NUREG/CR-4042

A 3-Dimensional Computer Model to Simulate Fluid Flow and Contaminant Transport Through a Rock Fracture System

Prepared by C. Huang, D. D. Evans

Department of Hydrology and Water Resources
University of Arizona

Prepared for
U.S. Nuclear Regulatory
Commission

NOTICE

Availability of Reference Materials Cited in NRC Publications

Most documents cited in NRC publications will be available from one of the following sources:

1. The NRC Public Document Room, 1717 H Street, N.W.
   Washington, DC 20555
2. The NRC/GPO Sales Program, U.S. Nuclear Regulatory Commission,
   Washington, DC 20555
3. The National Technical Information Service, Springfield, VA 22161

Although the listing that follows represents the majority of documents cited in NRC publications, it is not intended to be exhaustive.

Referenced documents available for inspection and copying for a fee from the NRC Public Document Room include NRC correspondence and internal NRC memoranda; NRC Office of Inspection and Enforcement bulletins, circulars, information notices, inspection and investigation notices; Licensee Event Reports; vendor reports and correspondence; Commission papers; and applicant and licensee documents and correspondence.

The following documents in the NUREG series are available for purchase from the NRC/GPO Sales Program: formal NRC staff and contractor reports, NRC-sponsored conference proceedings, and NRC booklets and brochures. Also available are Regulatory Guides, NRC regulations in the *Code of Federal Regulations*, and *Nuclear Regulatory Commission Issuances*.

Documents available from the National Technical Information Service include NUREG series reports and technical reports prepared by other federal agencies and reports prepared by the Atomic Energy Commission, forerunner agency to the Nuclear Regulatory Commission.

Documents available from public and special technical libraries include all [illegible] literature items, such as books, journal and periodical articles, and transactions. *Federal Register* notices, federal and state legislation, and congressional reports can usually be obtained from these libraries.

Documents such as theses, dissertations, foreign reports and translations, and non-NRC conference proceedings are available for purchase from the organization sponsoring the publication cited.

Single copies of NRC draft reports are available free, to the extent of supply, upon written request to the Division of Technical Information and Document Control, U.S. Nuclear Regulatory Commission, Washington, DC 20555.

Copies of industry codes and standards used in a substantive manner in the NRC regulatory process are maintained at the NRC Library, 7920 Norfolk Avenue, Bethesda, Maryland, and are available there for reference use by the public. Codes and standards are usually copyrighted and may be purchased from the originating organization or, if they are American National Standards, from the American National Standards Institute, 1430 Broadway, New York, NY 10018.

GPO Printed copy price: $5.00

NUREG/CR-4042
RW

A 3-Dimensional Computer Model to Simulate Fluid Flow and Contaminant Transport Through a Rock Fracture System

Manuscript Completed: October 1984
Date Published: January 1985

Prepared by
C. Huang, D. D. Evans

Department of Hydrology and Water Resources
University of Arizona
Tucson, AZ 85721

T. J. Nicholson, NRC Project Manager

Prepared for
Division of Radiation Programs and Earth Sciences
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, D.C. 20555
NRC FIN B7291
Under Contract No. NRC-04-81-224

ABSTRACT

A 3-dimensional fracture generating scheme is presented which can be used to simulate water flow and contaminant (solute) transport through fracture system of a rock. It is presently limited to water saturated conditions, zero permeability for the rock matrix, and steady state water flow, but allows for transient solute transport. The scheme creates finite planar plates of uniform thickness which represent fractures in 3-dimensional space. A given fracture (plate) has the following descriptors: center location, orientation, shape, areal extent and aperture. Each parameter can be described by an appropriate probability distribution. Individual fractures are generated to form an assemblage of a certain fracture density. All fracture intersections and boundary/fracture intersections are determined and deadend fractures are eliminated. Flow through the fracture assemblage is considered laminar and described by Poiseuille's law. The principle of mass conservation at each intersection is used to develop the global matrix equation, which is solved subject to specified boundary conditions to yield the head and flow distribution at each intersection. Solute transport is considered to be advective between intersections with complete mixing at each intersection. Solutes added to the flow system can be explicitly followed and concentration vs. time relationships can be determined anywhere in the system. Some examples are included.

TABLE OF CONTENTS

Page

ABSTRACT iii
LIST OF FIGURES vii

EXECUTIVE SUMMARY 1

1. INTRODUCTION 3

2. CODE DEVELOPMENT 4

2.1 Generation of the Fracture Network 4
2.2 Isolation of Samples 12
2.3 Design of Flow Experiments 15
2.4 Design of Mass Transport Experiments 20

3. EXAMPLE OF EXPERIMENTS 23

4. DISCUSSION 23

APPENDIX A. PROGRAM LISTING 26

A.1 Coordinate System Used in the Model 26

A.1.1 The Global Coordinate System 26
A.1.2 The Sample Coordinate System 26
A.1.3 The Fracture Local Coordinate System 26

A.2 Arguments Used in the Main Program 27

A.3 Program Listing of Experiment 1 32

A.4 Sample Input/Output for Experiment 1 77

A.5 Program Listing of Part 1 of Experiment 2 79

A.6 Program Listing of Part 2 of Experiment 2 87

A.7 Sample Input/Output of Experiment 2 105

LIST OF FIGURES

Figure		Page
1	Code subroutines and designated functions.	5
2	A finite fracture in 3-dimensional space where R_o is the center of the fracture and R_oN is the normal vector	6
3	The definition of two angles of rotation, α and β, used in specifying the orientation of the normal vector	6
4	The definition on an in-plane rotational angle, γ, used in specifying the orientation of a local coordination system (x_1, y_1)	8
5	The intersecting line L_{12} (represented by two distinct points T_1 and T_2) between two infinite planes encompassing Fractures 1 and 2 intersects: (a) only one fracture; and (b) neither fracture	13
6	The intersecting line L_{12} (represented by two distinct points T_1 and T_2) between two infinite planes encompassing Fractures 1 and 2, intersecting both fractures, but (a) does not have a finite section common to both fractures; and (b) has a common segment	13
7	The global fracture network generating coordinate system (X_G, Y_G, Z_G) and the sample coordinate system (X_S, Y_S, Z_S)	13
8	Diagram of one fracture intersected by two other fractures to illustrate the basic flow equation between fracture intersects.	16
9	A sample fracture network to illustrate the derivation of the global mass balance equation	16
10	A sample flow network showing inflow and outflow components at intersection X	19
11	A cubic block sample of size L isolated from the global fracture region	22
12	Schematic drawing of Experiment 1 showing a rock sample submerged in water with the top surface pressurized to a pressure head equal to the size of the block	23
13	Schematic drawing of Experiment 2.	24

A 3-DIMENSIONAL COMPUTER MODEL TO SIMULATE FLUID FLOW AND CONTAMINANT TRANSPORT THROUGH A ROCK FRACTURE SYSTEM

EXECUTIVE SUMMARY

A 3-dimensional fracture generating scheme has been developed to simulate fracture systems in a rock mass. Numerical experiments can be performed on the generated fracture system to examine its hydraulic and contaminant transport characteristics. The computer code is now limited to a fracture system filled with a single fluid, to an impermeable medium between fractures, and to steady state flow conditions. The code is being extended to simulate unsaturated fractured rock and where the rock matrix has a finite permeability.

The major purpose of performing numerical experiments on artificially generated fracture networks is to improve the understanding of flow and contaminant movement through a fractured medium, such as that surrounding a waste repository. Because the geometry of the fracture network determines flow paths and relative travel times for contaminants, recreating these networks with a computer allows the analysis of the isolation capabilities of a specified geologic environment. These types of analyses can also provide assessment of various measurement techniques in relation to effective sample size of measurement. When used in conjunction with a field testing program, the model is useful for planning, performance and analysis of experimental results.

Discrete fracture generators described in the literature are limited to 2-dimensional systems. In such a network, a fracture is represented by a straight line of infinite extent in the third dimension. A major criticism of these 2-dimensional models is the constraint of flow in only two dimensions. The problems associated with using a 2-dimensional model are even more severe when simulating contaminant transport. In some cases, the regional flow system can be viewed as 1-dimensional or 2-dimensional, but the movement of contaminants through a porous medium is always 3-dimensional. This is demonstrated by the advection-dispersion phenomena. Any study of dispersion in a 3-dimensional fracture system using a 2-dimensional model will lead to moot results. Thus the development of a 3-dimensional fracture network generator is motivated by the need to more accurately model contaminant flow.

The formulated fracture generator scheme creates finite planar fractures in 3-dimensional space. In this model, a fracture is simulated by a plate of uniform thickness. Thus the thickness of the plate represents the apparent hydraulic aperture of a fracture. To define a finite plate (or fracture) in 3-dimensional space, the following geometric input parameters are required: location, orientation, shape, areal extent and thickness. In addition, the density of the fractures (the number of fractures per unit volume) is required to complete the network. Each of these parameters can be described by an appropriate probability distribution. These

probability distributions can be independent of each other or correlated. Thus one realization of a network is the assemblage of a certain number of randomly generated finite plates of which the geometric parameters describing each individual plate are sampled from its respective probability distribution.

To generate a discrete network, a global coordinate system is first defined. The location parameters specify the center of the plate in global coordinates. Two angles are required to specify the orientation of a surface in the global, 3-dimensional space. The shape of each fracture can be circular, rectangular, elliptical, or any other computable shape provided that the boundaries can be expressed by analytic functions. Areal extent is given by the length scale associated with the specific shape, e.g., a radius defines a circular disk. Within each plate, a local 2-dimensional coordinate system is established. One additional angle is required to denote the rotation of the local 2-dimensional coordinate system when a non-circular shaped fracture is used in the network.

Once the desired number of plates have been generated, the program will define all intersections and delete isolated fractures (those that do not connect to any boundary or other fractures). The intersection of two finite surfaces is a finite line. Because the flow takes place from fracture to fracture through the intersections, a fracture must be intersected by at least two others to become conductive. Thus dead-end fractures (those only intersecting one other fracture) also need to be identified.

This program can also be used to isolate an arbitrarily oriented sub-volume of sample at any location within the global network. For this isolated sample, all information regarding boundary and interior intersections is explicitly defined. This procedure simulates the sampling process in actual field conditions.

Once a realization of the fracture network has been formed and a sub-volume of sample has been isolated and defined, experiments can be performed on the sample to study flow and transport characteristics.

Although the simulated fracture network is in 3-dimensional space, flow within each fracture is considered 2-dimensional in the local coordinate system defined for each individual fracture. The actual flow behavior between intersections within each fracture is an area of research in itself. For now a simplified flow condition is used. This restricton can be relaxed to incorporate a more complicated flow analysis within each fracture when a procedure becomes available.

The fractures are idealized as smooth parallel plates, and fluid flow in the fracture is assumed laminar and described by Poiseuille's law. Thus, the hydraulic conductivity and the flow region between all insections for each fracture can be determined. This simplified treatment allows the representation of a line intersection by a nodal point at its center. The principle of mass conservation, which ensures the net flux through each internal intersection (or nodal point) is zero, is used to assemble the global matrix equation. When proper boundary conditions are imposed on the external surfaces, the global matrix equation is solved and the pressure (or head) distribution within the sample is defined.

Once the flow domain is defined, the analysis is carried one step further to study mass transport characteristics. Mass transport is assumed advective between intersections with complete mixing at each intersection. The flow analysis provides a complete set of information in terms of velocity, travel time and flux between every two intersections. Thus, at any nodal point, the tree-type network structure of the inflow is defined. This enables the examination of all possible flow routes in the network.

A slug (either continuous or finite in time) of tracer introduced into the system can be explicitly followed. This technique, which is different from conventional particle-tracking techniques, provides an explicit concentration vs. time relationship anywhere in the system without any probabilistic influence or limitation due to space/time discretization.

Several experiments have been designed to study the scale effects and flow/transport properties of a fracture network. The results of these experiments are not included in this report but will be presented in technical papers. The purpose of this report is to present the details of the computer code for the fracture generator scheme and for some examples of numerical experiments using generated fracture systems.

1. INTRODUCTION

A 3-dimensional fracture generator scheme has been formulated and used to study fluid flow and contaminant transport characteristics of rock fracture systems. In this paper, the detailed development of the computer code and examples of flow and mass transport numerical experiments are presented.

Examination of flow and contaminant transport through the use of artificially-generated fracture systems provides insight and improved understanding of these processes in real fracture systems. Also, such examinations can provide useful information on sample size and measurement techniques for field assessments. With sufficient and reliable fracture data on a real rock system, the real system may be adequately simulated to yield useful estimates of contaminant travel times and release rates to a location at some distance from a given source.

Discrete fracture generators described in the literature are limited to 2-dimensions and use oversimplifing assumptions. For a 2-dimensional fracture network generator, a fracture is represented by a straight line which is of infinite extent in the third dimension. This assumption results in a no flow condition in the third direction which is unrealistic for real fracture systems. For problems involving the simulation of mass transport, a 2-dimensional network model creates even more severe departures from expected values. While pressure heads might be adequately modelled using a 1- or 2-dimensional network, the dimensionality of contaminant transport models can not be so reduced. It is a moot point whether a 2-dimensional representation of dispersion in a discrete fracture system will yield accurate results. To understand transport mechanisms, there exists an obvious need for the ability to model solute transport in 3-dimensional space.

In general, the code presented here generates planar fractures located in a 3-dimensional region. The generated fracture system is referred to as a global fracture network. A subvolume sample can be isolated from the global network and subjected to pre-designed fluid flow and mass transport experiments. Results of these experiments can then be used in the interpretation of the hydraulic and mass transport characteristics of the fracture network.

The main logic of program development can be divided into four categories: 1) generation of the fracture network; 2) isolation of the study sample; 3) design and performance of flow experiments; and 4) design and performance of transport experiments.

The code is written in standard ANSI FORTRAN-77 and has been used on the CDC Cyber-175 and 6600/7600 series computer systems. A program listing is presented in Appendix A.

2. CODE DEVELOPMENT

The 3-D fracture generator and test codes consist of an assembly of several major driving subprograms to perform different tasks. The major functions of these driving subprograms are listed in Figure 1.

To minimize storage requirements, subroutines 1 - 8 are called under one main program to set up the fracture network and perform the hydraulic tests, while subroutines 9-10 are called under a second main program to perform the mass transport experiments. A temporary data file is used to transfer information from the first to the second program.

2.1 Generation of Fracture Network

In the fracture generator, a fracture is represented by a 2-dimensional finite plate of a fixed thickness. A simulated fracture network is a collection of a selected number of individual finite plates in a 3-dimensional volume. The spaces bounded by these fracture planes are considered to be an impermeable rock mass.

To define each finite planar fracture in the 3-dimensional volume, the following geometric parameters are required: 1) center location, 2) orientation, 3) shape and areal extent, and 4) aperture. Figure 2 illustrates how a fracture is defined in the global region. In Figure 2, the center of the fracture is designated by point R_o having a

Major Subroutine	Function
1. INPFRCT	Generate geometric parameters for each fracture plane.
2. SUBFRCT	Find line intersections between fractures.
3. INPCUT	Generate geometric parameters for boundary surfaces of rectangular block of samples.
4. SUBCUT	Find line intersections between fracture and boundary surface.
5. SUBDEAD	Delete dead end fractures from the network.
6. SUBBOND	Assign pressure boundary conditions on boundary surfaces.
7. SUBMTRX	Assemble the global matrix equation and find the head distribution.
8. SUBMSBL	Calculate fluxes across boundary surfaces.
9. FLOWNET	Create flow network to be used in TRACER subroutine.
10. TRACER	Generate breakthrough curves for mass transport.

Figure 1. Code subroutines and designated functions.

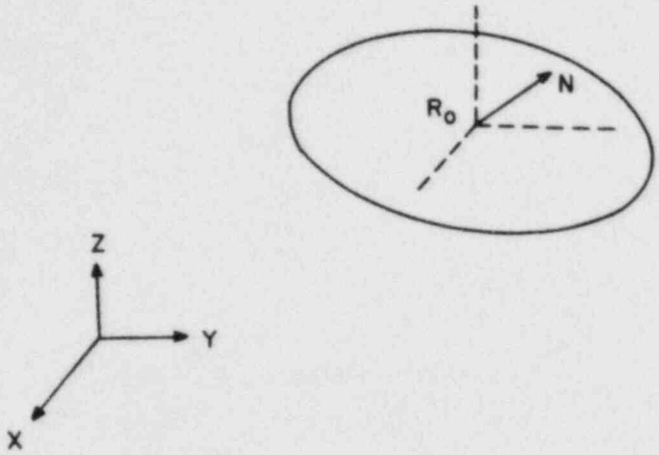

Figure 2. A finite fracture in 3-dimensional space where R_o is the center of the fracture and R_oN is the normal vector.

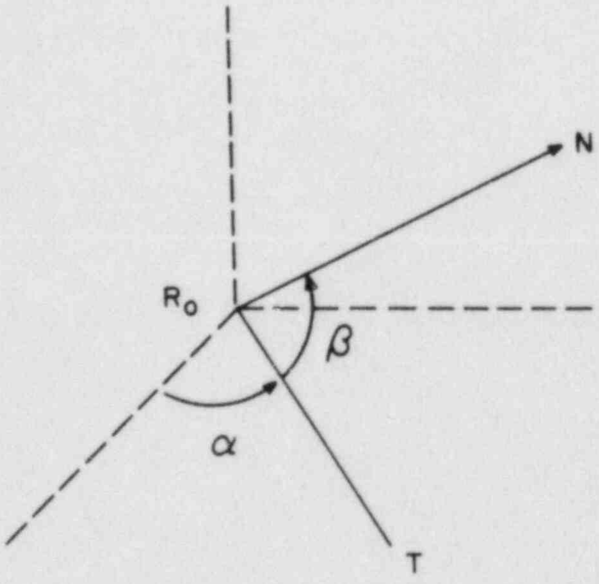

Figure 3. The definition of two angles of rotation, α and β, used in specifying the orientation of the normal vector.

global coordinate of (X_o, Y_o, Z_o). The orientation of the plane is defined by two angles of rotation, α and β (Figure 3). In fact, α and β are used in specifying the orientation of a vector located at point R_o and normal to the fracture plane. α is the horizontal angle measured counterclockwise from the +X axis. β is the elevation angle in the plane with R_oT and measured from the XY plane. The equation of an infinite plane encompassing the finite fracture is:

$$aX + bY + cZ = aX_o + bY_o + cY_o \tag{1}$$

where $a = \cos\beta\cos\alpha$

$b = \cos\beta\sin\alpha$

$c = \sin\beta$.

To define a fixed shaped finite region in an infinite plane, a local coordinate system is needed (Figure 4). This local coordinate system is 2-dimensional with the origin at the center of the fracture. The local coordinate system (x',y') is selected such that its axes coincide with the characteristic axes of a particular shaped fracture. The local coordinate system is specific to each fracture plane, and has to be defined for every fracture. One additional angle, γ, is required to define the orientation of the local (x',y') coordinates. This additional angle is not necessary when a circular shaped fracture is generated due to its axisymmetrical nature.

In Figure 4, since R_oN is a normal vector of the ι ure plane, a vector origin at R_o having a horizontal angle of α and vertical angle of β -90 must lie on the fracture plane. Let this vector be denoted as R_oX. This temporary vector R_oX is used as a reference to define the orientation of a local (x',y') system. The +x' axis of the local x',y' coordinate system is rotated by γ angle measured counterclockwise from the R_oX temporary axis. The transformation between the global network generating coordinate system (X,Y,Z) and the local fracture based coordinate system (x',y') is given below:

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix} = \begin{Bmatrix} X_o \\ Y_o \\ Z_o \end{Bmatrix} + [A]\ [B] \begin{Bmatrix} x' \\ y' \end{Bmatrix} \tag{2}$$

where

$$[A] = \begin{bmatrix} \sin\beta\cos\alpha, & -\sin\alpha \\ \sin\beta\cos\alpha, & \cos\alpha \\ -\cos\beta, & 0 \end{bmatrix} \tag{2a}$$

and

$$[B] = \begin{bmatrix} \cos\gamma, & -\sin\gamma \\ \sin\gamma, & \cos\gamma \end{bmatrix} \tag{2b}$$

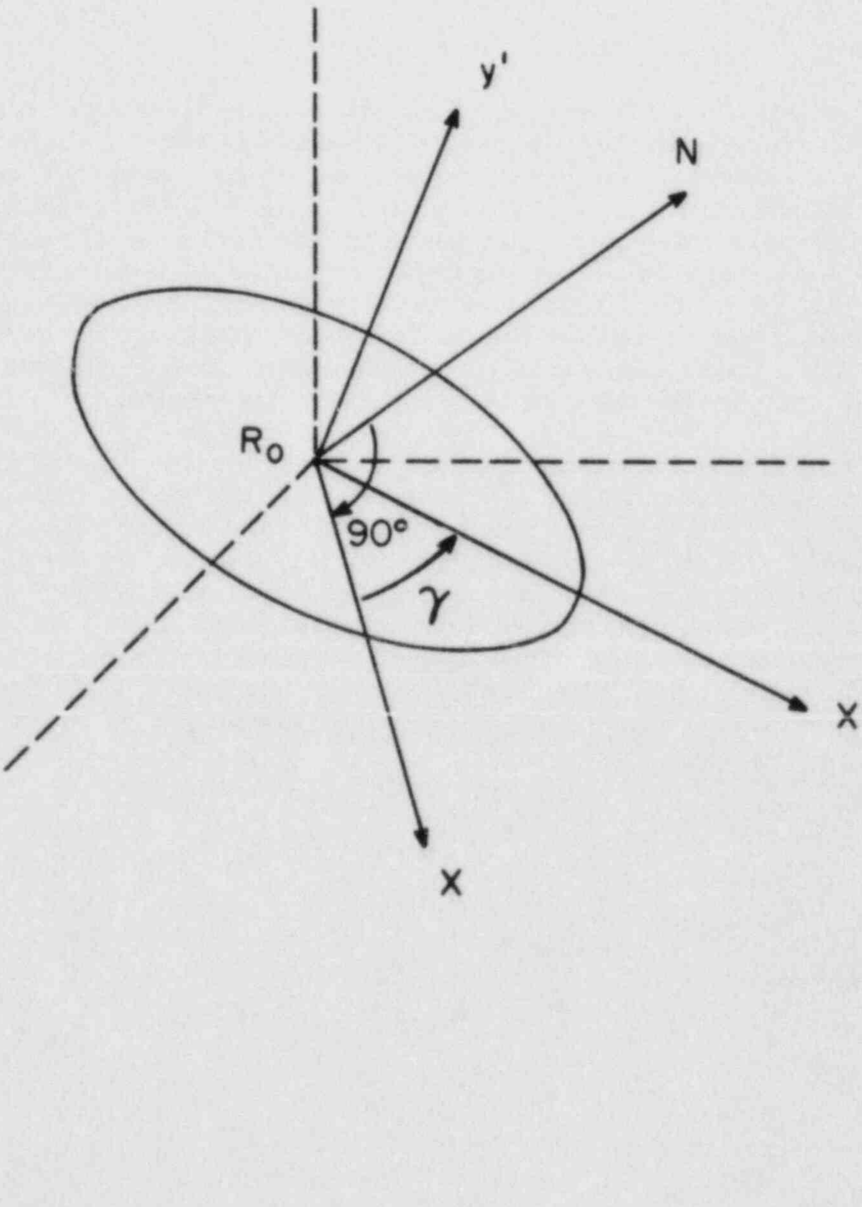

Figure 4. The definition on an in-plane rotational angle, γ, used in specifying the orientation of a local coordination system (x', y')

Once the local coordinate system is established, the bounded regions of some regular shapes can be defined easily. For example, equation 3 defines the region of an ellipse and equation 4 defines the region of a rectangle.

$$\frac{(x')^2}{r_1^2} + \frac{(y')^2}{r_1^2} \leq \qquad (3)$$

$$x' \leq r_1 \text{ and } y' \leq r_2. \qquad (4)$$

Note that a circle and a square are special cases of an ellipse and a rectangle.

