UR1983_30

1/7

1983/30. A FORTRAN program for selecting data from inside or outside a specified area (Revision 2).

R.G. Richardson

Abstract

This program outputs the data for sample points which lie inside or outside a polygonal area specified in the same co-ordinate system as used for the sample points. The output may be printed or saved for further processing.

THE PROGRAM

STNOUT (Appendix 1)

This program uses a point-in-polygon subroutine (Salomon, 1978) to determine if a sample point lies inside the boundary of a specified polygonal area or on, or outside, the boundary. Although the program requests co-ordinates in kilometres, similar co-ordinate systems may be used. The program reads the inside/outside option and the area co-ordinates from logical unit 5 and the data from logical unit 4. The co-ordinates of the area are written on logical unit 7 and the data requested from either inside or outside the area is written on logical unit 6.

Control data read from logical unit 5 is:

AOK - I for selecting data from inside the area, O otherwise.
NCORD - the number of co-ordinate pairs to define the closed polygonal area.

For each co-ordinate:

XTMP, YTMP - the co-ordinate pair specifying the vertex. The vertices are given in clockwise order around the polygon.

Data input from logical unit 4 is:

A - format (20A4)
  - a complete line is read from the file. This is then written to a buffer and the sample point co-ordinates re-read from there.
X, Y - format (10X, 2F9.1)
  - the east and north co-ordinates of the sample point in metres.

Output on logical unit 6 is the data in the selected area in the same format as the original data on logical unit 4. The polygon co-ordinates are listed on logical unit 7.

REFERENCE

SALOMON, K.B. 1978. An efficient point-in-polygon algorithm. *Computers and Geosciences*. 4:173-178.

[12 July 1983]

