1982

ALGORITHM: A Two Dimensional Domain Processor

John R. Rice
Purdue University, jrr@cs.purdue.edu

Report Number:
81-417

ALGORITHM: A TWO DIMENSIONAL DOMAIN PROCESSOR

John R. Rice

Mathematical Sciences
Purdue University

CSD-TR 417
October 11, 1982

ABSTRACT

This paper presents a Fortran algorithm for processing general two dimensional domains through the use of a rectangular grid overlay. The motivation and techniques of this algorithm are presented in the companion paper: *Numerical Computation with General Two Dimensional Domains*. This paper describes the assumptions used by the algorithm, describes a set of 20 parameterized domains useful for testing such algorithms and documents the two principal subprograms of the algorithm.

1. APPLICABILITY OF THE DOMAIN PROCESSOR

This algorithm uses a large number of heuristics and it is impractical (if not impossible) to precisely define those domains for which the algorithm is intended to work. A careful reading of the companion paper [Rice, 1982] reveals many of the assumptions used, we list the more important ones here.

1. The domain has more than 2 or 3 interior grid points.
2. The boundary does not enter a grid element several times.
3. The boundary is fairly smooth on the scale of the grid.
4. The coordinates and parameters of the problem are all ordinary sized numbers.
5. The parameterization of the boundary pieces is made by smooth, monotonic and well-behaved functions.
6. There are no corners on the boundary except at the ends of boundary pieces.
7. Holes, if any, are separated so that there is always an interior grid point between boundaries of different subdomains and the main domain.
8. The interior grid points of the domain are connected through the grid.
9. The grid is reasonably uniform.

Most of these assumptions are automatically satisfied if the grid is fine enough.

2. A SET OF TEST DOMAINS

This algorithm contains so many heuristics that we cannot evaluate its reliability except by actual testing. It and its predecessors have been used for several years on a wide variety of domains and each time it failed, the situation was examined to see whether the assumptions were violated or the heuristics were ineffective. Thus the algorithm has evolved to become more robust through this use. In addition, a test set of 20 parameterized domains has been constructed with the objectives:

1. To exercise the domain processor in a general sense.
2. To include special situations which are likely to cause trouble for the domain processor or one of its heuristics.
3. To provide a bench mark against which to compare as the algorithm is modified.

This test set is included with the algorithm along with a driver to set parameters and exercise the algorithm for any particular member of the set. Figure 1 shows two cases of each domain in the test set along with the parameterization.

3. FILE ORGANIZATION

The algorithm, test drivers are organized as follows:

File 1: ALGORITHM
- SUBROUTINE REGION
- SUBROUTINE DOMAIN
- Other subprograms in alphabetical order

File 2: DRIVER1 (with set of 20 test domains)
- Main program
- SUBROUTINE BCOORD (boundary parameterizations)
- SUBROUTINE SETRNG (initialize domain constants)

File 3: DATA (uses each test domain 4 times)

File 4: DRIVER2 (domain with two holes)

Reference

J. R. Rice, Numerical computation with general two dimensional domains, CSD-TR 416, Computer Science, Purdue University, 1982.

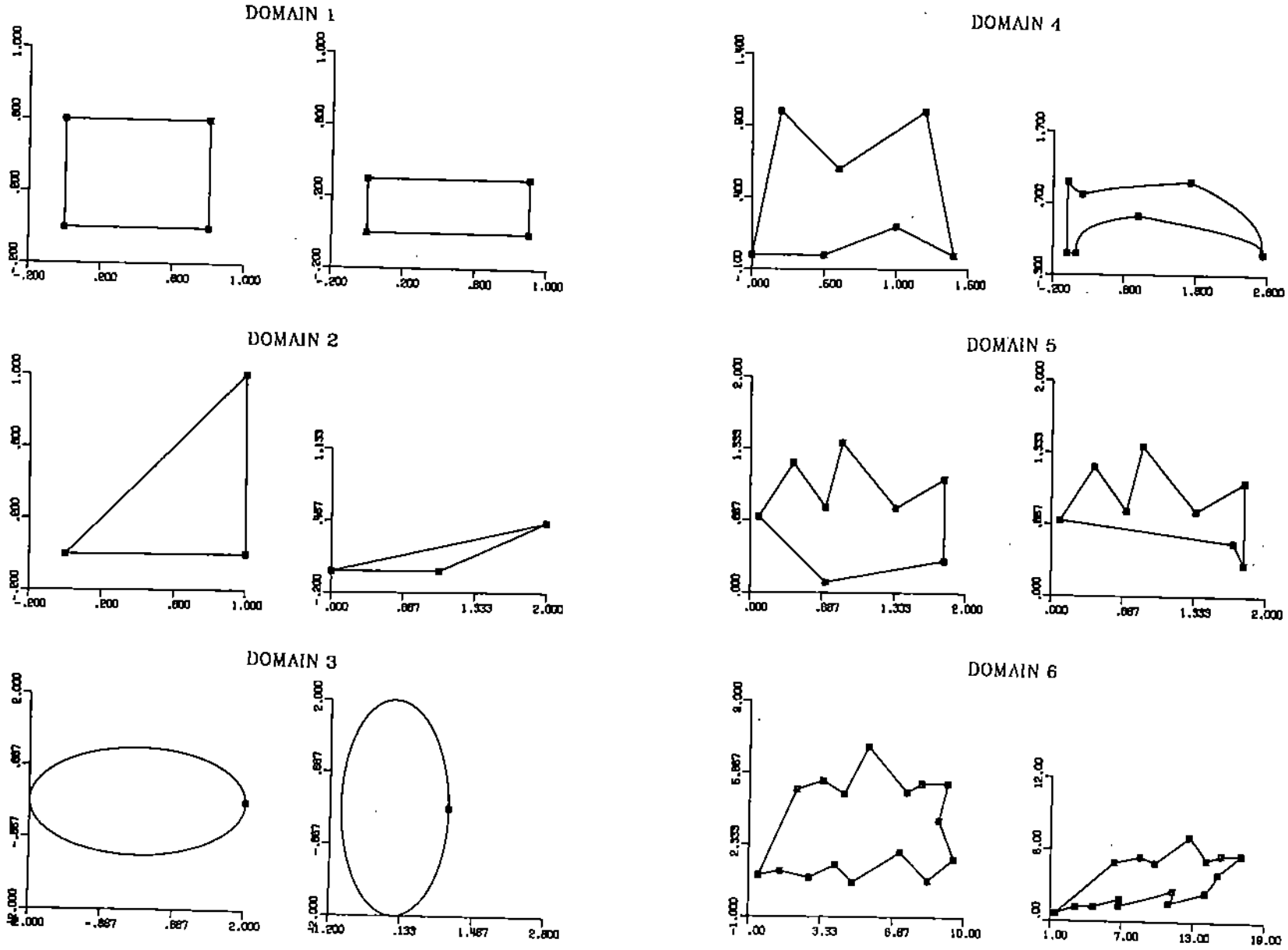


Figure 1. Examples of the 20 domains in the test set. The first example in each case uses the default parameters for the domains.