The areal extent of a particular shaped fracture is defined by characteristic lengths associated with that shape. The r_1 and r_2 are the characteristic lengths for an ellipse and a rectangle but they bear different meanings for different shapes. The present computer program can handle elliptical and rectangular shaped fractures. Extension to other shapes are straightforward as long as the boundaries can be expressed by analytical functions.

A single value is assigned to each fracture to represent the aperture. Although the aperture is variable in a natural setting, this single value represents an averaged equivalent hydraulic aperture.

Field observations have suggested that some geometric parameters may follow certain statistical distributions. Subroutine INPFRCT uses random number generators of different distributions to create a fracture network. It has four random number generating subroutines to create normal, log-normal, uniform, and exponential distributions.

Once a set of individual fractures is generated, subroutine SUPFRC is called to find the line intersects among the fractures. The procedure of finding the intersecting finite line segments between two finite planes consists of two steps: 1) find the intersecting line between two infinite planes containing these two finite fractures, and 2) truncate the infinite line to a finite segment such that it is common to both finite fractures.

Let equation 5 be the equation of an infinite plane containing finite fracture 1:

$$a_1X + b_1Y + c_1Z = d_1 \qquad (5)$$

Similarily, equation 6 describes a second plane containing fracture 2:

$$a_2X + b_2Y + c_2Z = d_2 \qquad (6)$$

The equation of a line, L_{12}, common to both planes is given as:

$$\bar{R} = \bar{R}_1 t + \bar{R}_2 \tag{7}$$

or

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \end{Bmatrix} t + \begin{Bmatrix} X_2 \\ Y_2 \\ Z_2 \end{Bmatrix} \tag{7a}$$

where t is a scaler.

If $e = c_2b_1 - c_1b_2 \neq 0$, then

$$\bar{R}_1 = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \end{Bmatrix} = \begin{Bmatrix} 1 \\ (a_2c_1 - a_1c_2)/e \\ (b_2a_1 - b_1a_2)/e \end{Bmatrix} \tag{8a}$$

and

$$\bar{R}_2 = \begin{Bmatrix} X_2 \\ Y_2 \\ Z_2 \end{Bmatrix} = \begin{Bmatrix} 0 \\ (c_2d_1 - c_1d_2)/e \\ (d_2b_1 - d_1b_2)/e \end{Bmatrix} \tag{8b}$$

or if $e = b_2a_1 - a_2b_1 \neq 0$.

$$\bar{R} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \end{Bmatrix} = \begin{Bmatrix} (c_2b_1 - c_1b_2)/e \\ (a_2c_1 - a_1c_2)/e \\ 1 \end{Bmatrix} \tag{9a}$$

$$\bar{R} = \begin{Bmatrix} X_2 \\ Y_3 \\ Z_2 \end{Bmatrix} = \begin{Bmatrix} (b_2d_1 - d_2b_1)/e \\ (d_2a_1 - d_1a_2)/e \\ 0 \end{Bmatrix} \tag{9b}$$

or if $e = a_2c_1 - a_1c_2 \neq 0$

$$\bar{R}_1 = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \end{Bmatrix} = \begin{Bmatrix} (b_2a_1 - b_1a_2)/e \\ 1 \\ (c_2b_1 - c_1b_2)/e \end{Bmatrix} \tag{10a}$$

$$\bar{R}_2 = \begin{Bmatrix} X_2 \\ Y_2 \\ Z_2 \end{Bmatrix} = \begin{Bmatrix} (a_2d_1 - a_1d_2)/e \\ 0 \\ (d_2c_1 - d_1c_2)/e \end{Bmatrix} \qquad (10b)$$

Once R_1 and R_2 are found, the intersecting line L_{12} can be represented by two distinct points on the line by choosing two different values of t. Procedures to truncate this line to a finite line segment common to both fractures are:

1) Transform T_1 and T_2 [Assume that points T_1 and T_2 are two distinct points on L_{12}], to local coordinates defined on fracture 1 and find the two boundary point intersects, P_{11} and P_{12}, between the line L_{12} and the boundary of fracture 1, if they exist.

2) Similar to step 1, find the two boundary points, P_{21} and P_{22}, representing the intersection between line L_{12} and fracture 2. If line P_{12} does not intersect either one of the two fracture boundaries, then the two fractures do not share a common line (Figures 5a and 5b).

3) If line L_{12} intersects both fractures, then points P_{11} and P_{12} are checked to see if they are contained within the boundary of fracture 2. Similarly, points P_{12} and P_{22} are checked on fracture 1. If the two fractures share a common line segment, then two of the four points should be common to both fracture regions (Figures 6a and 6b). These two points are two end points of the finite line segment.

A complete fracture network is defined after all fracture intersections are determined.

2.2 Isolation of Samples

After the creation of a global fracture network, a subvolume contained inside the global region can be isolated.

Subroutines INPCUT and SUPCUT are designed to perform the task of sample isolation. A rectangular block of sample is defined by its center location (CORØ), the rotation angles (COAL) and its size (COSZ). A sample coordinate system is defined at the center of the sample with an orientation that each sample boundary is perpendicular to one of the new coordinate axes. The rotation angles (COAL) are horizontal and vertical angles of rotation between the new X_s-axis of the sample coordinate system and the global X_g-axis (Figure 7).

Within an isolated sample, the finite line segments between fractures are further truncated to the boundary of the sample, these are called internal intersects. The line intersections between rectangular boundary surfaces and fracture planes are called external intersects. These internal and external intersects are expressed in sample

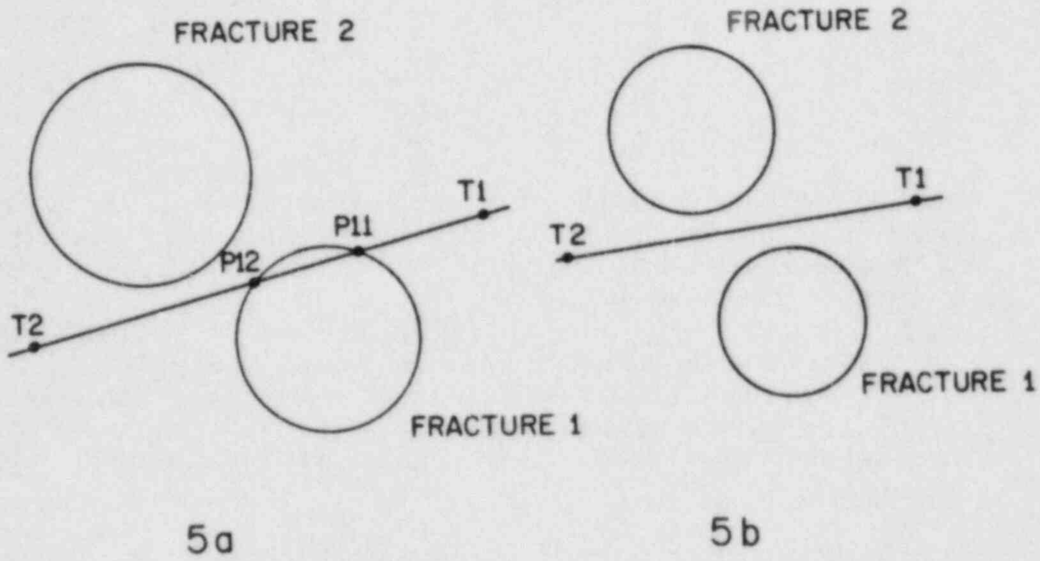

Figure 5. The intersecting line L_{12} (represented by two distinct points T_1 and T_2) between two infinite planes encompassing Fractures 1 and 2 intersects: (a) only one fracture; and (b) neither fracture.

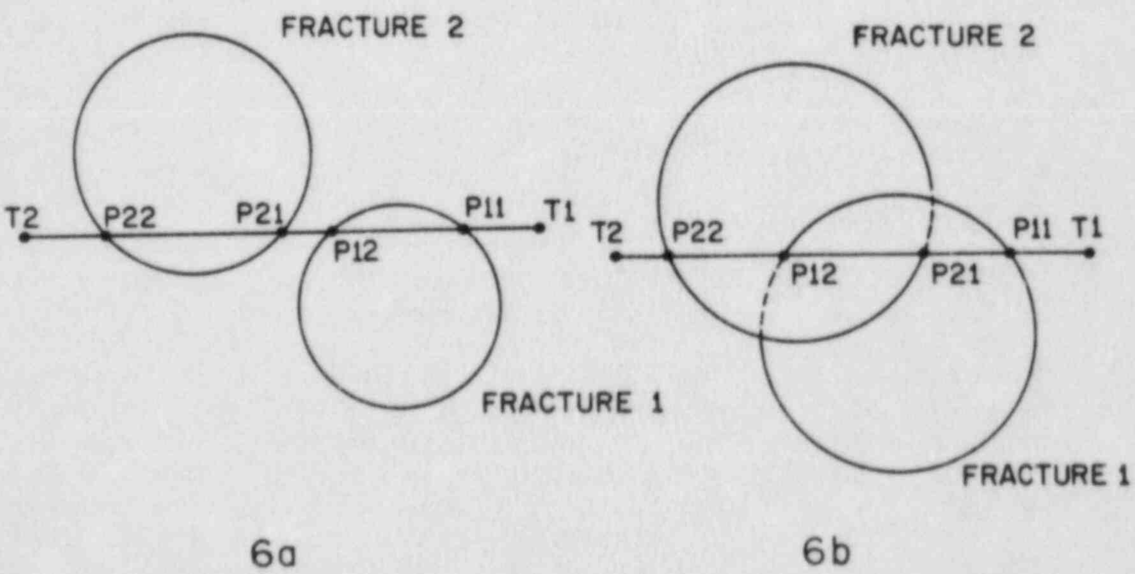

Figure 6. The intersecting line L_{12} (represented by two distinct points T_1 and T_2) between two infinite planes encompassing Fractures 1 and 2, intersecting both fractures, but (a) does not have a finite section common to both fractures; and (b) has a common segment.

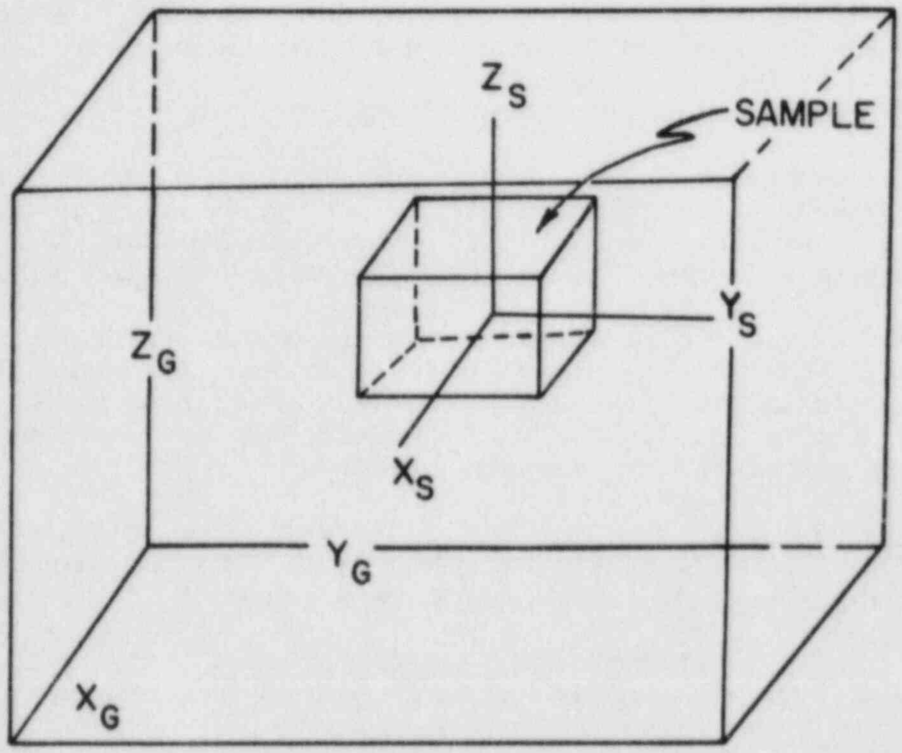

Figure 7. The qlobal fracture network generating coordinate system (X_G, Y_G, Z_G(and the sample coordinate system (X_S, Y_S, Z_S).

coordinates and the transformation from the global coordinate system to the sample coordinate system is given below:

$$\begin{Bmatrix} X_s \\ Y_s \\ Z_s \end{Bmatrix} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\alpha\cos\beta & \sin\beta \\ -\sin\alpha & \cos\alpha & 0 \\ -\cos\alpha\sin\beta & -\sin\alpha\sin\beta & \cos\beta \end{bmatrix} \begin{Bmatrix} X_g - X_o \\ Y_g - Y_o \\ Z_g - Z_o \end{Bmatrix} \quad (11)$$

where subscripts s and g, denote sample, and global coordinate systems, α and β are horizontal and vertical angles of rotation for the $+X_s$ axis of the sample coordinate system measured from the $+X_g$ axis of the global coordinate system. X_o, Y_o and Z_o are the center of the sample expressed in the global coordinate system.

2.3 Design of Flow Experiment

The formulation of flow experiments consists of the following steps: 1) assemble the flow network; 2) assign boundary conditions; 3) solve the matrix equation to obtain the pressure/hydraulic head distribution; and 4) calculate fluxes across boundary surfaces.

The flow network is assembled from individual fractures where flow takes place from one intersection to another. The actual flow pattern in an individual fracture is not well understood and is an area of research in itself. The simplification made in the present code can be relaxed when more knowledge is gained.

Figure 8 shows a fracture being intersected by two other fractures, having intersections A and B. To calculate the flow between A and B, the following simplified assumptions are used:

1) flow between A and B is laminar and uniform;
2) the pressure head along A and B is constant.

These assumptions allow us to simplify the representation of a finite line segment to a point at the center of the line and its length. The steady state flux from A to B, $F_{A,B}$, can then be calculated by:

$$F_{A,B} = \frac{gb^3}{12} \; \frac{L_A + L_B}{2} \; \frac{P_A + Z_A - P_B - Z_B}{D_{A,B}} \quad (12)$$

where ρ is fluid density, g is the gravitational constant, μ is dynamic viscosity, b is aperture, L_A and L_B are length of line segment A and B, P_A and P_B are pressure heads along A and B, Z_A and Z_A and Z_B are elevation heads at the midpoints of A and B, $D_{A,B}$ is the distance between midpoints at A and B. Equation 12 can be simplified to:

$$\begin{aligned} F_{A,B} &= C_{A,B} (H_A - H_B) \quad (13) \\ &= C_{A,B} (P_A - P_B) + C_{A,B} (Z_A - Z_B) \\ &= C_{A,B} (P_A - P_B + C_{A,B,Z}) \end{aligned}$$

where $H_A = P_A + Z_A$, $H_B = P_B + Z_B$; and $C_{A,B,Z} = C_{A,B} (Z_A - Z_B)$.

The principle of mass conservation is used to assemble the global flow network from formulations between individual intersections. Figure 9 is used to illustrate a simple flow network and is used as an example to demonstrate the formation of the global network flow equations. In Figure 9, there are 4 fractures, forming a network of 3 internal intersections and 3 external intersections. The principle of mass conservation implies that: 1) the sum of all fluxes across all boundary surfaces must be zero; and 2) the sum of all fluxes through any internal intersection must be zero.

If F_j equals the total flux across the jth intersection, and $F_{i,j}$ is a component flux of F_j representing the flux from the ith intersection to the jth intersection. The following mass balance equations are obtained:

$$F_1 = F_{2,1} + F_{3,1} + F_{4,1} = \emptyset \qquad (14a)$$

$$F_2 = F_{1,2} + F_{3,2} + F_{5,2} = \emptyset \qquad (14b)$$

$$F_3 = F_{1,3} + F_{2,3} + F_{6,3} = \emptyset \qquad (14c)$$

$$F_4 = F_{1,4} \qquad (14d)$$

$$F_5 = F_{2,5} \qquad (14e)$$

$$F_6 = F_{3,6} \qquad (14f)$$

$$F_4 + F_5 + F_6 = \emptyset \qquad (14g)$$

Expanding equation 14a, using the relationship given in equation 13, we obtain:

$$F_1 = C_{2,1}(P_2-P_1) + C_{2,1,Z} + C_{3,1}(P_3-P_1) + C_{3,1,Z} + C_{4,1}(P_4-P_1) + C_{4,1,Z} = \emptyset \qquad (15)$$

If the boundary pressure, (P_4), is known, the equation can be rearranged to contain all the known values on the right hand side:

$$[-C_{2,1}-C_{3,1}-C_{4,1}]P_1 + C_{2,1}P_2 + C_{3,1}P_3 = -C_{2,1,Z} - C_{3,1,Z} - C_{4,1}P_4 - C_{4,1,Z} \qquad (16)$$

Equations 14b and 14c can be rewritten in the same fashion to obtain:

$$C_{1,2}P_1 + (-C_{1,2} - C_{3,2} - C_{5,2})P_2 + C_{3,2}P_3 = -C_{1,2,Z} - C_{3,2,Z} - C_{5,2}P_5 - C_{5,2,Z} \qquad (17)$$

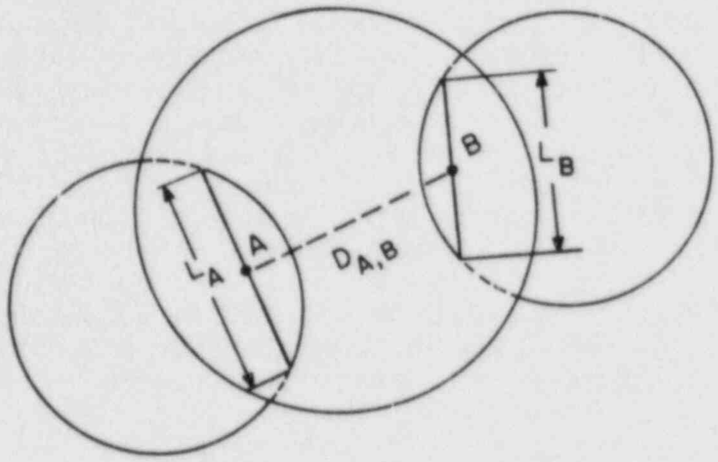

Figure 8. Diagram of one fracture intersected by two other fractures to illustrate the basic flow equation between fracture intersects.

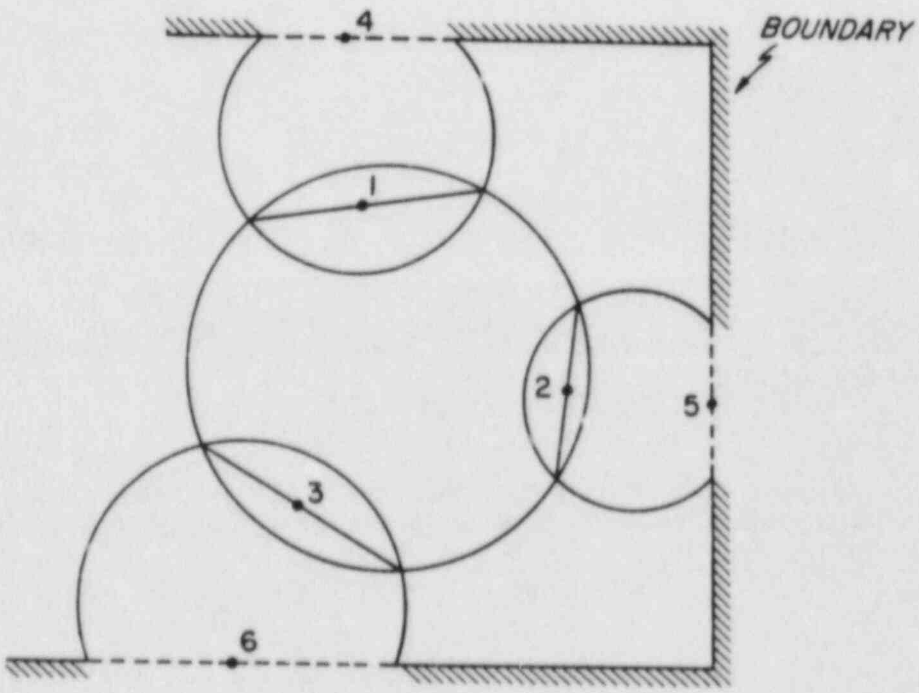

Figure 9. A sample fracture network to illustrate the derivation of the global mass balance equation.

and

$$C_{1,3}P_1 + C_{2,3}P_2 + (-C_{1,3} - C_{2,3} - C_{6,3})P_3 \tag{18}$$
$$= -C_{1,3,2} - C_{2,3,z} - C_{6,3}P_3 - C_{6,3,z}$$

Combining equations 16, 17 and 18, a matrix equation of order 3 is obtained:

$$[A]\ \{P\} = \{B\} \tag{19}$$

Matrix [A] is symmetric, positive, and definite. The Choleski's square root method can be used to decompose [A] to a product of two triangular matrices, [L] and [U],

$$[A] = [L]\ [U] \tag{20}$$

where [L] is a lower triangular matrix, and [U] is an upper triangular matrix, and [U] = [L] - transpose.

