U. S. DEPARTMENT OF THE INTERIOR

U.S. GEOLOGICAL SURVEY

Cartographic Production for the Louisiana Barrier Island Erosion Study: 1. Processing Land Features

by Dorothy Hopkins and Jeffrey H. List

Open-File Report #91-212

Cartographic Production for the Louisiana Barrier Island Erosion Study: 1. Processing Land Features

by Dorothy Hopkins and Jeffrey H. List

Introduction

This report is the first 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 land features, i.e. the lines delineating the boundary between water and subaerial features such as islands and other coastal shorelines. These features are shown as color filled areas on the bathymetric maps of Atlas I-2150-B.

Processing these land features involved two main tasks: digitizing the linework representing the land feature boundaries with the Interactive Surface Modeling software package (ISM version 6.93B), and processing for color fill of these features with ARC/INFO software (version 5.0).

Digitizing with ISM

The first step is to inventory, and create an index map, of all the hydrographic (soundings) and topographic (shorelines) source maps that are available for the study area. The index map will be used for locating adjacent source maps when digitizing, and helpful in determining the approximate limit of data. The exact data limit must be drawn on, and digitized from, each of the individual source maps. Since it will be used in ARC/INFO to define the color filled areas, it will look best if drawn with smooth curves and without sharp angles wherever possible. This data limit boundary is an important feature since it will be used at every stage of processing, and it should be drawn very carefully. The data limit boundary should be digitized as an ISM polygon file since polygon sections can be reshaped as individual source maps are digitized, rather than as a polygon annotation file which, once completed, cannot be modified.

The data limit polygon should be smoothly drawn, and digitized, along the ends of the bathymetric track lines for the offshore area and, in the nearshore area, then it should extend to the edge of the bathymetric data, cutting across land features where necessary (Figure 1). The data limit would ideally include every data point but, in areas of sparse data, some points may be

excluded to ensure the smoothness of the data limit polygon. The boundary of topographic data would be included in this data limit polygon.

In some areas, to orient the map reader or where bathymetric data in the nearshore is not available, topographic data may be shown beyond the extent of the bathymetric data. In this case, more than one data limit boundary will be created. The area where there is bathymetric data should be digitized as an ISM polygon file. To supplement this data limit polygon file, vertical fault line files will be used to limit, or define, the areas having topographic data (Figure 2). The index map will be useful again here to determine areas which have bathymetric or topographic data, but not both.

Once the data limit boundary is completed, the study area can be divided into land regions. This is necessary when the study area is large, or when many land features are within the study area, to ensure that the number of polygon and fault points does not surpass the limitations of 42,144 points for a polygon file and 13,678 points for a fault file (ISM version 6.93B). When needed, these divisions should be made where land formations are naturally separated (between passes, channels, etc.), rather than along lines of longitude, so that islands and lakes can be digitized wholly in one regional file. It is very important for ARC/INFO processing to ensure that each feature is digitized only once and only in one file.

Before digitizing land or water features, all source materials covering the section to be digitized, and adjacent source maps, must be assembled to determine what type of file will be used

when digitizing each feature. ISM polygon and vertical fault files are the types that will be used to digitize all features, since sections of both these types of files can be reshaped as needed. Land features which are completely within the data limit boundary should be digitized as ISM polygons; land features which extend past, or are broken by, the data limit polygon should be digitized as ISM vertical fault lines (Figure 3). Water features (such as lakes and ponds which are surrounded by land) should also be divided into lines and polygons, and digitized in separate files. Some islands or lakes will be completely within the data limit polygon but not shown completely on one source map. These features should be digitized as polygons stopping at the edge of the source map, and then the polygon can be reshaped as the adjacent source maps are digitized since each feature can be digitized in only one file. The distinction between fault lines and polygons will be important when determining the order of generating the ARC/INFO coverages.

When digitizing land or water features, there should be no overlapping, duplicate linework (figure 4). This type of linework must be avoided since all crossing lines are broken at every intersection in ARC/INFO, and overlapping lines would break apart into thousands of individual arcs. We found that, as a general guideline, digitizing linework within a range of 1/1000 of the mapscale in ground units, and closer when possible, was effective for this project (example: source map scale 1:20,000, linework should be digitized to within 20 meters). Before beginning to digitize any polygon, a pencil line should be drawn on the source map to show where to begin and end each digitized

line so that the polygon is not digitized twice, or digitized past the origin point (figure 5A and 5B). Also, when determining the begin/end nodes of adjacent polygons, do not place nodes in close proximity (figure 6) or they might snap together, connecting the two polygons as one, during ARC/INFO processing. For continuing fault lines, nodes should be as close as possible without overlapping the linework (figure 7). Fault lines must be digitized to extend past the data limit polygon (figure 8) to ensure that, in ARC/INFO, they can be converted into closed polygons.