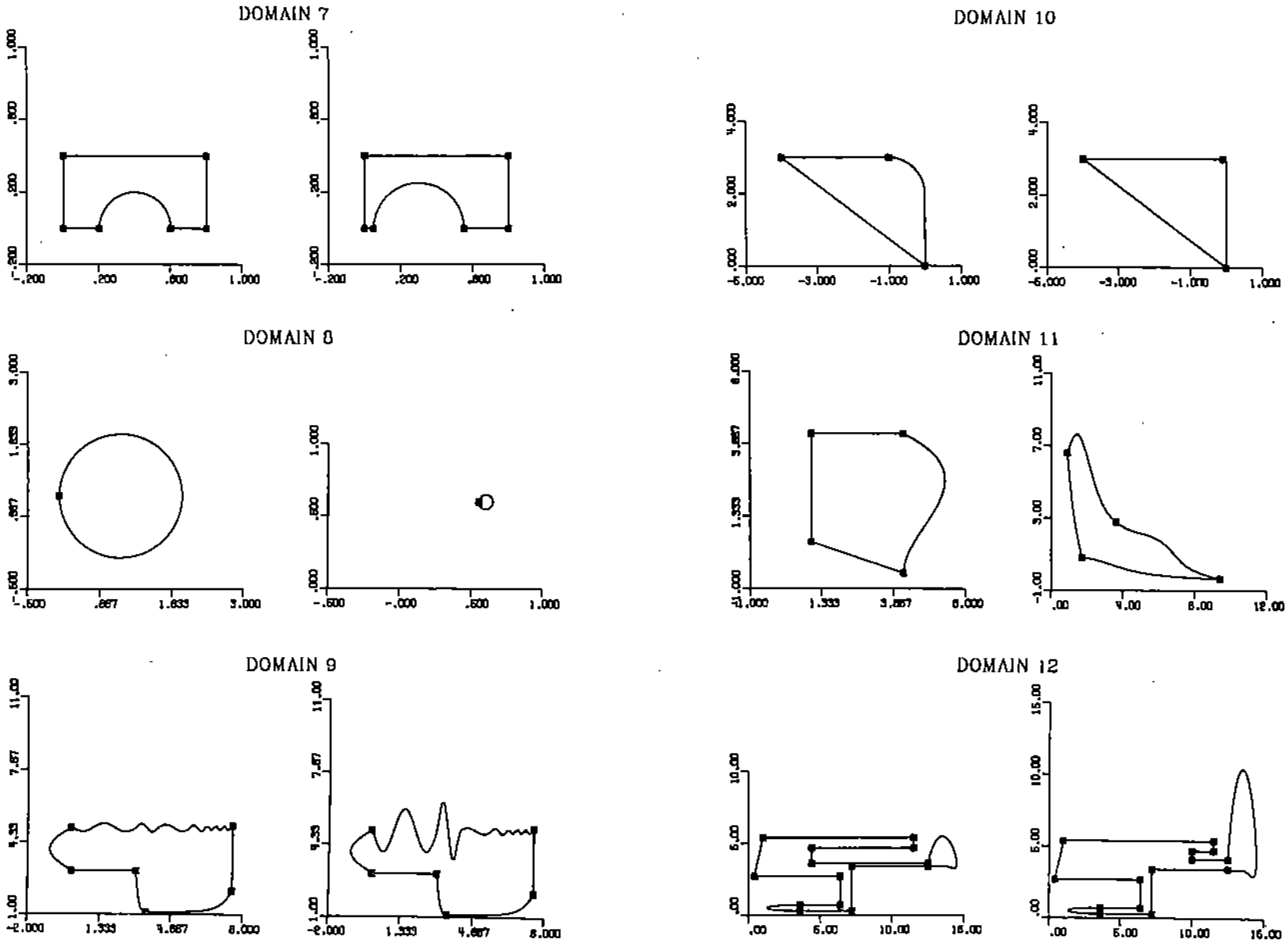


Figure 1. (continued)