The entry of matrix [L], M_{kj}, is given by:

$$M_{kj} = \frac{a_{kj} - \sum_{p=1}^{j-1} M_{kp}M_{jp}}{M_{jj}},\ j=1,\ 2,\ K-1 \tag{21a}$$

$$M_{kk} = (a_{kk} - \sum_{p=1}^{k-1} M^2{}_{kp})^{1/2} \tag{21b}$$

Once [L] and [U] are determined, equation 20 can be rewritten as:

$$[L]\ [U]\ \{P\} = \{B\} \tag{22a}$$

Let $$[U]\ \{P\} = \{R\} \tag{22b}$$

then $$[L]\ \{R\} = \{B\} \tag{22c}$$

To solve for {P}, a temporary vector {R} of equation 22c is solved first by forward substitution. {P} is then determined by back substitution using equation 22b. Once the pressure heads of internal nodes are determined, the fluxes across the boundary surfaces can be calculated.

Subroutines SUBBOND, SUBMTRX and SUBMSBL are designed to perform the tasks described in this section, namely assign boundary conditions, assemble the network matrix equation and solve for the pressure head along internal intersections and calculate fluxes across the boundary surfaces.

Before the assemblage of the flow network, the dead-end fractures must be identified. Dead-end fractures are those that do not intersect any boundary surface, or only intersect one other fracture. Based on the physical formulation, these dead-end fractures do not contribute to flow, thus they should not be included in the flow network. Mathematically, they result in a singular matrix, which hampers the analysis. Subroutine SUBDEAD is designed to delete dead-end fractures from the network.

2.4 Design of Mass Transport Experiments

Hydraulic analysis provides the pressure head distribution in the fracture network. From information on head distribution and fracture network, it is possible to reconstruct the flow network. For every intersection, the flow network contains the details of fluxes from all directly connected intersections and their corresponding travel time. Once the flow network is defined, it is possible to perform mass transport experiments based on the two following assumptions:

1) a piston type convective transport between intersections in the fracture, and
2) a complete mixing at the intersection.

A tracer source of any time variant function introduced anywhere in the system can be tracked explicitly. This capability allows the study of hydrodynamic dispersion in a fracture network.

According to the imposed assumptions, mass attenuation occurs at the intersection only. Figure 10 is used as an example. In Figure 10, the intersecting line segment is represented by a node at its midpoint. Intersects A, B and C provide inflow to X, and D and E are outflow intersects from X. $F_{A,X}$ denotes the flux from A to X, and can be calculated from equation 12. Assume $M_{A,X}$ denotes the tracer concentration in the stream $F_{A,X}$, the principle of mass conservation implies that:

$$F_{A,X} + F_{B,X} + F_{C,X} = -(F_{D,X} + F_{E,X}) \quad (23a)$$

$$M_{A,X}F_{A,X} + M_{B,X}F_{B,X} + M_{C,X}F_{C,X} = -(M_{D,X}F_{D,X} + M_{E,X}F_{E,X}) \quad (23b)$$

The assumption of complete mixing at the intersection, X, gives the following relation:

$$M_{D,X} = M_{E,X} = \frac{M_{A,X}F_{A,X} + M_{B,X}F_{B,X} + M_{C,X}F_{C,X}}{F_{A,X} + F_{B,X} + F_{C,X}} \quad (24)$$

The portion of $M_{D,X}$ or $M_{E,X}$ which is contributed from the specific inflow path, A - X, is given as:

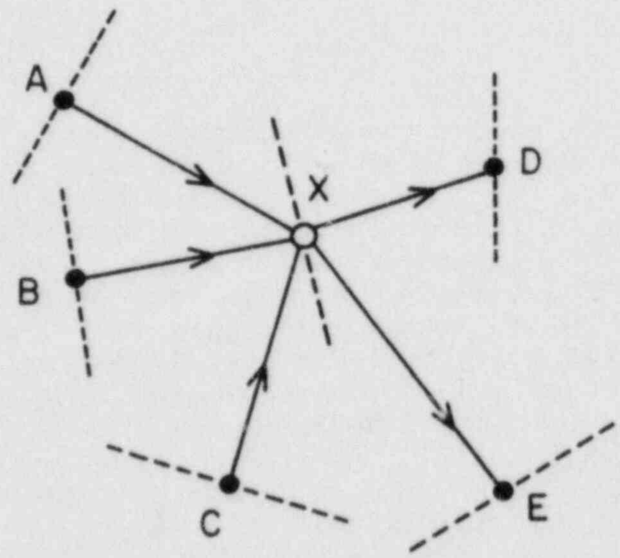

Figure 10. A sample flow network showing inflow and outflow components at intersection X.

$$L_{A,X} = \frac{M_{A,X}F_{A,X}}{F_{A,X} + F_{B,X} + F_{C,X}} = M_{A,X}R_{A,X} \tag{25a}$$

where

$$R_{A,X} = \frac{F_{A,X}}{F_{A,X} + F_{B,X} + F_{C,X}} \tag{25b}$$

and

$$\begin{aligned} M_{D,X} &= L_{A,X} + L_{B,X} + L_{C,X} \\ &= M_{A,X}R_{A,X} + M_{B,X}R_{B,X} + M_{C,X}R_{C,X} \end{aligned} \tag{26}$$

$R_{A,X}$ can be viewed as an attenuation coefficient. It represents the fraction of mass from a specific inflow route actually being reflected at an outflow route. Combining the concept of mass attenuation at the intersection and a piston type flow between intersections, the travel time and total mass attenuation can be calculated from the source point to the point of observation. Assume that between the tracer source and an observation point, there is a total of NR possible flow routes. For the jth flow route, it passes through K number of intersections, (or attenuation points). The total attenuation for this particular flow route is:

$$RT_j = \prod_{n=1}^{K} R_n \tag{27}$$

where R_n is the attenuation at the nth intersection. The travel time through route j is:

$$T_j = \sum_{n=1}^{K} t_n \tag{28}$$

where t_n is the travel time from intersection n-1 to intersection n. A mass change of DM at the source will be reflected through route j at a time lag of T_j units later at the point of observation with a change of dm_j, where

$$dm_j = DM * RT_j . \tag{29}$$

An analysis including all possible flow routes will give a complete breakthrough history at the point of observation. Note that the point of observation can be anywhere in the network as long as it is downstream from the tracer source. The convective piston type transport assumption does not allow mass to be transported against the direction of flow.

Subroutines FLOWNET and TRACER are designed to perform the mass transport experiments. To minimize the core memory requirement, the fracture network structure and hydraulic head distribution are generated first and stored in a data file. FLOWNET and TRACER subroutines are then called from a second main program to carry out the determination of breakthrough curves.

3. EXAMPLE OF EXPERIMENTS

To demonstrate the usage of the code, two experiments are presented. The complete code listing and sample input/output are given in Appendix A.

In both experiments, the global fracture network consists of 200 circular fractures with their centers located inside a cubic volume of 200 X 200 X 200 length units. These 200 circular fractures are uniformly distributed inside the cubic volume. The uniform distribution is also used to generate the angles of rotation, radius, and aperture of each fracture. Thus the generated fracture network represents a homogeneous system.

In Experiment 1, a cubic sample of size (L x L x L) is isolated from the fracture network (Figure 11). This sample is then placed in water such that the top surface is flush with the water surface, creating hydrostatic boundary conditions on all submerged boundary surfaces. The top surface is then pressurized to maintain a constant pressure of magnitude L length units (Figure 12). The pressure head distribution inside the sample is solved and the fluxes across all boundary surfaces are calculated. This experiment is used to examine the scale and boundary effects in the hydraulic conductivity structure.

Experiment 2 is a simulation of a hydraulic - tracer injection test (Figure 13). A cubic sample of size 180 length units, centered at the fracture generating region is isolated. At the center of the sample, a cubic volume of 20 length units on the side is identified as the injection zone. The sample is submerged in water to maintain a hydrostatic condition before injection starts. During the hydraulic injection test, the injection zone is pressurized to a constant head 100 units above its hydrostatic head. A steady state flow regime can be established. A continuous slug of tracer of concentration Co is introduced at the injection zone. The breakthrough curves are determined at every boundary (or outflow) intersects to study the mass transport characteristics of the simulated fracture sample.

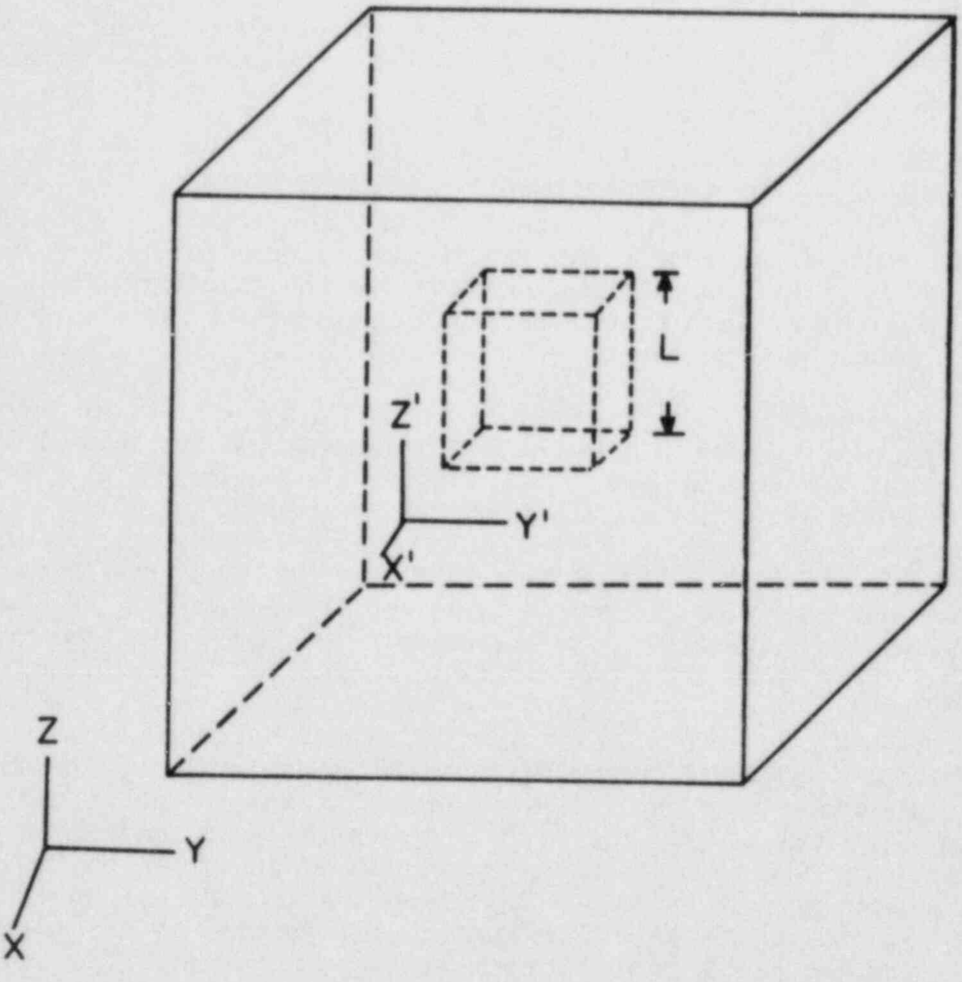

Figure 11. A cubic block sample of size L isolated from the global fracture region.

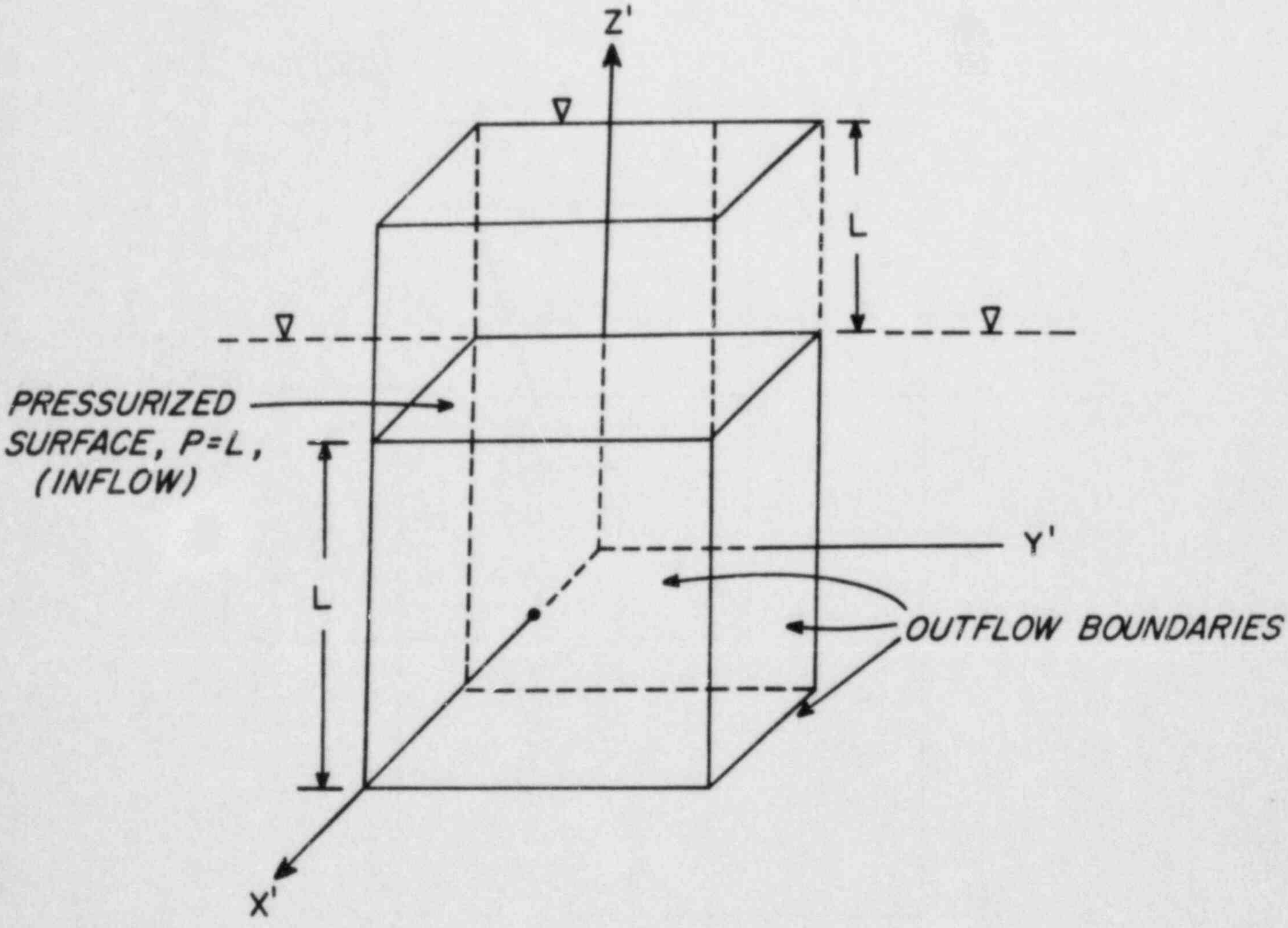

Figure 12. Schematic drawing of Experiment 1 showing a rock sample submerged in water with the top surface pressurized to a pressure head equal to the size of the block.

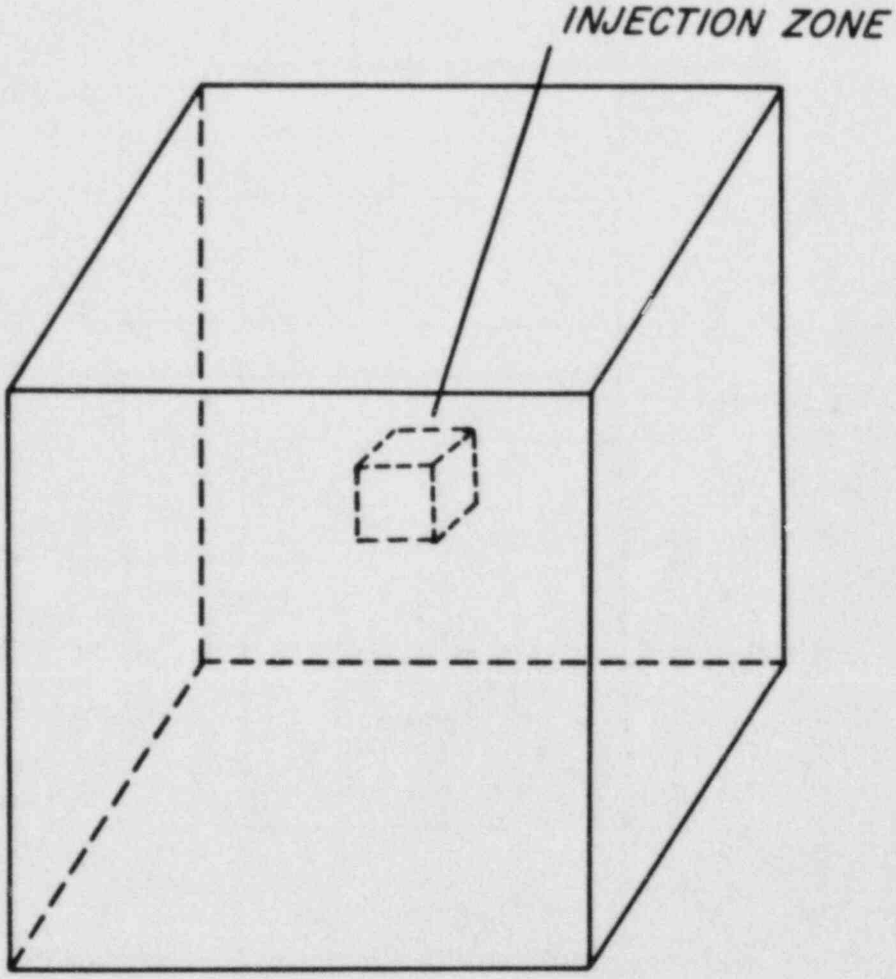

Figure 13. Schematic drawing of Experiment 2.

4. DISCUSSION

The fracture generating scheme presented is a useful tool for examining flow and transport through fractured systems. The present code is limited to steady state, saturated flow through a simulated fracture network where the rock matrix is impermeable. The code is being extended to simulate unsaturated fracture networks in rock of finite permeability.

Results of the two types of experiments are not included in this report. They are to be published in technical papers to follow. The purpose of this report is to present a detailed description of the computer code.

APPENDIX A: PROGRAM LISTING

A.1 Coordinate Systems Used in the Model

A.1.1 The Global Coordinate System:

The Global Coordinate System is a 3-D Cartesian coordinate system used in the creation of random fracture networks and in defining an isolated sample from the global network. The following arguments in the main program are expressed in the Global Coordinate System: XINP, XCUT, CORØ and COAL.

A.1.2 The Sample Coordinate System:

The Sample Coordinate System is a 3-D Cartesian coordinate system used to describe the intersections (or nodal points) contained inside a rectangular block of sample.

The coordinate system is defined at the center of the rectangular block, as specified by CORØ. The new +X axis is oriented COAL(1,1) degrees horizontally and COAL(2,1) degrees vertically from the +X axis of the Global Coordinate System (see Fig. X). The following arguments in the main program are expressed in the Sample Coordinate System: XYZI and XYZX.

A.1.3 The Fracture Local Coordinate System:

The Fracture Local Coordinate System is a 2-D Cartesian coordinate system defined at the center of each fracture or boundary surface, and is used to find the intersections (or nodal points) among fractures and boundary surfaces. For every surface, there is one such coordinate system defined.

For fracture K, the origin of the coordinate system is specified by XINP(J,K), J=1,3. XINP(4,K) and XINP(5,K) are the horizontal and vertical angles specifying the direction of the normal vector for the fracture plane.The +X axis of the Fracture Local Coordinate System is rotated XINP(6,K) degrees from a temperory axis having an orientation of XINP(4,K) degrees horizontally and XINP(5,K)-90 degrees vertically from the Global Coordinate Systems (see Fig. Y). This Coordinate System is used internally and should be transparent to the user.