APPENDIX 1

Program STNOUT

```
$TITL  STNOUT.FTN - STATIONS INSIDE OR OUTSIDE A COORDINATED AREA
C USED TO SEARCH A DATA FILE (LU 4) FOR STATIONS INSIDE AND OUTSIDE
C A COORDINATED AREA
C CONTROL INPUT LU  5
C COORDINATE LIST  LU 7
C STATIONS EITHER INSIDE OR OUTSIDE AS REQUESTED ON LU  6
C REQUIRES SUBROUTINE FSTJDN
      DIMENSION X(100),Y(100),ANUM(2),A(20),YINTVL(100),RSLOPE(100)
      DIMENSION TEXT(20)
C USE AS A BUFFER
      LOGICAL INOUT
      INTEGER*2 AOK,IN,OUT,SWATH(100,25)
      LOGICAL WITHIN,IER
      DATA IN/1HI/,OUT/1HO/
C USED TO SET INSIDE OR OUTSIDE
      OPEN(UNIT=5,FILE='CON:')
C OPEN THE CONSOLE AS LU 5
      REWIND 4
      XMAX=-9.9E+9
      YMAX=XMAX
      XMIN=-XMAX
      YMIN=XMIN
C USE FOR RANGE OF COORDS
    9 WRITE(5,200)
  200 FORMAT(' DATA INSIDE OR OUTSIDE AREA?')
      READ(5,201) AOK
  201 FORMAT(A1)
      IF (AOK .NE. IN .AND. AOK .NE. OUT) GOTO 9
      WITHIN=AOK .EQ. IN
   10 WRITE(5,102)
  102 FORMAT(' NO. OF COORDINATE PAIRS TO CLOSE AREA?')
      READ(5,*) NCORD
      IF (NCORD .LE. 2) GOTO 10
      WRITE(5,103)
  103 FORMAT(' ENTER COORDS. (KM) IN ORDER'/' EAST   NORTH')
      DO 15 I=1,NCORD
      READ(5,*) XTMP,YTMP
      XTMP=XTMP*1000.
      YTMP=YTMP*1000.
C CONVERT TO METRES
      XMAX=AMAX1(XMAX,XTMP)
      XMIN=AMIN1(XMIN,XTMP)
      YMAX=AMAX1(YMAX,YTMP)
      YMIN=AMIN1(YMIN,YTMP)
      X(I)=XTMP
      Y(I)=YTMP
   15 CONTINUE
C READ THE COORDS OF THE AREA
      IF (X(1) .EQ. X(NCORD) .AND. Y(1) .EQ. Y(NCORD)) GOTO 18
      NCORD=NCORD+1
      X(NCORD)=X(1)
      Y(NCORD)=Y(1)
C REQUIRE THAT AREA IS CLOSED
   18 WRITE(7,100)
  100 FORMAT(' COORDS. ARE')
      WRITE(7,101) (X(I),Y(I),I=1,NCORD)
  101 FORMAT(3(F12.0,F9.0))
      NCORD=NCORD-1
```

```
C ROUTINE FOR LOCATING POINTS DOESN'T REQUIRE CLOSURE
      CALL PREPLY(X,Y,NCORD,YINTVL,INTVLS,SWATH,RSLOPE)
C THE INITIAL SET UP
      WRITE(7,107)
  107 FORMAT(//)
   20 READ(4,104,END=25) A
  104 FORMAT(20A4)
      ENCODE(TEXT,104) A
C WRITE TO BUFFER TO ALLOW GETTING COORDS EASILY
      DECODE(TEXT,105) XTMP,YTMP
  105 FORMAT(10X,2F9.1)
      IF (XTMP .LT. XMIN .OR. XTMP .GT. XMAX) GOTO 23
      IF (YTMP .LT. YMIN .OR. YTMP .GT. YMAX) GOTO 23
C NOT POSSIBLY IN AREA
      LOC=-1
C DEFAULT TO BE OUTSIDE AREA
      IF (INOUT(XTMP,YTMP,X,Y,YINTVL,INTVLS,SWATH,RSLOPE)) LOC=+1
      IF (LOC .GE. 0 .AND. WITHIN) GOTO 24
C WITHIN AREA AND WANT WITHIN AREA
      IF (LOC .GE. 0 .AND. .NOT. WITHIN) GOTO 20
C WITHIN AREA AND WANTED OUTSIDE AREA
C ALL THAT IS LEFT IS OUTSIDE AREA
C IF IN AREA GO DIRECT TO THE PRINT STATEMENT
   23 IF (WITHIN) GOTO 20
C NOT INSIDE AREA
   24 WRITE(6,104) A
      GOTO 20
   25 CONTINUE
      ENDFILE 6
      STOP
      END
$TITL  FSTJDN.FTN - POINT IN POLYGON
C PROGRAM COLLECTION FASTJORDAN
C   SALOMON, K.B., 1978. AN EFFICIENT POINT-IN-POLYGON ALGORITHM
C     COMPUTERS AND GEOSCIENCES, V4, NO. 2, P. 173-178
C
C USE BY READING VERTICES OF POLYGON IN ORDER (BUT NOT CLOSING IT)
C AND CALLING PREPLY ONCE BEFORE THE SEARCH IS BEGUN.  THE ACTUAL
C SEARCHING IS DONE BY MEANS OF THE LOGICAL FUNCTION INOUT.
C
      SUBROUTINE PREPLY(X,Y,NUVERT,YINTVL,INTVLS,SWATH,RSLOPE)
C
