and filled with the palest blue shade, and all polygons with DEPTH values LESS THAN -15 are selected, and filled with the deepest blue shade.

Using software developed by Bob Marks (USGS, Menlo Park, California), color images of an ISM grid file are created on a Tektronix terminal and a hardcopy can be prepared with a Tektronix Inkjet printer. These color image plots code the grid nodes according to their elevation value; then, hardcopy plots can be used to compare the actual ISM grid node values with the corresponding label point values in the ARC/INFO coverage. The coverage should be edited carefully, with special attention to areas of rapid change (concentric contours which look like bullseyes), and to corners and map edges. Corrections should be made in ARCEDIT and then the BUILD command should be used to update the PAT.

Creating Scitex Data Files

The Scitex computer system was originally intended for use in the textile industry to create, and color separate, patterns. It is now also used in cartography, to produce color separated, high resolution negatives with screen patterns. The screen pattern consists of a super cell, made up of a few basic cells, in a continuously repeating pattern, where the resolution and angle of the screen is determined by the height and width, in pixels, of the basic cell. Thus, upon magnification, the screen appears to be made of tiny, jagged asterisk shapes, rather than round dots.

Once the plots have been edited, and all corrections have been made, Scitex data files can be created. For this step, a font table file must be created to assign Scitex channel numbers to all symbols used in the plot file. The Scitex has 10 available color channels, plus one reserved as a background color and one used to show linework. For plots with more than 10 solid color fill symbols additional font tables must be created. Channels 1 through 10 may be assigned to the first 10 symbols in the plot file with the rest of the symbols assigned to channel 12 as the background color in the first font table file; in the second font table file, symbols 11 through 20 may be assigned to channels 1 through 10, with all other symbols assigned to channel 12, and so on.

The pixel resolution is an important consideration on the Scitex since high resolution makes interactive editing very slow, and low resolution gives the map a digital appearance and increases the occurrence of 'choked' polygons. Choked polygons are narrow sections which when rasterized are converted into diagonally adjacent pixels, breaking the polygon (Figure 1).

These errors cannot be detected in advance and must be corrected by manually tagging the polygon with the appropriate color channel. Therefore the Scitex operator will require a plot of each region for comparison and manual error checking on the Scitex.

In ARC, we used the PLOTSCITEX command with a resolution of 40 pixels per inch as a compromise between a digital appearance and lengthy editing time. The PLOTSCITEX command creates the Scitex output files: DIGIT, ARF, and RPT. The DIGIT file contains the linework forming the outline of every polygon in the plot file. The ARF, or area reference file, contains a reference fill point, with a channel code from the font table, in the geographic center of each polygon in the plot file. The RPT file is a report on the size of all the Scitex files, for example, it lists the number of area reference points. In ARC, to send Scitex data files to the Scitex R-280 system, the SCITEXWRITE command is used. This command automatically sends the Scitex data files to a 9-track tape without using any host system commands.

Acknowledgments

The authors wish to thank Rob Wertz and Keith Dalziel for their comments in reviewing this paper, and Doug Aitken for his assistance in devising the Scitex procedure. We also thank Greg Green for the invaluable programs he wrote for us, which are essential to our processing.

References

Hopkins, D., and List, J. H., 1991, Cartographic Production for the Louisiana Barrier Island Erosion Study: 1. Processing Land Features, USGS Open-file Report #91-212

Figure 1 Scitex processing errors: choked polygons

Attachment 1 ISM contour output file

Attachment 2 CONWIN.FOR program to limit the area of contour output files.