A.2 Arguments Used in the Main Program:

NFRCT	Total number of fractures.
NSHP(I)	Vector of length NFRCT, containing the shape index of each individual fracture, I. Different shapes used in the program are: circle (NSHP=1), ellipse (NSHP=2), square (NSHP=3), rectangle (NSHP=4).
XINP(I,J)	Matrix of 9*NFRCT containing the geometric parameters describing each individual fracture, J. I=1,3 contain the global X-Y-Z coornidates specifying the center of the fracture. I=4,6 contain the angles (in drgrees) used to specify the orientation of the fracture plane. I=7,8 are characteristic length scales specifying the areal extent of the finite fracture. I=9 contains the aperture of the fracture.
CORØ(I,J)	Matrix of 3*N, where N is the total number of rectangular blocks to be isolated simultaneously from the global fracture generating region, containing the center location (I=1,3) expressed in the Global Coordinates for the Jth sample block.
COAL(I,J)	Matrix of 2*N, where N is the total number of rectangular blocks to be isolated simultaneously from the global fracture generating region, containing the angles of rotation (I=1,2) for the Jth sample block.
COSZ(I,J)	Matrix of 3*N, where N is the total number of rectangular blocks to be isolated simultaneously from the global fracture generating region, containing the sizes (I=1,3) of the Jth sample block.
CORA(I,J,K)	Matrix of 3*3*N, where N is total number of rect-
CORB(I,J,K)	angular blocks existing simultaneously within the global fracture generating region, containing the Jacobian coefficients to be used for coordinate transformation between the Global and the Kth Sample Coordinate Systems. If XYZGLB(I), I=1,3 defines the location of a point in the Golbal Coordinates, and XYZSAM(I,K) is the same point expressed in the Jth Sample Coodinates, then:

XYZSAM(I,K) = CORB(I,J,K) * (XYZGLB(J,K)-CORØ(J,K))

XYZGLB(I,K) = CORØ(I,K) + CORA(I,J,K)*XYZSAM(J,K)

Note: these two expressions are matrix equations.

NCUT — Total number of rectangular boundary surfaces. Each rectangular block of sample has 6 boundary surfaces.

XCUT(I,J) — Matrix of 8*NCUT containing the geometric parameters describing each rectangular boundary surface, J.
I=1,3 contain the global X-Y-Z coornidates specifying the center of the boundary surface.
I=4,6 contain the angles (in drgrees) used to specify the orientation of the boundary surface.
I=7,8 are characteristic length scales specifying the areal extent of the boundary surface.

NFR(I) — Vector of length NFRCT containing information about the number of intersecting fractures for fracture I.
If INFR(J) denotes the total number of intersecting fractures for fracture J, then

INFR(J) = NFR(J) - NFR(J-1) ,

and

NFR(J) = INFR(1) + INFR(2) + ... + INFR(J) .

NFRN(I) — Vector of length NFR(NFRCT) (or 2*NNODE) containing the fractures intersected by a fracture.
If INFR(J) denotes the total number of intersecting fractures for fracture J, then NFRN(NFR(J-1)+1) contains the fracture number representing the intersected fracture; and NFRN(NRF(J-1)+INFR(J)) or NFRN(NFR(J)) contains the fracture number representing the intersection with fracture J.

NCT(I) — Vector of length NCUT containing information about the number of intersecting fractures for each rectangular boundary surface J.
If INCT(J) denotes the total number of intersecting fractures for boundary surface J, then

INCT(J) = NCT(J) - NCT(J-1) ,

and

NCT(J) = INCT(1) + INCT(2) + ... + INCT(J) .

NCTN(I) Vector of length NCT(NCUT) (or NBOND) containing the fracture numbers that intersect each individual rectangular boundary surface.
If INCT(J) denotes the total number of intersecting fractures for boundary surface J, then NCTN(NCT(J-1)+1) contains the fracture number of the first intersecting fracture; and NCTN(NCT(J-1)+INCT(J)) or NCTN(NCT(J)) contains the fracture number of the INFR(J)th intersecting fracture.

NCF(I) Vector of length NFRCT containing information about the number of intersecting boundary surfaces for fracture I.
If INCF(J) denotes the total number of intersecting fractures for fracture J, then

INCF(J) = NCF(J) - NCF(J-1) ,

and

NCF(J) = INCF(1) + INCF(2) + ... + INCF(J) .

NCFN(I) Vector of length NCF(NFRCT) (or NBOND) containing the boundary surfaces numbers that intersect each individual fracture.
If INCF(J) denotes the total number of intersecting boundary surfaces for fracture J, then NCFN(NCF(J-1)+1) contains the boundary surface number of the first intersecting boundary; and NCFN(NCF(J-1)+INCF(J)) or NCFN(NCF(J)) contains the boundary surface number of the INFR(J)th intersecting boundary.

NNODE Total number of fracture-fracture intersections contained inside the rectangular block of sample.
Note that each each intersecting line segment is represented by a nodal point located at the center of the line segment. The fracture-fracture intersections are referred as internal nodes.

NDID(I,J) Matrix of length NNODE*2 containing the intersecting fracture numbers at internal node I. Thus, internal node I is the intersection of fracture NDID(I,1) and fracture NDID(I,2).

NDFR(I) Vector of length NFR(NFRCT) (or 2*NNODE) containing the internal nodal numbers of fracture intersects.
If INFR(J) denotes the total number of intersecting fractures for fracture J, then NDFR(NFR(J-1)+1) contains the internal nodal number representing the intersection of fracture J and fracture

NFRN(NFR(J-1)+1) [and NDFR(NFR(J-1)+INFR(J)) or NDFR(NFR(J)) contains the nodal number representing the intersection between fracture number J and fracture number NFRN(NFR(J))].

NBOND — Total number of fracture-boundary surface intersections. Note that each each intersecting line segment is represented by a nodal point located at the center of the line segment. The fracture-boundary surface intersections are referred as external nodes.

NDXD(I,J) — Matrix of length NBOND*2 containing the intersecting fracture number (J=1) and boundary surface number (J=2) for external node I. Thus, external node I is the intersection of fracture NDXD(I,1) anf boundary surface NDXD(I,2).

NDCF(I) — Vector of length NCFR(NFRCT) (or NBOND) containing the external nodal numbers of fracture-boundary intersections.
If INCF(J) denotes the total number of intersecting boundary surfaces for fracture J, then NDCF(NCF(J-1)+1) contains the external nodal number representing the intersection of fracture J and boundary surface NCFN(NCF(J-1)+1); and NDCF(NFR(J-1)+INCF(J)) or NDCF(NFR(J)) contains the nodal number representing the intersection between fracture number J and boundary surface number NCFN(NCF(J)).

NDCT(I) — Vector of length NCT(NCUT) (or NBOND) containing the external nodal numbers of fracture-boundary intersections.
If INCT(J) denotes the total number of intersecting fractures for boundary surface J, then NDCT(NCT(J-1)+1) contains the external nodal number representing the intersection of boundary surface J and fracture NCTN(NCT(J-1)+1); and NDCT(NCT(J-1)+INCT(J)) or NDCT(NCT(J)) contains the nodal number representing the intersection between boundary surface number J and fracture number NCTN(NCT(J)).

XYZI(I,J) — Matrix of NNODE*4 containing the location (J=1,3) and length (J=4) of fracture-fracture intersections. the fracture-fracture intersection is represented by a nodal point located at the center of the line segment. The location is expressed in the 3-D Sample Coordinate System.

XYZX(I,J) — Matrix of NBOND*4 containing the location (J=1,3) and length (J=4) of fracture-boundary surface intersections. The fracture-boundary surface intersection is represented by a nodal point located at the center of the line segment. The location is expressed in the 3-D Sample Coordinate System.

PO(I) — Vector of length NBOND containing the pressure heads, expressed in length units, on the external (fracture-boundary surface intersection) nodes.

PX(I) — Vector of length NNODE containing the pressure heads, expressed in length units, on the internal (fracture-fracture intersection) nodes.

