27
10/15/79
240 NTIS

MASTER

**TRIFEM 1: A Two-Dimensional Finite
Element Mesh Generator and Preprocessor
for Fluid Flow Problems
in Straight Channels**

C. Kleinstreuer
M. R. Patterson
J. L. Bledsoe

OAK RIDGE NATIONAL LABORATORY
OPERATED BY UNION CARBIDE CORPORATION · FOR THE DEPARTMENT OF ENERGY

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
Price: Printed Copy \$5.25; Microfiche \$3.00

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, contractors, subcontractors, or their employees, makes any warranty, express or implied, nor assumes any legal liability or responsibility for any third party's use or the results of such use of any information, apparatus, product or process disclosed in this report, nor represents that its use by such third party would not infringe privately owned rights.

Contract No. W-7405 eng 26

ENERGY DIVISION

TRIFEM 1: A TWO-DIMENSIONAL FINITE ELEMENT MESH
GENERATOR AND PREPROCESSOR FOR FLUID
FLOW PROBLEMS IN STRAIGHT CHANNELS

C. Kleinstreuer
Energy Division

M. R. Patterson
J. L. Bledsoe
Computer Sciences Division

Date Published - September 1979

NOTICE This document contains information of a preliminary nature.
It is subject to revision or correction and therefore does not represent a
final report.

OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37830
operated by
UNION CARBIDE CORPORATION
for the
DEPARTMENT OF ENERGY

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

**THIS PAGE
WAS INTENTIONALLY
LEFT BLANK**

TABLE OF CONTENTS

	PAGE
ACKNOWLEDGMENTS.	v
ABSTRACT	1
1. INTRODUCTION.	3
1.1 Literature Review	4
2. TRIFEM 1 - A Mesh Generator for Simple Flow Regimes . . .	7
2.1 MAIN.	7
2.2 CØØRD	9
2.3 NØDES	10
2.4 ERRØR	10
2.5 PTRIL	10
2.6 CRØSS	10
2.7 KØUPLE.	11
2.8 BAND.	11
2.9 QUADEL.	11
2.10 SØRTER.	12
3. A MESH GENERATION EXAMPLE	13
4. INPUT DATA FOR TRIFEM	23
4.1 Input Data for the 2-d Mesh	26
5. RECOMMENDATIONS	27
REFERENCES	29
APPENDIX A. OUTPUT LISTING FROM TRIFEM 1 RUN	31
APPENDIX B. PROGRAM LISTING OF TRIFEM 1	47

**THIS PAGE
WAS INTENTIONALLY
LEFT BLANK**

ACKNOWLEDGMENTS

Development of this code has progressed with the assistance of persons other than the authors. Discussions with Mark Reeves and Mary Ballou provided much insight into the desired capabilities of mesh generators for finite element codes. Jonas T. Holdeman contributed substantial ideas about programming logic for triangular meshes.

TRIFEM 1: A TWO-DIMENSIONAL FINITE ELEMENT MESH
GENERATOR AND PREPROCESSOR FOR FLUID
FLOW PROBLEMS IN STRAIGHT CHANNELS

C. Kleinstreuer, M. R. Patterson, and J. L. Bledsoe

ABSTRACT

This report describes a graphics mesh generator and preprocessor system that can be used as a preprocessor for two-dimensional flow problems. TRIFEM 1 provides a simple means of generating triangular finite element meshes of variable density for regions which can be conceptualized as rectangular and/or trapezoidal. A number of options for local mesh refinements, deleting and adding triangles in an interactive fashion, makes this program versatile without complicating its execution. The routines in this code have been implemented on the DECsystem PDP-10 computer with graphic display on the Tektronix 4014 or 4006 terminals. Much of the plotting in these codes is accomplished through the use of DISSPLA software. The TRIFEM 1 routines are written in standard FORTRAN.

**THIS PAGE
WAS INTENTIONALLY
LEFT BLANK**

1. INTRODUCTION

A characteristic feature of the finite element method is the discretization of a given flow domain into multiple subregions and subsequent assemblage of the individual contributions for each element.

If more than 50 elements appear to be necessary for approximating the flow region, preparation of the input data file becomes cumbersome and is, in general, prone to errors. Hence, for moderate-to-large domains, a semiautomatic mesh generator is desirable. Such an interactive program should execute the following tasks:

- drawing of solid boundaries (flow domain)
- designing initially coarse, rectangular grid
- local refinement options for rectangles
- forming triangles (boundary triangles utilize encoded line segments which approximate the shorelines, islands, etc.)
- numbering elements and nodal points
- local refinement option for triangles
- deleting and adding on triangles
- renumbering of all nodal points (vertices and midpoints) in order to minimize the maximum difference between assigned nodal point numbers considering all finite elements.

From this point on, the finite element mesh generator functions as a preprocessor in providing the following set of input data for the source program:

- x- and y-coordinates of all nodal points
- identified boundary nodes on shorelines with zero velocities representing the conditions of "no-slip" and "no fluid" penetration through solid walls

- identified boundary nodes on kinematic boundaries (inlets and outlets) for prescribed principle variables (velocities and pressure)
- counterclockwise sequences of numbered nodes for each finite element
- number of elements, vertices, midpoints, degrees of freedom, as well as the maximum bandwidth

A mesh generator/preprocessor code with such a list of features is useful only when the following overall requirements can be met:

- low computer running time and core occupation
- easy manual preparations and flexibility for 1) designing the initial grid, 2) encoding the flow domain and 3) commanding the interactive mesh generating procedure
- satisfactory approximation of curved open and closed natural boundaries as well as irregularities in the flow domain
- smooth transition between subregions of different mesh density; smallest angle of any triangle should not be less than 20° ; i.e., equilateral triangles are most appropriate.
- error-free optimal generation of a specified input data file for the main program.

1.1 Literature Review

Existing two-dimensional finite element mesh generators and data preprocessors are almost exclusively designed for structural analysis problems, see (1) - (7). However, the main features of these mesh generators are also applicable for two-dimensional fluid flow problems.

Frederick et al. (5) presented a graphical/computer-based method for discretizing irregular, nonhomogeneous two-dimensional continua into triangular elements. It is a two-stage procedure where first all node points are manually plotted and recorded using a magnetic pen, a graph-table and a card punch. Secondly, a computer program combines

the nodal points forming almost equilateral triangles. Boundaries of the structure are approximated with line segments which serve as sides of boundary triangles. A special scheme using "ghost" points outside the structure is used to avoid the identification of external boundary nodes--just the opposite is desirable in fluid flow problems.

Fukada and Suhara (6) presented an algorithm where the external boundaries are discretized manually with piece-wise linear arcs which compose a simple connected region. Additional interior nodes are generated automatically based on a density distribution furnished by the user: the entire region is covered by simply connected polygonal zones which form subregions of specific mesh density to be prescribed by a zonal node density factor. In the second phase of the algorithm nearly equilateral triangles are produced.

Cavendish (3) and Cavendish and Wixom (2) modified the Fukada-Suhara code so that it is applicable to planar and shell type structures with material interfaces.

Kamel and McCable (7) presented a pre- and postprocessor, GIFTS 4, which is a collection of program modules (operating on a unified data base) that is able to generate nodal coordinates and freedom numbers, element connectivities and properties, as well as loads and boundary conditions. The program consists of the following main packages: 1) the element library containing elements ranging from simple quadrilaterals and triangles for linear interpolation functions to three-dimensional elements for high-order shape functions, 2) a bulk model generator, model editor and display program, 3) bandwidth optimizer, stiffness assembler, deflection and stress computation and result display, and 4) a vibration and transient response module.

It is apparent that GIFTS #4 can be utilized for common stress analysis and vibration problems upon specification of the structure, material properties, and boundary conditions involved. On the other hand, such a mammoth program loses a certain flexibility necessary for semiautomatic preprocessors suitable for problems in hydraulics and hydrology. Furthermore, the expertise, costs and time required to execute even a few modules for a particular problem are often prohibitive.

A number of routines described in the literature can be utilized in building a versatile finite element mesh generation/preprocessor for two-dimensional fluid flow problems see (8) - (13).

Bykat (12) presented an automatic reordering scheme which helps to reduce the bandwidth. Unfortunately, as with other such schemes, it is rather sensitive to the particular element of the discretized flow domain that is selected as a starting element.

Holdeman (11) developed a triangulation algorithm, PTRIL, which starts with two given nodal points that form a line segment. The code then searches for a third node closest to these that forms the largest angle with that line segment.

Carey (4) and Gartling and Becker (9) reported mesh refinement schemes for quadrilateral and triangular grids.

Durfee and Edwards (13) described an interactive semiautomated digitization system which allows the user to encode the geometry of arbitrary flow domains from maps, photographs, etc., on a data file.

2. TRIFEM 1 - A MESH GENERATOR FOR SIMPLE FLOW REGIMES

TRIFEM 1, an acronym for TRIangular Finite Element Mesh-generator is a computer code which essentially produces specific geometric input data files for any two-dimensional fluid flow problem. The program consists of a simple, economical code developed for flow regions with straight natural boundaries.

TRIFEM 1 is deliberately designed to function in an interactive fashion to give the user an opportunity to create a mesh according to specifications dictated by a particular flow problem. For example, local mesh refinements are possible even at a later stage if suggested by preliminary computer runs of the preprocessor and source program together. Preparations and input data requirements for the TRIFEM 1 codes are rather simple; core storage and computer running time are low. However, as indicated previously, the determination of subregions of higher mesh densities demand a fundamental understanding and anticipation of fluid flow patterns involved.

The general considerations discussed in Section 1, Introduction, were used as a guide in the development of the mesh generator/preprocessor TRIFEM 1. The program is written in FORTRAN IV and structured in a modular fashion. Descriptions of the routines in the program are given below.

2.1 MAIN

This routine allows for interactive input of mesh data, primary deletion of regions, and plotting the original mesh system. (See Fig. 2.1). This original mesh system is refined by either halving or quartering specified elements in the network.