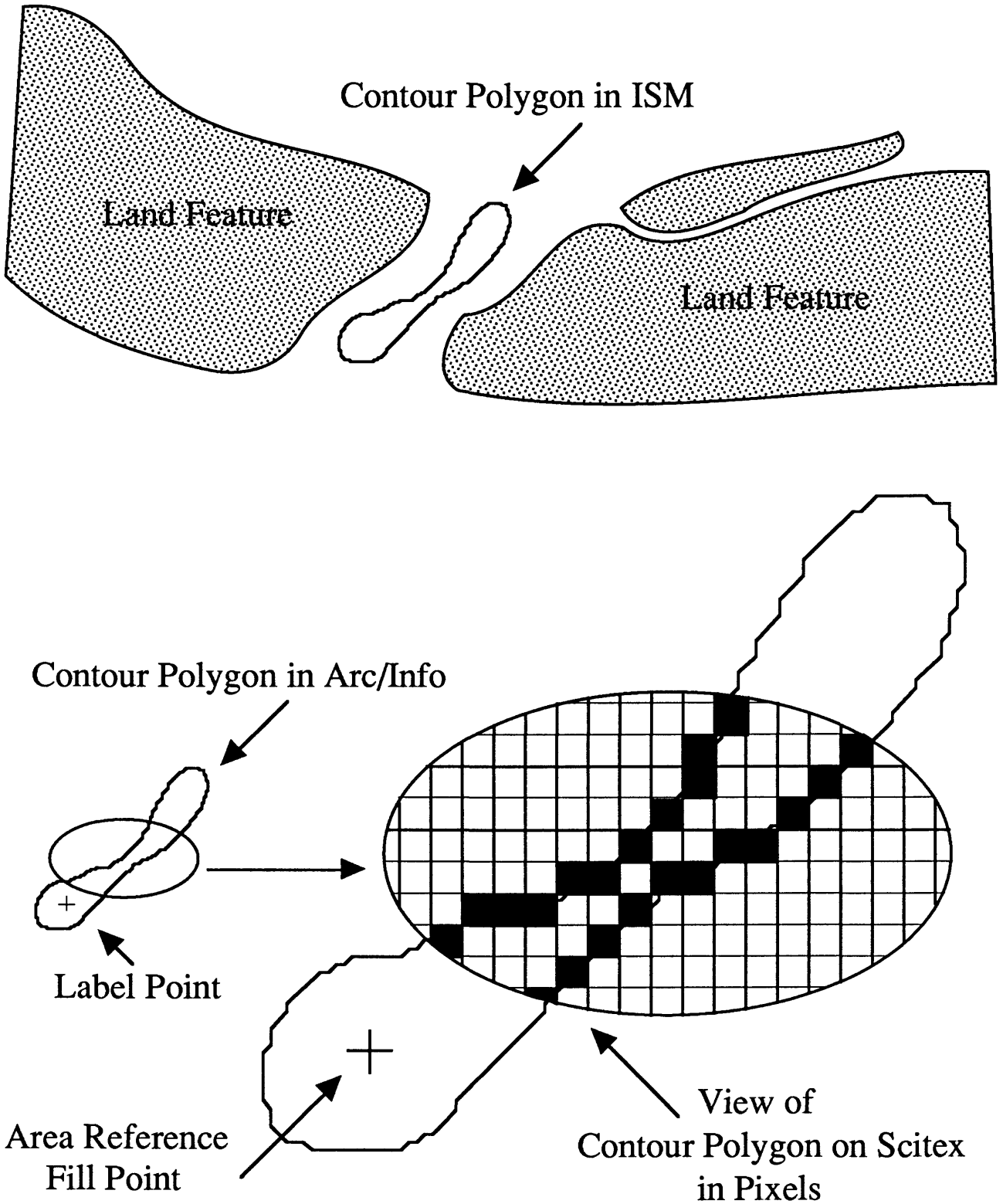
Attachment 3 Contour line generate file (ARC/INFO format)

Attachment 4 Contour line attribute file (ARC/INFO format)

Attachment 5 POLYWIN.FOR program to define limits of x and y data in the format (F12.2,F12.2), usually used for ISM Polygon files.

Attachment 6 XYZ2ARC.C program to convert ISM Scattered Data (xyz) type file to a generate (ID, x and y) file and an attribute (ID and z) file.

Figure 1


Contour Polygon in ISM
Land Feature
Land Feature
Contour Polygon in Arc/Info
Label Point
Area Reference
Fill Point
View of
Contour Polygon on Scitex
in Pixels