A.3 PROGRAM LISTING OF EXPERIMENT 1

```
      PROGRAM NEWEXP1(INPUT,OUTPUT,TAPE1=INPUT,TAPE8=OUTPUT,TAPE9)
C
C   PURPOSE:     THIS IS THE MAIN DRIVING PROGRAM TO PERFORM EXPERIMENT
C                1, WHERE THE HYDRAULIC CONDUCTIVITY IS DETERMINED FOR
C                A BLOCK OF SAMPLE ISOLATED FROM THE GLOBAL FRACTURE
C                GENERATING REGION.
C
C
C   REMARKS:     THIS SAMPLE PROGRAM IS SET UP TO PERFORM 6 FLOW TESTS
C                ON ONE ISOLATED RECTANGULAR BLOCK OF SAMPLE.  THE TEST
C                CONDITION IS DESCRIBED BELOW:
C
C                  THE SAMPLE BLOCK IS SUBMERGED IN WATER WITH THE TOP
C                  SURFACE FLUSH WITH THE WATER LEVEL.  THE TOP SURFACE
C                  IS THEN PRESSURED TO MAINTAIN A CONSTANT HEAD.  THE
C                  PRESSURE (OR HEAD) DISTRIBUTION INSIDE THE BLOCK IS
C                  SOLVED AND THE FLUXES ACROSS THE BOUNDARY SURFACES
C                  ARE CALCULATED.  THE EXPERIMENT IS REPEATED 6 TIMES
C                  VIA RE-ORIENTING THE SAMPLE BLOCK SUCH THAT EACH OF
C                  THE BOUNDARY SURFACE IS ACTED AS THE PRESSURIZED TOP
C                  SURFACE.  THE CONVENTION USED IN NUMBERING THE BOUNDARY
C                  SURFACE IS:
C                     1  FOR +X SURFACE        2  FOR -X SURFACE
C                     3      +Y                4      -Y
C                     5      +Z                6      -Z
C
C   ARGUMENTS:
C
C       NSET             TOTAL NUMBER OF SAMPLES TO BE ISOLATED FROM
C                          THE GLOBAL FRACTURE GENERATING REGION (INPUT).
C
C   SUBROUTINES USED:
C
C       INPFRCT          GENERATE PARAMETERS DESCRIBING EACH FRACTURE.
C
C       INPCUT           GENERATE PARAMETERS DESCRIBING THE RECTANGULAR
C                          BOUNDARY SURFACES OF AN ISOLATED BLOCK OF SAMPLE.
C
C       SUPFRC           FIND FRACTURE-FRACTURE INTERSECTIONS.
C
C       SUPCUT           FIND FRACTURE-BOUNDARY SURFACE INTERSECTIONS.
C
C       SUPDEAD          DELETE DEAD-END FRACTURES.
C
C       SUBMTRX          PERFORM THE FOLLOWING TASKS:
C                          1:  ASSIGN THE BOUNDARY CONDITIONS;
C                          2:  ASSEMBLE THE GLOBAL MATRIX EQUATION;
C                          3:  SOLVE THE MATRIX EQUATION;
C                          4:  CALCULATE FLUXES ACROSS BOUNDARY SURFACES
C                              OR CHECK MASS BALANCE.
C
```

```
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CFCF/ NCF(200),NCFN(300)
      COMMON /NODE/ NNODE,NDID(400,2),NDFR(1000)
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      COMMON /MTRX/ PO(400), PX(400)
      CALL INPFRCT
      READ (1,*) NSET
      DO 10 J=1,NSET
      CALL INPCUT
      CALL SUPFRC
      CALL SUPCUT
      CALL SUPDEAD
      CALL SUBMTRX
10    CONTINUE
      STOP
      END
```

```
C
C
C
      SUBROUTINE INPFRCT
C
C   PURPOSE:     THIS SUBROUTINE READS IN THE CHARACTERISTICS OF A
C                DESIRED FRACTURE NETWORK AND USES PSEUDO-RANDOM NUMBER
C                GENERATORS TO GENERATE THE GEOMETRIC PARAMETERS FOR EACH
C                INDIVIDUAL FRACTURE.  THIS SUBPROGRAM CHANGES THE
C                CONTENTS OF THE FOLLOWING VARIABLES: NFRCT, NSHP, XINP.
C
C   REMARKS:     AVAILABLE PSEUDO-RANDOM NUMBER GENERATORS ARE:
C                1: RANDUFM - UNIFORM (0,1) RANDOM NUMBER
C                2: RANDNRM - NORMAL (0,1) RANDOM NUMBER
C                3: RANDLGN - LOG-NORMAL RANDOM NUMBER
C                4: RANDEXP - EXPONENTIAL RANDOM NUMBER
C
C   ARGUMENTS:
C
C       FRX,FRY,FRZ      SCALE FACTORS USED TO SPECIFY THE GLOBAL FRACTURE
C                          GENERATING REGION.
C                        FOR THIS PARTICULAR EXAMPLE, AN UNIFORM DISTRIBUTION
C                          IS USED TO SPECIFY THE CENTER LOCATION OF EACH
C                          FRACTURE, THUS THE FRACTURE CENTERS ARE CONTAINED
C                          INSIDE A RECTANGULAR BLOCK OF SIZE FRX*FRY*FRX.
C
C       FRCTLTH          SCALE FACTOR USED IN DEFINING THE AREAL EXTENT OF
C                          OF A FRACTURE.
C
C       FRCAPTR,XFCT     SCALE FACTORS USED IN DEFINING THE APERTURE OF A
C                          FRACTURE.
C
C       DS1,..,DS4       SEED NUMBERS USED IN RANDOM NUMBER GENERATORS.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      DOUBLE PRECISION DDS1,DDS2,DDS3,DDS4
      REAL WORK(1020),WORK1(220),WORK2(220)
      DATA XFCT / 1.0E-3 /
      READ (1,*) NFRCT, FRX, FRY, FRZ
      READ (1,*) FRCTLTH, FRCAPTR
      READ (1,*) DS1,DS2,DS3,DS4
      WRITE (8,100) NFRCT,FRX,FRY,FRZ
100   FORMAT ('1NFRCT,FRX,FRY,FRZ      :  ',I8,3F8.1)
      WRITE (8,105) FRCTLTH, FRCAPTR
105   FORMAT (' FRCTLTH, FRCAPTR       :  ',2F8.1)
      WRITE (8,110) DS1,DS2,DS3,DS4
110   FORMAT (' SEED--DS1,DS2,DS3,DS4 :  ',4F8.1)
```

```
C ----------
C     GENERATE RANDOM FRACTURES -- CIRCULAR DISC
C ----------
      DDS1=DBLE(DS1)
      DDS2=DBLE(DS2)
      DDS3=DBLE(DS3)
      DDS4=DBLE(DS4)
      CALL RANDUFM (DDS1,NFRCT+5,WORK1)
      CALL RANDUFM (DDS4,NFRCT+5,WORK2)
      CALL RANDUFM (DDS3,5*NFRCT+5,WORK)
      DO 1Ø J=1,NFRCT
      NSHP(J) = 1
      XINP(1,J)=REAL(INT(FRX*WORK(         J+2)))
      XINP(2,J)=REAL(INT(FRY*WORK(   NFRCT+J+2)))
      XINP(3,J)=REAL(INT(FRZ*WORK(2*NFRCT+J+2)))
      XINP(4,J)=REAL(INT(36Ø.*WORK(3*NFRCT+J+2)))
      XINP(5,J)=REAL(INT(36Ø.*WORK(4*NFRCT+J+2)))
      XINP(6,J)=Ø.Ø
      XINP(7,J)=REAL(INT(FRCTLTH*WORK1(J+2)))
      XINP(8,J)=XINP(7,J)
      XINP(9,J)=(REAL(INT(FRCAPTR*WORK2(J+2)))+1.Ø) * XFCT
1Ø    CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE INPCUT
C
C   PURPOSE:     THIS SUBROUTINE READS IN THE CENTER LOCATION (CORØ),
C                ROTATION (COAL), AND SIZE (COSZ) OF A RECTANGULAR BOLCK
C                AND GENERATES THE GEOMETRIC PARAMETERS FOR EACH OF THE
C                RECTANGLE BOUNDARY SURFACES.   THIS SUBPROGRAM CHANGES THE
C                CONTENTS OF THE FOLLOWING VARIABLES: NCUT, CORØ, COAL, COSZ,
C                XCUT, CORA, CORB.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(2ØØ),XINP(9,2ØØ),NFR(2ØØ),NFRN(1ØØØ)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(3ØØ)
      COMMON /CORD/ CORØ(3,2),COSZ(3,2),COAL(2,2),
     1              CORA(3,3,2),CORB(3,3,2)
      REAL WKALP(12),WKBTA(12),WKX(3),WKYZ(18)
      INTEGER NWAL(6), NWBT(6)
      DATA PI,DG2RAD / 3.141592654,Ø.Ø174532925199433/
      DATA WKALP /1.,Ø., 1.,18Ø., 1.,9Ø., 1.,27Ø., 1.,Ø.,1.,Ø./
      DATA WKBTA /1.,Ø.,-1.,Ø., Ø.,Ø., Ø.,Ø., 1.,9Ø., 1.,27Ø./
      DATA NWAL  / 3, 3, 3, 3, 1, 1 /
      DATA NWBT  / 2, 2, 1, 1, 2, 2 /
      DATA WKYZ  /1.,Ø.,Ø., -1.,Ø.,Ø., Ø.,1.,Ø., Ø.,-1.,Ø.,
     1            Ø.,Ø.,1., Ø.,Ø.,-1. /
      NCUT = 6
      JTOT = 1
      READ (1,*)     CORØ(1,1),CORØ(2,1),CORØ(3,1),
     1               COAL(1,1),COAL(2,1),
     1               COSZ(1,1),COSZ(2,1),COSZ(3,1)
      WRITE (8,1ØØ)  CORØ(1,1),CORØ(2,1),CORØ(3,1),
     1               COAL(1,1),COAL(2,1),
     1               COSZ(1,1),COSZ(2,1),COSZ(3,1)
1ØØ   FORMAT (/,' SAMPLE CENTER (CORØ,1) :  ',3F8.1/
     1          ' SAMPLE ANGLE  (COAL,1) :  ',2F8.1/
     2          ' SAMPLE SIZE   (COSZ,1) :  ',3F8.1,/)
      DO 1Ø K1=1,JTOT
      ALPP = COAL(1,K1) * DG2RAD
      BETT = COAL(2,K1) * DG2RAD
      CORA(1,1,K1)= COS(ALPP) * COS(BETT)
      CORA(2,1,K1)= SIN(ALPP) * COS(BETT)
      CORA(3,1,K1)=             SIN(BETT)
      CORA(1,2,K1)=-SIN(ALPP)
      CORA(2,2,K1)= COS(ALPP)
      CORA(3,2,K1)= Ø.
      CORA(1,3,K1)=-COS(ALPP) * SIN(BETT)
      CORA(2,3,K1)=-SIN(ALPP) * SIN(BETT)
      CORA(3,3,K1)=             COS(BETT)
```

```
      DO 9 I=1,3
      DO 9 J=1,3
9     CORB(I,J,K1) = CORA(J,I,K1)
10    CONTINUE
      DO 30 K1=1,JTOT
      DO 25 J=1,NCUT
      J4 = J + (K1-1)*NCUT
      DO 20 J1=1,3
      WKKK = 0.
      DO 15 J2=1,3
15    WKKK= WKKK + CORA(J1,J2,K1)*COSZ(J2,K1)*WKYZ(3*(J-1)+J2)/2.
20    XCUT(J1,J4)= COR0(J1,K1)+WKKK
      XCUT(4,J4) = COAL(1,K1)*WKALP(2*(J-1)+1) + WKALP(2*(J-1)+2)
      XCUT(5,J4) = COAL(2,K1)*WKBTA(2*(J-1)+1) + WKBTA(2*(J-1)+2)
      XCUT(6,J4) = 0.0
      XCUT(7,J4) = 0.5 * COSZ(NWAL(J),K1)
      XCUT(8,J4) = 0.5 * COSZ(NWBT(J),K1)
      XCUT(9,J4) = 0.0
25    CONTINUE
30    CONTINUE
      NCUT = NCUT * JTOT
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUBMTRX
C
C
C   PURPOSE:     THIS SUBROUTINE PERFORMS THE FOLLOWING TASKS:
C                1: SET UP BOUNDARY CONDITION BY CALLING *SUBBOND*;
C                2: ASSEMBLE THE GLOBAL MATRIX EQUATION;
C                3: SOLVE THE MATRIX EQUATION;
C                4: CALCULATE FULXES ACROSS BOUNDARY SURFACES OR
C                   PERFORM MASS BALANCE BY CALLING *SUBMSBL*.
C
C
C   ARGUMENTS:
C
C      NTS              INDEX INDICATING THE TEST NUMBER AND THE BOUNDARY
C                         SURFACE NUMBER WHICH IS BEING PRESSUREIZED IN
C                         THE CURRENT TEST.
C
C      FLUX(I)          VECTOR OF LENGTH NCUT CONTAINING THE STEADY-STATE
C                         FLUX ACROSS THE BOUNDARY SURFACE I.  THIS VECTOR
C                         IS RETURNED FROM THE SUBPROGRAM *SUBMSBL*.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CFCF/ NCF(200),NCFN(300)
      COMMON /NODE/ NNODE,NDID(400,2),NDFR(1000)
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      COMMON /MTRX/ PO(400), PX(400)
      REAL X0(4),X1(4),AA(310),BB,BBX(310,6),FLUX(12)
      REAL AMTRX(47200)
      DATA M,IB / 6, 310/
      DATA RHO,UN,UNITCON /9810., 0.001, 0.01 /
      DO 60 NTS=1,M
      CALL SUBBOND (NTS)
      NSD=INT(REAL(NTS+1)/2.)
      SIGN=1.0
      IF( NSD*2 .EQ. NTS) SIGN = -1.0
      J5=0
      DO 50 I=1,NNODE
      BB=0.
      DO 5 I1=1,NNODE
5     AA(I1)=0.
      J1=NDID(I,1)
      J2=NDID(I,2)
      DO 40 III=1,2
```

```
      IF (III .EQ. 2) THEN
        JTEMP=J1
        J1=J2
        J2=JTEMP
      ENDIF
      DO 15 I1=1,IFN(J1,NFR)
      ITT = IFK(J1,NFR) + I1
      IF (J2 .EQ. NFRN(ITT) ) THEN
        JJ2=I1
        DO 1Ø I2=1,4
1Ø      XØ(I2)= XYZI (NDFR(ITT), I2)
      ENDIF
15    CONTINUE
      DO 25 I1=1,IFN(J1,NFR)
      ITT = IFK(J1,NFR) + I1
      IF (I1 .NE. JJ2) THEN
        T2=Ø.
        DO 2Ø I2=1,3
        X1(I2) = XYZI (NDFR(ITT), I2)
        T2=T2+(X1(I2)-XØ(I2))**2.
2Ø      CONTINUE
        X1(4)= XYZI (NDFR(ITT), 4)
        XØX1 = SQRT(T2)
        XX   = (X1(4) + XØ(4))/2.
        XCOEF= XX * (XINP(9,J1)**3.) * RHO
     1          / ( 12. * UN * XØX1 )
        AA(NDFR(ITT))=AA(NDFR(ITT)) + XCOEF
        AA(I)= AA(I) - XCOEF
        BB  = BB + SIGN * (XØ(NSD)-X1(NSD))*XCOEF
      ENDIF
25    CONTINUE
      DO 35 I1=1,IFN(J1,NCF)
        T2=Ø.
        ITT = IFK(J1,NCF) + I1
        DO 3Ø I2=1,3
          X1(I2) = XYZX (NDCF(ITT), I2)
          T2=T2+(X1(I2)-XØ(I2))**2.
3Ø      CONTINUE
        X1(4)= XYZX (NDCF(ITT), 4)
        XØX1 = SQRT(T2)
        XX   = (X1(4) + XØ(4))/2.
        XCOEF= XX * (XINP(9,J1)**3.) * RHO
     1          / ( 12. * UN * XØX1 )
        AA(I)= AA(I) - XCOEF
        BB  = BB + (SIGN*(XØ(NSD)-X1(NSD))-PO(NDCF(ITT)))*XCOEF
35    CONTINUE
4Ø    CONTINUE
      DO 45 I1=1,I
      AMTRX(J5+I1) = -AA(I1)
45    CONTINUE
```

```
      J5=J5+I
      BBX(I,NTS) = -BB
50    CONTINUE
60    CONTINUE
      IER=0
      CALL SUBLEQS(AMTRX,M,NNODE,BBX,IB,D1,D2,IER)
      IF ( IER .GT. 0 ) THEN
      WRITE (8, *) 'ERROR MSG FROM  *SUBLEQS*  IER =',IER
      RETURN
      ENDIF
      DO 70 NTS = 1, M
      CALL SUBBOND(NTS)
      DO 65 I1 = 1, NNODE
65    PX(I1) = BBX(I1,NTS)
      CALL SUBMSBL(NTS,FLUX)
      WRITE (8,100) NTS, (FLUX(J),J=1,6)
100   FORMAT(2X,'NTS=',I1,3X,6F10.2)
70    CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUBBOND (NT)
C
C
C   PURPOSE:     THIS SUBROUTINE ASSIGNS THE BOUNDARY CONDITIONS FOR
C                A SPECIFIC TEST.  THE CONTENTS OF THE *PO* VECTOR
C                IS CHANGED.
C
C
C   ARGUMENTS:
C
C      NT                INDEX INDICATING THE TEST NUMBER AND THE BOUNDARY
C                          SURFACE NUMBER WHICH IS BEING PRESSUREIZED IN
C                          THE CURRENT TEST.
C
C

      COMMON /BOND/ NBOND,NDXD(4ØØ,2),NDCF(3ØØ),NDCT(3ØØ)
      COMMON /XYZZ/ XYZI(4ØØ,4), XYZX(4ØØ,4)
      COMMON /MTRX/  PO(4ØØ), PX(4ØØ)
      COMMON /CORD/ CORØ(3,2),COSZ(3,2),COAL(2,2),
     1              CORA(3,3,2),CORB(3,3,2)
      NSD=INT(REAL(NT+1)/2.)
      SIGN=1.Ø
      IF( NSD*2 .EQ. NT) SIGN = -1.Ø
      DO 1Ø J=1,NBOND
      PRES= Ø.Ø
      IF ( NDXD(J,2) .EQ. NT) PRES = COSZ(NSD,1)
      PO(J) = PRES + COSZ(NSD,1)/2. - SIGN * XYZX(J,NSD)
1Ø    CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUBMSBL (NTS,FLUX)
C
C
C   PURPOSE:    THIS SUBROUTINE CALCULATES FULXES ACROSS BOUNDARY SURFACES
C               (OR PERFORMS MASS BALANCE).
C
C   ARGUMENTS:
C
C       NTS             INDEX INDICATING THE TEST NUMBER AND THE BOUNDARY
C                         SURFACE NUMBER WHICH IS BEING PRESSUREIZED IN
C                         THE CURRENT TEST.
C
C       FLUX(I)         VECTOR OF LENGTH NCUT CONTAINING THE STEADY-STATE
C                         FLUX ACROSS THE BOUNDARY SURFACE I.
C
C
C

      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CFCF/ NCF(200),NCFN(300)
      COMMON /NODE/ NNODE,NDID(400,2),NDFR(1000)
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      COMMON /MTRX/ PO(400), PX(400)
      REAL FLUX(12), X0(4),X1(4)
      DATA RHO,UN,UNITCON /9810., 0.001, 0.01 /
      NSD=INT(REAL(NTS+1)/2.)
      SIGN=1.0
      IF( NSD*2 .EQ. NTS) SIGN = -1.0
      DO 40 J=1,NCUT
      FLUX(J)=0.
      DO 35 J1=1,IFN(J,NCT)
      ITT = IFK(J,NCT) + J1
      J2=NCTN(ITT)
      DO 10 J4=1,4
10    X0(J4) = XYZX (NDCT(ITT), J4)
      DO 20 J3=1,IFN(J2,NCF)
        ITS = IFK(J2,NCF) + J3
      IF (J .NE. NCFN(ITS)) THEN
        T2=0.
        DO 15 J4=1,3
        X1(J4) = XYZX (NDCF(ITS), J4)
        T2=T2+(X1(J4)-X0(J4))**2.
15      CONTINUE
        X1(4)= XYZX (NDCF(ITS), 4)
        X0X1 = SQRT(T2)
```

```
         XX    = (X1(4) + XØ(4))/2.
         XCOEF= XX * ((XINP(9,J2)*1.ØØ)**3.) * RHO
     1           / ( 12. * UN * XØX1 )
         FLUX(J)=FLUX(J)+ XCOEF * ( PO(NDCF(ITS)) + SIGN*X1(NSD)
     1                            - PO(NDCT(ITT)) - SIGN*XØ(NSD) )
       ENDIF
 2Ø    CONTINUE
       DO 3Ø J3=1,IFN(J2,NFR)
         T2=Ø.
         ITS = IFK(J2,NFR) + J3
         DO 25 J4=1,3
         X1(J4) = XYZI (NDFR(ITS), J4)
         T2=T2+(X1(J4)-XØ(J4))**2.
 25      CONTINUE
         X1(4) = XYZI (NDFR(ITS), 4)
         XØX1 = SQRT(T2)
         XX    = (X1(4) + XØ(4))/2.
         XCOEF= XX * ((XINP(9,J2)*1.ØØ)**3.) * RHO
     1           / ( 12. * UN * XØX1 )
         FLUX(J)=FLUX(J)+ XCOEF * ( PX(NDFR(ITS)) + SIGN*X1(NSD)
     1                            - PO(NDCT(ITT)) - SIGN*XØ(NSD) )
 3Ø    CONTINUE
 35    CONTINUE
 4Ø    CONTINUE
       RETURN
       END
```

```
C
C
C
      SUBROUTINE SUPFRC
C
C
C   PURPOSE      FIND ALL INTERSECTIONS AMONG FRACTURES CONTAINED INSIDE
C                A RECTANGULAR BLOCK OF SAMPLE.
C
C   REMARKS      OUTPUT OF THIS SUBROUTINE IS STORED IN THE FOLLOWING
C                VARIABLES: NFR, NFRN, NNODE, NDID, NDFR, XYZI.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(2ØØ),XINP(9,2ØØ),NFR(2ØØ),NFRN(1ØØØ)
      COMMON /NODE/ NNODE,NDID(4ØØ,2),NDFR(1ØØØ)
      COMMON /XYZZ/ XYZI(4ØØ,4), XYZX(4ØØ,4)
      REAL X1(9),X2(9),TA(3,2)
      DATA PI,DG2RAD / 3.141592654,Ø.Ø174532925199433/
      DATA TERR / 1.E-9/
      NNODE=Ø
      ITT = Ø
      DO 1Ø J1=1,NFRCT
1Ø    NFR(J1)=Ø
      DO 5Ø J1=1,NFRCT
      IA = NSHP(J1)
      DO 15 J3=1,9
15    X1(J3)=XINP(J3,J1)
      DO 4Ø J2=1,NFRCT
      IB = NSHP(J2)
      DO 2Ø J3=1,9
2Ø    X2(J3)=XINP(J3,J2)
      CALL SUPDRV (IA,X1,IB,X2,ICK,TA)
      IF (ICK .EQ. 2) THEN
       CALL SUB1 ( TA, ICHK )
       IF (ICHK .EQ. Ø) THEN
        ITT = ITT + 1
        NFRN(ITT) = J2
        IF ( J2 .GT. J1 ) THEN
         NNODE = NNODE + 1
         NDFR(ITT) = NNODE
         NDID(NNODE,1) = J1
         NDID(NNODE,2) = J2
         TO=Ø.
         CALL SUB1AA (TA)
         DO 25 JJ=1,3
         XYZI(NNODE,JJ)=Ø.5*(TA(JJ,1)+TA(JJ,2))
         TO=TO+(TA(JJ,1)-TA(JJ,2))**2.
25       CONTINUE
         XYZI(NNODE,4)=SQRT(TO)
```

```
          ENDIF
          IF ( J1 .GT. J2 ) THEN
           JT = NFR(J2)
           IF ( J2 .EQ. 1 ) THEN
            JS = 1
           ELSE
            JS = NFR(J2-1) + 1
           ENDIF
           DO 3Ø J3 = JS, JT
           IF (NFRN(J3) .EQ. J1) NDFR(ITT) = NDFR(J3)
3Ø         CONTINUE
          ENDIF
         ENDIF
        ENDIF
4Ø      CONTINUE
        NFR(J1) = ITT
5Ø      CONTINUE
        RETURN
        END
```

```
C
C
C
      SUBROUTINE SUPCUT
C
C
C   PURPOSE      FIND ALL INTERSECTIONS BETWEEN FRACTURES AND THE BOUNDARY
C                SURFACES OF A RECTANGULAR BLOCK OF SAMPLE.
C
C   REMARKS      OUTPUT OF THIS SUBSROUTINE IS STORED IN THE FOLLOWING
C                VARIABLES: NCT, NCTN, NBOND, NXID, NDCF, NDCT, XYZX.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(2ØØ),XINP(9,2ØØ),NFR(2ØØ),NFRN(1ØØØ)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(3ØØ)
      COMMON /CFCF/ NCF(2ØØ),NCFN(3ØØ)
      COMMON /NODE/ NNODE,NDID(4ØØ,2),NDFR(1ØØØ)
      COMMON /BOND/ NBOND,NDXD(4ØØ,2),NDCF(3ØØ),NDCT(3ØØ)
      COMMON /XYZZ/ XYZI(4ØØ,4), XYZX(4ØØ,4)
      REAL X1(9),X2(9),TA(3,2)
      DATA PI,DG2RAD / 3.141592654,Ø.Ø174532925199433/
      DATA TERR / 1.E-9/
      NBOND=Ø
      ITT = Ø
      DO 1Ø J1=1,NFRCT
1Ø    NCF(J1)=Ø
      DO 15 J1=1,NCUT
15    NCT(J1)=Ø
      DO 5Ø J1=1,NCUT
      IA = 4
      DO 2Ø J3=1,9
2Ø    X1(J3)=XCUT(J3,J1)
      DO 4Ø J2=1,NFRCT
      IB = NSHP(J2)
      DO 25 J3=1,9
25    X2(J3)=XINP(J3,J2)
      CALL SUPDRV (IA,X1,IB,X2,ICK,TA)
      IF (ICK .EQ. 2) THEN
        ITT = ITT + 1
        NCTN(ITT) = J2
        NBOND = NBOND + 1
        NDCT(ITT) = NBOND
        NDXD(NBOND,1) = J2
        NDXD(NBOND,2) = J1
        TO=Ø.
        CALL SUB1AA (TA)