It is necessary to change the format of all ISM files, prior to conversion to the ARC/INFO format, since the ISM format has no standard number of decimal places (Attachment 1). All vertical fault files should be in the format (F12.2,F12.2,I12) for consistency (Attachment 2). The format of these files can be changed using the FLTFLT.FOR program (Attachment 3). Polygon files need the format (F12.2,F12.2), and can be changed with the POLYPOLY.FOR program (Attachment 4). When the fault and polygon files are in the proper format, the ISM2ARC.FOR program can be used to convert the files to the ARC/INFO format (Attachment 5).

Processing ARC/INFO Coverages

When the data is in the ARC/INFO format, the GENERATE command is used to create ARC/INFO coverages from the data files. The structure of the generated coverages will be the same as 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. A total of

five coverages will be generated and later will be combined into one main or total coverage. The data limit polygon should be generated first as one coverage, then all the land polygon files as a second coverage, and all the land fault files as a third coverage. The water coverages should be created in the same manner, with polygons the fourth coverage and fault lines as the fifth coverage. The water fault coverage can be COPYed as the main coverage, and arcs from all other coverages can be added to it, one at a time. After the coverages are generated, the BUILD command should be used with the LINE option, to create an Arc Attribute Table (AAT) for the coverage and update this table in the INFO directory. The Arc Attribute Table will hold values for defined items such as codes, etc. for each line feature, and will be updated during BUILD and CLEAN operations. Attributes, such as Depth and Code, should be added to the coverages with the ADDITEM command after BUILDing. These attributes will be useful when selecting features for color fill during ARC/INFO plotting.

In Arcedit, select the main (formerly the water fault) coverage as the EDITCOVERAGE, and EDITFEATURE ARCs. All arcs presently in the coverage (water fault lines) should have a value CALCULATEd for Code; we used code = 2 for all land features. Then the data limit arcs (linework) can be added to the water fault coverage using the GET command. While the new arcs from the data limit coverage are still selected, they should have an appropriate value CALCULATEd for any attributes such as Code or Depth; we used code = 9 for the data limit polygon, and code = 8 for data limit lines, if any. Then, the CLEAN command should be used with a tolerance of 1 as minimal FUZZY and DANGLE values to

prevent distortion at the edges of the coverage, and arcs from snapping together. The FUZZY distance refer to the minimum distance allowed between arcs before they snap 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 recognize arcs forming polygons and also create a Polygon Attribute Table (PAT) in the coverage and update the table in INFO. The coverage should then be checked for errors in Arcedit; arcs which fall short of the data limit line should be extended, and nodes which don't connect can be snapped together with the MOVE command. Using the GET command again, arcs from the water polygon coverage should now be added. Check to make sure all of the new polygons are closed, and that there are no node errors (dangling arcs).

The BUILD command is now used with the POLY option to recognized the new polygons and to update the PAT. The CREATELABELS command is used to create a label point inside every polygon (if the polygon does not already have one). At this time, attributes such as Code or Depth may be added to the PAT using the ADDITEM command. Then in Arcedit, values for the water polygon label attributes may be calculated. Before leaving Arcedit, using the GET command, add the arcs from the land fault and land polygon coverages, and make any necessary corrections. Then, BUILD the coverage again, and CREATELABELS for the new land polygons. The values for the land polygon labels will be calculated in Arcedit, and then BUILD the coverage to update the PAT.

Once all the linework errors have been corrected and the coverage is completely labeled, a solid color filled plot can be made. In Arcplot, polygons are recognized as a feature type and can be selected by values in the PAT. This is very helpful when checking the calculated values of each polygon attribute for errors. Thus, all polygons of a specified depth, or code, will be together in the selected set and can be filled with the same color, or pattern, in the final plot (Figure 9). Other features, such as contour lines, can also be added to the completed coverage and filled with patterns or color in Arcplot. The final color filled plot can show the entire study area with all land etc. features filled according to their feature type.

The final plot could also be further processed to produce color separated, screened negatives for printing. The procedure we used for this process is be outlined in another open file report in this series called, Processing Contours for Color Fill.

Acknowledgements

The authors wish to thank Rob Wertz and Keith Dalziel for their comments in reviewing this paper, and John Wilson for plotting the figures. We also owe thanks to Deborah L. Gillett of the Southwest Florida Water Management District for her assistance with Map Composition, the software used to draft the figures.

Figures 1-2 Data limit polygon with land features and bathymetric data.

Figure 3 Determine the type of file, for each feature to be digitized.

Figures 4-5 Digitizing techniques to be avoided.

Figures 6-8 Digitizing techniques to be avoided.

Figure 9 Final plot with pattern/color filled features.

Attachment 1 Data in non-standard format (raw ISM vertical fault file).

Attachment 2 Data in standard format (ISM vertical fault file after FLTFLT.FOR program).

Attachment 3 FLTFLT.FOR program to reformat ISM vertical fault files.

Attachment 4 POLYPOLY.FOR program to reformat ISM polygon files.

Attachment 5 ISM2ARC.FOR program to transform ISM files to ARC/INFO generate format.