760840.1	3181875.	-24.00000	1
760822.5	3181889.	-24.00000	1
760755.0	3181934.	-24.00000	1
760741.1	3181943.	-24.00000	1
760687.5	3181970.	-24.00000	1
760620.0	3181997.	-24.00000	1
760581.4	3182010.	-24.00000	1
760552.5	3182018.	-24.00000	1
760485.0	3182034.	-24.00000	1
760417.5	3182048.	-24.00000	1
760350.0	3182067.	-24.00000	1
760326.1	3182078.	-24.00000	1
760282.5	3182098.	-24.00000	1
760215.0	3182145.	-24.00000	1
760214.8	3182145.	-24.00000	1
760153.6	3182213.	-24.00000	1
760147.5	3182222.	-24.00000	1
760111.8	3182280.	-24.00000	1
760081.3	3182348.	-24.00000	1
760080.0	3182351.	-24.00000	1
760047.2	3182415.	-24.00000	1
760012.5	3182478.	-24.00000	1
760008.1	3182483.	-24.00000	1
759945.0	3182544.	-24.00000	1
759931.6	3182550.	-24.00000	1
759877.5	3182574.	-24.00000	1
760840.1	3181875.	-24.00000	2
761025.0	3182975.	-24.00000	3
761007.3	3182955.	-24.00000	3
760995.3	3182888.	-24.00000	3
761002.9	3182820.	-24.00000	3
761430.0	3183556.	-24.00000	4
761374.6	3183495.	-24.00000	4
761970.0	3184399.	-24.00000	5
761960.9	3184373.	-24.00000	5
761960.8	3184305.	-24.00000	5
762105.0	3184441.	-24.00000	6
762037.5	3184444.	-24.00000	6
762023.1	3184440.	-24.00000	6
765075.0	3188912.	-24.00000	7
765007.5	3188919.	-24.00000	7
764940.0	3188904.	-24.00000	7
764932.5	3188895.	-24.00000	7
764931.2	3188828.	-24.00000	7
764929.9	3188760.	-24.00000	7

```
C PROGRAM CONWIN.FOR--TO READ IN CONTOUR OUTPUT AND WRITE OUT
C                      A NEW FILE WITH A LIMITED WINDOW

      INTEGER NBAD,ID
      CHARACTER CONTIN*40,CONTOUT*40,REJECTS*40
      CHARACTER Q1*1
      REAL*8 XX,YY,ZZ

      PRINT*,'INPUT NAME OF SCATTERED DATA TO BE WINDOWED'
      READ(*,200)CONTIN
  200 FORMAT(A40)
      OPEN(31,FILE=CONTIN,STATUS='OLD')

      PRINT*,'INPUT WINDOW IN DATA UNITS:  XMIN,XMAX,YMIN,YMAX:'
      READ*,XMIN,XMAX,YMIN,YMAX

      PRINT*,'INPUT NAME OF WINDOWED DATA OUTPUT FILE:'
      READ(*,200)CONTOUT
      OPEN(32,FILE=CONTOUT,STATUS='NEW',CARRIAGECONTROL='LIST')

C      WRITE(*,624)
C  624 FORMAT(' ENTER FORMAT FOR WINDOWED OUTPUT DATA'/
C     &       ' X,Y AND Z. EXAMPLE: (F12.7,F12.7,F12.7)')
C      READ(*,401)FORM2
C  401 FORMAT(A60)

      PRINT*,'WANT TO PUT REJECT POINTS IN A SEPARATE FILE (Y/N)?'
      READ(*,625)Q1
  625 FORMAT(A1)

      IF(Q1.EQ.'Y'.OR.Q1.EQ.'y')THEN
        PRINT*,'INPUT NAME FOR REJECT POINTS FILE'
        READ(*,200)REJECTS
        OPEN(33,FILE=REJECTS,STATUS='NEW',CARRIAGECONTROL='LIST')
      END IF

      NBAD = 0

      DO 10 I=1,1000000
         READ(31,*,END=999)XX,YY,ZZ,ID

           IF(XX .EQ. 1.0E+20) THEN
              NBAD = NBAD + 1

              ELSEIF(XX.LE.XMAX.AND.XX.GE.XMIN
     &             .AND.YY.LE.YMAX.AND.YY.GE.YMIN)THEN
                  WRITE(32,401)XX,YY,ZZ,ID
  401 FORMAT(F11.1,F15.0,F15.5,9X,I5)

                        ELSEIF(Q1.EQ.'Y'.OR.Q1.EQ.'y')THEN
                        WRITE(33,401)XX,YY,ZZ,ID
```

```
          END IF

 10 CONTINUE
    WRITE(6,22)NBAD
 22 FORMAT(1X,'NUMBER OF BIG ONES IS ',I4)

999 CLOSE(31)
    CLOSE(32)
    CLOSE(33)
    STOP
    END
```

```
          1
760840.12500   3181875.00000
760822.50000   3181889.00000
760755.00000   3181934.00000
760741.12500   3181943.00000
760687.50000   3181970.00000
760620.00000   3181997.00000
760581.37500   3182010.00000
760552.50000   3182018.00000
760485.00000   3182034.00000
760417.50000   3182048.00000
760350.00000   3182067.00000
760326.12500   3182078.00000
760282.50000   3182098.00000
760215.00000   3182145.00000
760214.81250   3182145.00000
760153.62500   3182213.00000
760147.50000   3182222.00000
760111.81250   3182280.00000
760081.31250   3182348.00000
760080.00000   3182351.00000
760047.18750   3182415.00000
760012.50000   3182478.00000
760008.12500   3182483.00000
759945.00000   3182544.00000
759931.62500   3182550.00000
759877.50000   3182574.00000
END
          2
760840.12500   3181875.00000
END
          3
761025.00000   3182975.00000
761007.31250   3182955.00000
760995.31250   3182888.00000
761002.87500   3182820.00000
END
          4
761430.00000   3183556.00000
761374.62500   3183495.00000
END
          5
761970.00000   3184399.00000
761960.87500   3184373.00000
761960.81250   3184305.00000
END
```