C*************************************************************************
C  THIS ROUTINE PREPARES THE POLYGON CONSISTING OF THE NUVERT VERTICES
C  (X(I),Y(I)) BY FIRST SORTING THE SEGMENT Y-END POINTS INTO
C  DECREASING ORDER AND FORMING AN INTERVAL FOR EACH CONSECUTIVE PAIR:
C  (YINTVL(I),YINTVL(I+1)), I=1,INTVLS.  THIS IS PERFORMED BY CALLING
C  SORT.
C       THE CODE CONSISTING OF THE DO 100 AND DO 200 LOOPS CONSTRUCTS,
C  FOR EACH INTERVAL I, THE LIST OF SEGMENTS TO BE TESTED BY INOUT.
C  THIS LIST IS PLACED IN THE I-TH ROW OF SWATH.  THE FIRST ENTRY,
C  SWATH(I,1), WILL BE SET TO THE NUMBER OF SEGMENTS IN THE ROW.  NOTE
C  THAT AS YINTVL CONTAINS NO REDUNDANCIES, I.E. YINTVL(I) IS STRICTLY
C  GREATER THAN YINTVL(I+1), NO HORIZONTAL SEGMENTS WILL BE PLACED IN
C  THE LIST.
C       THE CODE CONSISTING OF THE DO 300 LOOP ESTABLISHES THE
C  RECIPROCAL SLOPE FOR EACH NON-HORIZONTAL SEGMENT.  THIS IS TO BE
C  USED BY INOUT.  FINALLY, THE SEGMENTS WITHIN A ROW OF SWATH ARE
C  ORDERED FROM LEFT-TO-RIGHT.
C
```

```
C*************************************************************************
C
      INTEGER*2 SWATH(100,25)
      REAL X(100),Y(100),YINTVL(100),RSLOPE(100)
      CALL SORT(Y,NUVERT,YINTVL,INTVLS)
      IF (INTVLS .LE. 0) GOTO 400
      X(NUVERT+1)=X(1)
      Y(NUVERT+1)=Y(1)
      DO 100 I=1,INTVLS
  100 SWATH(I,1)=0
      DO 200 I=1,INTVLS
         DO 200 J=1,NUVERT
         IF (Y(J).GE.YINTVL(I) .AND. YINTVL(I+1).GE.Y(J+1) .OR.
     *       Y(J+1).GE.YINTVL(I) .AND. YINTVL(I+1).GE.Y(J))
     *                                    CALL INCLUD(SWATH,I,J)
  200 CONTINUE
      DO 300 I=1,NUVERT
      IF (Y(I).EQ.Y(I+1)) GOTO 300
      RSLOPE(I)=(X(I+1)-X(I))/(Y(I+1)-Y(I))
  300 CONTINUE
      CALL ORDER(X,Y,YINTVL,INTVLS,SWATH,RSLOPE)
      RETURN
  400 WRITE(7,401)
  401 FORMAT(' ******* PREP OF POLYGON ABORTED SINCE NO INTERVALS',
     * ' CONSTRUCTED')
      STOP
      END
      SUBROUTINE SORT(Y,NUVERT,YINTVL,INTVLS)
C
C*************************************************************************
C
C  ROUTINE ESTABLISHES THE INTERVALS OF THE Y-AXIS DEFINED BY THE
C  ENDPOINTS OF THE POLYGON'S SEGMENTS.  THE DO 100 LOOP INITIALLISES
C  YSORT FROM THE SEGMENT Y-END POINTS.  THE DO 200 LOOPS SORT YSORT
C INTO DESCENDING ORDER.  THE DO 300 LOOP ELIMINATES REDUNDANCIES IN
C  YSORT AND PLACES IRREDUNDANT SORTED Y'S INTO YINTVL.  IT ALSO SETS
C  INTVLS TO THE TRUE NUMBER OF Y INTERVALS.  JUST PRIOR TO RETURNING
C  A FINAL INTERVAL EXTENDING TO '-INFINITY' IS ESTABLISHED.
C
C*************************************************************************
C
      REAL Y(100),YINTVL(100),YSORT(100)
      INTEGER*2 UPPER
      DO 100 I=1,NUVERT
  100 YSORT(I)=Y(I)
      UPPER=NUVERT-1
      DO 200 I=1,UPPER
         IPLS1=I+1
         DO 200 J=IPLS1,NUVERT
         IF (YSORT(I).GE.YSORT(J)) GOTO 200
             TEMP=YSORT(I)
             YSORT(I)=YSORT(J)
             YSORT(J)=TEMP
  200 CONTINUE
      YINTVL(1)=YSORT(1)
      INTVLS=0
      DO 300 I=1,UPPER
         IF (YSORT(I).EQ.YSORT(I+1)) GOTO 300
         INTVLS=INTVLS+1
         YINTVL(INTVLS+1)=YSORT(I+1)
```

```
  300 CONTINUE
      YINTVL(INTVLS+2)=-1.0E75
      RETURN
      END
      SUBROUTINE INCLUD(SWATH,I,J)