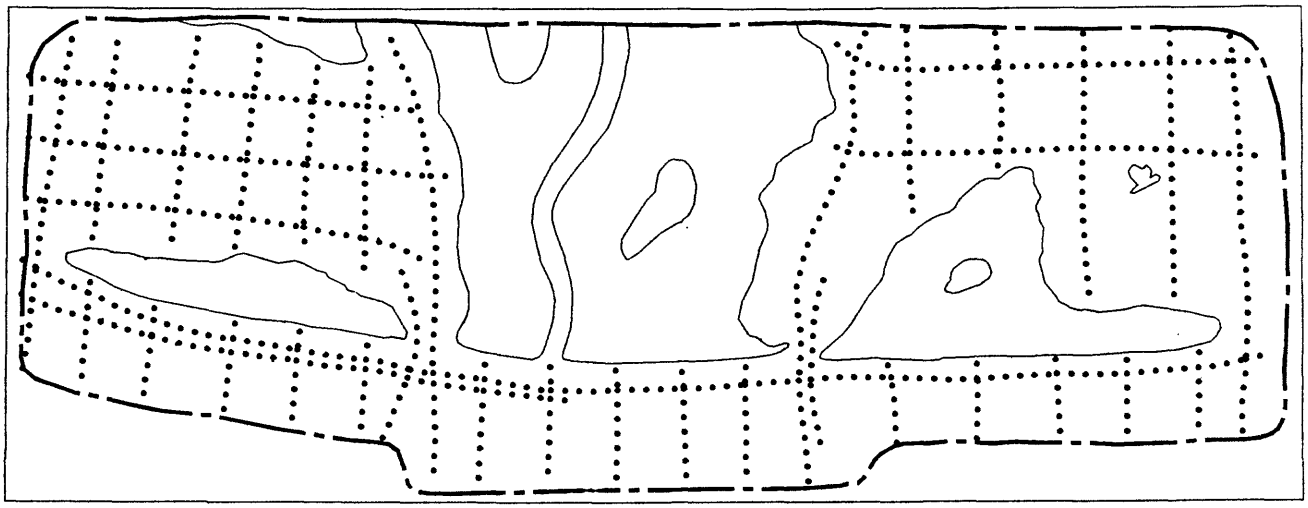

Figure 1

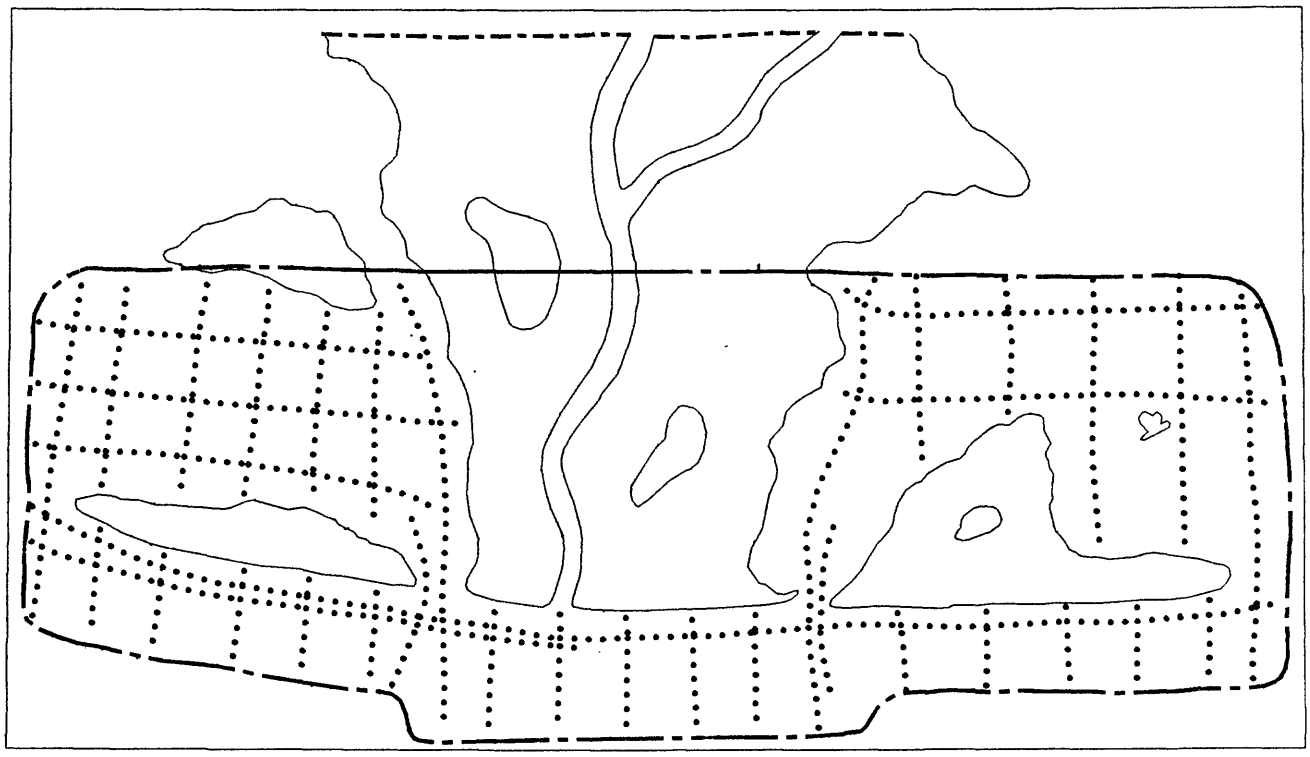

Figure 2

Topographic/Bathymetric Data Limit Polygon

Topographic Data limit Line

Bathymetric Track Lines

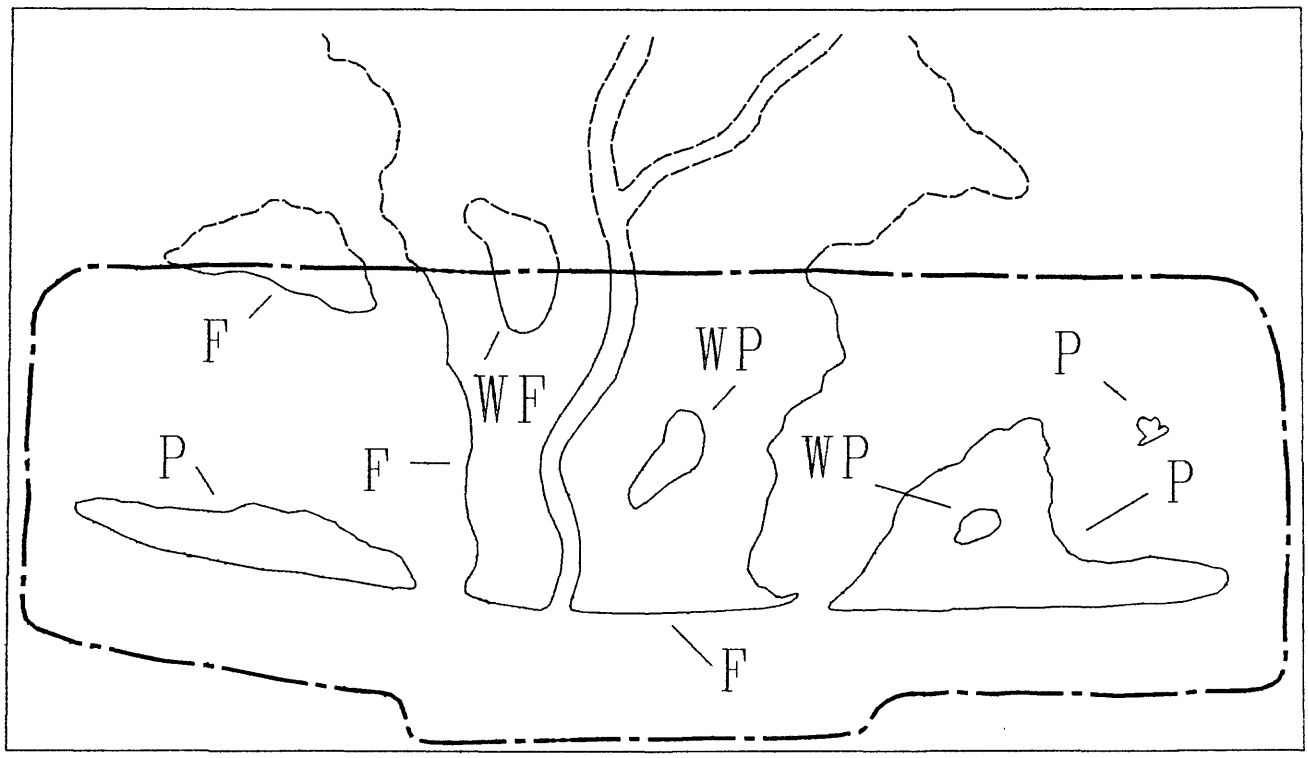

Figure 3