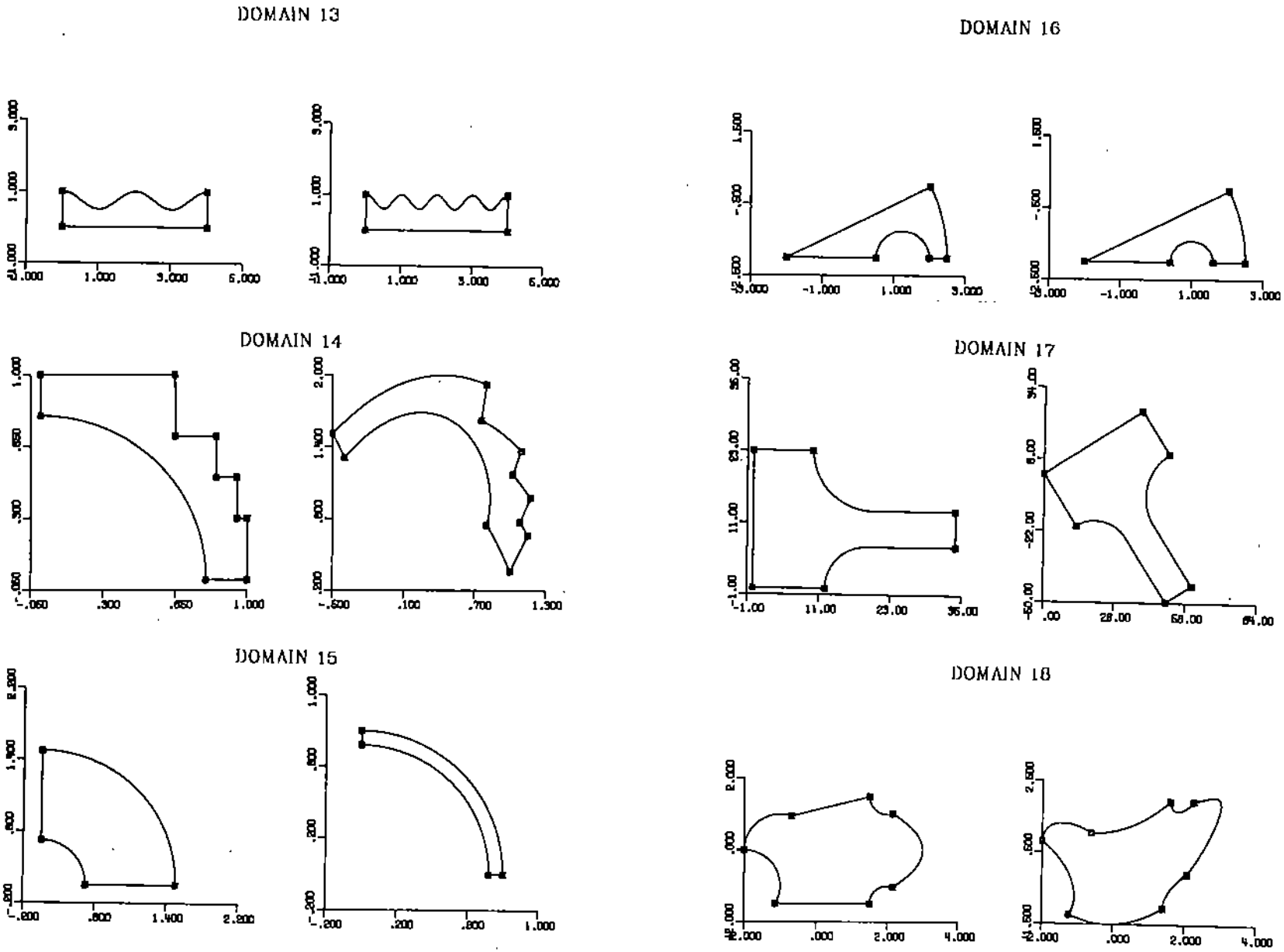
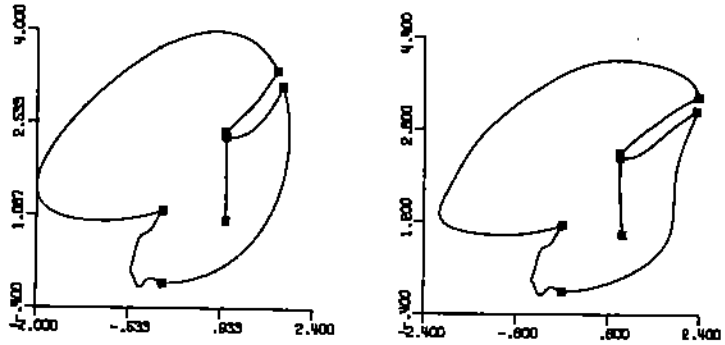


Figure 1. (continued)