        DO 3Ø JJ=1,3
        XYZX(NBOND,JJ)=Ø.5*(TA(JJ,1)+TA(JJ,2))
        TO=TO+(TA(JJ,1)-TA(JJ,2))**2.
```

```
30          CONTINUE
            XYZX(NBOND,4)=SQRT(TO)
          ENDIF
40        CONTINUE
          NCT(J1) = ITT
50        CONTINUE
          ITT = Ø
          DO 6Ø J1 = 1, NFRCT
          DO 55 J2 = 1, NBOND
          IF ( NDXD(J2,1) .EQ. J1) THEN
            ITT = ITT + 1
            NDCF(ITT) = J2
            NCFN(ITT) = NDXD(J2,2)
          ENDIF
55        CONTINUE
          NCF(J1) = ITT
6Ø        CONTINUE
          RETURN
          END
```

```
C
C
C
      SUBROUTINE SUB1(TQ,II)
C
C
C   PURPOSE     NUCLEAS CALLED BY SUBROUTINE *SUPFRC* TO TRUNCATE A FINITE
C               LINE SEGMENT AT THE BOUNDARY OF A RECTANGULAR BLOCK OF
C               SAMPLE.  OLNY THE PORTION OF THE LINE SEGMENT CONTAINED
C               INSIDE THE BLOCK IS RETURNED.
C
C   ARGUMENTS:
C
C     TQ(I,J)            INPUT MATRIX OF 3*2 CONTAINING THE GLOBAL X-Y-Z
C                          COORDINATES (I=1,3) OF 2 END POINTS (J=1,2) OF
C                          A LINE SEGMENT.
C                        AS OUTPUT, TQ(I,J) RETURNS THE COORDINATES OF THE
C                          TRUNCATED LINE SEGMENT.
C
C     II                 OUTPUT INDEX.
C                        II=Ø INDICATES THAT A PORTION OF THE INPUT LINE
C                          SEGMENT IS CONTAINED INSIDE THE RECT ANGULAR BLOCK.
C                        II=1 INDICATES THAT THE INPUT LINE SEGMENT IS LOCATED
C                          OUTSIDE THE RECTANGULAT BLOCK.
C
C
C
      COMMON /CORD/ CORØ(3,2),COSZ(3,2),COAL(2,2),
     1              CORA(3,3,2),CORB(3,3,2)
      REAL S(6),TP(3,2),TQ(3,2),TT(3)
      DO 1Ø I=1,3
      S(2*(I-1)+1)=-COSZ(I,1)/2.
      S(2*(I-1)+2)= COSZ(I,1)/2.
1Ø    CONTINUE
      DO 25 I=1,2
      DO 2Ø I1=1,3
      WRK=Ø.
      DO 15 I2=1,3
15    WRK=WRK+CORB(I1,I2,1)*(TQ(I2,I)-CORØ(I2,1))
2Ø    TP(I1,I)=WRK
25    CONTINUE
      II=Ø
      DO 3Ø I=1,3
      I1=I*2
      I2=I*2-1
      IF (((TP(I,1) .GT. S(I1)) .AND. (TP(I,2) .GT. S(I1))) .OR.
     1     ((TP(I,1) .LT. S(I2)) .AND. (TP(I,2) .LT. S(I2))))THEN
      II=1
      ENDIF
3Ø    CONTINUE
```

```
      IF (II .EQ. 0) THEN
      DO 50 I=1,3
      IJ=0
      I1=1*2
      I2=I*2-1
      IP1=I+1
      IP2=I+2
      IF(IP1 .GT. 3) IP1=IP1-3
      IF(IP2 .GT. 3) IP2=IP2-3
      IF((TP(I,1) .GT. S(I1)) .OR. (TP(I,2) .GT. S(I1))) THEN
      IF((TP(I,1) .GT. TP(I,2)) .AND. (TP(I,1) .GT. S(I1)))THEN
        TT(I)=S(I1)
        IJ=1
      ENDIF
      IF((TP(I,2) .GT. TP(I,1)) .AND. (TP(I,2) .GT. S(I1)))THEN
        TT(I)=S(I1)
        IJ=2
      ENDIF
      TT(IP1)=TP(IP1,1)+(TT(I)-TP(I,1))*(TP(IP1,2)-TP(IP1,1))
     1                  /(TP(I,2)-TP(I,1))
      TT(IP2)=TP(IP2,1)+(TT(I)-TP(I,1))*(TP(IP2,2)-TP(IP2,1))
     1                  /(TP(I,2)-TP(I,1))
      IF((TT(IP1) .GT. S(IP1*2)) .OR. (TT(IP1) .LT. S(IP1*2-1)) .OR.
     1   (TT(IP2) .GT. S(IP2*2)) .OR. (TT(IP2) .LT. S(IP2*2-1)))THEN
        II=1
      ELSE
        II=0
        DO 35 IK=1,3
35      TP(IK,IJ)=TT(IK)
      ENDIF
      ENDIF
      IF((TP(I,1) .LT. S(I2)) .OR. (TP(I,2) .LT. S(I2))) THEN
      IF((TP(I,2) .GT. TP(I,1)) .AND. (TP(I,1) .LT. S(I2)))THEN
        TT(I)=S(I2)
        IJ=1
      ENDIF
      IF((TP(I,1) .GT. TP(I,2)) .AND. (TP(I,2) .LT. S(I2)))THEN
        TT(I)=S(I2)
        IJ=2
      ENDIF
      TT(IP1)=TP(IP1,1)+(TT(I)-TP(I,1))*(TP(IP1,2)-TP(IP1,1))
     1                  /(TP(I,2)-TP(I,1))
      TT(IP2)=TP(IP2,1)+(TT(I)-TP(I,1))*(TP(IP2,2)-TP(IP2,1))
     1                  /(TP(I,2)-TP(I,1))
      IF((TT(IP1) .GT. S(IP1*2)) .OR. (TT(IP1) .LT. S(IP1*2-1)) .OR.
     1   (TT(IP2) .GT. S(IP2*2)) .OR. (TT(IP2) .LT. S(IP2*2-1)))THEN
        II=1
      ELSE
        II=0
        DO 40 IK=1,3
```

```
40        TP(IK,IJ)=TT(IK)
        ENDIF
        ENDIF
50      CONTINUE
        ENDIF
        DO 65 I=1,2
        DO 60 I1=1,3
        WRK=0.
        DO 55 I2=1,3
55      WRK=WRK+CORA(I1,I2,1)*TP(I2,I)
60      TQ(I1,I)=WRK+CORØ(I1,1)
65      CONTINUE
        RETURN
        END
```

```
C
C
C
      SUBROUTINE SUB1AA (TP)
C
C
C   PURPOSE      PERFORM COORDINATE TRANSFORMATION FROM THE GLOBAL 3-D
C                COORDINATES DEFINED FOR THE FRACTURE GENERATING REGION TO
C                THE SAMPLE 3-D COORDINATE SYSTEM DEFINED AT THE CENTER OF
C                THE RECTANGULAR BLOCK.
C
C   ARGUMENTS:
C
C
C      TP(I,J)           INPUT MATRIX OF 3*2 CONTAINING THE GLOBAL X-Y-Z
C                          COORDINATES (I=1,3) OF 2 POINTS (J=1,2).
C                        AS OUTPUT, TP(I,J) RETURNS THE TRANSFORMED 3-D
C                          COORDINATES BASED ON THE CENTER OF THE RECTANGULAR
C                          BLOCK.
C
C
C
      COMMON /CORD/ CORØ(3,2),COSZ(3,2),COAL(2,2),
     1              CORA(3,3,2),CORB(3,3,2)
      REAL TP(3,2),TQ(3,2)
      DO 1Ø I=1,2
      DO 1Ø J=1,3
1Ø    TQ(J,I)=TP(J,I)
      DO 4Ø I=1,2
      DO 3Ø I1=1,3
      WRK=Ø.
      DO 2Ø I2=1,3
2Ø    WRK=WRK+CORB(I1,I2,1)*(TQ(I2,I)-CORØ(I2,1))
3Ø    TP(I1,I)=WRK
4Ø    CONTINUE
      RETURN
      END
```

```
C
C
C
      FUNCTION IFN(I,NF)
C
C   REMARKS:      NUCLEAS CALLED BY VARIOUS SUBPROGRAMS TO UNPACK A
C                 ONE-DIMENSIONAL ARRAY WHERE DATA OF TWO-DIMENSIONAL
C                 NATURE HAVE BEEN STORED.
C
C
C
      DIMENSION NF(300)
      IF ( I .GT. 1 ) THEN
        IFN = NF(I) - NF(I-1)
      ELSE
        IFN = NF(1)
      ENDIF
      RETURN
      END
C
C
C
      FUNCTION IFK(I,NF)
C
C   REMARKS:      NUCLEAS CALLED BY VARIOUS SUBPROGRAMS TO UNPACK A
C                 ONE-DIMENSIONAL ARRAY WHERE DATA OF TWO-DIMENSIONAL
C                 NATURE HAVE BEEN STORED.
C
C
C
      DIMENSION NF(300)
      IF ( I .GT. 1 ) THEN
        IFK = NF(I-1)
      ELSE
        IFK = 0
      ENDIF
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUPDEAD
C
C
C   PURPOSE      DELETE DEAD-END FRACTURES FROM THE FRACTURE NETWORK
C                CONTAINED INSIDE THE RECTANGULAR BLOCK OF SAMPLE.
C                A DEAD-END FRACTURE IS THE ONE THAT INTERSECTS ONLY ONE
C                OTHER FRACTURES AND DOES NOT INTERSECT ANY BOUNDARY SURFACES.
C
C   REMARKS      RESULTS OF THIS SUBSROUTINE MAY CHANGE THE CONTENTS OF THE
C                FOLLOWING VARIABLES: NFR, NFRN, NNODE, NDID, NDFR, XYZI.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CFCF/ NCF(200),NCFN(300)
      COMMON /NODE/ NNODE,NDID(400,2),NDFR(1000)
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      INTEGER IX(2)
5     CONTINUE
      ISOL=0
      DO 40 I=1,NFRCT
      IF( (IFN(I,NFR) .EQ. 1) .AND. (IFN(I,NCF) .EQ. 0)) THEN
      ISOL = 1
      JNODE= NDFR(NFR(I))
      JFR1 = I
      JFR2 = NFRN(NFR(I))
      DO 15 I1 = JNODE+1, NNODE
      NDID(I1-1,1)=NDID(I1,1)
      NDID(I1-1,2)=NDID(I1,2)
      DO 10 I2=1,4
10    XYZI(I1-1,I2)=XYZI(I1,I2)
15    CONTINUE
      IIJ = 0
      DO 20 I2=1,NFR(NFRCT)
      IF (NDFR(I2) .EQ. JNODE) THEN
       IIJ = IIJ +1
       IX(IIJ) = I2
      ENDIF
      IF (NDFR(I2) .GT. JNODE) NDFR(I2)=NDFR(I2)-1
20    CONTINUE
      IF (IX(1) .GT. IX(2)) THEN
        IT    = IX(1)
        IX(1) = IX(2)
        IX(2) = IT
      ENDIF
```

```
      DO 31 I3 = IX(2)+1, NFR(NFRCT)
      NDFR(I3-1)=NDFR(I3)
31    NFRN(I3-1)=NFRN(I3)
      DO 32 I3 = IX(1)+1, NFR(NFRCT)
      NDFR(I3-1)=NDFR(I3)
32    NFRN(I3-1)=NFRN(I3)
      DO 33 I3 = JFR1, NFRCT
33    NFR(I3) = NFR(I3) - 1
      DO 34 I3 = JFR2, NFRCT
34    NFR(I3) = NFR(I3) - 1
      NNODE = NNODE -1
      ENDIF
40    CONTINUE
      IF (ISOL .GT. 0) GO TO 5
      RETURN
      END
```

```
C
C
C
      SUBROUTINE RANDLGN  (DSEED,NR,XM,S,R)
C
C
C   PURPOSE      LOG-NORMAL RANDOM DEVIATE GENERATOR.
C
C   REMARKS      NORMAL (Ø,1) PSEUDO RANDOM DEVIATES U ARE GENERATED BY
C                CALLING SUBROUTINE *RANDNRM* AND TRANSFORMED TO LOG-NORMAL
C                DEVIATES X, AS X = EXP(XM + S*U).
C
C   ARGUMENTS:
C
C     DSEED             INPUT/OUTPUT DOUBLE PRECISION SEED NUMBER HAVING
C                         AN INTEGER VALUE IN THE EXCLUSIVE RANGE
C                         (1.DØ, 2147483647.DØ).  DSEED IS REPLACED BY
C                         A NEW VALUE TO BE USED IN A SUBSEQUENT CALL.
C
C     NR                NUMBER OF DEVIATES TO BE GENERATED (INPUT).
C
C
C     XM,S              INPUT PARAMETERS, SEE REMARKS.
C
C     R(I)              OUTPUT VECTOR OF LENGTH NR+1 CONTAINING THE LOG-
C                         NORMAL DEVIATES IN THE FIRST NR LOCATIONS.
C                         R(NR+1) IS WORK STORAGE.
C
C
C
      INTEGER NR
      REAL R(1),XM,S
      DOUBLE PRECISION DSEED
      INTEGER I,NN,M
      NN = NR
      M = MOD(NR,2)
      IF(M.NE.Ø) NN = NN+1
      CALL RANDNRM(DSEED,NN,R)
      DO 5 I = 1,NR
      R(I) = EXP(XM+S*R(I))
5     CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE RANDNRM (DSEED,NR,R)
C
C   PURPOSE     NORMAL RANDOM DEVIATE GENERATOR VIA THE POLAR METHOD.
C
C   ARGUMENTS:
C
C       DSEED             INPUT/OUTPUT DOUBLE PRECISION SEED NUMBER HAVING
C                           AN INTEGER VALUE IN THE EXCLUSIVE RANGE OF
C                           (1.DØ, 2147483647.DØ).  DSEED IS REPLACED BY
C                           A NEW VALUE TO BE USED IN A SUBSEQUENT CALL.
C
C       NR                NUMBER OF DEVIATES TO BE GENERATED (INPUT).
C                           NR MUST BE 2 OR GREATER.  IF NR IS ODD, NR-1
C                           DEVIATES ARE GENERATED.
C
C       R(I)              OUTPUT VECTOR OF LENGTH NR CONTAINING THE NORMAL
C                           DEVIATES. IF NR IS ODD, R(NR) IS NOT USED.
C
C
C
      INTEGER NR,I,NN,M
      REAL R(NR)
      DOUBLE PRECISION DSEED
      REAL U,V,SUM,SLN,Z,D2P31M,D2PN31
      DATA D2P31M/2147483647.DØ/
      DATA D2PN31/2147483648.DØ/
      NN = NR
      M = MOD(NR,2)
      IF (M .NE. Ø) NN = NN-1
      Z = DSEED
      DO 1Ø I = 1,NN,2
5      Z = AMOD(168Ø7.EØ*Z,D2P31M)
       U = Z/D2PN31
       Z = AMOD(168Ø7.EØ*Z,D2P31M)
       V = Z/D2PN31
       U = U+U-1.Ø
       V = V+V-1.Ø
       SUM = U*U+V*V
       IF(SUM .GE. 1.Ø) GO TO 5
       SLN = ALOG(SUM)
       SLN = SQRT((-SLN-SLN)/SUM)
       R(I) = U*SLN
       R(I+1) = V*SLN
1Ø    CONTINUE
      DSEED = DBLE(Z)
      RETURN
      END
```

```
C
C
C
      SUBROUTINE RANDUFM (DSEED,NR,R)
C
C
C   PURPOSE     BASIC UNIFORM (Ø,1) PSEUDO-RANDOM NUMBER GENERATOR.
C
C   ARGUMENTS:
C
C      DSEED           INPUT/OUTPUT DOUBLE PRECISION SEED NUMBER HAVING
C                        AN INTEGER VALUE IN THE EXCLUSIVE RANGE OF
C                        (1.DØ, 2147483647.DØ).  DSEED IS REPLACED BY
C                        A NEW VALUE TO BE USED IN A SUBSEQUENT CALL.
C
C      NR              NUMBER OF DEVIATES TO BE GENERATED (INPUT).
C
C      R(I)            OUTPUT VECTOR OF LENGTH NR CONTAINING THE PSEUDO-
C                        RANDOM UNIFORM (Ø,1) DEVIATES.
C
C
C
      INTEGER NR
      REAL R(NR)
      DOUBLE PRECISION DSEED
      INTEGER I
      REAL S2P31,S2P31M,SEED
C                                 S2P31M = (2**31) - 1
C                                 S2P31 = (2**31)
      DATA S2P31M/2147483647.EØ/
      DATA S2P31 /2147483648.EØ/
      SEED = DSEED
      DO 5 I=1,NR
       SEED = AMOD(168Ø7.EØ*SEED,S2P31M)
5     R(I) = SEED / S2P31
      DSEED = SEED
      RETURN
      END
```

```
C
C
C
      SUBROUTINE RANDEXP  (DSEED,XM,NR,R)
C
C   PURPOSE      EXPONENTIAL RANDOM DEVIATE GENERATOR
C
C   REMARKS      UNIFORM (Ø,1) PSEUDO RANDOM DEVIATES U ARE GENERATED BY
C                CALLING SUBROUTINE *RANDUFM* AND TRANSFORMED TO EXPONENTIAL
C                DEVIATES X, AS X = -XM * ALOG(U).
C
C   ARGUMENTS:
C
C      DSEED              INPUT/OUTPUT DOUBLE PRECISION SEED NUMBER HAVING
C                           AN INTEGER VALUE IN THE EXCLUSIVE RANGE
C                           (1.DØ, 2147483647.DØ).  DSEED IS REPLACED BY
C                           A NEW VALUE TO BE USED IN A SUBSEQUENT CALL.
C
C      XM                 INPUT MEAN VALUE.
C
C      NR                 NUMBER OF DEVIATES TO BE GENERATED (INPUT).
C
C      R(I)               OUTPUT VECTOR OF LENGTH NR CONTAINING THE
C                           EXPONENTIAL DEVIATES.
C
C
C
C
      INTEGER NR,I
      REAL XM,R(NR)
      DOUBLE PRECISION DSEED
      CALL RANDUFM(DSEED,NR,R)
      DO 5 I=1,NR
5     R(I) = -XM*ALOG(R(I))
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUPDRV(N1,X1,N2,X2,ICK,P)
C
C
C   PURPOSE      FIND THE FINITE LINE INTERSECTION BETWEEN TWO FINITE PLANES.
C                ON OUTPUT, THE LINE SEGMENT IS REPRESENTED BY TWO END POINTS
C                OF THE LINE.
C
C   ARGUMENTS:
C
C      N1,N2              INDEX DEFINING THE SHAPE OF THE FINITE PLANE;
C                         CIRCLE = 1, ELLIPSE = 2, SQUARE = 3, RECTANGLE = 4.
C
C      X1(I),X2(I)        INPUT VECTORS CONTAINING THE GEOMETRY PARAMETERS
C                           USED TO SPECIFY TWO FINITE PLANES.
C                         I=1,3 ARE GLOBAL X-Y-Z COORDINATES DENOTING THE
C                           THE CENTER LOCATION OF THE PLANES.
C                         I=4,6 ARE ANGLES (IN DEGREES) SPECIFYING THE
C                           ORIENTATION OF THE PLANES.
C                         I=7,8 ARE CHARACTERISTIC LENGTH SCALES ASSOCIATED
C                           WITH THE SHAPE OF THE FINITE PLANE.
C
C      ICK                OUTPUT INDEX, ICK = 2 IF AN INTERSECTING LINE SEGMENT
C                           EXISTS; ICK = Ø IF TWO FINITE PLANES DO NOT
C                           INTERSECT; ICK = 1 INDICATES POSSIBLE ERROR
C                           CONIDITION IN THIS SUBROUTINE.
C
C      P(I,J)             OUTPUT MATRIX OF 3*2 CONTAINING THE GLOBAL X-Y-Z
C                           COORDINATES (I=1,3) OF TWO END POINTS (J=1,2)
C                           OF THE INTERSCTING LINE SEGMENT.
C
C
C
      REAL X1(9),X2(9),P(3,2)
      REAL TG1(3,2),TG2(3,2),TL1(2,2),TL2(2,2),TX(3,2)
      REAL TP1(2,2),TP2(2,2),TOUT(3,4)
      TERR = 1.E-9
      CALL SUPINT(X1,X2,ICK,TX)
      IF (ICK .EQ. Ø) RETURN
      CALL SUPGTL(2,TX,TL1,X1)
      CALL SUPGTL(2,TX,TL2,X2)
      IF (N1 .LE. 2) THEN
      CALL SUPCIR(TL1,X1(7),X1(8),IC1,TP1)
      ELSE
      CALL SUPRET(TL1,X1(7),X1(8),IC1,TP1)
      ENDIF
      IF (N2 .LE. 2) THEN
      CALL SUPCIR(TL2,X2(7),X2(8),IC2,TP2)
```

```
      ELSE
      CALL SUPRET(TL2,X2(7),X2(8),IC2,TP2)
      ENDIF
      ICK = IC1 * IC2
      IF (ICK .EQ. Ø) RETURN
      ICK = Ø
      CALL SUPLTG(2,TP1,TG1,X1)
      CALL SUPLTG(2,TP2,TG2,X2)
      CALL SUPGTL(2,TG1,TP1,X2)
      CALL SUPGTL(2,TG2,TP2,X1)
      DO 1Ø J = 1 , 2
C    TP1 ON 2
      IIC = Ø
      IF (N2 .LE. 2) THEN
      IIC = ICKCIR ( TP1(1,J),TP1(2,J),X2(7),X2(8) )
      ELSE
      IIC = ICKRET ( TP1(1,J),TP1(2,J),X2(7),X2(8) )
      ENDIF
      IF ( IIC .EQ. 1) THEN
      ICK = ICK + 1
      DO 12 J1 = 1, 3
12    TOUT(J1,ICK) = TG1(J1,J)
      ENDIF
C    TP2 ON 1
      IIC = Ø
      IF (N1 .LE. 2) THEN
      IIC = ICKCIR ( TP2(1,J),TP2(2,J),X1(7),X1(8) )
      ELSE
      IIC = ICKRET ( TP2(1,J),TP2(2,J),X1(7),X1(8) )
      ENDIF
      IF ( IIC .EQ. 1) THEN
      ICK = ICK + 1
      DO 14 J1 = 1, 3
14    TOUT(J1,ICK) = TG2(J1,J)
      ENDIF
1Ø    CONTINUE
      IF (ICK .EQ. 2) THEN
      DO 2Ø J1 = 1, 3
      P(J1,1) = TOUT(J1,1)
2Ø    P(J1,2) = TOUT(J1,2)
      ENDIF
      IF (ICK .GT. 2) THEN
      P(1,1) = TOUT(1,1)
      P(2,1) = TOUT(2,1)
      P(3,1) = TOUT(3,1)
      IIC = 1
      DO 22 J1 = 2, ICK
      IF ( (ABS(TOUT(1,J1)-P(1,1)) .LT. TERR)  .AND.
     1     (ABS(TOUT(2,J1)-P(2,1)) .LT. TERR)  .AND.
     1     (ABS(TOUT(3,J1)-P(3,1)) .LT. TERR)       ) THEN
```

```
      IIC = IIC + 1
      JC  = J1
      ENDIF
22    CONTINUE
      IF ( IIC .EQ. 2 ) THEN
      P(1,2) = TOUT(1,JC)
      P(2,2) = TOUT(2,JC)
      P(3,2) = TOUT(3,JC)
      ICK = IIC
      ELSE
      ICK = Ø
      WRITE(9,1ØØ)
1ØØ   FORMAT(2X,'*****  WARNING ON SUBROUTINE SUPDRV  ***** ')
      ENDIF
      ENDIF
      IF (ICK .EQ. 1) WRITE (9,1ØØ)
      RETURN
      END
```

```
C
C
C
      FUNCTION ICKCIR(X,Y,R1,R2)
C
C
C   PURPOSE      CHECK IF A POINT (X,Y) IS LOCATED INSIDE AN ELLIPSE.
C                IF TRUE, ICKCIR = 1; OTHERWISE ICKCIR = 0.
C
C   ARGUMENTS:
C
C       X,Y               COORDINATES OF A POINT (INPUT).
C
C       R1,R2             CHARACTERISTIC LENGTH SCALE DEFINING THE ELLIPSE.
C                           THE BOUNDARY THE ELLIPSE IS GIVEN BY:
C                           (X*X)/(R1*R1) + (Y*Y)/(R2*R2) = 1  (INPUT).
C
C
C
      REAL X,Y,R1,R2,TEM
      ICKCIR = 0
      TEM = X*X / (R1*R1)  +  Y*Y / (R2*R2)
      IF ( TEM .LE. 1. ) ICKCIR = 1
      RETURN
      END


C
C
      FUNCTION ICKRET(X,Y,R1,R2)
C
C
C   PURPOSE      CHECK IF A POINT (X,Y) IS LOCATED INSIDE A RECTANGLE.
C                IF TRUE, ICKRET = 1, OTHERWISE ICKRET = 0.
C
C   ARGUMENTS:
C
C       X,Y               COORDINATES OF A POINT (INPUT).
C
C       R1,R2             CHARACTERISTIC LENGTH SCALE DEFINING THE RECTANGLE.
C                           THE AREA OF THE RECTANGLE IS BOUND BY
C                           ABS(X) <= R1  AND  ABS(Y) <= R2  (INPUT).
C
C
C
      REAL X,Y,R1,R2
      ICKRET = 0
      IF ( (ABS(X) .LE. R1)  .AND. (ABS(Y) .LE. R2) ) ICKRET = 1
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUPLTG(NI,XYL,XYZG,XINP)
C
C
C   PURPOSE      PERFORM COORDINATE TRANSFORMATION FROM A LOCAL 2-D
C                COORDINATE SYSTEM DEFINED FOR EACH INDIVIDUAL FINITE
C                PLANE TO THE GLOBAL 3-D COORDINATES.
C
C   ARGUMENTS:
C
C      NI               NO. OF TRANSFORMATIONS (INPUT).
C
C      XYL(I,J)         INPUT MATRIX OF 2*NI CONTAINING THE LOCAL X-Y
C                         COORDINATES (I=1,2) OF POINT J.
C
C      XYZG(I,J)        OUTPUT MATRIX OF 3*NI CONTAINING THE GLOBAL X-Y-Z
C                         COORDINATES (I=1,3) OF POINT J.
C
C      XINP(I)          INPUT VECTOR CONTAINING THE GEOMETRY PARAMETERS