——·—— Data Limit Polygon

———— Land inside Data Limit

- - - - - - - - Land outside Data Limit

P = Polygon File

F = Fault File

WP = Water Polygon File

WF = Water Fault File

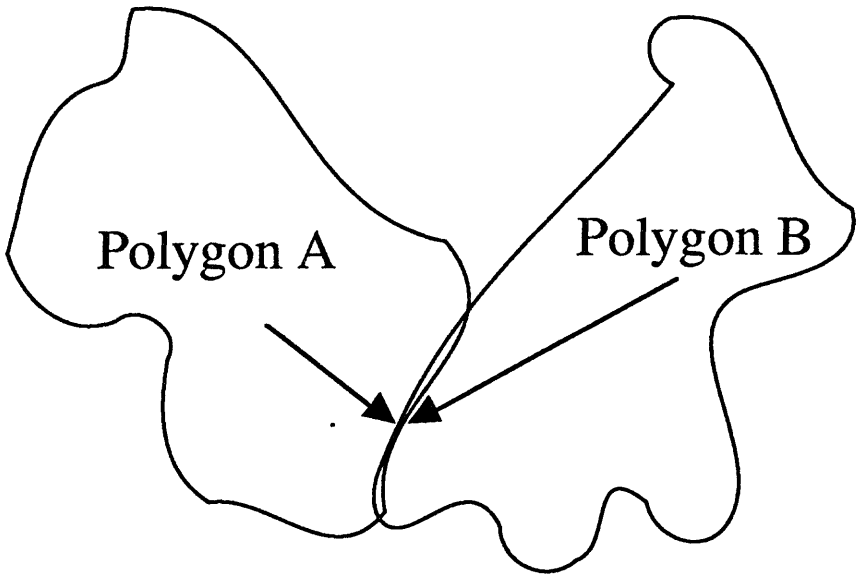

Figure 4
Overlapping, or duplicate lines.