DOMAIN 19



DOMAIN 20

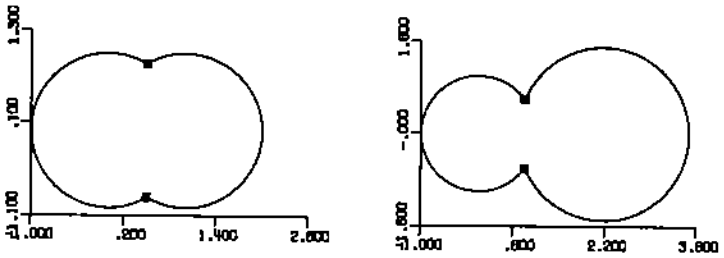


Figure 1. (continued)

```
      SUBROUTINE REGION(XGRID,YGRID,NGDIMX,NGDIMY,BRANGE,NPDIM,BCOORD,
     A                  SCLOCK,SARC,SLEVEL,
     B                  GTYPE,XBOUND,YBOUND,PIECE,BPTYPE,BNEIGH,
     C                  BGRID,BPARAM,NBDIM,NBPTS,FAIL)
C
C    DOMAIN PROCESSOR.  WRITTEN BY JOHN R. RICE, PURDUE UNIVERSITY.
C
C    THE PURPOSE OF THIS ALGORITHM IS TO RELATE A GENERAL TWO DIMENSIONAL
C    DOMAIN TO A RECTANGULAR GRID LAID OVER IT.  THE DOMAIN BOUNDARY IS
C    GIVEN IN PARAMETRIC FORM AND THE PROGRAM PRODUCES ARRAYS WHICH
C    DESCRIBE THE RELATIONSHIP BETWEEN THE TWO OBJECTS.  THESE ARRAYS
C    CONTAIN INFORMATION POINTING IN BOTH DIRECTIONS AND WHICH SHOULD
C    FACILITATE APPLICATIONS( E.G. NUMERICAL QUADRATURE, SURFACE
C    FITTING, DISCRETIZING PDE=S ) WHICH INVOLVE SUCH DOMAINS.
C
C
C ------------------- INPUT INFORMATION FOR DOMAIN PROCESSOR --------
C
C 1** OUTPUT LEVEL CONTROL *******
C         SLEVEL = LEVEL = OUTPUT CONTROL SETTING
C                          DETAILS GIVEN IN SUBROUTINE DOMAIN
C
C 2** GRID DEFINITION ************
C
C         NGDIMX,NGDIMY  = NUMBER OF GRID LINES IN X AND Y COORDINATES
C                          NGRIDX = NGDIMX, NGRIDY = NGDIMY
C         XGRID(IX),YGRID(JY)  FOR  IX = 1 TO NGRIDX, JY = 1 TO NGRIDY
C                        = ARRAYS CONTAINING THE GRID LINE COORDINATES
C
C 3** BOUNDARY DEFINITION ********
C
C         NPDIM  = NBOUND = NUMBER OF BOUNDARY PIECES
C         NBDIM  = ARRAY DIMENSION FOR BOUNDARY POINTS
C         BCOORD = PARAMETERIZED DEFINITION OF THE BOUNDARY.
C                  BCOORD(P,X,Y,IPIECE) GIVES THE X,Y VALUES OF THE
C                  POINT ON PIECE IPIECE WITH PARAMETER VALUE = P.
C                  SEE NOTE BELOW FOR DISCRETE BOUNDARY DEFINITION
C
C         BRANGE(2,I) = FIRST AND LAST VALUES OF PARAMETERS DEFINING
C                       THE I-TH BOUNDARY PIECE
C         SCLOCK = CLOCK = SWITCH TO SPECIFY BOUNDARY ORIENTATION
C                  .TRUE.  MEANS BOUNDARY IS CLOCKWISE
C                  .FALSE. MEANS BOUNDARY IS COUNTER-CLOCKWISE
C         SARC   = ARC = .TRUE. MEANS DOMAIN IS AN ARC WITH NO INTERIOR
C
C ---------------------------------------------------------------------
C ---------  OUTPUT INFORMATION OF THE DOMAIN PROCESSOR IS IN TWO PARTS
C             ( THERE IS ALSO A FAILURE FLAG = FAIL )
C
C PART 1 ********** GRID POINT TYPES ***********
C
C   GTYPE(IX,JY) FOR  IX = 1 TO NGRIDX, JY = 1 TO NGRIDY
C     THE VALUES IN THIS ARRAY GIVE THE TYPE OF THE GRID POINTS AND
C     INFORMATION ABOUT THEIR RELATION TO THE BOUNDARY. DETAILS AND
C     EXAMPLES ARE GIVEN IN THE SUBROUTINE DOMAIN.
C     POSSIBLE VALUE ARE( ASSUMING PACKING FACTOR IPACKB = 1000)
C
C   = INTEGER OVER 1000
C     GRID POINT IS NEXT TO THE BOUNDARY AND THE GTYPE VALUE IS
C     GTYPE = K + 1000*J WHICH ENCODES THE LOWEST NUMBERED BOUNDARY
C     NEIGHBOR (K) AND POINTER (J) TO NEIGHBORING BOUNDARY POINTS.
C     SEE DOMAIN COMMENTS FOR DETAILS.
C
C   = 999
C     GRID POINT IS INTERIOR TO REGION AND NOT CLOSE TO BOUNDARY
C
C   = INTEGER LESS THAN 999
C     GRID POINT IS ALSO BOUNDARY PT., GTYPE IS ITS INDEX
C
C   = 0
C     GRID POINT IS EXTERIOR AND FAR FROM THE BOUNDARY
C
```

```
C     = NEGATIVE INTEGER
C       GRID POINT IS EXTERIOR NEXT TO THE BOUNDARY, ITS LOCATION
C       RELATIVE TO THE BOUNDARY IS ENCODED AS FOR INTERIOR POINTS
C       NEAR THE BOUNDARY
C
C
C  PART 2 ********* BOUNDARY/GRID INTERSECTIONS **********
C
C         NBNDPT = NUMBER OF BOUNDARY POINTS ACTUALLY FOUND
C                  BOUNDARY POINT INDEX RANGE = 1 TO NBNDPT+1
C                  THE FIRST BOUNDARY POINT IS ADDED TO THE LIST TO
C                  MAKE IT CIRCULAR, HENCE THE +1 IN THE INDEX RANGE
C         XBOUND(I),YBOUND(I) = COORDINATES OF I-TH BOUNDARY POINT
C         BPARAM(I) = PARAMETER VALUE OF I-TH BOUNDARY POINT
C         PIECE(I)  = INDEX OF BOUNDARY PIECE TO WHICH PT. BELONGS
C                        SMALLEST NUMBER FOR CORNER POINTS
C         BPTYPE(I) = TYPE OF BOUNDARY POINT
C                   = HORZ  IF POINT IN ON A Y GRID LINE
C                   = VERT                 X
C                   = BOTH  IF POINT IS ALSO A GRID POINT
C                   = INTE  IF POINT IS NOT ON A GRID LINE
C                           HAPPENS ONLY FOR CORNERS OF THE BOUNDARY
C                   = JUMP  IF POINT PRECEEDS A HOLE
C         BNEIGH(I) = POINTER TO THE INTERIOR POINTS FROM THE I-TH
C                     BOUNDARY POINT. SAME SCHEME IS USED TO ENCODE