Figure 2.1 Organization of Subroutines in TRIFEM 1

When refinement is complete, the nodes (vertices) for each element (not including midpoints) are used for triangulating the region. At this point, the user is allowed the choices of adding and/or deleting triangular elements from the network.

The final set of triangular elements is searched for the element that gives the overall minimum of the maximum difference between node numbers within the same element. This element is assigned to be the new starting element in the mesh. Thus, the numbering of the triangles is reordered.

Midpoints located between vertices of the reordered mesh are calculated. Thus, the "new" network contains a reordered set of vertex and midpoint nodes.

2.2 CØØRD

For each rectangular element in the original and altered meshes, this routine determines the corners X_{left} , X_{right} , Y_{bottom} , and Y_{top} . Since the order of the four corners is counterclockwise, the pairing of the X's and Y's is $(X_{\text{left}}, Y_{\text{bottom}})$, $(X_{\text{right}}, Y_{\text{bottom}})$, $(X_{\text{right}}, Y_{\text{top}})$, and $(X_{\text{left}}, Y_{\text{top}})$ (see Figure 2.2). The centroids are calculated for each

Figure 2.2 Example of a Rectangular Cell Drawn by TRIFEM

element and written into an output file. Centroids are used mainly for location of the element number on the plotted displays.

2.3 NØDES

Within this routine, all possible nodal points (vertices) are generated for all elements in the mesh. Points common to adjoining triangles are retained only once in the set of nodes.

2.4 ERRØR

This routine prints a message wherever it is referenced to indicate the need for user corrections.

2.5 PTRIL

This routine for triangulating a region searches for a starting segment among the nodes from MAIN. Upon finding this first segment, the code will look at the perimeter list for the newly found segment. The list of nodes will be examined for feasibility of forming a new triangle.

Function CRØSS will be accessed to test for overlapping triangles. Segments on the perimeter list will be deleted if these are also internal segments. Triangles not on the segment list will be added to it. Simultaneously, all triangles that are valid will be displayed as soon as these are found. The displaying consists of a drawing made by plotting software.

2.6 CRØSS

Function CRØSS's purpose is detecting the intersection of interior line segments. Given two points (X_1, Y_1) and (X_2, Y_2) , and two other

points (X_{p_1}, Y_{p_1}) and (X_{p_2}, Y_{p_2}) , CRØSS will detect if these two segments have a point or points in common.

2.7 KØUPLE

KØUPLE has a two-fold purpose--completing the mesh by determining the midpoint nodes and outputting the data file necessary for the two-dimensional finite element code, RIVINCA (14).

The optimized element mesh that comes from subroutine BAND is passed to KØUPLE. Nodes in this optimized mesh are renumbered in array NØDSIX, along with the calculated midpoints. Boundary nodes are stored in segment array ISEG and sorted into contiguous segments.

KØUPLE allows the user to define boundary values at those nodes stored in ISEG.

Finally, the data file for RIVINCA (14) is output. This file contains the number of triangles (NT), number of nodes (NNM), the number of boundary nodes (NSEG), X and Y locations of each node, prescribed velocity profiles and the array NØDSIX.

2.8 BAND

Subroutine BAND uses the Bykat algorithm (12) for determining the starting element necessary for the minimized maximum node difference in an element mesh. After finding this element, the node numbers in the mesh are reordered and triangle numbers are reassigned.

2.9 QUADEL

Rectangular cells in the original and refined element meshes are drawn by this routine. Plotting these cells requires the use of DISSPLA (15) software that draws vectors (straight lines), labels nodes, and elements.

2.10 SØRTER

This short routine sorts the element numbers for those cells that are being refined. The element numbers are accompanied by refinement operation code numbers, 1 (vertical halving), 2 (horizontal halving) and 4 (quartering). Arrays containing element numbers are sorted in ascending order and the operation codes are placed in the same order.

3. A MESH GENERATION EXAMPLE

Application of TRIFEM 1 as a mesh generator and preprocessor is demonstrated below. Planar regions consisting of a geometry which can be readily decomposed into rectangular and trapezoidal subdomains constitute the most likely cases. An example of a possible region is given in Figure 3.1.

The flowchart in Figure 3.2 gives the procedure for interactively creating a mesh.

In Figure 3.3, the input data have been used to draw an initial mesh with rectangular elements. TRIFEM 1 used the input (see Section 4.1) to determine the coordinates of each node in the mesh, both vertices and centroids. Also, note that two rectangular regions have been deleted to approximate the shape of an intake.

Refinement of larger cells near the intake and the intake itself is displayed in the next phase of generation, Figure 3.4. Cell number 7, the intake, was quartered, number 8 was halved vertically, and number 9 was halved horizontally and later, halved vertically. (These numbers refer to those cells in Figure 3.3.)

Completing the refinement stage (rectangular), the mesh system is now triangulated (see Figure 3.5). All elements and vertices are numbered. One new triangle is formed from the nodes 2, 16, and 9 (counterclockwise order). A land mass is approximated by deleting the triangles 40, 39 and 35. After additions and deletions have been made on the initial triangular mesh, the system is reordered by the Bykat algorithm. This "new" mesh is shown in Figure 3.6.

Figure 3.1 Straight River Stretch with Surface Intake Canal

Figure 3.2 Flow Diagram for Mesh Generation Using the Code TRIFEM-1 for Straight Channels

Figure 3.2 Cont.

Figure 3.3 Initial Coarse Grid for Straight River Stretch Canal

YELLOWSTONE

Figure 3.4 Refinement Options for the Rectangular Grid

YELLOWSTONE

ORNL-DWG 78-2514

Figure 3.5 Triangulation, Numbering of Elements and Vertices

YELLOWSTONE

Figure 3.6 Deleting Interior Triangles and Adding On Exterior Elements

Since the RIVINCA (14) code requires data input with six nodes per element, the midpoint of each line is used to represent a node. The set of new node numbers is recalculated and displayed in Figure 3.7.

YELLOWSTONE

Figure 3.7 Ordering Scheme for All Nodal Point Numbers, including Midpoints

4. INPUT DATA FOR TRIFEM 1

TRIFEM 1 reads data via a remote terminal and prompts the operator at the keyboard at the appropriate time to type specific data. The amount of data needed to generate the mesh has been reduced by requiring the operator only to define what types of operations are to be performed with the current set of elements in order to generate a new set of elements. For example, a rectangular element can be quartered to yield four new elements having a higher spatial resolution. Five new nodal points may be created by this one operation requiring the entry of only two integers on a part of the user.

Integral to this mesh generator is the fact that the current mesh is replotted on the scope of the terminal at the completion of each type of operation. This visual display of the nodes and elements provides an easy means of determining whether further refinement of the mesh is needed. To be most effective, the display should be shown instantaneously on the scope of the terminal rather than plotted on a pen-and-ink device at the central computer.

The input for TRIFEM 1 is simple enough that it can be done quite conveniently at the computer terminal used to execute the program. Because of the importance of visual display in presenting the results, the terminal should have graphics capability. In the following description of the input data, the word "ACCEPT" means "READ" from the terminal. The * symbol means that the input is free format, a very convenient mode in which to make the input.

MAIN

- 1) Input the one heading used for all plots

prompting message: 'INPUT HEADING'

ACCEPT 8003, (HEADER(I), I=1,9)

8003 FORMAT (16A5)

The nine storage words allotted to HEADER allow for reading as many as 45 characters for the heading.

- 2) Input the number of X, Y intervals.

Prompting Messages: 'BEGIN INPUTTING DATA',

'GIVE NO. XS AND YS'

ACCEPT*, NX, NY

- 3) Give the increasing X values.

Prompting messages: 'GIVE CUM NX + 1 values of X'

ACCEPT*, (X(I), I=1, NX1)

- 4) Give the increasing Y values.

Prompting messages: 'GIVE CUM NY + 1 values of Y'

ACCEPT*, (Y(I), I=1, NY1)

- 5) Give the number of rectangular regions to be deleted.

Prompting message: 'HOW MANY REGIONS WILL BE DELETED?'

ACCEPT*, NREG

- 6) Give the X, Y minimum and maximum of each region.

Prompting message: 'GIVE XMIN, XMAX, YMIN, YMAX FOR EACH REGION'

ACCEPT*, XMINR(N), XMAXR(N), YMINR(N), YMAXR(N)

- 7) Give the number of the cell and the code for the refinement operation.

Prompting message: 'ID ØF ELT TØ BE REFINED, ØPERATION 0 0 STØP'

Entering two separate zeros terminates refinement of the elements.

ACCEPT*, IDØUB, IØP

- 8) If instructions are complete, end with zero; otherwise, end with 1.

Prompting message: '0 IF COMPLETE, 1 IF FURTHER REFINEMENT'

ACCEPT*, IREFIN

- 9) Give the three node numbers for additional triangles in counterclockwise order. Entering three separate zeros terminates generation of additional triangles.

Prompting message: 'GIVE NØDES FØR TRIANG IN CCW ØRDER'

ACCEPT*, N1, N2, N3

- 10) Give the element number for the triangle to be deleted. Entering a single zero terminates deletion of triangles.

Prompting message: 'GIVE NUMBER ØF ØLD TRIANG TØ BE DELETED'

ACCEPT*, ITNØ

- 11) Have all the extra elements been added? If all extra elements have been added, type 0; otherwise, type 1.

Prompting message: 'ARE THERE ADDITIØNAL ELEMENTS?'

ACCEPT*, IANSWR

KØUPLE

- 12) Give the starting and ending nodes for the fixed boundary conditions in counterclockwise order.

Prompting message: 'TYPE NØDE1 NØDE2 CCW FØR FIXED BC'

ACCEPT*, N1, N2

4.1 Input Data for the 2-D Mesh

This example corresponds to Figures 3.3 - 3.7 and demonstrates the small amount of input data necessary to generate this geometry.