1,	-24.00000
2,	-24.00000
3,	-24.00000
4,	-24.00000
5,	-24.00000
6,	-24.00000
7,	-24.00000
8,	-24.00000
9,	-24.00000
10,	-24.00000
11,	-24.00000
12,	-23.00000
13,	-23.00000
14,	-23.00000
15,	-23.00000
16,	-23.00000
17,	-22.00000
18,	-22.00000
19,	-22.00000
20,	-22.00000
21,	-21.00000
22,	-21.00000
23,	-20.00000
24,	-20.00000
25,	-20.00000
26,	-20.00000
27,	-19.00000
28,	-19.00000
29,	-19.00000
30,	-19.00000
31,	-19.00000
32,	-19.00000
33,	-19.00000
34,	-18.00000
35,	-18.00000
36,	-18.00000
37,	-18.00000
38,	-18.00000
39,	-17.00000
40,	-17.00000
41,	-17.00000
42,	-17.00000
43,	-17.00000
44,	-17.00000
45,	-17.00000
46,	-17.00000

```
C PROGRAM POLYWIN.FOR--TO READ IN POLYGON DATA AND WRITE OUT
C                      A NEW FILE WITH A LIMITED WINDOW.  DOES NOT
C                      INCLUDE POLYGON COMMAND LINES IN WINDOWING

      CHARACTER SCATIN*40,SCATOUT*40
      CHARACTER CHRREC*80
      CHARACTER FORM*60,FORM2*60
      REAL*8 XX,YY

      PRINT*,'INPUT NAME OF POLYGON DATA TO BE WINDOWED'
      READ(*,200)SCATIN
  200 FORMAT(A40)
      OPEN(31,FILE=SCATIN,STATUS='OLD')
      WRITE(*,623)
  623 FORMAT(' ENTER FORMAT FOR POLYGON DATA'/
     &       ' X AND Y.  EXAMPLE:  (F12.2,F12.2)')
      READ(*,401)FORM
  401 FORMAT(A60)

      PRINT*,'INPUT WINDOW IN DATA UNITS:  XMIN,XMAX,YMIN,YMAX:'
      READ*,XMIN,XMAX,YMIN,YMAX

      PRINT*,'INPUT NAME OF OUTPUT FILE:'
      READ(*,300)SCATOUT
  300 FORMAT(A40)
      OPEN(32,FILE=SCATOUT,STATUS='NEW',CARRIAGECONTROL='LIST')

      READ(31,207,END=999)CHRREC
  207 FORMAT(A80)
      WRITE(32,207)CHRREC
      IFLAG=1
      DO 10 I=1,1000000
         READ(31,207,END=999)CHRREC

         IF(CHRREC(1:7).EQ.'POLYGON'.OR.CHRREC(1:7).EQ.'polygon'.OR.
     *      CHRREC(2:8).EQ.'POLYGON'.OR.CHRREC(2:8).EQ.'polygon')THEN
           IF(IFLAG.EQ.1)THEN
             WRITE(32,207)CHRREC
             IFLAG=2
           END IF

         ELSE
           READ(CHRREC(1:80),FMT=FORM)XX,YY

           IF(XX.LE.XMAX.AND.XX.GE.XMIN
     &               .AND.YY.LE.YMAX.AND.YY.GE.YMIN)THEN

             WRITE(32,705)XX,YY
  705        FORMAT(F12.2,F12.2)
             IFLAG=1
           ELSEIF(XX.EQ.1.0E+20.AND.IFLAG.EQ.1)THEN
```

```
            WRITE(32,706)XX,YY
706         FORMAT(E12.5,E12.5)
            IFLAG=1
          END IF

        END IF

 10 CONTINUE

999 CLOSE(31)
    CLOSE(32)
    STOP
    END
```

```
/* Program xyz2arc.c
       Converts ISM x,y,z ASCII scattered data format files into an
ARC/INFO  point data input format. Program reads from standard input
(stdin) or a filename given as the first of exactly two command line
arguments,  and writes to two separate output files. The output
filenames are made from a basename given as a single command line
argument (if stdin used for input) or the second of exactly two command
line arguments (if a file is used for input). Filename #1 consists of
the basename followed by ".xy", and filename #2 consists of the basename
followed by ".z". The program terminates upon receipt of EOF from input.
ISM scattered data input record format:
               One record per line, 3 fields (x,y,z) per line, separated
               by at least one blank.
ARC/INFO point data output record format:
       File #1: One record per line, 3 fields per line (IDNum, x, y),
               separated by at least one blank.
       File #2: One record per line, 2 fields per line (IDNum, z),
               separated by a comma.
The IDNum field for each file is simply a sequential number of points
processed so far so each point is given a unique IDNum value. Each file
must have as its last line a line containing the characters "END".

Designed for a Sun-4/280 system running Sun O.S. (ver. 4.03), but uses
only standard 'C' I/O calls, so should be portable to any UNIX system.
Author: John O. Wilson, Jr.  01 March 1990
Written for U.S. Geological Survey, Center for Coastal Geology