C                     DIRECTIONS AS FOR THE J PART OF GTYPE ABOVE
C         BGRID(I) = IX + IPACKB*JY WHEN PT. I IS IN GRID SQUARE IX,JY
C
C  ************************************************************************
C  **************  DOMAIN PROCESSOR ALGORITHM STRUCTURE  ******************
C
C     THERE ARE TWO BASIC PARTS TO THE ALGORITHM, PROCESSING THE BOUNDARY
C     AND TYPING THE GRID POINTS. AN OUTLINE FOLLOWS
C
C    MARK ALL GRID POINTS AS EXTERIOR
C
C********   PART 1 *** PROCESS BOUNDARY
C
C    LOCATE FIRST BOUNDARY POINT
C
C    DO LOOP OVER BOUNDARY PIECES
C     1 DO WHILE NOT AT END OF PIECE
C     1   1 LOCATE NEXT INTERSECTION WITH GRID LINE( SUBROUTINE BWALK )
C     1   1 DETERMINE TYPE FOR INTERSECTION POINT FOUND
C     1   1 CHECK FOR CHANGING BOUNDARY PIECE(SUBROUTINE CHANGE DOES IT)
C     1   1 CHECK CONTINUITY OF BOUNDARY
C     1   +---
C     +------
C
C    CHECK CLOSING OF BOUNDARY
C    FINISH UP FIRST POINT
C
C******** PART 2 *** TYPE GRID POINTS AND MARK INTERIOR POINTS
C
C    PASS OVER GRID MARKING THE GRID POINT TYPES(SUBROUTINE GVALUS )
C    MARK INTERIOR POINTS( SUBROUTINE FILL )
C          LOCATE FIRST INTERIOR POINT
C          MARK THE INTERIOR POINTS
C    SET POINTERS FROM BOUNDARY TO INTERIOR( SUBROUTINE NEIGH )
C
C
C   **** NOTE. IF THE BOUNDARY IS DESCRIBED DISCRETELY BY THE SET OF
C   BOUNDARY-GRID INTERSECTIONS, THEN THE FOLLOWING FUNCTION PROGRAM
C   ALLOWS THE DOMAIN PROCESSOR TO BE APPLIED.  IT ESSENTIALLY PROVIDES
C   LINEAR INTERPOLATION BETWEEN THE DISCRETE POINTS
C
C    *     FUNCTION BCOORD(P,X,Y,IPIECE)
C    *     REAL XBOUND(***),YBOUND(***)
C    *     LOGICAL STARTD
C    *     DATA STARTD/.FALSE./
C    *     IF( STARTD )              GO TO 20
C    *           OBTAIN DATA FOR DISCRETE POINTS THROUGH A COMMON BLOCK
```

```
C     *          OR BY READING FROM A FILE.  THE READ CASE IS SHOWN.
C     *        READ(INFILE,10) NBNDPT,(XBOUND(I),YBOUND(I),I=1,NBNDPT-1)
C     *   10   FORMAT( ****** )
C     *        XBOUND(NBNDPT) = XBOUND(1)
C     *        YBOUND(NBNDPT) = YBOUND(1)
C     *
C     *           OBTAIN COORDINATES FROM ARRAYS OF DISCRETE DATA
C     *   20 IP = P
C     *      X = (P-IP)*XBOUND(IP+1) + (1.+P-IP)*XBOUND(IP)
C     *      Y = (P-IP)*YBOUND(IP+1) + (1.+P-IP)*YBOUND(IP)
C     *      RETURN
C     *      END
C
C  ***********************************************************************
C  ***** REGION IS AN INTERFACE SUBROUTINE FOR THE DOMAIN PROCESSING
C           MAIN SUBPROGRAM = DOMAIN.
C     REGION MUST BE THE FIRST ROUTINE CALLED FOR PROCESSING
C           MULTIPLE REGIONS( E.G. WITH HOLES)
C     THE FUNCTIONS OF THIS INTERFACE PROGRAM ARE AS FOLLOWS.
C
C     1.  INITIALIZE OR COMPUTE VARIOUS CONSTANTS AND PLACE THEM IN
C         COMMON BLOCKS.  THESE ARE
C         A. I/O CONSTANT = MOUTPT
C         B. NUMERICAL CONTROL CONSTANTS = EPSGRD,EPSTAN
C         C. 5 CHARACTER STRING CONSTANTS
C         D. COUNTERS = N1BND,N1BDPT = 1
C         E. INTERIOR MARKER = NSIDE
C         F. PACKING FACTOR = IPACKB( SHOULD BE NSIDE+1 )
C
C     2.  CHECK THAT THE GRID IS PROPERLY DEFINED.
C
C     3.  MOVE VARIABLES FROM SUBROUTINE ARGUMENT LIST INTO COMMON.
C         A. DIMENSION SPECIFICATIONS = NGDIMX,NGDIMY,NBOUND
C         B. CONTROL VARIABLES = LEVEL, CLOCKW, ARC
C
C     4.  CALL DOMAIN.
C
C     5.  SET FAILURE FLAG FAIL, FINAL NUMBER NBPTS OF BOUNDARY POINTS
C
C
      REAL XGRID(NGDIMX),YGRID(NGDIMY),BRANGE(2,NPDIM)
      LOGICAL SCLOCK, SARC, FAIL
      INTEGER SLEVEL
      DIMENSION GTYPE(NGDIMX, NGDIMY),XBOUND(NBDIM),YBOUND(NBDIM),
     A     BPTYPE(NBDIM),BNEIGH(NBDIM),BGRID(NBDIM),BPARAM(NBDIM),
     B     PIECE(NBDIM)
      EXTERNAL BCOORD
C
C     ******** IF HOLE IS ALSO USED THEN THE CALLING PROGRAM  *****
C     ******** MUST HAVE THE FOLLOWING COMMON BLOCKS IN ORDER *****
C     ******** TO CONFORM TO FORTRAN STANDARDS                *****
C
      INTEGER N1BND, N1BDPT, NSIDE, IPACKB, MOUTPT,
     A        NGRIDX, NGRIDY, LEVEL, QLIMIT,QX(250),QY(250),
     B        GTYPE, PIECE, BGRID, BNEIGH
      REAL             PARAM, EPSGRD, EPSTAN
      LOGICAL CLOCKW, ARC, FATAL, INHOLE
      INTEGER TYPE, BPTYPE, HORZ, VERT, BOTH, INTER, JUMP
      COMMON / DMCINT / N1BND, N1BDPT, NSIDE, IPACKB, MOUTPT,
     A                  NGRIDX, NGRIDY, LEVEL, QLIMIT, QX, QY
      COMMON / DMCREL / PARAM, EPSGRD, EPSTAN
      COMMON / BNDRYI / IPIECE, NBOUND, NBNDPT
      COMMON / BNDRYL / CLOCKW, ARC, FATAL, INHOLE
      COMMON / DMCHAR / TYPE, HORZ, VERT, BOTH, INTER, JUMP
C
C  CONSTANT CHARACTER STRINGS
C
      DATA IQHORZ, IQVERT, IQBOTH, IQINTE, IQJUMP