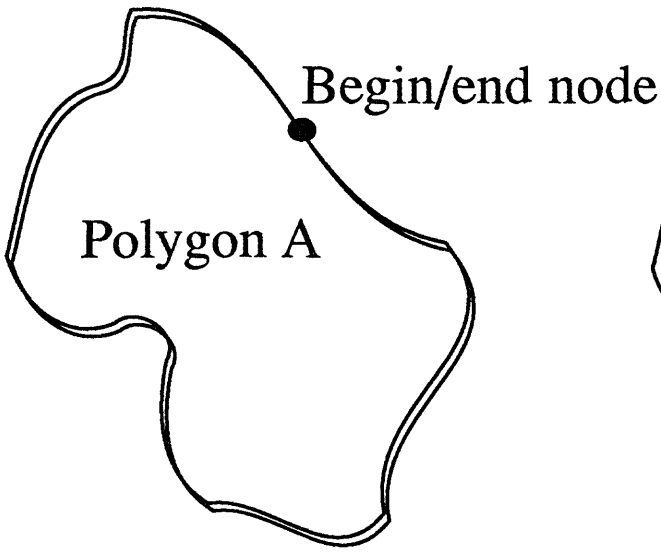

Figure 5A
Polygon digitized twice.

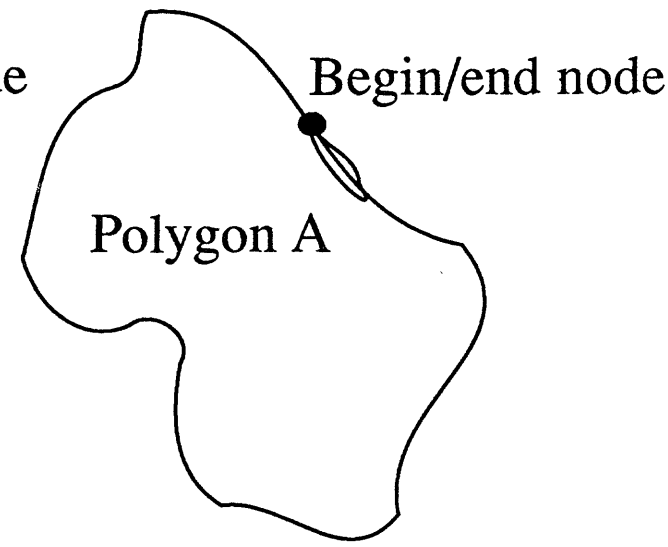

Figure 5B
Polygon digitized past begin node.

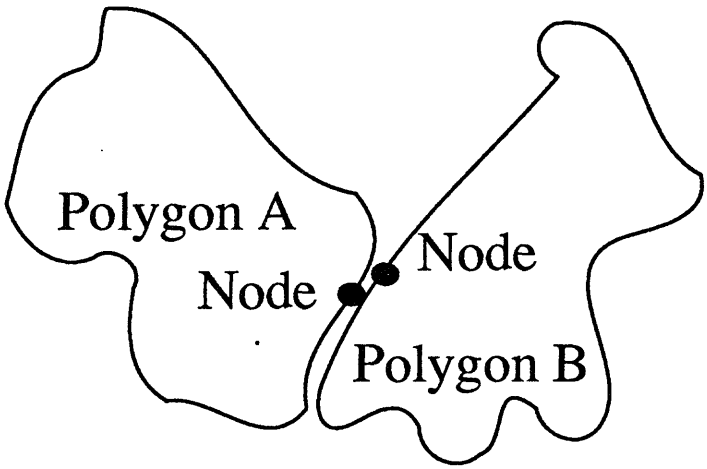

Figure 6
Poor node placement. (Polygon)