C                         USED TO SPECIFY A FINITE PLANE.
C                       I=1,3 ARE GLOBAL X-Y-Z COORDINATES DENOTING THE
C                         THE CENTER LOCATION OF THE PLANE.
C                       I=4,6 ARE ANGLES (IN DEGREES) SPECIFYING THE
C                         ORIENTATION OF THE FINITE PLANE.
C
C
      INTEGER NI
      REAL XYL(2,NI),XYZG(3,NI),XINP(9),XA2(3),XA3(3)
      DATA PI,DG2RAD / 3.141592654,0.0174532925199433/
      AALP = XINP(4) * DG2RAD
      ABTA = XINP(5) * DG2RAD
      ATHT = XINP(6) * DG2RAD
      XA2(1)= SIN(ABTA)*COS(AALP)
      XA2(2)= SIN(ABTA)*SIN(AALP)
      XA2(3)=-COS(ABTA)
      XA3(1)=-SIN(AALP)
      XA3(2)= COS(AALP)
      XA3(3)= 0.0
      DO 10 N = 1,NI
      XB = XYL(1,N)*COS(ATHT) - XYL(2,N)*SIN(ATHT)
      YB = XYL(1,N)*SIN(ATHT) + XYL(2,N)*COS(ATHT)
      DO 10 J = 1, 3
10    XYZG(J,N) = XINP(J) + XB*XA2(J) + YB*XA3(J)
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUPGTL(NI,TP,XYL,XCN)
C
C
C   PURPOSE      PERFORM COORDINATE TRANSFORMATION FROM THE GLOBAL 3-D
C                COORDINATES TO A LOCAL 2-D SYSTEM DEFINED FOR EACH
C                INDIVIDUAL FINITE PLANE.
C
C   ARGUMENTS:
C
C      NI              NO. OF TRANSFORMATIONS (INPUT).
C
C      TP(I,J)         INPUT MATRIX OF 3*NI CONTAINING THE GLOBAL X-Y-Z
C                        COORDINATES (I=1,3) OF POINT J.
C
C      XYL(I,J)        OUTPUT MATRIX OF 2*NI CONTAINING THE LOCAL X-Y
C                        COORDINATES (I=1,2) OF POINT J.
C
C      XCN(I)          INPUT VECTOR CONTAINING THE GEOMETRY PARAMETERS
C                        USED TO SPECIFY A FINITE PLANE.
C                      I=1,3 ARE GLOBAL X-Y-Z COORDINATES DENOTING THE
C                        THE CENTER LOCATION OF THE PLANE.
C                      I=4,6 ARE ANGLES (IN DEGREES) SPECIFYING THE
C                        ORIENTATION OF THE FINITE PLANE.
C
C
C
      INTEGER NI
      REAL XYL(2,NI),TP(3,NI),XCN(9),XC2(3),XC3(3)
      DATA TERR / 1.E-9/
      DATA PI,DG2RAD / 3.141592654,0.0174532925199433/
      AALP = XCN(4) * DG2RAD
      ABTA = XCN(5) * DG2RAD
      ATHT = XCN(6) * DG2RAD
      XC2(1)= SIN(ABTA)*COS(AALP)
      XC2(2)= SIN(ABTA)*SIN(AALP)
      XC2(3)=-COS(ABTA)
      XC3(1)=-SIN(AALP)
      XC3(2)= COS(AALP)
      XC3(3)= 0.0
      DO 10 N = 1, NI
      IF ( ABS(XC2(3)) .GT. TERR ) THEN
      XB=(TP(3,N)-XCN(3))/XC2(3)
      ELSE
      XB=(XC3(2)*(TP(1,N)-XCN(1))-XC3(1)*(TP(2,N)-XCN(2)))
     1     / (XC3(2)*XC2(1)-XC3(1)*XC2(2))
      ENDIF
      IF ( ABS(XC[illegible])) .GT. TERR ) THEN
```

```
      YB=(TP(1,N)-XCN(1)-XB*XC2(1))/XC3(1)
      ELSE
      YB=(TP(2,N)-XCN(2)-XB*XC2(2))/XC3(2)
      ENDIF
      XYL(1,N) = XB*COS(ATHT) + YB*SIN(ATHT)
      XYL(2,N) =-XB*SIN(ATHT) + YB*COS(ATHT)
10    CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUPRET(P,R1,R2,ICUT,CX)
C
C
C   PURPOSE      FIND POINT INTERSECTIONS BETWEEN A LINE AND THE BOUNDARY
C                OF A RECTANGLE.
C
C   ARGUMENTS:
C
C       P(I,J)           INPUT MATRIX OF 2*2 CONTAINING THE LOCAL X-Y
C                          COORDINATES (I=1,2) OF TWO DISTINCT POINTS (J=1,2)
C                          ON THE LINE.
C
C       R1,R2            CHARACTERISTIC LENGTH SCALE DEFINING THE RECTANGLE.
C                          THE AREA OF THE RECTANGLE IS BOUND BY
C                          ABS(X) <= R1  AND  ABS(Y) <= R2  (INPUT).
C
C       ICUT             OUTPUT INDEX, ICUT = 1 IF THE LINE INTERSECTS THE
C                          RECTANGLE; OTHERWISE ICUT = Ø.
C
C       CX(I,J)          OUTPUT MATRIX OF 2*2 CONTAINING THE LOCAL X-Y
C                          COORDINATES (I=1,2) OF TWO POINT INTERSECTIONS
C                          (J=1,2) BETWEEN THE LINE AND THE RECTANGLE.
C                          CX(I,J) IS NOT USED WHEN ICUT = Ø.
C
C
C
      REAL P(2,2),CX(2,2)
      REAL CT(2,4),XX(4),YY(4)
      TERR = 1.E-9
      IF((ABS(R1) .LT. TERR) .OR. (ABS(R2) .LT. TERR)) THEN
      ICUT = Ø
      RETURN
      ENDIF
      A = P(2,2) - P(2,1)
      B = P(1,1) - P(1,2)
      C = -A*P(1,1) + (-B)*P(2,1)
      ICUT=Ø
      IF ( ABS(A) .LT. TERR) THEN
       Y = -C / B
       IF ( ABS(Y) .LE. R2) THEN
        ICUT=1
        CX(1,1) = R1
        CX(1,2) =-R1
        CX(2,1) = Y
        CX(2,2) = Y
       ELSE
        ICUT=Ø
```

```
      ENDIF
     ELSE
      IF ( ABS(B) .LT. TERR) THEN
       X = -C / A
       IF ( ABS(X) .LE. R1) THEN
        ICUT=1
        CX(1,1) = X
        CX(1,2) = X
        CX(2,1) = R2
        CX(2,2) =-R2
       ELSE
        ICUT=Ø
       ENDIF
      ELSE
       XX(1) = R1
       YY(1) = (-A*R1-C)/B
       XX(2) =-R1
       YY(2) = ( A*R1-C)/B
       YY(3) = R2
       XX(3) = (-B*R2-C)/A
       YY(4) =-R2
       XX(4) = ( B*R2-C)/A
       ICCT=Ø
       DO 1Ø IC = 1,4
       IF((ABS(XX(IC)).LE. R1) .AND. (ABS(YY(IC)).LE. R2)) THEN
       ICCT = ICCT + 1
       CT(1,ICCT) = XX(IC)
       CT(2,ICCT) = YY(IC)
       ENDIF
1Ø     CONTINUE
       IF (ICCT .EQ. Ø) THEN
       ICUT = Ø
       ELSE
       CX(1,1) = CT(1,1)
       CX(2,1) = CT(2,1)
       JCT = 1
       DO 15 J= 2,ICCT
       IF ( CT(1,J) .NE. CT(1,1)) THEN
        CX(1,2) = CT(1,J)
        CX(2,2) = CT(2,J)
        JCT = 2
       ENDIF
15     CONTINUE
       ICUT = 1
       IF (JCT .EQ. 1) ICUT=Ø
       ENDIF
      ENDIF
     ENDIF
     RETURN
     END
```

```
C
C
C
      SUBROUTINE SUPCIR(P,R1,R2,ICUT,CT)
C
C
C   PURPOSE      FIND POINT INTERSECTIONS BETWEEN A LINE AND THE BOUNDARY
C                OF A ELLIPSE.
C
C   ARGUMENTS:
C
C     P(I,J)           INPUT MATRIX OF 2*2 CONTAINING THE LOCAL X-Y
C                        COORDINATES (I=1,2) OF TWO DISTINCT POINTS (J=1,2)
C                        ON THE LINE.
C
C     R1,R2            CHARACTERISTIC LENGTH SCALE DEFINING THE ELLIPSE.
C                        THE BOUNDARY THE ELLIPSE IS GIVEN BY:
C                        (X*X)/(R1*R1) + (Y*Y)/(R2*R2) = 1  (INPUT).
C
C     ICUT             OUTPUT INDEX, ICUT = 1 IF THE LINE INTERSECTS THE
C                        ELLIPSE; OTHERWISE ICUT = Ø.
C
C     CT(I,J)          OUTPUT MATRIX OF 2*2 CONTAINING THE LOCAL X-Y
C                        COORDINATES (I=1,2) OF TWO POINT INTERSECTIONS
C                        (J=1,2) BETWEEN THE LINE AND THE ELLIPSE.
C                        CT(I,J) IS NOT USED WHEN ICUT = Ø.
C
C
C
      REAL P(2,2),CT(2,2)
      TERR = 1.E-9
      IF((ABS(R1) .LT. TERR) .OR. (ABS(R2) .LT. TERR)) THEN
      ICUT = Ø
      RETURN
      ENDIF
      A = P(2,2) - P(2,1)
      B = P(1,1) - P(1,2)
      C = -A*P(1,1) + (-B)*P(2,1)
      IF ( ABS(B) .GT. TERR ) THEN
      R2B = R2*R2*B*B
      AX = 1./(R1*R1) + A*A / R2B
      BX = 2.*A*C / R2B
      CX = C*C / R2B  - 1
      BAC = BX*BX - 4.*AX*CX
      IF (BAC .GT. Ø) THEN
      ICUT=1
      CT(1,1) = (-BX + SQRT(BAC))/ (2.*AX)
      CT(2,1) = (- A*CT(1,1) - C) / B
      CT(1,2) = (-BX - SQRT(BAC))/ (2.*AX)
      CT(2,2) = (- A*CT(1,2) - C) / B
```

```
ELSE
ICUT=Ø
ENDIF
ELSE
R1A = R1*R1*A*A
AX = 1./(R2*R2) + B*B / R1A
BX = 2.*B*C / R1A
CX = C*C / R1A  - 1
BAC = BX*BX - 4.*AX*CX
IF (BAC .GT. Ø) THEN
ICUT=1
CT(2,1) = (-BX + SQRT(BAC))/ (2.*AX)
CT(1,1) = (- B*CT(2,1) - C) / A
CT(2,2) = (-BX - SQRT(BAC))/ (2.*AX)
CT(1,2) = (- B*CT(2,2) - C) / A
ELSE
ICUT=Ø
ENDIF
ENDIF
RETURN
END
```

```
C
C
C
      SUBROUTINE SUPINT(XAN,XBN,INCT,P)
C
C
C   PURPOSE     FIND THE LINE INTERSECTION BETWEEN TWO INFINITE PLANES.
C               ON OUTPUT, THE LINE INTERSECTION IS REPRESENTED BY TWO
C               DISTINCT POINTS ON THE LINE.
C
C   ARGUMENTS:
C
C      XAN(I),XBN(I)    INPUT VECTORS CONTAINING THE GEOMETRY PARAMETERS
C                         USED TO SPECIFY TWO INFINITE PLANES.
C                       I=1,3 ARE GLOBAL X-Y-Z COORDINATES DENOTING THE
C                         THE CENTER LOCATION OF THE PLANES.
C                       I=4,6 ARE ANGLES (IN DEGREES) SPECIFYING THE
C                         ORIENTATION OF THE PLANES.
C
C      INCT             OUTPUT INDEX, INCT = 1 IF A LINE INTERSECTION EXISTS;
C                         OTHERWISE INCT = 0.
C
C      P(I,J)           OUTPUT MATRIX OF 3*2 CONTAINING THE GLOBAL X-Y-Z
C                         COORDINATES (I=1,3) OF TWO DISTINCT POINTS (J=1,2)
C                         ON THE LINE INTERSCTION.
C
C
C
      REAL XAN(9),XBN(9),P(3,2)
      REAL XA(4),XB(4),R1(3),R2(3)
      DATA TERR / 1.E-9/
      DATA PI,DG2RAD / 3.141592654,0.0174532925199433/
      XA (1)=COS(XAN(5)*DG2RAD)*COS(XAN(4)*DG2RAD)
      XA (2)=COS(XAN(5)*DG2RAD)*SIN(XAN(4)*DG2RAD)
      XA (3)=SIN(XAN(5)*DG2RAD)
      XA (4)=XA(1)*XAN(1)+XA(2)*XAN(2)+XA(3)*XAN(3)
      XB (1)=COS(XBN(5)*DG2RAD)*COS(XBN(4)*DG2RAD)
      XB (2)=COS(XBN(5)*DG2RAD)*SIN(XBN(4)*DG2RAD)
      XB (3)=SIN(XBN(5)*DG2RAD)
      XB (4)=XB(1)*XBN(1)+XB(2)*XBN(2)+XB(3)*XBN(3)
      IF ((ABS(XA(1)*XB(2)-XB(1)*XA(2)) .LT. TERR) .AND.
     1     (ABS(XA(2)*XB(3)-XB(2)*XA(3)) .LT. TERR) .AND.
     2     (ABS(XA(3)*XB(1)-XB(3)*XA(1)) .LT. TERR)) THEN
        INCT=0
      ELSE
        INCT=1
        R00=XA(1)*XB(2)-XB(1)*XA(2)
      IF (ABS(R00) .GT. TERR) THEN
      R1(1)=(XB(3)*XA(2)-XA(3)*XB(2))/R00
      R1(2)=(XB(1)*XA(3)-XA(1)*XB(3))/R00
```

```
      R1(3)=1.
      R2(1)=(XB(2)*XA(4)-XA(2)*XB(4))/R00
      R2(2)=(XB(4)*XA(1)-XA(4)*XB(1))/R00
      R2(3)=0.
      ELSE
      R00=XA(2)*XB(3)-XB(2)*XA(3)
      IF (ABS(R00) .GT. TERR) THEN
      R1(1)=1.
      R1(2)=(XB(1)*XA(3)-XA(1)*XB(3))/R00
      R1(3)=(XB(1)*XA(2)-XA(1)*XB(2))/R00
      R2(1)=0.
      R2(2)=(XB(3)*XA(4)-XA(3)*XB(4))/R00
      R2(3)=(XB(4)*XA(2)-XA(4)*XB(2))/R00
      ELSE
      R00=XA(3)*XB(1)-XB(3)*XA(1)
      R1(1)=(XB(3)*XA(2)-XA(3)*XB(2))/R00
      R1(2)=1.
      R1(3)=(XB(2)*XA(1)-XA(2)*XB(1))/R00
      R2(1)=(XB(4)*XA(3)-XA(4)*XB(3))/R00
      R2(2)=0.
      R2(3)=(XB(1)*XA(4)-XA(1)*XB(4))/R00
      ENDIF
      ENDIF
      DO 10 J=1,3
      P(J,1)= R1(J) * 1.0 + R2(J)
      P(J,2)=-R1(J) * 1.0 + R2(J)
10    CONTINUE
      ENDIF
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUBLEQS (A,M,N,B,IB,D1,D2,IER)
C
C
C   PURPOSE      LINEAR EQUATION SOLUTION OF A SYMMETRIC, POSITIVE DEFINITE
C                MATRIX.
C
C   ARGUMENTS:
C
C      A(I)                INPUT VECTOR OF LENGTH N(N+1)/2 CONTAINING THE N BY N
C                            COEFFICIENT MATRIX OF THE EQUATION AX = B.  A IS
C                            A POSITIVE DEFINITE SYMMETRIC MATRIX STORED IN
C                            SYMMETRIC STORAGE MODE.
C
C      M                   NUMBER OF RIGHT HAND SIDES OR COLUMNS IN B (INPUT).
C
C      N                   ORDER OF A AND NUMBER OF ROWS IN B (INPUT).
C
C      B(I,J)              INPUT MATRIX OF N * M CONTAINING THE RIGHT-HAND SIDES
C                            OF THE EQUATION AX = B.
C                          ON OUTPUT, THE N BY M SOLUTION MATRIX X REPLACES B.
C
C      IB                  ROW DIMENSION OF B EXACTLY AS SPECIFIED IN THE
C                            DIMENSION STATEMENT IN THE CALLING PROGRAM (INPUT).
C
C      D1,D2               COMPONENTS OF THE DETERMINANT OF A, WHERE:
C                            DETERMINANT(A) = D1*2.**D2   (OUTPUT).
C
C      IER                 ERROR PARAMETER (OUTPUT).
C                            IER = 1 INDICATES THAT THE INPUT MATRIX A IS
C                            ALGORITHMICALLY NOT POSITIVE DEFINITE.
C
C
C
      DIMENSION A(1),B(IB,1)
      IER = Ø
C                                 DECOMPOSE A
      CALL SUBDECP (A,A,N,D1,D2,IER)
      IF (IER.NE.Ø) RETURN
C                                 PERFORM ELIMINATION
      DO 5 I = 1,M
       CALL SUBLUEM (A,B(1,I),N,B(1,I))
5     CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUBDECP (A,UL,N,D1,D2,IER)
C
C
C   PURPOSE      DECOMPOSITION OF A SYMMETRIC POSITIVE DEFINITE MATRIX
C                USING THE CHOLESKI'S METHOD.
C
C
C   ARGUMENTS:
C
C      A(I)              INPUT VECTOR OF LENGTH N(N+1)/2 CONTAINING THE N*N
C                          POSITIVE DEFINITE SYMMETRIC MATRIX STORED IN
C                          SYMMETRIC STORAGE MODE.
C
C      UL(I)             OUTPUT VECTOR OF LENGTH N(N+1)/2 CONTAINING THE
C                          DECOMPOSED MATRIX L SUCH THAT A = L*L-TRANSPOSE.
C                          L IS STORED IN SYMMETRIC STORAGE MODE.
C
C      N                 ORDER OF A  (INPUT).
C
C      D1,D2             COMPONENTS OF THE DETERMINANT OF A, WHERE:
C                          DETERMINANT(A) = D1*2.**D2  (OUTPUT).
C
C      IER               ERROR PARAMETER (OUTPUT).
C                          IER = 1 INDICATES THAT THE INPUT MATRIX A IS
C                          ALGORITHMICALLY NOT POSITIVE DEFINITE.
C
C
C
      DIMENSION A(1),UL(1)
      DATA  ZERO,ONE,FOUR,SIXTN,SIXTH/
     1        0.0, 1.,  4.,  16.,.0625/
      D1=ONE
      D2=ZERO
      RN = ONE/(N*SIXTN)
      IP = 1
      IER=0
      DO 45 I = 1,N
        IQ = IP
        IR = 1
        DO 40 J = 1,I
          X = A(IP)
          IF (J .EQ. 1) GO TO 10
          DO 5  K=IQ,IP1
            X = X - UL(K) * UL(IR)
            IR = IR+1
5         CONTINUE
10        IF (I.NE.J) GO TO 30
```

```
      D1 = D1*X
      IF (A(IP) + X*RN .LE. A(IP)) GO TO 50
15    IF (ABS(D1).LE.ONE) GO TO 20
      D1 = D1 * SIXTH
      D2 = D2 + FOUR
      GO TO 15
20    IF (ABS(D1) .GE. SIXTH) GO TO 25
      D1 = D1 * SIXTN
      D2 = D2 - FOUR
      GO TO 20
25    UL(IP) = ONE/SQRT(X)
      GO TO 35
30    UL(IP) = X * UL(IR)
35    IP1 = IP
      IP = IP+1
      IR = IR+1
40   CONTINUE
45  CONTINUE
    RETURN
50  IER = 1
    RETURN
    END
```

```
C
C
C
      SUBROUTINE SUBLUEM (A,B,N,X)
C
C
C   PURPOSE      ELIMINATION PART OF THE SOLUTION OF AX=B, WHERE A IS A
C                SYMMETRIC, POSITIVE DEFINITE MATRIX.
C
C   ARGUMENTS:
C
C     A(I)              INPUT VECTOR OF LENGTH N(N+1)/2 CONTAINING THE N * N
C                         MATRIX L WHERE A = L*L-TRANSPOSE.  L IS A LOWER
C                         TRIANGULAR MATRIX STORED IN SYMMETRIC STORAGE MODE.
C
C     B(I)              INPUT VECTOR OF LENGTH N CONTAINING THE RIGHT HAND
C                         SIDE OF THE EQUATION AX = B.
C
C     N                 ORDER OF A AND THE LENGTH OF B AND X (INPUT).
C
C     X(I)              OUTPUT VECTOR OF LENGTH N CONTAINING THE SOLUTION TO
C                         THE EQUATION AX = B.
C
C
C
      DIMENSION A(1),B(1),X(1)
      DATA ZERO/0.0/
C                                  SOLUTION OF LY = B
      IP=1
      IW = 0
      DO 15 I=1,N
        T=B(I)
        IM1 = I-1
        IF (IW .EQ. 0) GO TO 9
        IP=IP+IW-1
        DO 5 K=IW,IM1
          T = T-A(IP)*X(K)
          IP=IP+1
5       CONTINUE
        GO TO 10
9       IF (T .NE. ZERO) IW = I
        IP = IP+IM1
10      X(I)=T*A(IP)
        IP=IP+1
15    CONTINUE
C                                  SOLUTION OF UX = Y
      N1 = N+1
      DO 30 I = 1,N
        II = N1-I
        IP=IP-1
```

```
          IS=IP
          IQ=II+1
          T=X(II)
          IF (N.LT.IQ) GO TO 25
          KK = N
          DO 20 K=IQ,N
            T = T - A(IS) * X(KK)
            KK = KK-1
            IS = IS-KK
20        CONTINUE
25      X(II)=T*A(IS)
30      CONTINUE
        RETURN
        END
```

```
C
C        A.4  SAMPLE INPUT/OUTPUT FOR EXPERIMENT 1
C
C
C        SAMPLE OF INPUT DECK SETUP FOR EXPERIMENT 1
C
C
#EOR                                                (END OF RECORD)
200, 200., 200., 200.                               (NFRCT, FRX, FRY, FRZ)
50.,  10.                                           (FRCTLTH, FRCAPTR    )
918.,  35., 763.,  849.                             (DS1, DS2, DS3, DS4  )
3                                                   (NSET                )
100.,100.,130.,  0.,    0., 100., 100., 100.  (CORØ,COAL,COSZ)
100., 100., 125.,  0.,   0., 110., 110., 110.  (CORØ,COAL,COSZ)
100., 100., 120.,  0.,   0., 120., 120., 120.  (CORØ,COAL,COSZ)
#EOR                                                (END OF RECORD )
#EOF                                                (END OF FILE   )
```

```
*
*          SAMPLE OUTPUT FROM EXPERIMENT 1
*
*
*
*          ECHO INPUT PARAMETERS
*

NFRCT,FRX,FRY,FRZ     :        200    200.0    200.0    200.0
FRCTLTH, FRCAPTR      :       50.0     10.0
SEED--DS1,DS2,DS3,DS4 :      918.0     35.0    763.0    849.0

*
* ISOLATED SAMPLE # 1 AND THE FLUXES ACROSS THE BOUNDARY SURFACES.
* NTS INDICATES DIFFERENT SAMPLE ORIENTATIONS, NEGATIVE FLUX IMPLIES
* THE FLUX INTO THE SAMPLE BLOCK.
*

SAMPLE CENTER  (COR0,1) :      100.0    100.0    130.0
SAMPLE ANGLE   (COAL,1) :         .0       .0
SAMPLE SIZE    (COSZ,1) :      100.0    100.0    100.0

(BONDARY          +X          -X          +Y          -Y          +Z          -Z )

 NTS=1      -2668.66      127.62      489.19     1183.36      739.63      128.86
 NTS=2        127.62    -1594.47      168.68      387.45      435.84      474.89
 NTS=3        489.19      168.68    -1165.08       28.51      445.85       32.85
 NTS=4       1183.36      387.45       28.51    -3228.39     1070.14      558.93
 NTS=5        739.63      435.84      445.85     1070.14    -2821.24      129.78
 NTS=6        128.86      474.89       32.85      558.93      129.78    -1325.30


*
* ISOLATED SAMPLE # 2 AND THE TEST RESULTS.
*

SAMPLE CENTER  (COR0,1) :      100.0    100.0    125.0
SAMPLE ANGLE   (COAL,1) :         .0       .0
SAMPLE SIZE    (COSZ,1) :      110.0    110.0    110.0

(BONDARY          +X          -X          +Y          -Y          +Z          -Z )

 NTS=1      -1940.17       33.40      393.26      712.43      697.57      103.51
 NTS=2         33.40    -1405.49      155.85      496.35      274.94      444.95
 NTS=3        393.26      155.85    -1564.32       29.73      623.12      362.37
 NTS=4        712.43      496.35       29.73    -3296.71     1382.42      675.77
 NTS=5        697.57      274.94      623.12     1382.42    -3144.93      166.88
 NTS=6        103.51      444.95      362.37      675.77      166.88    -1753.49
```