C
C*****************************************************************************
C  ROUTINE PLACES THE J-TH POLYGON SEGMENT INTO THE NEXT AVAILABLE
C  LOCATION IN ROW I OF SWATH.
C
C*****************************************************************************
C
      INTEGER*2 SWATH(100,25),POINTR
      SWATH(I,1)=SWATH(I,1)+1
      POINTR=SWATH(I,1)
      SWATH(I,POINTR+1)=J
      RETURN
      END
      SUBROUTINE ORDER(X,Y,YINTVL,INTVLS,SWATH,RSLOPE)
C
C*****************************************************************************
C
C  FOR EACH INTERVAL, A HORIZONTAL LINE IS PASSED THROUGH THE MIDDLE
C  (YMID) OF THE INTERVAL.  THE DO 100 LOOP PLACES THE X-INTERSECTION
C  OF EACH SEGMENT IN THIS SWATH SO THAT THESE INTERSECTIONS OCCUR
C  FROM LEFT-TO-RIGHT.
C
C*****************************************************************************
C
      REAL X(100),Y(100),YINTVL(100),RSLOPE(100),XINTSC(25)
      INTEGER*2 SWATH(100,25),POINTR,SEGNO,UPPER
      LOGICAL VERTSG
      DO 200 INTVAL=1,INTVLS
         NMBSEG=SWATH(INTVAL,1)
         YMID=(YINTVL(INTVAL)+YINTVL(INTVAL+1))/2.0
         DO 100 POINTR=1,NMBSEG
            SEGNO=SWATH(INTVAL,POINTR+1)
            VERTSG=ABS(X(SEGNO+1)-X(SEGNO)) .LT.  1.0E-5
            IF (VERTSG) XINTSC(POINTR)=X(SEGNO)
            IF (.NOT.  VERTSG) XINTSC(POINTR)=X(SEGNO)+
     *                                    RSLOPE(SEGNO)*(YMID-Y(SEGNO))
  100    CONTINUE
         IF (NMBSEG.LT.2 .OR.  MOD(NMBSEG,2).NE.0) GOTO 300
         UPPER=NMBSEG-1
         DO 200 I=1,UPPER
            IPLS1=I+1
            DO 200 J=IPLS1,NMBSEG
               IF (XINTSC(I).LE.XINTSC(J)) GOTO 200
               TEMP=XINTSC(I)
               XINTSC(I)=XINTSC(J)
               XINTSC(J)=TEMP
               ITEMP=SWATH(INTVAL,I+1)
               SWATH(INTVAL,I+1)=SWATH(INTVAL,J+1)
               SWATH(INTVAL,J+1)=ITEMP
  200 CONTINUE
      RETURN
  300 WRITE(7,301) INTVAL
  301 FORMAT(' ** PREP OF POLYGON ABORTED.  INTERVAL ',I5/
     * ' HAS EITHER LESS THAN TWO SEGMENTS OR AN ODD NUMBER OF THEM')
      STOP
```

```
      END
      LOGICAL FUNCTION INOUT(XP,YP,X,Y,YINTVL,INTVLS,SWATH,RSLOPE)
C
C*****************************************************************************
C
C  THE FOUR LINES ENCLOSED IN DASHES DETERMINE THE INTERVAL CONTAINING
C  YP.  THE DO 400 LOOP CONTINUES UNTIL THE FIRST SEGMENT WITHIN THE
C  INTERVAL FALL TO THE LEFT OF (XP,YP).  IN THIS EVENT, INOUT IS SET
C  .TRUE.  IFF AN EVEN NUMBER OF SEGMENTS HAS BEEN TESTED.
C
C*****************************************************************************
C
      REAL X(100),Y(100),YINTVL(100),RSLOPE(100)
      INTEGER*2 SWATH(100,25),SEGNO
      INOUT=.FALSE.
C-------------------------------------------------
      INTVAL=0
  100 INTVAL=INTVAL+1
      IF (YINTVL(INTVAL) .GT.  YP) GOTO 100
      INTVAL=INTVAL-1
C-------------------------------------------------
  300 IF (INTVAL.LT.1 .OR.  INTVAL.GT.INTVLS) RETURN
      NMBSEG=SWATH(INTVAL,1)+1
      DO 400 I=2,NMBSEG
         SEGNO=SWATH(INTVAL,I)
         IF(XP-X(SEGNO).LE.(YP-Y(SEGNO))*RSLOPE(SEGNO)) GOTO 500
  400 CONTINUE
      RETURN
  500 INOUT=MOD(I,2) .EQ.  1
      RETURN
      END
```