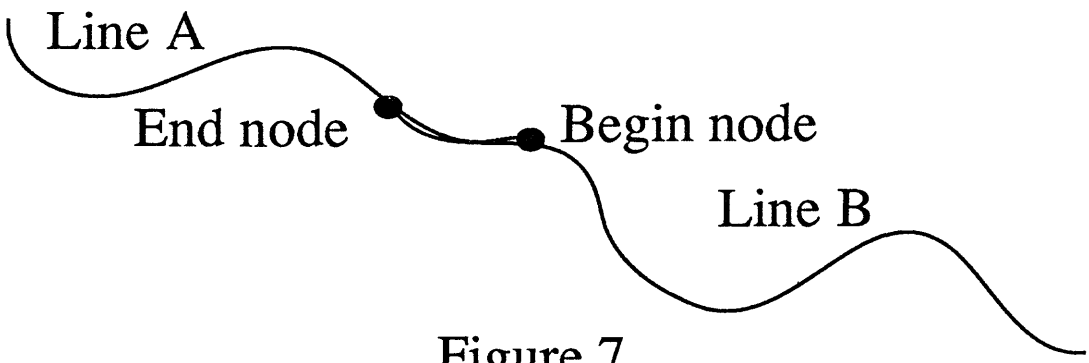

Figure 7
Poor node placement. (Fault Line)

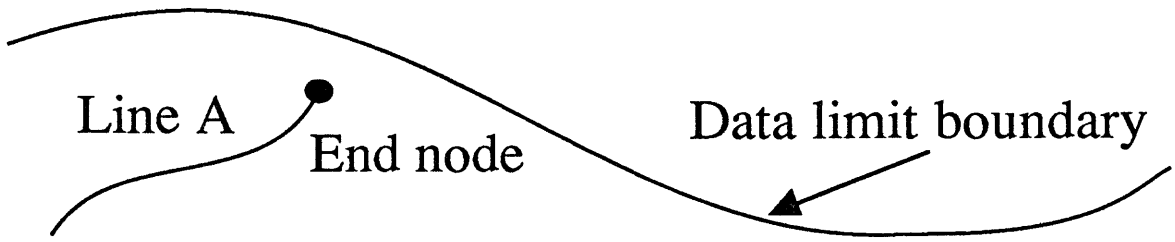

Figure 8
Poor node placement. (Fault Line)

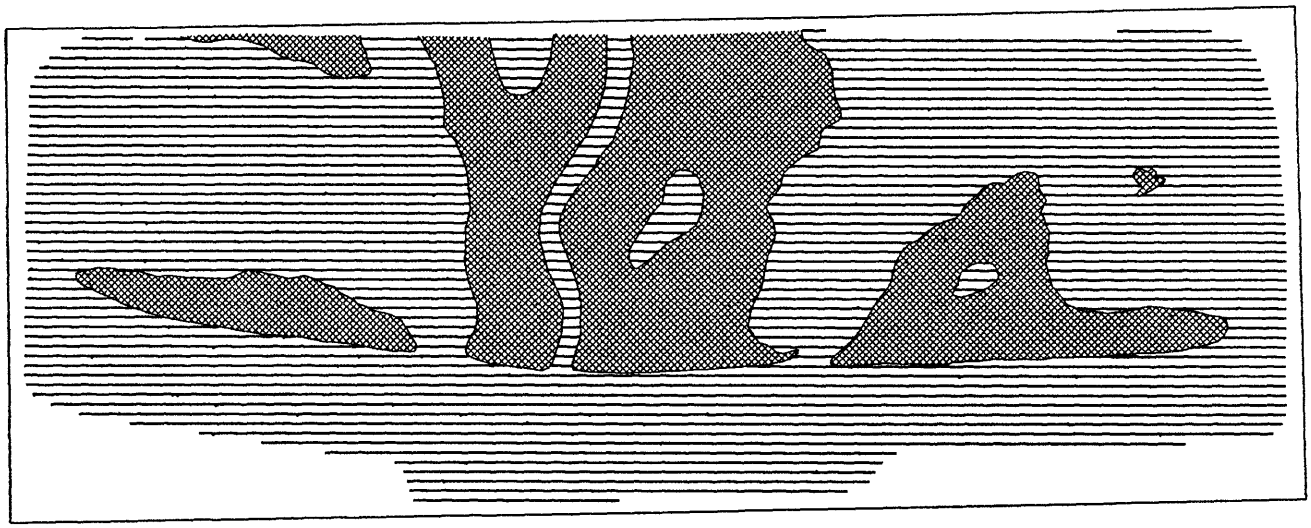

Figure 9

Topographic Data

Bathymetric Data

692103.2	3229412	1
692207.3	3229244	1
692139.1	3229011	1
692291.5	3228797	1
692650	3229034	1
693070.3	3229039	1
693281.2	3228638	1
693615	3228461	1
693891.3	3228497	1
694342.8	3228391	1
694829.6	3228439	1
695216	3228476	1
695293.5	3228688	1
695609.5	3228572	1
696393.8	3228550	1
696858.3	3228813	1
697551.2	3229174	1
698067.5	3229490	1
698072.3	3229770	1
697741.2	3229596	1
697377.5	3229639	1
697094.6	3229699	1
696747.7	3229790	1
697086.2	3230168	1
697710.2	3230276	1
698371	3230154	1
698741.7	3229746	1
698888.2	3229647	1
698959.9	3229270	1
699259.5	3228830	1
699560.8	3228725	1
699987.3	3228765	1
700293.5	3229079	1
700466	3228584	1
700153.9	3228100	1
699846.5	3227752	1
700177.2	3227486	1
700550.5	3227433	1
700808.5	3227234	1
700748.1	3227081	1
701310.5	3226862	1
701579.1	3226678	1
701779.3	3226402	1
701984.7	3226125	1
702447.4	3226059	1
702544	3225821	1
702720.9	3225730	1
703006.2	3225596	1
702980.4	3225432	1
703296	3225306	1
703448.7	3224956	1