600 - 4th. St. South, St. Petersburg, FL  33701
(813) 893-3100.
*/
#include <stdio.h>
#include <malloc.h>
#include <string.h>


main(argc, argv)
       int argc;
       char *argv [];
{
       FILE *xyout, *zout;
        char *end = "END\n";
        char *linein, *lineout, fnamexy[20], fnamez[20];
        float x, y, z;
       int segnum = 1;

        if (argc == 3)
              {
               if (freopen(argv[1], "r", stdin) == NULL)
                     {
                     fputs("Error opening input file\n", stderr) ;
                     exit(1);
                     }
```

```
                strcpy(fnamexy, argv[2]);
                strcpy(fnamez, argv[2]);
                strcat(fnamexy, ".xy");
                strcat(fnamez, ".z");
                 if ((xyout = fopen(fnamexy, "w")) == NULL)
                        {
                         fputs("Error opening output file: ", stderr);
                        fputs(fnamexy, stderr);
                        fputs(".\n", stderr);
                         exit(1);
                         }
                 if ((zout = fopen(fnamez, "w")) == NULL)
                        {
                         fputs("Error opening output file: ", stderr);
                        fputs(fnamez, stderr);
                        fputs(".\n", stderr);
                         exit(1);
                         }
                }

        if (argc == 2)
                {
                strcpy(fnamexy, argv[1]);
                strcpy(fnamez, argv[1]);
                strcat(fnamexy, ".xy");
                strcat(fnamez, ".z");
                 if ((xyout = fopen(fnamexy, "w")) == NULL)
                        {
                         fputs("Error opening output file: ", stderr);
                        fputs(fnamexy, stderr);
                        fputs(".\n", stderr);
                         exit(1);
                         }
                 if ((zout = fopen(fnamez, "w")) == NULL)
                        {
                         fputs("Error opening output file: ", stderr);
                        fputs(fnamez, stderr);
                        fputs(".\n", stderr);
                         exit(1);
                         }
                }
       if (argc == 1)
                {
                fputs("Program: ", stderr);
                fputs(argv[0], stderr);
                fputs(".\nTransforms ISM x,y,z scattered data input ",
stderr);
                fputs("files\ninto 2 ARC/INFO point data output ",
stderr);
                fputs("format files.\n\nUsage:\n", stderr);
                fputs(argv[0], stderr);
                fputs(" [in_file] out_files_basename\n\n", stderr);
                exit(1);
                }
```

```
        linein = malloc(1024);
       lineout = &linein[512];

       /* get first valid x,y,z record for loop */
        gets(linein);

       /* start of main loop */
        while (!feof(stdin))
                {
               /* remove whitespace from x,y,z fields */
                sscanf(linein, "%f %f %f", &x, &y, &z);
               /* replace in buffer using correct ARC/INFO spacing */
                sprintf(lineout, "%8i%15.6f%15.6f\n", segnum, x, y);
                fputs(lineout, xyout);
               /* repeat for second file containing z value */
                sprintf(lineout, "%8i,%10.4f\n", segnum, z);
                fputs(lineout, zout);
               segnum++;
               /* get next line from stdin */
                gets(linein);
                }       /* end while */

       /* need to write a trailing ARC/INFO "END" record */
       fputs(end, xyout);
       fputs(end, zout);

        fclose(stdin);
        fclose(xyout);
       fclose(zout);

        free(linein);

        return(0);

}        /* main */

/*************************************************************************
****/
```