EXAMPLE

YELLOWSTONE

4	4				
	0.0	1.0	2.0	3.0	4.0
	0.0	1.0	1.5	2.5	3.0
2	0.0	2.0	0.0	1.0	
	3.0	4.0	0.0	1.0	
<hr/>					
8	1				
7	4	Computer draws Fig. 3.3			
9	2				
0	0				
<hr/>					
1		Computer draws Fig.3.4			
13	1				
0	0				
0					
<hr/>					
2	16	9	Computer draws Fig. 3.5		
0	0	0			
39					
40					
35					
0					
0					
<hr/>					
78	20	Computer draws Fig. 3.6			
24	72	Computer draws Fig. 3.7			
99	96				
0	0				

5. RECOMMENDATIONS

The current TRIFEM 1 code described in this report provides an excellent means of generating finite element meshes for regions that can be conceptualized as primarily rectangular. The sharp edges of the rectangles can be broken and considerable tailoring of the region can be accomplished. Flow regimes requiring extra resolution can be emphasized by generating more elements in that vicinity. Simple boundary conditions around the periphery can be specified and more complicated boundary conditions could be incorporated as necessary. As has been mentioned throughout this report, the TRIFEM 1 code is interactive and provides the user with rapid feedback on the effect of the specified operations.

A further need exists within fluid dynamical calculations for mesh generators for more complicated flow geometries. For example the coastline of an estuary or the shoreline of an irregular lake may require one to specify considerably more data, but the need for accurate simulation may require that elements internal to such regions be generated. We recommend that a mesh generator having the capability to incorporate such boundaries be implemented.

The mesh generators discussed thus far are two-dimensional finite element models. In a wide range of calculations it is not possible to idealize the flow as two-dimensional, and it becomes necessary to generate three-dimensional elements for finite-difference or finite element calculations. The types of surfaces and boundary conditions in these fluid dynamical problems are different from those encountered in similar methods of stress calculation. We recommend the development of a relatively simple three-dimensional mesh generator which will use tetrahedrons as the basic shape. This code will be primarily useful for finite element

codes that now require a great amount of hand preparation of the input data.

Initial estimates of mesh sizes are based on the intuitive knowledge of the behavior of the flow field on the part of the user. Occasionally, these estimates of the mesh size need to be changed as the flow field develops. We recommend the study of creating a self-adjusting dynamic mesh generation program which would change the size of the mesh according to gradients of the velocity field calculated in previous steps until network density and flow patterns are established.

Future work in the direction of finite element mesh generators should concentrate on: a) "TRIFEM 2", a code for complicated flow regimes, b) a mesh generator for three-dimensional flow domains using tetrahedrons, and c) a self-adjusting dynamic mesh generation program which changes the size of the finite elements according to gradients of the velocity field calculated in previous steps until network density and flow patterns are adjusted.

REFERENCES

1. W. R. Buell and B. A. Bush, "Mesh Generation--A Survey," *J. of Eng. Ind., ASME Trans.*, 332 (1973).
2. J. C. Cavendish and J. A. Wixom, "Finite Element Mesh Generation for Planar and Shell Type Structures," General Motors Research Lab, Warren, MI (1975).
3. J. C. Cavendish, "Automatic Triangulation of Arbitrary Planar Domains for the Finite Element Method," *Int. J. Numer. Methods Eng.* 8: 679 (1974).
4. G. F. Carey, "A Mesh-Refinement Scheme for Finite Element Computations," *Comput. Methods Appl. Mech. Eng.* 7: 93 (1976).
5. C. O. Frederick, T. C. Wong, and F. W. Edge, "Two-Dimensional Automatic Mesh Generation for Structural Analysis," *Int. J. Numer. Methods Eng.* 2: 133 (1970).
6. J. Fukuda and J. Suhara, "Automatic Mesh Generation for Finite Element Analysis: in *Advances in Computational Methods in Structural Mechanics and Design*, Eds. J. T. Oden et al., UAH Press, Huntsville, AL (1972).
7. H. A. Kamel and M. W. McCable, *GIFTS 4 Users Manual Graphics-Oriented Interactive Finite Element Time-Sharing System*, AME Dept., University of Arizona, Tucson, AZ (1974).
8. H. J. Barten, *Automated Finite Element Grid Break-Up Method--A Verification of the Six Node Averaging Approach in Applications of Computer Graphics in Engineering*, NASA SP-360, October 1975.
9. D. K. Gartling and E. B. Becker, "Finite Element Analysis of Viscous Incompressible Fluid Flow," *Comput. Methods Appl. Mech. Eng.* 8: 51 (1976).
10. J. T. Holdeman, *PREPR 2: A Program to Aid in the Preparation of Input Data for the Farout Hydraulic Transport Code*, ORNL/CSD/TM-19, May 1977.
11. J. T. Holdeman, Union Carbide Corp., Nuclear Division, Oak Ridge, TN, personal communication, July 1977.
12. A. Bykat, "A Note on an Element Ordering Scheme--Short Communication," *Int. J. Numer. Methods Eng.* 11: 194 (1977).
13. R. C. Durfee and R. G. Edwards, *Geographical Data Systems: Overview and Concepts*, ORNL/RUS-27 (in preparation).

14. C. Kleinstreuer, F. Ferdowsi, J. L. Bledsoe, and E. C. Long,
*Theoretical Analysis and Computer Codes for the Biota Entrainment
at Cooling Water Intakes of Power Plants Located at Non-Tidal
Rivers. Part I. Entrainment of Passive Biota*, ORNL/TM Report
(in preparation).
15. DISSPLA User's Manual, Integrated Software Systems Corp., San
Diego CA (1970).

Appendix A. SAMPLE OUTPUT FROM TRIFEM 1 RUN

ORNL-DWG 78-6246

[LNKXCT TRIFEM Execution]

INPUT HEADING
YELLOWSTONE

YELLOWSTONE
BEGIN INPUTTING DATA
GIVE NO. XS AND YS
4 4

NX= 4 NY= 4
GIVE CUM NX+1 VALUES OF X
0. 1. 2. 3. 4.

X GRID 0.00000 1.00000 2.00000 3.00000 4.00000
GIVE CUM NY+1 VALUES OF Y
0. 1. 1.5 2.5 3.0

Y GRID 0.00000 1.00000 1.50000 2.50000 3.00000
HOW MANY REGIONS WILL BE DELETED ?
2

2.
GIVE XMIN,XMAX,YMIN,YMAX FOR EACH REGION
0. 2. 0. 1.

0.000 2.000 0.000 1.000
GIVE XMIN,XMAX,YMIN,YMAX FOR EACH REGION
3. 4. 0. 1.

ORNL-DWG 78-6247

ID OF ELT TO BE REFINED, OPERATION 0 0 STOP
8 1

ID OF ELT TO BE REFINED, OPERATION 0 0 STOP
7 4

ID OF ELT TO BE REFINED, OPERATION 0 0 STOP
9 2

ID OF ELT TO BE REFINED, OPERATION 0 0 STOP
0 0

TYPE 0 IF COMPLETE, 1 IF FURTHER REFINMENT
1

ID OF ELT TO BE REFINED, OPERATION 0 0 STOP
13 1

ID OF ELT TO BE REFINED, OPERATION 0 0 STOP
0 0

TYPE 0 IF COMPLETE, 1 IF FURTHER REFINMENT
0

ORNL-DWG 78-6248

GIVE NODES FOR NEW TRIANG IN CCW ORDER
2 16 9

2, 16, 9,
GIVE NODES FOR NEW TRIANG IN CCW ORDER
0 0 0

0, 0, 0,
GIVE NUMBER OF OLD TRIANG TO BE DELETED
39

39,
GIVE NUMBER OF OLD TRIANG TO BE DELETED
40

40,
GIVE NUMBER OF OLD TRIANG TO BE DELETED
35

35,
GIVE NUMBER OF OLD TRIANG TO BE DELETED
0

0,
ARE THERE ADDITIONAL ELEMENTS?
0

ORNL-DWG 78-6249

IT,NODSIX	1	1	2	3	4	5	6
IT,NODSIX	2	3	7	8	9	5	4
IT,NODSIX	3	1	6	5	10	11	12
IT,NODSIX	4	3	13	14	15	8	7
IT,NODSIX	5	5	9	8	16	17	18
IT,NODSIX	6	11	10	5	18	17	19
IT,NODSIX	7	20	21	1	12	11	22
IT,NODSIX	8	3	23	24	25	14	13
IT,NODSIX	9	14	26	27	28	8	15
IT,NODSIX	10	29	30	11	19	17	31
IT,NODSIX	11	32	33	20	22	11	34
IT,NODSIX	12	14	35	36	37	27	26
IT,NODSIX	13	32	34	11	30	29	38
IT,NODSIX	14	36	39	40	41	27	37
IT,NODSIX	15	32	38	29	42	43	44
IT,NODSIX	16	36	45	46	47	40	39
IT,NODSIX	17	27	41	40	48	49	50
IT,NODSIX	18	51	52	43	42	29	53
IT,NODSIX	19	54	55	32	44	43	56
IT,NODSIX	20	46	57	58	59	40	47
IT,NODSIX	21	60	61	49	48	40	62
IT,NODSIX	22	63	64	43	52	51	65
IT,NODSIX	23	51	53	29	66	49	67
IT,NODSIX	24	54	56	43	68	69	70
IT,NODSIX	25	46	71	72	73	58	57
IT,NODSIX	26	58	74	60	62	40	59
IT,NODSIX	27	60	75	51	67	49	61
IT,NODSIX	28	63	76	69	68	43	64
IT,NODSIX	29	63	65	51	75	60	77
IT,NODSIX	30	78	79	54	70	69	80
IT,NODSIX	31	58	81	82	83	60	74
IT,NODSIX	32	63	84	85	86	69	76
IT,NODSIX	33	87	88	63	77	60	89