692103.20	3229412.00	1
692207.30	3229244.00	1
692139.10	3229011.00	1
692291.50	3228797.00	1
692650.00	3229034.00	1
693070.30	3229039.00	1
693281.20	3228638.00	1
693615.00	3228461.00	1
693891.30	3228497.00	1
694342.80	3228391.00	1
694829.60	3228439.00	1
695216.00	3228476.00	1
695293.50	3228688.00	1
695609.50	3228572.00	1
696393.80	3228550.00	1
696858.30	3228813.00	1
697551.20	3229174.00	1
698067.50	3229490.00	1
698072.30	3229770.00	1
697741.20	3229596.00	1
697377.50	3229639.00	1
697094.60	3229699.00	1
696747.70	3229790.00	1
697086.20	3230168.00	1
697710.20	3230276.00	1
698371.00	3230154.00	1
698741.70	3229746.00	1
698888.20	3229647.00	1
698959.90	3229270.00	1
699259.50	3228830.00	1
699560.80	3228725.00	1
699987.30	3228765.00	1
700293.50	3229079.00	1
700466.00	3228584.00	1
700153.90	3228100.00	1
699846.50	3227752.00	1
700177.20	3227486.00	1
700550.50	3227433.00	1
700808.50	3227234.00	1
700748.10	3227081.00	1
701310.50	3226862.00	1
701579.10	3226678.00	1
701779.30	3226402.00	1
701984.70	3226125.00	1
702447.40	3226059.00	1
702544.00	3225821.00	1
702720.90	3225730.00	1
703006.20	3225596.00	1
702980.40	3225432.00	1
703296.00	3225306.00	1
703448.70	3224956.00	1

```
c PROGRAM FLTFLT.FOR-- Converts vertical fault file to new format.

      INTEGER ZZ
      CHARACTER FLTIN*40,FLTOUT*40
      CHARACTER FORM*60,FORM2*60,Q1*1
      REAL*8 XX,YY

      PRINT*,'Enter the name of the vertical fault file :'
      READ(*,200)FLTIN
  200 FORMAT(A40)
      OPEN(31,FILE=FLTIN,STATUS='OLD')

      PRINT*,'Enter the name of the re-formatted output file :'
      READ(*,300)FLTOUT
  300 FORMAT(A40)
      OPEN(32,FILE=FLTOUT,STATUS='NEW',CARRIAGECONTROL='LIST')

      WRITE(*,624)
  624 FORMAT(' Enter the format for the output file :'/
     &       ' X,Y AND Z. EXAMPLE: (F12.2,F12.2,I12)')
      READ(*,401)FORM2
  401 FORMAT(A60)

      DO 10 I=1,1000000
         READ(31,*,END=999)XX,YY,ZZ
         WRITE(32,FMT=FORM2)XX,YY,ZZ
   10 CONTINUE

  999 CLOSE(31)
      CLOSE(32)

      STOP
      END
```

```
C PROGRAM POLYPOLY.FOR-- Converts polygon file to new format

      CHARACTER POLYIN*40,POLYOUT*40
      CHARACTER TEXT*60,POLYPOLY*14,FORM*40
      INTEGER NUMBER
      REAL XX,YY

      POLYPOLY='POLYGON'
      FORM='(F12.0,F12.0)'

      PRINT*,'Enter the name of the polygon file to be converted:'
      READ(*,200)POLYIN
  200 FORMAT(A40)

      PRINT*,'Enter the name of the output polygon file:'
      READ(*,300)POLYOUT
  300 FORMAT(A40)

      OPEN(31,FILE=POLYIN,STATUS='OLD')
      OPEN(32,FILE=POLYOUT,STATUS='NEW',CARRIAGECONTROL='LIST')

      DO 10 I=1,10000000
         READ(31,100,END=999)TEXT
  100    FORMAT(A60)

         IF(TEXT(2:8).EQ.'polygon'.OR.TEXT(1:7).EQ.'polygon'.OR.
     *      TEXT(2:8).EQ.'POLYGON'.OR.TEXT(1:7).EQ.'POLYGON')THEN
            WRITE(32,150)POLYPOLY
  150       FORMAT(A14)
            GO TO 10
         ELSE
            READ(TEXT(1:60),FMT=FORM)XX,YY
               WRITE(32,400)XX,YY
  400          FORMAT(F12.2,F12.2)
         END IF

   10 CONTINUE

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

```
C PROGRAM ISM2ARC.FOR--TO READ IN AN ISM ANNOTATION OR VERTICAL FAULT FILE
C                       AND PRODUCE AN ARC/INFO FORMAT INPUT FILE.