     A  /  4HHORZ, 4HVERT, 4HBOTH, 4HINTE, 4HJUMP /
C
C  INITIALIZE COMMON BLOCKS FROM SUBROUTINE ARGUMENTS
      NGRIDX = NGDIMX
```

```
      NGRIDY = NGDIMY
      NBOUND = NPDIM
      LEVEL  = SLEVEL
      CLOCKW = SCLOCK
      ARC    = SARC
C
C  INITIALIZE CONSTANT CHARACTER STRINGS FROM DATA STATEMENTS
      HORZ   = IQHORZ
      VERT   = IQVERT
      BOTH   = IQBOTH
      INTER  = IQINTE
      JUMP   = IQJUMP
C
C                 SET UNIT NUMBER FOR OUTPUT
      MOUTPT = 6
C                 SET MARKER FOR INTERIOR POINTS OF REGION
C                 MUST EXCEED MAX NUMBER OF BOUNDARY POINTS
      NSIDE = 999
      IPACKB = NSIDE + 1
C                 SET LIMIT ON THE QUEUE LENGTH IN ROUTINE FILL
C                 QLIMIT = 250 SHOULD ALLOW WELL OVER 10,000
C                 INTERIOR POINTS IN THE DOMAIN
      QLIMIT = 250
C                 MARK THAT WE ARE NOT IN HOLE
      INHOLE = .FALSE.
C                 INITIALIZE BOUNDARY LIST COUNTERS
      N1BND  = 1
      N1BDPT = 1
C
C                SET GEOMETRY TOLERANCE PARAMETER EPSGRD
C     ***************************************************************
C     ******                                                  ******
C     ******  THIS IS A MACHINE AND PROBLEM DEPENDENT CONSTANT  ******
C     ******                                                  ******
C     ***************************************************************
C
C     EPSGRD SHOULD BE LARGE ENOUGH TO INSULATE THE COMPUTATIONS FROM
C     MACHINE ROUND-OFF.  ALL POINTS AND LINES WITHIN EPSGRD OF ONE
C     ANOTHER ARE ASSUMED TO BE EQUAL.  IT IS PROBABLY SAFE TO TAKE
C     EPSGRD AS 50 UNITS IN THE LAST PLACE, IT MUST BE AT LEAST 20
C     UNITS IN THE LAST PLACE. THE CONVERGENCE TEST IN SECANT IS
C     .2*EPSGRD
C
C     ******  THIS PARAMETER IS RELATIVE TO THE DOMAIN SIZE  ******
C
C     EPSGRD SHOULD BE SMALL ENOUGH SO THAT THE ACCURACY IN THE PROBLEM
C     SOLUTION IS NOT AFFECTED BY AN UNCERTAINTY IN THE GEOMETRY OF
C     EPSGRD.
C
C        EPSGRD = 1.E-8 IS APPROPRIATE FOR MOST LONG WORD LENGTH
C                       MACHINES AND PROBLEMS
C                 2.E-5 IS APPROPRIATE( BUT NOT FAIL-SAFE ) FOR
C                       32 BIT MACHINES.
      XWIDTH = XGRID(NGRIDX) - XGRID(1)
      YWIDTH = YGRID(NGRIDY) - YGRID(1)
      EPSGRD = 2.E-5*AMAX1(XWIDTH,YWIDTH)
C
C         CHECK THAT THE GRIDS ARE PROPERLY DEFINED
      IF( NGRIDX .GE. 2 )                                  GO TO 4
C               FATAL ERROR, GRID TOO SMALL
    2     IF( LEVEL .GE. 0 ) WRITE(MOUTPT,9002)
 9002     FORMAT(5(3H **),38HFATAL ERROR, MUST HAVE AT LEAST 2 GRID
     A           ,17H LINES IN X AND Y)
          FAIL = .TRUE.
          RETURN
    4 IF( NGRIDY .LE. 1 )                                  GO TO 2
C
C          CHECK THAT THE GRID INCREASES IN BOTH DIRECTIONS
C          FIND THE MINIMUM GRID WIDTHS
      XGMIN = XWIDTH
      DO 10 I = 2,NGRIDX
   10     XGMIN = AMIN1(XGMIN,XGRID(I)-XGRID(I-1))
```

```
      YGMIN = YWIDTH
      DO 12 I = 2,NGRIDY
   12    YGMIN = AMIN1(YGMIN,YGRID(I)-YGRID(I-1))
      IF( AMIN1(XGMIN,YGMIN) .GT. EPSGRD )          GO TO 20
C
C         FATAL ERROR, HAVE ZERO OR NEGATIVE GRID WIDTH
          FAIL = .TRUE.
          IF( LEVEL .GE. 0 ) WRITE(MOUTPT,9015)
 9015     FORMAT(//5(3H **),36HFATAL ERROR, X AND Y GRID LINES MUST,
     A                      14H BE INCREASING )
          RETURN
C
C           SET PARAMETER FOR TANGENCY TEST
C           SEE SUBROUTINE CHKTAN FOR DETAILS
   20 EPSTAN = .1E0*AMIN1(XGMIN,YGMIN,15.E0*SQRT(EPSGRD))
C               PRINT INITIAL HEADER
C
      IF (LEVEL .GT. 0) WRITE(MOUTPT,9000)
 9000 FORMAT(///1H , 19(1H-)
     A     / 1H , 18H DOMAIN PROCESSOR
     B     / 1H , 19(1H-)
     C     //5X,33H D O M A I N   P R O C E S S O R )
C
C  CALL DOMAIN PROCESSOR
      CALL DOMAIN(XGRID,YGRID,NGDIMX,NGDIMY,BRANGE,NPDIM,BCOORD,
     A            GTYPE,XBOUND,YBOUND,PIECE,BPTYPE,BNEIGH,
     B            BGRID,BPARAM,NBDIM)
C
C           SET FAILURE FLAG AND FINAL NUMBER OF BOUNDARY POINTS
      FAIL  = FATAL
C           THESE NUMBERS WILL BE USED BY HOLE IF IT IS CALLED
      NBPTS = NBNDPT
      N1BND = NBOUND+1
C
      RETURN
      END

      SUBROUTINE DOMAIN(XGRID,YGRID,NGDIMX,NGDIMY,BRANGE,NPDIM,BCOORD,
     A                  GTYPE,XBOUND,YBOUND,PIECE,BPTYPE,BNEIGH,
     B                  BGRID,BPARAM,NBDIM)
C
C  ***** THIS SUBROUTINE PROCESSES THE RECTANGULAR GRID AND SPECIFIED
C  BOUNDARY. IT APPLIES TO ONE CLOSED LOOP OR ARC OF THE BOUNDARY AND
C  MAY BE CALLED SEVERAL TIMES FOR A COMPLEX DOMAIN
C
C  ------------------- INPUT INFORMATION FOR DOMAIN -----------------