ORNL-DWG 78-6250

IT,NODSIX	34	90	91	78	80	69	92
IT,NODSIX	35	87	89	60	83	82	93
IT,NODSIX	36	87	94	85	84	63	88
IT,NODSIX	37	85	95	96	97	69	86
IT,NODSIX	38	96	98	90	92	69	97
IT,NODSIX	39	99	100	87	93	82	101
1	4.00000	1.50000					
2	4.00000	2.00000					
3	4.00000	2.50000					
4	3.50000	2.25000					
5	3.00000	2.00000					
6	3.50000	1.75000					
7	3.50000	2.50000					
8	3.00000	2.50000					
9	3.00000	2.25000					
10	3.00000	1.75000					
11	3.00000	1.50000					
12	3.50000	1.50000					
13	3.50000	2.75000					
14	3.00000	3.00000					
15	3.00000	2.75000					
16	2.75000	2.25000					
17	2.50000	2.00000					
18	2.75000	2.00000					
19	2.75000	1.75000					
20	4.00000	1.00000					
21	4.00000	1.25000					
22	3.50000	1.25000					
23	4.00000	2.75000					
24	4.00000	3.00000					
25	3.50000	3.00000					
26	2.50000	2.75000					

ORNL-DWG 78-6251

27	2.00000	2.50000
28	2.50000	2.50000
29	2.50000	1.50000
30	2.75000	1.50000
31	2.50000	1.75000
32	3.00000	1.00000
33	3.50000	1.00000
34	3.00000	1.25000
35	2.50000	3.00000
36	2.00000	3.00000
37	2.00000	2.75000
38	2.75000	1.25000
39	1.50000	2.75000
40	1.00000	2.50000
41	1.50000	2.50000
42	2.50000	1.25000
43	2.50000	1.00000
44	2.75000	1.00000
45	1.50000	3.00000
46	1.00000	3.00000
47	1.00000	2.75000
48	1.50000	2.25000
49	2.00000	2.00000
50	2.00000	2.25000
51	2.00000	1.50000
52	2.25000	1.25000
53	2.25000	1.50000
54	3.00000	0.50000
55	3.00000	0.75000
56	2.75000	0.75000
57	0.50000	2.75000
58	0.00000	2.50000

ORNL-DWG 78-6252

59	0.50000	2.50000
60	1.00000	1.50000
61	1.50000	1.75000
62	1.00000	2.00000
63	2.00000	1.00000
64	2.25000	1.00000
65	2.00000	1.25000
66	2.25000	1.75000
67	2.00000	1.75000
68	2.50000	0.75000
69	2.50000	0.50000
70	2.75000	0.50000
71	0.50000	3.00000
72	0.00000	3.00000
73	0.00000	2.75000
74	0.50000	2.00000
75	1.50000	1.50000
76	2.25000	0.75000
77	1.50000	1.25000
78	3.00000	0.00000
79	3.00000	0.25000
80	2.75000	0.25000
81	0.00000	2.00000
82	0.00000	1.50000
83	0.50000	1.50000
84	2.00000	0.75000
85	2.00000	0.50000
86	2.25000	0.50000
87	1.00000	1.00000
88	1.50000	1.00000
89	1.00000	1.25000
90	2.50000	0.00000
91	2.75000	0.00000
92	2.50000	0.25000
93	0.50000	1.25000
94	1.50000	0.75000
95	2.00000	0.25000

96	2.00000	0.00000
97	2.25000	0.25000
98	2.25000	0.00000
99	0.00000	1.00000
100	0.50000	1.00000
101	0.00000	1.25000

BOUNDARY	1	1	2
BOUNDARY	2	2	3
BOUNDARY	3	3	23
BOUNDARY	4	23	24
BOUNDARY	5	24	25
BOUNDARY	6	25	14
BOUNDARY	7	14	35
BOUNDARY	8	35	36
BOUNDARY	9	36	45
BOUNDARY	10	45	46
BOUNDARY	11	46	71
BOUNDARY	12	71	72
BOUNDARY	13	72	73
BOUNDARY	14	73	58
BOUNDARY	15	58	81
BOUNDARY	16	81	82
BOUNDARY	17	82	101
BOUNDARY	18	101	99
BOUNDARY	19	99	100
BOUNDARY	20	100	87
BOUNDARY	21	87	94
BOUNDARY	22	94	85
BOUNDARY	23	85	95
BOUNDARY	24	95	96
BOUNDARY	25	96	98
BOUNDARY	26	98	90
BOUNDARY	27	90	91
BOUNDARY	28	91	78
BOUNDARY	29	78	79
BOUNDARY	30	79	54

ORNL DWG 79-15565

BOUNDARY	31	54	55
BOUNDARY	32	55	32
BOUNDARY	33	32	33
BOUNDARY	34	33	20
BOUNDARY	35	20	21
BOUNDARY	36	21	1
BOUNDARY	37	49	50
BOUNDARY	38	50	27
BOUNDARY	39	27	28
BOUNDARY	40	28	8
BOUNDARY	41	8	16
BOUNDARY	42	16	17
BOUNDARY	43	17	31
BOUNDARY	44	31	29
BOUNDARY	45	29	66
BOUNDARY	46	66	49

NODE	78	BC	0.00000	0.00000	0.00000
NODE	79	BC	0.00000	0.00000	0.00000
NODE	54	BC	0.00000	0.00000	0.00000
NODE	55	BC	0.00000	0.00000	0.00000
NODE	32	BC	0.00000	0.00000	0.00000
NODE	33	BC	0.00000	0.00000	0.00000
NODE	20	BC	0.00000	0.00000	0.00000

TYPE NODE1 NODE2 CCW FOR FIXED BC

24 72

NODE	24	BC	0.00000	0.00000	0.00000
NODE	25	BC	0.00000	0.00000	0.00000
NODE	14	BC	0.00000	0.00000	0.00000
NODE	35	BC	0.00000	0.00000	0.00000
NODE	36	BC	0.00000	0.00000	0.00000
NODE	45	BC	0.00000	0.00000	0.00000
NODE	46	BC	0.00000	0.00000	0.00000
NODE	71	BC	0.00000	0.00000	0.00000
NODE	72	BC	0.00000	0.00000	0.00000

TYPE NODE1 NODE2 CCW FOR FIXED BC

99 96

NODE	99	BC	0.00000	0.00000	0.00000
NODE	100	BC	0.00000	0.00000	0.00000
NODE	87	BC	0.00000	0.00000	0.00000
NODE	94	BC	0.00000	0.00000	0.00000
NODE	85	BC	0.00000	0.00000	0.00000
NODE	95	BC	0.00000	0.00000	0.00000
NODE	96	BC	0.00000	0.00000	0.00000

TYPE NODE1 NODE2 CCW FOR FIXED BC

0 0

**THIS PAGE
WAS INTENTIONALLY
LEFT BLANK**

Appendix B. PROGRAM LISTING OF TRIFEM 1

```

C      T R I F E M :
C *****
C      MESH GENERATOR FOR FINITE ELEMENT
C      CODE RIVINCA-PB
C *****
      DIMENSION X(50),Y(50),XL(250),XR(250),
1     YB(250),YT(250),IDBL(250),IOPER(250),
2     XLN(250),XRN(250),YBN(250),YTN(250),
3     XP(1500),YP(1500),XN(1000),YN(1000),
4     IVERT(3,750),ISEG(2,250),ISIDE(2,250),HEADER(10)

C      DIMENSION XA(1000),YA(1000),NVERT(3,750),ION(1000),NEW(750)

C
C      DIMENSION ICOD(1500),NODSIX(6,250),IBC(1500),LOC(1500)
      COMMON/REG/XMINR(2),YMINR(2),XMAXR(2),YMAXR(2),NREG,EPS
      DATA HEADER(10)/'S'  '/'
      CALL TKTRN(120,0)
      NTDIM=750
C      NUMBER OF TRIANGLES      MAXIMUM DIMENSION IVERT(3,750)
      TYPE 8002
8002 FORMAT(' INPUT HEADING')
      ACCEPT 8003,(HEADER(I),I=1, 9)
8003 FORMAT(16A5)
      TYPE 8003,HEADER
      TYPE 8000
8000 FORMAT(' BEGIN INPUTTING DATA ',/' GIVE NO. XS AND YS')
      ACCEPT *,NX,NY
      TYPE 8005,NX,NY
8005 FORMAT(' NX=',I3,5X,'NY=',I3)
      NX1=NX+1
      NY1=NY+1
      TYPE 8010
8010 FORMAT(' GIVE CUM NX+1 VALUES OF X')
      ACCEPT *,(X(I),I=1,NX1)
      TYPE 8015,(X(I),I=1,NX1)
8015 FORMAT(' X GRID',7F10.5)
C
      TYPE 8020
8020 FORMAT(' GIVE CUM NY+1 VALUES OF Y')
      ACCEPT *,(Y(I),I=1,NY1)
      TYPE 8025,(Y(I),I=1,NY1)
8025 FORMAT(' Y GRID',7F10.5)
C
C
      TYPE 8030
8030 FORMAT(' HOW MANY REGIONS WILL BE DELETED ?')
      ACCEPT *,NREG
      TYPE *, NREG
      DO 100 N=1,NREG
C
      TYPE 8035
8035 FORMAT(' GIVE XMIN,XMAX,YMIN,YMAX FOR EACH REGION ')
      ACCEPT *,XMINR(N),XMAXR(N),YMINR(N),YMAXR(N)
      TYPE 8040,XMINR(N),XMAXR(N),YMINR(N),YMAXR(N)
8040 FORMAT(4F10.3)
100 CONTINUE
      XAXIS=11.0
      YAXIS=8.
      HGT = .10
      XOFSET=0.10
      YOFSET=0.10
      CALL BGNPL(1)