      CHARACTER INFILE*40,OUTFILE*40
      CHARACTER BRKTXT*4,FTYPE*4
      CHARACTER CLONG*13,CLAT*13
      REAL*8 LONG,LAT
      LOGICAL BRKFLAG

      BRKTXT='END '
      BRKFLAG=.TRUE.
      ICNT=1

      PRINT*,' '
      PRINT*,'Enter name of ISM Polygon or Vertical Fault input file:'
      READ(*,100)INFILE
  100 FORMAT(A40)

  102 PRINT*,' '
      PRINT*,'Type of file?'
      PRINT*,'FLT - Vertical fault data'
      PRINT*,'PLY - Polygon data'
      PRINT*,'ANN - Annotation data (Line or Polygon)'
      PRINT*,'Enter the abbreviation for the type of file:'
      READ(*,106)FTYPE
      ITMP=ICHAR(FTYPE)                    !Look at first char. of string.
      IF(ITMP.GT.90)ITMP=ITMP-32           !Convert to upper case.
      IF(ITMP.NE.70.AND.ITMP.NE.80.AND.ITMP.NE.65) THEN
         PRINT*,'ERROR--Please enter one of the requested abbreviations'
         GO TO 102
      END IF
      ITYP=0                                    !ITYP is a flag value: 0 for
                                                annot.,
      IF(ITMP.EQ.70)ITYP=1                      !ITYP=1 for vertical fault
                                                files.

      PRINT*,' '
      PRINT*,'Enter name of ARC/INFO format output file:'
      READ(*,100)OUTFILE

      OPEN(31,FILE=INFILE,STATUS='OLD')
      OPEN(32,FILE=OUTFILE,STATUS='NEW',CARRIAGECONTROL='LIST')

c Common format statements for remainder of routine.
  106 FORMAT(A4)
  108 FORMAT(I10)
  204 FORMAT (1X,F12.5,1X,F15.5)

c Write initial segment counter.
      WRITE(32,108)ICNT
```

```
c Test which type of file.
      IF(ITYP)102,300,400                        !Branch on neg,zero,or pos
ITYP.

c Branch here for Polygon files.
c Line breakpoints indicated by non-numeric records or "hole" values.

  300 DO WHILE (.TRUE.)
        READ(31,302,END=999)CLONG,CLAT           !Read record in char. format.
  302   FORMAT(A12,A12)
        IF(ICHAR(CLONG).GT.57) THEN              !Is record non-numeric?
            IF(.NOT.BRKFLAG) THEN                !Have we just written 'END'?
                BRKFLAG=.TRUE.                   !If not, set flag, and
                WRITE(32,106)BRKTXT              !write out line break
                                                  command.
                ICNT=ICNT+1                      !Also, increment next counter
                WRITE(32,108)ICNT                !value and write it out.
            END IF
        ELSE                                     !Record has numeric fields.
            READ(CLONG,*)LONG                    !Convert values to floating
            READ(CLAT,*)LAT                      !point internal
                                                  representation.
            IF(LONG.GT.1000000000.) THEN         !Test for a "hole" value.
                IF(.NOT.BRKFLAG) THEN            !Again, don't duplicate
                    BRKFLAG=.TRUE.               !break commands if one
                    WRITE(32,106)BRKTXT          !was just written.
                    ICNT=ICNT+1                  !Also, increment next counter
                    WRITE(32,108)ICNT            !value and write it out.
                END IF
            ELSE                                 !If numeric and not hole,
                WRITE(32,204)LONG,LAT            !write the record
                BRKFLAG=.FALSE.                  !and reset break flag.
            END IF
         END IF
      END DO

c Vertical Fault type files: 3 fields. X, Y, and line segment number.

  400 READ(31,402)LONG,LAT,ICONT1                !Initial record read
  402 FORMAT(F12.2,F12.2,T28,I9)

      WRITE(32,204)LONG,LAT                      !and write.

      DO WHILE (.TRUE.)
         READ(31,402,END=999)LONG,LAT,ICONT2

         IF(ICONT2.NE.ICONT1)THEN                !Has line number changed?
             WRITE(32,106)BRKTXT                 !Write out line break
                                                  command.
             ICNT=ICNT+1                         !Also, increment next counter
             WRITE(32,108)ICNT                   !value and write it out.
             ICONT1=ICONT2                       !New current line number.
         END IF
         WRITE(32,204)LONG,LAT                   !Use same write format for
                                                  all.
```

```
    END DO

999 WRITE(32,106)BRKTXT                    !Write out last line break.
    CLOSE(31)
    CLOSE(32)
    PRINT*,' '
    PRINT*,'Done!'
    END
```