C     THE INPUT INFORMATION IS IN THE COMMON BLOCKS AND ARGUMENTS
C               SEE THE MAIN DRIVER REGION FOR MORE DETAILS
C
C 1** OUTPUT LEVEL CONTROL ********
C         LEVEL           = CONTROL SETTING, DETAILS GIVEN BELOW
C
C 2** GRID DEFINITION ************
C
C       NGDIMX,NGDIMY = NGRIDX,NGRIDY  = GRID LINES IN X AND Y COORDINATES
C       XGRID(IX),YGRID(JY)  FOR  IX = 1 TO NGRIDX, JY = 1 TO NGRIDY
C
C 3** BOUNDARY DEFINITION ********
C
C         N1BND  = NUMBER OF THE FIRST BOUNDARY PIECE
C                  WILL DIFFER FROM 1 AFTER FIRST CALL OF DOMAIN
C         NPDIM  = ARRAY DIMENSION FOR BOUNDARY PIECES
C         N1BDPT = NUMBER OF THE FIRST BOUNDARY POINT
C                  WILL DIFFER FROM 1 AFTER FIRST CALL OF DOMAIN
C         NBDIM  = ARRAY DIMENSION FOR BOUNDARY POINTS
C         BCOORD = PARAMETERIZED DEFINITION OF THE BOUNDARY.
C                  BCOORD(P,X,Y,IPIECE) GIVES THE X,Y VALUES OF THE
C                  POINT ON PIECE IPIECE WITH PARAMETER VALUE = P.
C         BRANGE(2,I) = FIRST AND LAST VALUES OF PARAMETERS DEFINING
C                       THE I-TH BOUNDARY PIECE
C         CLOCKW = SWITCH TO SPECIFY BOUNDARY ORIENTATION
C                  .TRUE. MEANS BOUNDARY IS CLOCKWISE
```

```
C                .FALSE.MEANS BOUNDARY IS COUNTER-CLOCKWISE
C        ARC    = .TRUE. MEANS DOMAIN IS AN ARC WITH NO INTERIOR
C
C  ----------      THE OUTPUT OF DOMAIN IS IN TWO PARTS ----------
C
C  1** GRID SPECIFICATION **********
C
C    GTYPE(IX,JY) FOR  IX = 1 TO NGRIDX, JY = 1 TO NGRIDY
C      THE VALUES IN THIS ARRAY GIVE THE TYPE OF THE GRID POINTS AND
C      INFORMATION ABOUT THEIR RELATION TO THE BOUNDARY.
C      THERE IS A PACKING FACTOR IPACKB WHICH IS NORMALLY 1000. FOR
C      VERY LARGE PROBLEMS, IPACKB AND RELATED CONSTANT NSIDE =
C      IPACKB - 1 MUST BE INCREASED SO THAT NSIDE .GT. NBNDPT.
C      POSSIBLE VALUES OF GTYPE ARE( ASSUMING IPACKB = 1000)
C
C    = INTEGER OVER 1000
C      GRID POINT IS NEXT TO THE BOUNDARY AND THE GTYPE VALUE IS
C      GTYPE = K + 1000*J   WHERE
C            K IS THE INDEX OF THE LOWEST NUMBERED BOUNDARY NEIGHBOR
C                MUST DOUBLE CHECK USE WHEN K = 1
C            J = FOUR BITS TO NOTE LOCATION OF BOUNDARY POINTS
C                 0001 - BOUNDARY NEIGHBOR TO NORTH  (NOON)
C                 0010 - BOUNDARY NEIGHBOR TO EAST   (3 O-CLOCK)
C                 0100 - BOUNDARY NEIGHBOR TO SOUTH  (6 O-CLOCK)
C                 1000 - BOUNDARY NEIGHBOR TO WEST   (9 O-CLOCK)
C              THUS J=9 (1001 IN BINARY) IMPLIES THAT THERE ARE BOUNDARY
C              NEIGHBORS TO THE NORTH AND WEST
C           EXAMPLES ( X = BNDRY PT., 0 = GRID PT )
C
C                          X          X       X  0   X
C
C                       X  0          0  X       X
C                         J=9        J=3        J=14
C
C           ***** NOTE THAT GTYPE IS INITIALLY SET NEG. AND THEN MADE
C                 POSITIVE WHEN THE INTERIOR IS FILLED
C
C    = 999
C      MEANS GRID POINT IS INTERIOR TO THE REGION AND NOT CLOSE
C      TO THE BOUNDARY.
C
C    = INTEGER LESS THAN 1000
C      GRID POINT IS ALSO BOUNDARY POINT, GTYPE IS ITS INDEX
C
C    = 0
C      GRID POINT IS EXTERIOR AND FAR FROM THE BOUNDARY
C
C    = NEGATIVE INTEGER
C      GRID POINT IS EXTERIOR NEXT TO THE BOUNDARY, ITS LOCATION
C      RELATIVE TO THE BOUNDARY IS ENCODED AS FOR INTERIOR POINTS
C      NEAR THE BOUNDARY
C
C  2** BOUNDARY SPECIFICATION **********
C
C         NBNDPT = NUMBER OF BOUNDARY POINTS ACTUALLY FOUND
C         XBOUND(I),YBOUND(I) = COORDINATES OF I-TH BOUNDARY POINT
C         BPARAM(I) = PARAMETER VALUE OF I-TH BOUNDARY POINT
C         PIECE(I)  = INDEX OF BOUNDARY PIECE TO WHICH PT. BELONGS
C                        SMALLEST NUMBER FOR CORNER POINTS
C         BPTYPE(I) = TYPE OF BOUNDARY POINT
C                   = HORZ,VERT,BOTH,INTE OR JUMP
C         BNEIGH(I) = POINTER TO THE INTERIOR POINTS FROM THE I-TH
C                     BOUNDARY POINT. SAME SCHEME IS USED TO ENCODE
C                     DIRECTIONS AS FOR THE J PART OF GTYPE ABOVE
C         BGRID(I) = IX + IPACKB*JY WHEN PT. I IS IN GRID SQUARE IX,JY
C
C         MAXIMUM NUMBER OF BOUNDARY POINTS = NBDIM
C         THIS IS ESTIMATED BY THE CALLING PROGRAM FOR DIMENSIONING
C         THE VARIOUS ARRAYS.
C
C **********    DOMAIN PROCESSOR SUBPROGRAMS    **********************
C
```

```
C     REGION     - MAIN DRIVER AND USER INTERFACE
C     HOLE       - ALTERNATE DRIVER, INSERTS HOLES IN FIRST DOMAIN
C        REMOVH  - REMOVES HOLE FROM GRID TYPES, UPDATES ALL INFO
C     DOMAIN     - MAIN PROGRAM TO PROCESS A DOMAIN