```

```

      CALL PAGE(XAXIS+3.0,YAXIS+3.0)
      CALL TITLE(HEADER,100,'CHANNEL LENGTH$',100,
1    'CHANNEL WIDTH$',100,XAXIS,YAXIS)
      CALL GRAF(X(1),'SCALE',X(NX1),Y(1),'SCALE',Y(NY1))
      CALL HEIGHT(HGT)
C    DRAW A PRIMARY AXIS
C    GENERATE INITIAL RECTANGULAR ELEMENTS
      NE=0
      DO 125 IX=1,NX
      DO 120 IY=1,NY
      NE=NE+1
      XL(NE)=X(IX)
      XR(NE)=X(IX+1)
      YB(NE)=Y(IY)
      YT(NE)=Y(IY+1)
      XC=(XL(NE)+XR(NE))/2.0
      YC=(YB(NE)+YT(NE))/2.0
C
      IF(NREG .LE. 0) GO TO 110
      DO 105 N=1,NREG
      IF((XC .GE. XMINR(N) .AND. XC .LE. XMAXR(N)) .AND.
1    (YC .GE. YMINR(N) .AND. YC .LE. YMAXR(N)))
2    GO TO 115
105 CONTINUE
C
110 CALL QUADEL(NE,XL,YB,XR,YT,XC,YC)
C
      GO TO 120
115 NE=NE-1
120 CONTINUE
125 CONTINUE
      CALL ENDPL(1)
130 NDOUB=0
135 TYPE 8050
8050 FORMAT(' ID OF ELT TO BE REFINED, OPERATION 0 0 STOP')
      ACCEPT*,IDOUB,IOP
      IF(IDOUB.LE.0)GO TO 140
      NDOUB=NDOUB+1
      IDBL(NDOUB)=IDOUB
      IOPER(NDOUB)=IOP
      GO TO 135
140 IF(NDOUB.LE.0)GO TO 175
      CALL SCRTER(IDBL,NDOUB,IOPER)
      KE=0
      INEW=1
      DO 165 IE=1,NE
      IF(IE.EQ.IDBL(INEW))GO TO 145
      KE=KE+1
      CALL COORD(KE,XL(IE),XR(IE),YB(IE),YT(IE),XLN,XRN,YBN,YTN)
      GO TO 165
C
C
145 XMID=(XL(IE)+XR(IE))/2.0
      YMID=(YB(IE)+YT(IE))/2.0
      IF(IOPER(INEW) .LE. 0) CALL ERROR
      IF(IOPER(INEW) .GT. 1) GO TO 150
C    HALVE ELEMENT VERTICALLY    IOP=1
      KE = KE +1
      CALL COORD(KE,XL(IE),XMID,YB(IE),YT(IE),XLN,XRN,YBN,YTN)
      KE = KE +1
      CALL COORD(KE,XMID,XR(IE),YB(IE),YT(IE),XLN,XRN,YBN,YTN)
      GO TO 160
150 IF(IOPER(INEW) .GT. 2) GO TO 155

```

```

C      HALVE ELEMENT HORIZONTALLY,      IOP=2
      KE = KE + 1
      CALL COORD(KE,XL(IE),XR(IE),YB(IE),YMID,XLN,XRN,YBN,YTN)
      KE = KE + 1
      CALL COORD(KE,XL(IE),XR(IE),YMID,YT(IE),XLN,XRN,YBN,YTN)
      GO TO 160
C      QUARTER ELEMENT,      IOP >= 3
C      LOWER LEFT RECTANGLE
155    KE = KE + 1
      CALL COORD(KE,XL(IE),XMID,YB(IE),YMID,XLN,XRN,YBN,YTN)
C      UPPPER LEFT RECTANGLE
      KE=KE+1
      CALL COORD(KE,XL(IE),XMID,YMID,YT(IE),XLN,XRN,YBN,YTN)
C      LOWER RIGHT RECTANGLE
      KE=KE+1
      CALL COORD(KE,XMID,XR(IE),YB(IE),YMID,XLN,XRN,YBN,YTN)
C      UPPER RIGHT RECTANGLE
      KE=KE+1
      CALL COORD(KE,XMID,XR(IE),YMID,YT(IE),XLN,XRN,YBN,YTN)
160    INEW=INEW+1
      IF(INEW.GT.NDOUB) INEW=NDOUB
165    CONTINUE
      NE=KE
      CALL BGNPL(2)
      CALL PAGE(XAXIS+3.0,YAXIS+3.0)
      CALL TITLE(HEADER,100,'CHANNEL LENGTH$',100,
1      'CHANNEL WIDTH$',100,XAXIS,YAXIS)
      CALL GRAF(X(1),'SCALE',X(NX1),Y(1),'SCALE',Y(NY1))
      DO 170 I=1,KE
      IE=I
      XL(IE)=XLN(IE)
      XR(IE)=XRN(IE)
      YB(IE)=YBN(IE)
      YT(IE)=YTN(IE)
C
      XC=(XL(IE)+XR(IE))/2.0
      YC=(YB(IE)+YT(IE))/2.0
C
      CALL QUADDEL(IE,XL,YB,XR,YT,XC,YC)
C
170    CONTINUE
C
      CALL ENDPL(2)
175    TYPE      8055
8055    FORMAT('O TYPE 0 IF COMPLETE, 1 IF FURTHER REFINMENT')
      ACCEPT *,IREFIN
      IF(IREFIN.GT.0) GO TO 130
C
C
      EPS=AMAX1(X(NX1),Y(NY1))*1.0E-5
      CALL NODES(EPS,NE,XL,XR,YB,YT,NP,XP,YP,NN,XN,YN)
      HGT2=.800*HGT
      CALL BGNPL(3)
      CALL PAGE(XAXIS+3.0,YAXIS+3.0)
      CALL TITLE(HEADER,100,'CHANNEL LENGTH$',100,
1      'CHANNEL WIDTH$',100,XAXIS,YAXIS)
      CALL GRAF(X(1),'SCALE',X(NX1),Y(1),'SCALE',Y(NY1))
      CALL HEIGHT(HGT2)
      CALL PTRIL(NN,XN,YN,NTDIM,IVERT,NTRI,ISEG,NS,ISIDE)
C
C      IDENTIFY TRIANGLES
C
C      NUMBER EACH TRIANGLE

```

```

DO 180 ITR1=1,NTRI
  ITR=ITR1
  XCT=(XN(IVERT(1,ITR))+XN(IVERT(2,ITR))+XN(IVERT(3,ITR)))/3.0
  YCT=(YN(IVERT(1,ITR))+YN(IVERT(2,ITR))+YN(IVERT(3,ITR)))/3.0
C
  CALL RLINT(ITR,XCT,YCT)
C
180 CONTINUE
C
  CALL SHIFT(XOFSET,YOFSET)
  DO 185 I1=1,NN
    I=I1
    CALL RLINT(I,XN(I),YN(I))
185 CONTINUE
  CALL ENDPL(3)
C
  GENERATE NEW TRIANGLES IN COUNTERCLOCKWISE
190 TYPE 8060
8060 FORMAT(' GIVE NODES FOR NEW TRIANG IN CCW ORDER')
  ACCEPT *,N1,N2,N3
  TYPE *,N1,N2,N3
  IF(N1 .LE. 0) GO TO 195
  NTRI=NTRI+1
  IVERT(1,NTRI)=N1
  IVERT(2,NTRI)=N2
  IVERT(3,NTRI)=N3
  GO TO 190
C
  DELETE OLD TRIANGLES
195 TYPE 8065
8065 FORMAT(' GIVE NUMBER OF OLD TRIANG TO BE DELETED')
  ACCEPT *,ITNO
  TYPE *,ITNO
  IF(ITNO .LE. 0) GO TO 205
  DO 200 IN=1,3
    IVERT(IN,ITNO)=IVERT(IN,NTRI)
200 CONTINUE
  NTRI=NTRI-1
  GO TO 195
205 TYPE 8070
8070 FORMAT(' ARE THERE ADDITIONAL ELEMENTS?')
  ACCEPT *, IANSWR
  TYPE *,IANSWR
  IF(IANSWR .GT. 0) GO TO 190
C
  BAND OPTIMIZATION
C
  FIND STARTING ELT FOR MIN BANDWIDTH WITH BYKAT SCHEME
  MAXDIF= 9999
  NT= NTRI
  DO 210 IT2 = 1,NTRI
    IT=IT2
    CALL BAND(IT,NN,XN,YN,NT,IVERT,
1    NODTOT,XA,YA,NVERT,ION,NEW,MBW)
    IF(MBW .GE. MAXDIF) GO TO 210
    ISTART = IT
    MAXDIF= MBW
210 CONTINUE
  TYPE 8075,ISTART
8075 FORMAT(' STARTING ELT FOR MIN BW IS ',I3)
  IF(ISTART .LE. 0) GO TO 235
  CALL BAND(ISTART,NN,XN,YN,NTRI,IVERT,
1    NODTOT,XA,YA,NVERT,ION,NEW,MBW)
C
  HGT3=HGT2
  CALL BGNPL(5)