C     BWALK      - WALK ALONG BOUNDARY TO FIND GRID INTERSECTION
C        CHKTAN  - CHECK FOR BOUNDARY TANGENT TO A GRID LINE
C        CROSS2  - CHECK FOR DOUBLE CROSSING OF A GRID LINE
C        DBACK   - FOLLOW BOUNDARY FOR A DOUBLE CROSSING OF A GRID LINE
C        REGULA  - MODIFIED REGULA FALSI METHOD
C        SECANT  - SECANT METHOD
C     CHANGE     - MAKE CHANGE OF BOUNDARY PIECE
C     FILL       - LOCATE AND FILL INTERIOR OF THE DOMAIN
C        EXPAND  - EXPAND INTERIOR OF DOMAIN BY 1 POINT
C     GVALUS     - SET TYPES OF ALL GRID POINTS
C        ISETGT  - COMPUTE GTYPE VALUE FOR A GRID POINT, SET IT
C     INSIDE     - UTILITY: CHECKS POINT INSIDE SPECIFIED SUBGRID
C     LOCATE     - UTILITY: LOCATE POINT IN GRID, TYPE IT
C     NEIGH      - COMPUTE POINTERS FROM BOUNDARY TO GRID POINTS
C     TABLGT     - TABLE THE GTYPE VALUES FOR THE GRID
C
C ***************************************************************
C ************  OUTPUT CONTROL  *********************************
C
C    LEVEL 0  ***** FATAL ERROR MESSAGES ************************
C
C      BWALK  - BOUNDARY GOES OUTSIDE DOMAIN
C             - UNABLE TO FIND GRID WHERE BOUNDARY GOES
C             - UNABLE TO FIND GRID INTERSECTION WITH BOUNDARY
C      CHANGE - BOUNDARY PIECES DO NOT JOIN UP
C      DOMAIN - BOUNDARY PARAMETER NOT INCREASING
C             - OVERFLOW OF STORAGE ALLOCATED FOR BOUNDARY POINTS
C                 MUST CHANGE DECLARATIONS IN PROGRAM CALLING REGION
C             - ABNORMAL EXIT FROM BOUNDARY PROCESSING
C                 PROBABLY CANNOT HAPPEN
C             - BOUNDARY DOES NOT CLOSE
C      FILL   - FAILURE TO FIND INTERIOR OF DOMAIN
C             - OVERFLOW IN QUEUE FOR FILLING INTERIOR OF DOMAIN
C                 MUST INCREASE QLIMIT AND RECOMPILE SUBROUTINE FILL
C      GVALUS - HAVE ILLEGAL POINT TYPE. THINGS ARE REALLY MESSED UP
C      LOCATE - ASKED TO FIND POINT OUTSIDE DOMAIN
C                 INPUT ERROR OR PROGRAM BUG
C      REMOVH - HOLE IS TOO CLOSE TO BOUNDARY OF CONTAINING DOMAIN
C                 NEED TO USE FINER GRID
C      REGION - GRID LINES ARE NOT INCREASING, INPUT ERROR
C             - LESS THAN 2 GRID LINES IN X OR Y DIRECTION, INPUT ERROR
C
C   LEVEL 1   *****  MINIMAL MESSAGES *******************************
C
C      REGION - NEW PAGE, DOMAIN PROCESSING STARTS
C      HOLE   - NEW PAGE, HOLE PROCESSING STARTS
C      DOMAIN - NUMBER OF BOUNDARY/GRID INTERSECTIONS
C             - FATAL ERROR NOTE( IF PRESENT )
C
C   LEVEL 2   ***** SUMMARY TRACE ***********************************
C
C      BWALK  - PIECE ENDS IN INTERIOR OF GRID SQUARE
C      CHANGE - CHANGE OF BOUNDARY PIECE
C      CHKTAN - POINT REPLACEMENT DUE TO TANGENCY TO GRID LINE
C      DOMAIN - INITIAL BOUNDARY POINT
C             - BOUNDARY POINT FOUND
C             - TABLE OF GRID POINT TYPES( FROM TABLGT )
C             - SUMMARY OUTPUT OF BOUNDARY POINT ARRAYS
C      FILL   - WARNING, FIRST TRY TO LOCATE INTERIOR FAILS
C             - FAILURE INFORMATION ( IF NEEDED )
C      REMOVH - TABLE OF GRID POINT TYPES ( FROM TABLGT )
C
C   LEVEL 3   ***** SOME DETAILS OF THE PROGRAM OPERATION ***********
C
C      BWALK  - START
C             - WARNING, POINT FOUND IS NOT IN EXPECTED GRID
C             - BOUNDARY IS VERTICAL/HORIZONTAL
C             - FINISH
```

```
C       CHANGE - BOUNDARY PIECE CHANGE DATA
C       CHKTAN - DATA FOR THE TANGENCY TEST( WHEN IT SUCCEEDS )
C       CROSS2 - SIGNS FOR DOUBLE CROSSING CHECK
C              - SUCCESSFUL RESULTS
C       DOMAIN - BOUNDARY PIECE CHANGE DATA
C              - HEADING FOR TYPING OF GRID POINTS
C       FILL   - DATA ON SEARCH FOR INTERIOR POINTS
C       NEIGH  - SUMMARY OF EXECUTION
C
C    LEVEL 4   ***** MORE DETAILS ABOUT OPERATION **********************
C
C       BWALK  - EXPANSION OF PARAMETER STEP
C       DBACK  - SUMMARY OF EXECUTION
C       EXPAND - DETAILS OF IDENTIFICATION OF THE INTERIOR POINTS
C       GVALUS - DATA USED FOR TYPING GRID POINTS
C       LOCATE - DATA FOR POINTS LOCATED
C       NEIGH  - DATA FOR NEIGHBORS LOCATED
C       REGULA - SUMMARY OF EXECUTION
C       SECANT - START INFORMATION
C              - FINISH INFORMATION
C
C    LEVEL 5   ***** DETAILS OF MANY LOOPS ,ETC. ***********************
C
C       BWALK  - DATA ON INITIAL ROUGH GUESSES AT INTERSECTIONS
C       DBACK  - LOOP DETAILS
C       FILL   - DATA FOR LOOP TO FIND FIRST INTERIOR POINT
C              - DATA ON CLASSIFYING INTERIOR POINTS
C       GVALUS - DATA ON TYPING GRID POINTS
C       SECANT - BOUNDS USED ON VARIABLES
C              - DATA WHEN VARIABLE BOUNDS ARE USED
```