```

```

CALL PAGE(XAXIS+3.0,YAXIS+3.0)
CALL TITLE(HEADER,100,'CHANNEL LENGTH$',100,
1 'CHANNEL WIDTH$',100,XAXIS,YAXIS)
CALL GRAF(X(1),'SCALE',X(NX1),Y(1),'SCALE',Y(NY1))
CALL HEIGHT(HGT3)
C PLOT NEW TRIANGLES, IDENTIFY NODES AND ELTS
  CALL RESET('SHIFT')
  DO 215 IT=1,NTRI
    NOD1=NVERT(1,IT)
    NOD2=NVERT(2,IT)
    NOD3=NVERT(3,IT)
    CALL RLVEC(XA(NOD1),YA(NOD1),XA(NOD2),YA(NOD2),1100)
    CALL RLVEC(XA(NOD2),YA(NOD2),XA(NOD3),YA(NOD3),1100)
    CALL RLVEC(XA(NOD3),YA(NOD3),XA(NOD1),YA(NOD1),1100)
    XCB=(XA(NVERT(1,IT))+XA(NVERT(2,IT))+XA(NVERT(3,IT)))/3.0
    YCB=(YA(NVERT(1,IT))+YA(NVERT(2,IT))+YA(NVERT(3,IT)))/3.0
    ITT=IT
    CALL RLINT(ITT,XCB,YCB)
  215 CONTINUE
C
  CALL SHIFT(XOFSET,YOFSET)
  DO 220 I1=1,NODTOT
    II=I1
    CALL RLINT(II,XA(II),YA(II))
  220 CONTINUE
  CALL ENDPL(5)
  NSIX=6*NT
  CALL KOUPLE(NSIX,NODTOT,XA,YA,NT,NVERT,ION,IVERT,
1 NNM,XP,YP,ICOD,NODSIX,ISEG,IBC,LOC)
  HGTNEW=HGT3
  CALL BGNPL(6)
  CALL PAGE(XAXIS+3.0,YAXIS+3.0)
  CALL TITLE(HEADER,100,'CHANNEL LENGTH$',100,
1 'CHANNEL WIDTH$',100,XAXIS,YAXIS)
  CALL GRAF(X(1),'SCALE',X(NX1),Y(1),'SCALE',Y(NY1))
  CALL HEIGHT(HGTNEW)
C PLOT FINAL TRIANGLES, IDENTIFY NODES AND ELTS
  CALL RESET('SHIFT')
  DO 225 ITR=1,NT
    NOD1=NODSIX(1,ITR)
    NOD3=NODSIX(3,ITR)
    NOD5=NODSIX(5,ITR)
    CALL RLVEC(XP(NOD1),YP(NOD1),XP(NOD3),YP(NOD3),1100)
    CALL RLVEC(XP(NOD3),YP(NOD3),XP(NOD5),YP(NOD5),1100)
    CALL RLVEC(XP(NOD5),YP(NOD5),XP(NOD1),YP(NOD1),1100)
    XCN=(XP(NOD1)+XP(NOD3)+XP(NOD5))/3.0
    YCN=(YP(NOD1)+YP(NOD3)+YP(NOD5))/3.0
    I=ITR
    CALL RLINT(I,XCN,YCN)
  225 CONTINUE
C
C
  CALL SHIFT(XOFSET,YOFSET)
  DO 230 IN=1,NNM
    INM=IN
    CALL RLINT(INM,XP(INM),YP(INM))
  230 CONTINUE
  CALL ENDPL(6)
  235 CALL DONEPL
  STOP
  END

```

```

SUBROUTINE QUADDEL(II,XL,YB,XR,YT,XC,YC)
C
  DIMENSION XR(II),YB(II),XL(II),YT(II)
C
  CALL HEIGHT(0.10)
  CALL RLVEC(XL(II),YB(II),XR(II),YB(II),1100)
  CALL RLVEC(XR(II),YB(II),XR(II),YT(II),1100)
  CALL RLVEC(XR(II),YT(II),XL(II),YT(II),1100)
  CALL RLVEC(XL(II),YT(II),XL(II),YB(II),1100)
  CALL RLINT(II,XC,YC)
  RETURN
  END

SUBROUTINE SORTER(IDBL,NELTS,IOPER)
C
  DIMENSION IDBL(NELTS),IOPER(NELTS)
C
  IF(NELTS .LE. 1) GO TO 110
C NELTS <= 1; NO NEED TO SORT RETURN
  NM1=NELTS-1
  DO 105 I=1,NM1
    JKEEP=I
    JTHR=I+1
    DO 100 J=JTHR,NELTS
      IF(IDBL(J) .GE. IDBL(JKEEP)) GO TO 100
      JKEEP=J
100  CONTINUE
      IF(JKEEP .EQ. I) GO TO 105
C
      IDBL1=IDBL(I)
      IOPI=IOPER(I)
      IDBL(I)=IDBL(JKEEP)
      IOPER(I)=IOPER(JKEEP)
      IDBL(JKEEP)=IDBL1
      IOPER(JKEEP)=IOPI
105  CONTINUE
110  RETURN
  END

SUBROUTINE COORD(KE,XL,XR,YB,YT,XLN,XRN,YBN,YTN)
  DIMENSION XLN(1),XRN(1),YBN(1),YTN(1)
  XLN(KE)=XL
  XRN(KE)=XR
  YBN(KE)=YB
  YTN(KE)=YT
  XCN=(XL+XR)/2.0
  YCN=(YB+YT)/2.0
  RETURN
  END

```



```

SUBROUTINE NODES(EPS,NE,XL,XR,YB,YT,
1 NP,XP,YP,NN,XN,YN)
  DIMENSION XL(1),XR(1),YB(1),YT(1),
1 XP(1),YP(1),XN(1),YN(1)
C      GENERATE ALL POSSIBLE NODAL POINTS
      IP=0
      DO 100 IE=1,NE
      IP=IP+1
      XP(IP)=XL(IE)
      YP(IP)=YB(IE)
      IP=IP+1
      XP(IP)=XR(IE)
      YP(IP)=YB(IE)
      IP=IP+1
      XP(IP)=XR(IE)
      YP(IP)=YT(IE)
      IP=IP+1
      XP(IP)=XL(IE)
      YP(IP)=YT(IE)
100 CONTINUE
C      GENERATE NODE SPACE
      NP=IP
      DO 105 IP=1,NP
      XN(IP)=XP(IP)
      YN(IP)=YP(IP)
105 CONTINUE
C      RETAIN ONLY ONE POINT FOR EACH NODE
      JPL=NP
      JP=1
110 JPI=JP+1
      JN=JP
      DO 120 IP=JPI,JPL
      IF POINT IP SAME AS JP, DISCARD IP
      IF (ABS(XN(IP)-XN(JP)).LE.EPS. AND
1 ABS(YN(IP)-YN(JP)).LE.EPS) GO TO 120
C      PCINT IP DIFFERENT FROM JP, TABULATE IP
115 JN=JN+1
      XN(JN)=XN(IP)
      YN(JN)=YN(IP)
120 CONTINUE
      JPL=JN
      JP=JP+1
      IF (JP.GE.JPL) GO TO 125
      GO TO 110
C      ARRAY OF UNIQUE POINTS COMPLETE, ASSIGN
C      NODES TO RECTANGULAR ELEMENTS
125 NN=JPL
      RETURN
      END

```

```

SUBROUTINE PTRIL(NN,X,Y,NNT,TRANGL,NT,SEG,NS,SIDE)
C
C      . . . . .
C      * GIVEN NN DATA POINTS OR NODES IN A PLANE WITH
C      COORDINATES (X,Y), THIS SUBROUTINE PARTITIONS THE CONVEX
C      REGION INTO TRIANGLES WITH DATA POINTS AS VERTICES.
C      * UPON COMPLETION THE NODES OR VERTICES OF EACH TRIANGLE
C      ARE RETURNED IN THE ARRAY TRANGL, WHILE THE ARRAY SEG
C      CONTAINS THE NODES OR ENDPOINTS OF THE LINE SEGMENTS
C      MAKING UP THE PERIMETER OF THE CONVEX REGION.

```

```

C  * THE ARRAY TRANGL MUST BE DIMENSIONED AT LEAST
C  2*NN-5 IN THE CALLING PROGRAM.
C  * THE SECOND DIMENSION OF THE ARRAY SEG MUST BE AT LEAST NN+2
C  IN THE CALLING PROGRAM. A RIGOROUS BOUND ON THIS DIMENSION
C  IS NOT KNOWN BUT THIS BOUND HAS OCCURED WHEN TRIANGULATING
C  RANDOM POINTS IN THE PLANE FOR NN AROUND 12.
C  . . . . . JTH . . . . .
C
COMMON/REG/XMINR(2),YMINR(2),XMAXR(2),YMAXR(2),NREG,EPS
COMMON/STATS/MNS,MLVL,MNT
DIMENSION X(NN),Y(NN),TRANGL(3,NN),SEG(2,NN),SIDE(2,NN)
LOGICAL CROSS
INTEGER TRANGL,SEG,SIDE,P1,P2

C
C  FIND FIRST SEGMENT
C
100 I=1
   J=2
   DMIN=(X(I)-X(J))**2+(Y(I)-Y(J))**2
   DO 105 K=3,NN
   DTRY=(X(I)-X(K))**2+(Y(I)-Y(K))**2
   IF(DTRY.EQ.0.) GO TO 105
   IF(DTRY.GE.DMIN) GO TO 105
   DMIN=DTRY
   J=K
105 CONTINUE
   NT=0
   NS=1
   SEG(1,NS)=J
   SEG(2,NS)=I
   NS=NS+1
   MNS=NS
   SEG(1,NS)=I
   SEG(2,NS)=J
   IJ=NS
   LVL=1
   MLVL=LVL
   SIDE(1,LVL)=J
   SIDE(2,LVL)=I
C   SIDE FOUND ON PERIMETER LIST
110 CCDM=1.E20
C   EXAMINE EACH NODAL POINT FOR FEASIBILITY OF FORMING
C   A NEW TRIANGLE
   N=0
   DO 130 K=1,NN
C   SKIP IF POINT IS TO THE LEFT OF SEGMENT (I,J)
   XPD=(X(J)-X(I))*(Y(K)-Y(I))-(Y(J)-Y(I))*(X(K)-X(I))
   IF(XPD.GE.0.) GO TO 130
   CCD=-((Y(J)-Y(K))*(Y(I)-Y(K))+(X(J)-X(K))*(X(I)-X(K)))/XPD
   IF(CCD.GE.CCDM) GO TO 130
C   TEST FOR INTERSECTING SEGMENTS TO PREVENT OVERLAPING
C   TRIANGLES
   DO 115 LS=1,NS
   P1=SEG(1,LS)
   P2=SEG(2,LS)
   IF(CROSS(X(I),Y(I),X(K),Y(K),X(P1),Y(P1),X(P2),Y(P2)))GOTO130
   IF(CROSS(X(J),Y(J),X(K),Y(K),X(P1),Y(P1),X(P2),Y(P2)))GOTO130
115 CONTINUE
   IF( NREG .LE. 0 ) GO TO 125
   DO 120 NR=1,NREG
C  1ST CHECK IS FOR BOTTOM SIDE REGION
   IF(CROSS(X(I),Y(I),X(K),Y(K),
1XMINR(NR)+EPS,YMINR(NR)+EPS,XMAXR(NR)-EPS,YMAXR(NR)-EPS))GO TO 130

```

```

C      IF(CROSS(X(J),Y(J),X(K),Y(K),
1XMINR(NR)+EPS,YMINR(NR)+EPS,XMAXR(NR)-EPS,YMAXR(NR)-EPS))GO TO 130
C 2ND CHECK IS FOR RIGHT SIDE OF REGION
      IF(CROSS(X(I),Y(I),X(K),Y(K),
1MAXR(NR)-EPS,YMINR(NR)+EPS,XMINR(NR)+EPS,YMAXR(NR)-EPS)) GO TO 130
      IF(CROSS(X(J),Y(J),X(K),Y(K),
1MAXR(NR)-EPS,YMINR(NR)+EPS,XMINR(NR)+EPS,YMAXR(NR)-EPS)) GO TO 130
120 CONTINUE
C      HAVE FOUND FEASIBLE POINT
125 CCOM=CCD
      N=K
130 CONTINUE
C      WAS A POINT FOUND ?
      IF(N.EQ.0) GO TO 155
C      DELETE SEGMENT (I,J) FROM PERIMETER LIST
      SEG(1,IJ)=SEG(1,NS)
      SEG(2,IJ)=SEG(2,NS)
      NS=NS-1
      NT=NT+1
      MNT=NT
      TRANGL(1,NT)=I
      TRANGL(2,NT)=N
      TRANGL(3,NT)=J
      CALL RLVEC(X(I),Y(I),X(N),Y(N),1100)
      CALL RLVEC(X(N),Y(N),X(J),Y(J),1100)
      CALL RLVEC(X(J),Y(J),X(I),Y(I),1100)
C      IF SEGMENT (N,J) IS ON THE PERIMETER LIST THEN DELETE IT
      DO 135 K=1,NS
      IF(SEG(1,K).NE.J) GO TO 135
      IF(SEG(2,K).NE.N) GO TO 135
      SEG(1,K)=SEG(1,NS)
      SEG(2,K)=SEG(2,NS)
      NS=NS-1
      GO TO 140
135 CONTINUE
C      ELSE ADD IT TO THE PERIMETER LIST
      LVL=LVL+1
      MLVL=MAX0(MLVL,LVL)
      SIDE(1,LVL)=N
      SIDE(2,LVL)=J
      NS=NS+1
      MNS=MAX0(MNS,NS)
      SEG(1,NS)=N
      SEG(2,NS)=J
C      IS SEGMENT (I,N) ON THE PERIMETER LIST ?
140 DO 145 K=1,NS
      IF(SEG(1,K).NE.N) GO TO 145
      IF(SEG(2,K).EQ.I) GO TO 150
145 CONTINUE
C      NO, SO ADD IT TO THE PERIMETER LIST
      NS=NS+1
      MNS=MAX0(MNS,NS)
      SEG(1,NS)=I
      SEG(2,NS)=N
      J=N
      IJ=NS
      GO TO 110
C      YES, SO DELETE IT FROM THE PARAMETER LIST
150 SEG(1,K)=SEG(1,NS)
      SEG(2,K)=SEG(2,NS)
      NS=NS-1
C      NO FEASIBLE POINT AVAILABLE, GO BACK TO PREVIOUS TRIANGLE

```

```

155 IF(LVL.EQ.0) GO TO 165
    I=SIDE(1,LVL)
    J=SIDE(2,LVL)
    LVL=LVL-1
C    IS THIS SIDE STILL ON PERIMETER LIST ?
    DO 160 IJ=1,NS
    IF(SEG(1,IJ).NE.I) GO TO 160
    IF(SEG(2,IJ).EQ.J) GO TO 110
160 CONTINUE
C    NOT ON LIST, MUST HAVE BEEN PROCESSED
    GO TO 155
C    FINISHED THE TRIANGULATION
C
165 RETURN
    END

```

```

FUNCTION CROSS(XS1,YS1,XS2,YS2,XT1,YT1,XT2,YT2)
C
C    DETECT INTERIOR INTERSECTION OF LINE SEGMENTS
C
C    LOGICAL CROSS
C
    CROSS=.FALSE.
    DXS=XS2-XS1
    DYS=YS2-YS1
    DXT=XT2-XT1
    DYT=YT2-YT1
    C1=DYS*DXT
    C2=DXS*DYT
    CD=C1-C2
    ACD=ABS(CD)
    CTST=(ABS(C1)+ABS(C2))*1.E-6
    IF(ACD.LE.CTST) GO TO 100
    CNS=(XS1-XT1)*DYT-(YS1-YT1)*DXT
    IF(ABS(CNS-CD+CNS).GE.ACD-CTST) GO TO 100
    CNT=(XS1-XT1)*DYS-(YS1-YT1)*DXS
    IF(ABS(CNT-CD+CNT).GE.ACD-CTST) GO TO 100
    CROSS=.TRUE.
100 RETURN
    END

```

```

SUBROUTINE BAND(ISTART,N,X,Y,NT,NODES,
1  NODTOT,XNEW,YNEW,JSIDE,ION,NEW,MBW)
C    SUBROUTINE BAND TO OPTIMIZE MAT BAND WIDTH
C
    DIMENSION X(N),Y(N),NODES(3,NT),XNEW(N),YNEW(N),JSIDE(3,NT),
1  ION(N),NEW(NT)
C    IMPLEMENTATION OF A. BYKAT, ELEMENT ORDERING SCHEME
C    IDENTIFY SIDES
    MBW=0
    DO 105 IT=1,NT
    DO 100 ID=1,3
    N1=ID
    N2=MOD(ID,3)+1
    JSIDE(ID,IT)=KSIDE(NODES(N1,IT),NODES(N2,IT),N)
    NODE1=NODES(N1,IT)
    NODE2=NODES(N2,IT)

```

```

        MBW=MAX0(MBW,IABS(NODE2-NODE1))
100 CONTINUE
105 CONTINUE
        MBWOLD = MBW
C   REPLACE SIDES WITH NEGATIVE OF ADJACENT TRIANGLE NUMBER
        DO 130 IT=1,NT
        DO 125 IS=1,3
            IF(JSIDE(IS,IT).LE.0) GO TO 125
C   COUNT OCCURRENCE OF SIDE IS,IT
            ICOUNT=0
            DO 115 ITD=IT,NT
            DO 110 ISD=1,3
                IF(ITD.EQ. IT .AND. ISD.LE. IS) GO TO 110
                IF(JSIDE(ISD,ITD).NE. JSIDE(IS,IT)) GO TO 110
                ICOUNT=ICOUNT +1
                JSIDE(IS,IT)=-ITD
                JSIDE(ISD,ITD)=-IT
            GO TO 120
110 CONTINUE
115 CONTINUE
120 IF(ICOUNT .GT. 1) CALL ERROR
        IF(ICOUNT .EQ. 0) JSIDE(IS,IT) =0
125 CONTINUE
130 CONTINUE
C   RENUMBER TRIANGULAR ELEMENTS
        DO 135 IT=1,NT
C   ION(IT)=0 IMPLIES TRIANGLE IT IS NOT YET ASSIGNED
            ION(IT)=0
135 CONTINUE
C
        NEW(1)=ISTART
        ION(ISTART)=1
        INEW=1
        ICUR=1
140 IT=NEW(ICUR)
        DO 145 IS=1,3
            ITNEXT=-JSIDE(IS,IT)
            IF(ITNEXT.LE. 0) GO TO 145
            IF(ION(ITNEXT).GT. 0) GO TO 145
            INEW=INEW+1
            NEW(INEW)=ITNEXT
            ION(ITNEXT)=1
            IF(INEW.GE. NT) GO TO 150
145 CONTINUE
            ICUR=ICUR+1
            GO TO 140
C   RENUMBERING OF TRIANGLES COMPLETE IN NEW
150 CONTINUE
C   RENUMBER NODES
        DO 155 I=1,N
C   HERE ION(I)=0 IMPLIES NEW NODE NOT SET
            ION(I)=0
155 CONTINUE
            JN=0
            DO 165 ITN=1,NT
                IT=NEW(ITN)
                DO 160 ID=1,3
                    NXTNOD=NODES(ID,IT)
                    IF(ION(NXTNOD).GT. 0) GO TO 160
                    JN=JN+1
C   SET NEW NODE NUMBER IN ION(OLDNODE)
                ION(NXTNOD)=JN
                IF(JN.GT. N) CALL ERROR

```

```

160 CONTINUE
165 CONTINUE
    NODTOT=JN
C   RENUMBERING NODES COMPLETE
    DO 170 I=1,N
        NUNODE=IGN(I)
        IF(NUNODE .LE. 0) GO TO 170
        XNEW(NUNODE)=X(I)
        YNEW(NUNODE)=Y(I)
170 CONTINUE
C   REASSIGN NODE NUMBERS FOR TRIANGLES
    MBW=0
    DO 185 ITD=1,NT
        IT=NEW(ITD)
        DO 175 IND=1,3
            IN=NODES(IND,IT)
            NUN=IGN(IN)
            IF(NUN .LE. 0) CALL ERROR
            JSIDE(IND,ITD)=NUN
175 CONTINUE
        DO 180 IND=1,3
            N1=IND
            N2=MOD(IND,3)+1
            NODE1=JSIDE(N1,ITD)
            NODE2=JSIDE(N2,ITD)
            MBW=MAX0(MBW,IABS(NODE2-NODE1))
180 CONTINUE
185 CONTINUE
    TYPE 8000,ISTART,MBWOLD,MBW
8000 FORMAT(' STARTING ELT  OLD/  NEW BAND WIDTH',3I5)
    RETURN
    END

```

```

SUBROUTINE ERROR
TYPE      8000
8000 FORMAT(' AN ERROR HAS OCCURRED')
RETURN
END

```

```

FUNCTION KSIDE(N1,N2,NN)
IF(N1 .GT. N2) GO TO 100
I=N1
J=N2
GO TO 105
100 I=N2
    J=N1
105 KSIDE=J+(I-1)*NN
RETURN
END

```

```

SUBROUTINE KOUPLK(NSIX,NN,X,Y,NT,NVERT,ION,ISAVS,
1 NNM,XNM,YNM,ICOD,NOOSIX,ISEG,IBC,LOC)
  DIMENSION X(NN),Y(NN),ION(NN),ISAVS(3,NT),
1 NVERT(3,NT),XNM(NSIX),YNM(NSIX),ICOD(NSIX),
2 NODSIX(6,NT),ISEG(2,NSIX),IBC(NSIX),LOC(NSIX)
C      RENUMBER NODES INCLUDING MID POINTS
      DO 100 IN=1,NN
100  ION(IN)=0
      DO 110 IT=1,NT
      DO 105 IN=1,3
        N1=IN
        NOD1=NVERT(N1,IT)
        N3=MOD(N1,3)+1
        NOD3=NVERT(N3,IT)
        ISIDE=KSIDE(NOD1,NOD3,NN)
        ISAVS(IN,IT)=-ISIDE
105  CONTINUE
110  CONTINUE
      NNM=0
      DO 145 IT=1,NT
      ISIX=0
      DO 140 IN=1,3
        N1=IN
        NOD1=NVERT(N1,IT)
        N3=MOD(N1,3)+1
        NOD3=NVERT(N3,IT)
        IF(ION(NOD1).GT.0)GO TO 115
        NNM=NNM+1
        XNM(NNM)=X(NOD1)
        YNM(NNM)=Y(NOD1)
        ICOD(NNM)=3
        ION(NOD1)=NNM
C      NNM IS THE TOTAL NUMBER OF NODES AND MIDPOINTS
C      ION IS ZERO UNTIL SET TO NEW NODE NUMBER FOR VERTICES
C      ISAVS IS NEGATIVE UNTIL SET TO NEW NODE NUMBER FOR MIDPOINT
115 IF(ISAVS(IN,IT).GT.0)GO TO 135
      ISIDE=ISAVS(IN,IT)
      NNM=NNM+1
      ISAVS(IN,IT)=NNM
      DO 125 ITD=IT,NT
      IF(ITD.EQ.IT)GO TO 125
      DO 120 IND=1,3
      IF(ISAVS(IND,ITD).NE.ISIDE)GO TO 120
      ISAVS(IND,ITD)=NNM
      GO TO 130
120 CONTINUE
125 CONTINUE
130 ICOD(NNM)=2
      XNM(NNM)=(X(NOD1)+X(NOD3))/2.0
      YNM(NNM)=(Y(NOD1)+Y(NOD3))/2.0
135 ISIX=ISIX+1
      NODSIX(ISIX,IT)=ION(NOD1)
      ISIX=ISIX+1
      NODSIX(ISIX,IT)=ISAVS(IN,IT)
140 CONTINUE
      TYPE 8000,IT,(NODSIX(IDUM,IT),IDUM=1,6)
8000 FORMAT(' IT,NODSIX',7I5)
145 CONTINUE
      TYPE 8005,(IDUM,XNM(IDUM),YNM(IDUM),IDUM=1,NNM)
8005 FORMAT(I5,2F10.5)
C      SET BOUNDARY NODES IN SEGMENT ARRAY ISEG
      NSEG=0
      DO 165 IT=1,NT

```

```

DO 160 IN=1,3
MID=ISAVS(IN,IT)
DO 155 ITD=1,NT
IF(ITD.EQ.IT)GO TO 155
DO 150 IND=1,3
IF(ISAVS(IND,ITD).EQ.MID)GO TO 160
150 CONTINUE
155 CONTINUE
N1=IN
NOD1=NVERT(N1,IT)
N3=MOD(N1,3)+1
NOD3=NVERT(N3,IT)
NSEG=NSEG+1
ISEG(1,NSEG)=ION(NOD1)
ISEG(2,NSEG)=MID
NSEG=NSEG+1
ISEG(1,NSEG)=MID
ISEG(2,NSEG)=ION(NOD3)
160 CONTINUE
165 CONTINUE
C          SORT SEGMENTS INTO CONTIGUOUS SEGMENTS
C          CURRENT SEGMENT STORED IN NC
NC=1
170 N2=ISEG(2,NC)
NC1=NC+1
DO 175 NS=NC1,NSEG
IF(ISEG(1,NS).NE.N2)GO TO 175
ITEMP1=ISEG(1,NC1)
ITEMP2=ISEG(2,NC1)
ISEG(1,NC1)=ISEG(1,NS)
ISEG(2,NC1)=ISEG(2,NS)
ISEG(1,NS)=ITEMP1
ISEG(2,NS)=ITEMP2
GO TO 180
175 CONTINUE
180 NC=NC1
IF(NC.LT.NSEG)GO TO 170
IF(ISEG(2,NSEG).NE.ISEG(1,1))CALL ERROR
DO 185 NS=1,NSEG
TYPE 8010,NS,ISEG(1,NS),ISEG(2,NS)
8010 FORMAT(' BOUNDARY',3I5)
IBC(NS)=0
NOD=ISEG(1,NS)
LOC(NOD)=NS
185 CONTINUE
C          READ STARTING AND ENDING NODES FOR STATIONARY BC
190 TYPE 8015
8015 FORMAT(' TYPE NODE1  NODE2 CCW FOR FIXED BC')
ACCEPT *,N1,N2
IF(N1.LE.0)GO TO 200
LOC1=LOC(N1)
LOC2=LOC(N2)
IF(LOC2.LT.LOC1)LOC2=LOC2+NSEG
DO 195 LS=LOC1,LOC2
NS=MOD(LS-1,NSEG)+1
NUMNP=ISEG(1,NS)
XVEL=0.0
YVEL=0.0
ETA=0.0
IBC(NS)=1
TYPE 8020,NUMNP,XVEL,YVEL,ETA
8020 FORMAT(' NODE',I5,' BC',3F10.5)
195 CONTINUE

```



```

      GO TO 190
C      IDENTIFY REMAINING BOUNDARY NODES
200 DO 205 NS=1,NSEG
    IF(IBC(NS).GT.0)GO TO 205
    NOD=ISEG(1,NS)
    TYPE 8025,NOD,XNM(NOD),YNM(NOD)
8025 FORMAT(' NO BC FOR NODE', I5,' AT X,Y',2F10.5)
205 CONTINUE
C      OUTPUT FILE FOR RIVINCA
    IDATA2=10
    WRITE(IDATA2,8030) NT,NNM,NSEG
8030 FORMAT(3I5)
    DO 210 IP=1,NNM
    WRITE(IDATA2,8035) IP,XNM(IP),YNM(IP),ICOD(IP)
8035 FORMAT(I5,2F10.4,I5)
210 CONTINUE
    DO 220 NS=1,NSEG
    NUMNP = ISEG(1,NS)
    IF(IBC(NS) .EQ. 1) GO TO 215
    WRITE(ICATA2,8040) NUMNP
8040 FORMAT(I5,3F10.3)
    GO TO 220
215 WRITE(IDATA2,8040) NUMNP,XVEL,YVEL
220 CONTINUE
    DO 225 IT=1,NT
    WRITE(IDATA2,8045) (NODSIX(IDUM,IT),IDUM=1,6)
8045 FORMAT(6I5)
225 CONTINUE
    RETURN
    END

```

INTERNAL DISTRIBUTION

1. M. L. Ballou	26-31. M. R. Patterson
2-6. J. L. Bledsoe	32. H. Postma
7. C. R. Boston	33. R. G. S. Rao
8. A. A. Brooks	34-72. T. H. Row
9. J. B. Cannon	73. W. P. Staub
10. H. P. Carter/CSD X-10 Library	74. E. K. Triegel
11. J. S. Crowell	75. P. R. Vanstrum
12. A. H. Eraslan	76. G. T. Yeh
13. S. K. Fischer	77. A. J. Witten
14. W. Fulkerson	78. H. E. Zittel
15. R. F. Hibbs	79-80. Central Research Library
16. J. T. Holdeman	81. Document Reference
17. S. K. Iskander	Section Y-12
18. M. E. LaVerne	82-84. Laboratory Records
19. D. W. Lee	85. Laboratory Records - RC
20. E. C. Long	86. ORNL Patent Office
21-25. H. A. McLain	87. RSIC

EXTERNAL DISTRIBUTION

88. P. Crooke, Department of Mathematics, Vanderbilt University, Nashville, TN.
89. Chief, Mathematics and Geoscience Branch, Department of Energy, Washington, DC 20545.
90. J. E. Flaberty, Rensselaer Polytechnic Institute, Department of Applied Mathematics, Amos Eaton 407, Troy, NY 12181.
91. C. Kleinstreuer, Rensselaer Polytechnic Institute, Department of Chemical and Environmental Engineering, Richetts 205, Troy, NY 12181.
92. Prof. F. L. Parker, Environmental and Water Resources Engineering, Vanderbilt University, Nashville, TN.
93. M. Reeves, INTERA Environmental Consultants, Inc., 1201 Dairy Ashford, Suite 200, Houston, TX 77079.
94. J. N. Rogers, Division 8324, Sandia Laboratories, Livermore, CA 94550.
95. Assistant Manager, Energy Research and Development, DOE/ORO.
- 96-122. Technical Information Center, Department of Energy, Oak Ridge, TN 37830.