A.5 PROGRAM LISTING OF PART 1 OF EXPERIMENT 2

```
      PROGRAM EXP2R1(INPUT,OUTPUT,DATA,TAPE1=INPUT,TAPE10=DATA,
     1              TAPE8=OUTPUT)
C
C
C  PURPOSE:     THIS PROGRAM PERFORMS THE FIRST PART OF THE HYDRAULIC-
C               TRACER INJECTION EXPERIMENT.  A HYDRAULIC INJECTION
C               TEST IS SET UP AND THE HEAD DISTRIBUTION IS CALCULATED.
C               THE FRACTURE NETWORK INFORMATION AND THE HYDRAULIC DATA
C               ARE STORED IN A TEMPORARY DATA FILE, AND ARE LATER USED
C               BY A SECOND PROGRAM AS INPUT TO PERFORM THE BREAKTHROUGH
C               CALCULATION.
C
C  SUBROUTINES USED:
C
C       INPFRCT          GENERATE PARAMETERS DESCRIBING EACH FRACTURE.
C
C       INPCUT           GENERATE PARAMETERS DESCRIBING THE RECTANGULAR
C                          BOUNDARY SURFACES OF AN ISOLATED BLOCK OF
C                          SAMPLE AND A RECTANGULAR VOLUME INSIDE THE
C                          SAMPLE ACTING AS THE INJECTION ZONE.
C
C       SUPFRCT          FIND FRACTURE-FRACTURE INTERSECTIONS.
C
C       SUPCUT           FIND FRACTURE-BOUNDARY SURFACE INTERSECTIONS.
C
C       SUPDEAD          DELETE DEAD-END FRACTURES.
C
C       SUBBOND          ASSIGN BOUNDARY CONDITIONS.
C
C       SUBMTRX          ASSEMBLE THE GLOBAL MATRIX EQUATION AND SOLVE
C                          FOR THE HEAD DISTRIBUTION IN THE SAMPLE.
C
C       OUTRUN2          WRITE THE FRACTURE NETWORK INFORMATION AND THE
C                          HYDRAULIC DATA INTO A TEMPORARY FILE TO BE USED
C                          IN RUN2 OF THE EXPERIMENT.
C
C  REMARKS:     THE FIRST-ORDER SUBROUTINES (CALLED BY THE MAIN PROGRAM)
C               *INPFRCT*, *SUPFRCT*, *SUPCUT* AND *SUPDEAD* AND ALL
C               SECONDARIES (THOSE CALLED BY OTHER SUBROUTINES) ARE EXACTLY
C               THE SAME AS THOSE LISTED UNDER EXPERIMENT 1.   SUBROUTINES
C               *INPCUT*, *SUBBOND*  AND *SUBMTRX* DIFFER SLIGHTLY FROM
C               THOSE LISTED IN EXPERIMENT 1.  SUBROUTINE *OUTRUN2* IS
C               DESIGNED SPECIFICALLYIN FOR THE RUN1 OF THE TWO-PART
C               EXPERIMENT TO CARRY THE DATA OVER TO THE NEXT STAGE OF THE
C               EXPERIMENT.
C
C              ALL VARIABLES HAVE EXACTLY THE SAME MEANING AS GIVEN
C               IN PROGRAM 1.  THE EXCEPTIONS WILL BE IDENTIFIED WHEN
C               THEY APPEAR.
C
C
C
```

```
C
C
      COMMON /FRCT/ NFRCT,NSHF(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CFCF/ NCF(200),NCFN(300)
      COMMON /NODE/ NNODE,NDID(400,2),NDFR(1000)
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      COMMON /MTRX/ PO(400), PX(400)
      CALL INPFRCT
      CALL INPCUT
      CALL SUPFRC
      CALL SUPCUT
      CALL SUPDEAD
      CALL SUBBOND
      CALL SUBMTRX
      CALL OUTRUN2
      STOP
      END
```

```
C
C
C
      SUBROUTINE INPCUT
C
C   PURPOSE:     THIS SUBROUTINE READS IN THE CENTER LOCATION (CORØ),
C                ROTATION (COAL) AND SIZE (COSZ) OF A RECTANGULAR BLOCK
C                OF SAMPLE (K1=1) AND A RECTANGULAR ZONE OF INJECTION
C                (K1=2) AND GENERATES THE GEOMETRIC PARAMETERS FOR EACH
C                OF THE RECTANGULAR BOUNDARY SURFACES.   THUS, BOUNDARY
C                SURFACES 1-6 ARE THOSE OF THE SAMPLE BLOCK AND THE SURFACES
C                7-12 ARE INJECTION PLANES.   THIS SUB-PROGRAM CHANGES THE
C                CONTENTS OF THE FILLOWING VARIABLES: NCUT, CORØ, COAL,
C                COSZ, XCUT, CORA, CORB.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CORD/ CORØ(3,2),COSZ(3,2),COAL(2,2),
     1              CORA(3,3,2),CORB(3,3,2)
      REAL WKALP(12),WKBTA(12),WKX(3),WKYZ(18)
      INTEGER NWAL(6), NWBT(6)
      DATA PI,DG2RAD / 3.141592654,0.0174532925199433/
      DATA WKALP /1.,0., 1.,180., 1.,90., 1.,270., 1.,0.,1.,0./
      DATA WKBTA /1.,0.,-1.,0., 0.,0., 0.,0., 1.,90., 1.,270./
      DATA NWAL  / 3, 3, 3, 3, 1, 1 /
      DATA NWBT  / 2, 2, 1, 1, 2, 2 /
      DATA WKYZ  /1.,0.,0., -1.,0.,0., 0.,1.,0., 0.,-1.,0.,
     1            0.,0.,1., 0.,0.,-1. /
      NCUT = 6
      JTOT = 2
      READ (1,*)    CORØ(1,1),CORØ(2,1),CORØ(3,1),
     1              COAL(1,1),COAL(2,1),
     2              COSZ(1,1),COSZ(2,1),COSZ(3,1)
      WRITE (8,100) CORØ(1,1),CORØ(2,1),CORØ(3,1),
     1              COAL(1,1),COAL(2,1),
     2              COSZ(1,1),COSZ(2,1),COSZ(3,1)
100   FORMAT (//,' SAMPLE CENTER (CORØ,1) :  ',3F8.1/
     1           ' SAMPLE ANGLE  (COAL,1) :  ',2F8.1/
     2           ' SAMPLE SIZE   (COSZ,1) :  ',3F8.1)
      READ (1,*)    CORØ(1,2),CORØ(2,2),CORØ(3,2),
     1              COAL(1,2),COAL(2,2),
     2              COSZ(1,2),COSZ(2,2),COSZ(3,2)
      WRITE (8,110) CORØ(1,2),CORØ(2,2),CORØ(3,2),
     1              COAL(1,2),COAL(2,2),
     2              COSZ(1,2),COSZ(2,2),COSZ(3,2)
110   FORMAT (//,' INJECT CENTER (CORØ,2) :  ',3F8.1/
     1           ' INJECT ANGLE  (COAL,2) :  ',2F8.1/
     2           ' INJECT SIZE   (COSZ,2) :  ',3F8.1)
```

```
      DO 1Ø K1=1,JTOT
      ALPP = COAL(1,K1) * DG2RAD
      BETT = COAL(2,K1) * DG2RAD
      CORA(1,1,K1)= COS(ALPP) * COS(BETT)
      CORA(2,1,K1)= SIN(ALPP) * COS(BETT)
      CORA(3,1,K1)=             SIN(BETT)
      CORA(1,2,K1)=-SIN(ALPP)
      CORA(2,2,K1)= COS(ALPP)
      CORA(3,2,K1)= Ø.
      CORA(1,3,K1)=-COS(ALPP) * SIN(BETT)
      CORA(2,3,K1)=-SIN(ALPP) * SIN(BETT)
      CORA(3,3,K1)=             COS(BETT)
      DO 9 I=1,3
      DO 9 J=1,3
 9    CORB(I,J,K1) = CORA(J,I,K1)
 1Ø   CONTINUE
      DO 3Ø K1=1,JTOT
      DO 25 J=1,NCUT
      J4 = J + (K1-1)*NCUT
      DO 2Ø J1=1,3
      WKKK = Ø.
      DO 15 J2=1,3
 15   WKKK= WKKK + CORA(J1,J2,K1)*COSZ(J2,K1)*WKYZ(3*(J-1)+J2)/2.
 2Ø   XCUT(J1,J4)= CORØ(J1,K1)+WKKK
      XCUT(4,J4) = COAL(1,K1)*WKALP(2*(J-1)+1) + WKALP(2*(J-1)+2)
      XCUT(5,J4) = COAL(2,K1)*WKBTA(2*(J-1)+1) + WKBTA(2*(J-1)+2)
      XCUT(6,J4) = Ø.Ø
      XCUT(7,J4) = Ø.5 * COSZ(NWAL(J),K1)
      XCUT(8,J4) = Ø.5 * COSZ(NWBT(J),K1)
      XCUT(9,J4) = Ø.Ø
 25   CONTINUE
 3Ø   CONTINUE
      NCUT = NCUT * JTOT
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUBBOND
C
C   PURPOSE:     THIS SUBROUTINE ASSIGNS THE BOUNDARY CONDITIONS FOR
C                THE HYDRAULIC INJECTION TEST.  THE CONTENTS OF THE VECTOR
C                *PO* IS CHANGED.
C
C   ARGUMENTS:
C
C      PRES(I)          VECTOR OF LENGTH NCUT CONTAINING THE PRESSURE
C                         HEAD VALUES (IN LENGTH UNITS) FOR THE BOUNDARY
C                         SURFACE I.
C                       NOTE THAT BOUNDARY SURFACES 7-12 ARE INJECTION
C                         PLANES AND A PRESSURE HEAD OF 100 LENGTH UNITS
C                         IS ASSIGNED FOR THIS EXPERIMENT.
C
C
C
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      COMMON /MTRX/  PO(400), PX(400)
      COMMON /CORD/ COR0(3,2),COSZ(3,2),COAL(2,2),
     1              CORA(3,3,2),CORB(3,3,2)
      REAL PRES(12)
      DATA PRES /0.,0.,0.,0.,0.,0.,100.,100.,100.,100.,100.,100./
      DO 10 J=1,NBOND
      PO(J)=PRES(NDXD(J,2))+COSZ(3,1)/2.-XYZX(J,3)
10    CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE SUBMTRX
C
C   PURPOSE:     THIS SUBROUTINE ASSEMBLES THE GLOBAL MATRIX EQUATION
C                AND SOLVES THE HEAD DISTRIBUTION IN THE SAMPLE.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CFCF/ NCF(200),NCFN(300)
      COMMON /NODE/ NNODE,NDID(400,2),NDFR(1000)
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      COMMON /MTRX/ PO(400), PX(400)
      REAL X0(4),X1(4),AA(400),BB
      REAL AMTRX(50000)
      DATA M,IB / 1, 400/
      DATA RHO,UN,UNITCON /9810., 0.001, 0.01 /
      J5=0
      DO 50 I=1,NNODE
      BB=0.
      DO 5 I1=1,NNODE
5     AA(I1)=0.
      J1=NDID(I,1)
      J2=NDID(I,2)
      DO 40 III=1,2
      IF (III .EQ. 2) THEN
        JTEMP=J1
        J1=J2
        J2=JTEMP
      ENDIF
      DO 15 I1=1,IFN(J1,NFR)
      ITT = IFK(J1,NFR) + I1
      IF (J2 .EQ. NFRN(ITT) ) THEN
        JJ2=I1
        DO 24 I2=1,4
10      X0(I2)= XYZI (NDFR(ITT), I2)
      ENDIF
15    CONTINUE
      DO 25 I1=1,IFN(J1,NFR)
      ITT = IFK(J1,NFR) + I1
      IF (I1 .NE. JJ2) THEN
        T2=0.
        DO 20 I2=1,3
        X1(I2) = XYZI (NDFR(ITT), I2)
        T2=T2+(X1(I2)-X0(I2))**2.
20      CONTINUE
```

```
        X1(4)= XYZI (NDFR(ITT), 4)
        XØX1 = SQRT(T2)
        XX   = (X1(4) + XØ(4))/2.
        XCOEF= XX * (XINP(9,J1)**3.) * RHO
     1         / ( 12. * UN * XØX1 )
        AA(NDFR(ITT))=AA(NDFR(ITT)) + XCOEF
        AA(I)= AA(I) - XCOEF
        BB   = BB + (XØ(3)-X1(3))*XCOEF
      ENDIF
 25   CONTINUE
      DO 35 I1=1,IFN(J1,NCF)
        T2=Ø.
        ITT = IFK(J1,NCF) + I1
        DO 3Ø I2=1,3
          X1(I2) = XYZX (NDCF(ITT), I2)
          T2=T2+(X1(I2)-XØ(I2))**2.
 3Ø     CONTINUE
        X1(4)= XYZX (NDCF(ITT), 4)
        XØX1 = SQRT(T2)
        XX   = (X1(4) + XØ(4))/2.
        XCOEF= XX * (XINP(9,J1)**3.) * RHO
     1         / ( 12. * UN * XØX1 )
        AA(I)= AA(I) - XCOEF
        BB   = BB + (XØ(3)-X1(3)-PO(NDCF(ITT)))*XCOEF
 35   CONTINUE
 4Ø   CONTINUE
      DO 45 I1=1,I
      AMTRX(J5+I1) = -AA(I1)
 45   CONTINUE
      J5=J5+I
      PX(I)= -BB
 5Ø   CONTINUE
      IER = Ø
      CALL SUBLEQS(AMTRX,M,NNODE,PX,IB,D1,D2,IER)
      IF ( IER .GT. Ø) THEN
      WRITE (8, *) 'ERROR MSG FROM  *SUBLEQS*  IER =',IER
      ENDIF
      RETURN
      END
```

```
C
C
C
      SUBROUTINE OUTRUN2
C
C
C   PURPOSE:     THIS SUBROUTINE WRITEE THE FRACTURE NETWORK INFORMATION
C                AND THE HYDRAULIC DATA INTO A TEMPORARY FILE TO BE USED
C                IN RUN2 OF THE EXPERIMENT.
C
C
C
      COMMON /FRCT/ NFRCT,NSHP(200),XINP(9,200),NFR(200),NFRN(1000)
      COMMON /CUTT/ NCUT, XCUT(9,12),NCT(12),NCTN(300)
      COMMON /CFCF/ NCF(200),NCFN(300)
      COMMON /NODE/ NNODE,NDID(400,2),NDFR(1000)
      COMMON /BOND/ NBOND,NDXD(400,2),NDCF(300),NDCT(300)
      COMMON /XYZZ/ XYZI(400,4), XYZX(400,4)
      COMMON /MTRX/ PO(400), PX(400)
      WRITE (10,*) NFRCT,NFR(NFRCT),NCF(NFRCT),NNODE,NBOND
      DO 10 J = 1, NFRCT
      WRITE (10,*)  J,   XINP(9,J),
     1              IFN(J,NFR), (NDFR(K),K=IFK(J,NFR)+1,NFR(J)),
     2              IFN(J,NCF), (NDCF(K),K=IFK(J,NCF)+1,NCF(J))
10    CONTINUE
      DO 20 J = 1, NNODE
      WRITE (10,*) J, PX(J), NDID(J,1),NDID(J,2), (XYZI(J,K),K=1,4)
20    CONTINUE
      DO 30 J = 1, NBOND
      WRITE (10,*) J, PO(J), NDXD(J,1),NDXD(J,2), (XYZX(J,K),K=1,4)
30    CONTINUE
      RETURN
      END
```

```
C           A.6  PROGRAM LISTING OF PART 2 OF EXPERIMENT 2
C
      PROGRAM EXP2R2 (INPUT,OUTPUT,TAPE1=INPUT,TAPE10=OUTPUT)
C
C
C  PURPOSE:    THIS PROGRAM PERFORMS RUN2 OF THE HYDRAULIC-TRACER
C              EXPERIMENT.  THE FRACTURE NETWORK INFORMATION AND
C              THE HYDRAULIC HEAD DISTRIBUTION IS READ IN FROM A
C              TEMPORARY DATA FILE CREATED IN RUN1 OF THE EXPERIMENT.
C              THE COMPLETE FLOW NETWORK IS DEFINED BY CALLING
C              SUBROUTINE *FLOWNET*.  SUBROUTINE *TRACER* IS THEN
C              CALLED TO CALCULATE THE BREAKTHROUGH CURVE FOR A
C              SPECIFIED NODAL POINT.  FINALLY, CHARACTERISTIC TIMES
C              ASSOCIATED WITH THE BREAKTHROUGH (BT) CURVE, SUCH AS
C              INITIAL BT - T0, 50 PERCENT BT, T50, ETC., ARE DETERMINED.
C
C  ARGUMENTS:
C
C     NCHK           NUMBER OF BREAKTHROUGH (BT) CURVES TO BE
C                      DETERMINED.
C
C     NDCHK(I)       VECTOR OF LENGTH NCHK CONTAINING THE NODAL NUMBER
C                      OF WHICH A BT CURVE IS DESIRED.  NDCHK(I) .GT. 0
C                      INDICATES THAT NDCHK(I) IS AN INTERNAL NODE
C                      (FRACTURE-FRACTURE INTERSECTION).   A NEGATIVE
C                      ENTRY IMPLIES AN EXTERNAL (FRACTURE-BOUNDARY
C                      SURFACE INTERSECTION) NODE.
C
C     TBEG,TEND      THE BEGINNING (TBEG) AND ENDING (TEND) TIME
C                      INTERVAL OF A BREAKTHROUGH CURVE.  TIME 0 IS
C                      DEFINED AS THE BEGINNING OF STEP TRACER
C                      INJECTION AT THE SOURCE.
C
C     NSTP           NUMBER OF TIME DISCRETIZATIONS BETWEEN TBEG AND
C                      TEND.
C                    NOTE: THE TRACER SUB-PROGRAM HAS AN INFINITE TIME
C                      RESOLUTION FOR THE BT CURVE.
C
C     BTCV(I)        VECTOR OF LENGTH NSTP+2 CONTAINING THE CONCEN-
C                      TRATION (C/C0) HISTRORY AT THE END OF TIME STEP
C                      I (OR TIME UNIT  TBEG + I*(TBEG-TEND)/NSTP ).
C
C     CCHK(I)        VECTOR OF LENGTH N+1, WHERE N IS THE NUMBER OF
C                      CHECK POINTS, CONTAINING RELATIVE BT (C/C0)
C                      VALUES AT WHICH THE CORRESPONDING TIME SCALES
C                      ARE TO BE DETERMINED AND STORED IN THE *TCHK*
C                      VECTOR.
C                    NOTE :  CCHK(I) .LT. 1.0  IF I = 1, 2, ... N
C                            CCHK(N+1)  .GT. 1.0
C
```

```
C       TCHK(I)            VECTOR OF LENGTH N+1, WHERE N IS THE NUMBER OF
C                            CHECK POINTS, CONTAINING THE CHARACTERISTIC
C                            TIME SCALES CORRESPONDING TO THE RELATIVE BT
C                            VALUES DEFINED BY VEATOR *CCHK*.
C
C       NIJET              TOTAL NUMBER OF BOUNDARY SURFACES WHERE THE
C                            TRACER WILL BE APPLIED AT TIME UNIT 0.
C
C       NJET(I)            VECTOR OF LENGTH NIJET CONTAINING THE BOUNDARY
C                            SURFACE NUMBER OF WHICH A TRACER IS APPLIED AT
C                            TIME 0.
C
C       AVGVEL             AVERAGE VELOCITY OF THE FRACTURE NETWORK.
C
C       VOLSUM             TOTAL VOLUME OF THE FLOWING FLUID IN THE NETWORK.
C
C
C
        COMMON /FRCT/ NFRCT,APER(200),NFR(200),NCF(200)
        COMMON /NODE/ NNODE,NDID(300,2),PX(300),XYZI(300,4)
        COMMON /BOND/ NBOND,NDXD(200,2),PO(200),XYZX(200,4)
        COMMON /FLNT/ INET(300),FINND(300)
        COMMON /FBND/ INXT(200),FEXND(200)
        COMMON /RFDN/ NDFR(700),NDCF(300)
        COMMON /ERTI/ ITRE(1800),FTRE(1800),TYMIN(1800)
        COMMON /XRTI/ ITRX( 800),FTRX( 800),TYMEX( 800)
        COMMON /IJET/ NIJET, NJET(12)
        DIMENSION BTCV(2050),TCHK(7),CCHK(7),NDCHK(6)
        DATA NCHK,NDCHK       / 6, -1, -2, -3, -4, -5, -6/
        DATA TBEG,TEND,NSTP  / 0.0, 2000.0, 2000 /
        DATA CCHK          / 0.0, 0.1, 0.25, 0.5, 0.75, 0.90, 1.5/
        DATA NIJET,NJET  / 6, 7, 8, 9, 10, 11, 12, 0,0,0,0,0,0 /
        CALL INPDATA
        CALL FLOWNET (AVGVEL,VOLSUM)
        WRITE (10,100) AVGVEL,VOLSUM
100     FORMAT (//2X,'******  GLOBAL VOLUME AVERAGED VELOCITY = ',
     1   F14.4,/  2X,'******  GLOBAL VOLUME                    = ',
     2   F14.4,//)
        CALL OUTFLNT
        DO 20 I = 1, NCHK
        CALL TRACER ( NDCHK(I), TBEG, TEND, NSTP, BTCV)
        J1=1
        TYMST = (TBEG - TEND) / REAL(NSTP)
        DO 10 J0 = 2,NSTP+1
        IF( (BTCV(J0)-CCHK(J1)) .GT. 1.E-9 ) THEN
        TCHK(J1) = TYMST * (REAL (J0-2) + (CCHK(J1) - BTCV(J0-1))
     1                       / (BTCV(J0)-BTCV(J0-1)) )
        J1=J1+1
        ENDIF
10      CONTINUE
```

```
      J1=J1-1
      WRITE (10,110) NDCHK(I),TEND,BTCV(NSTP+1),
     1                (NINT(100.*CCHK(J0)),TCHK(J0),J0=1,J1)
110   FORMAT(/2X,'NODE, T-END, BT-TEND: ',I5,4X,F6.0,
     1   4X,F7.4/(3(2X,'T',I2,' = ',F6.1,2X)) )
20    CONTINUE
      STOP
      END
```

```
C
C
C
      SUBROUTINE INPDATA
C
C  PURPOSE:    THIS SUBROUTINE READS IN THE FRACTURE NETWORK INFOMATION,
C              INCLUDING HYDRAULIC HEAD DISTRIBUTIONS, FROM A DATA FILE.
C              THE DATA FILE IS CREATED IN PPOGRAM *EXP2R1*, OR RUN 1
C              OF THE HYDRAULIC-TRACER INJECTION EXPERIMENT.  INPUT
C              DATA ARE STORED IN COMMON BLOCKS /FRCT/, /NODE/, /BOND/
C              AND /RFND/.
C
C
C
      COMMON /FRCT/ NFRCT,APER(200),NFR(200),NCF(200)
      COMMON /NODE/ NNODE,NDID(300,2),PX(300),XYZI(300,4)
      COMMON /BOND/ NBOND,NDXD(200,2),PO(200),XYZX(200,4)
      COMMON /RFDN/ NDFR(700),NDCF(300)
      DIMENSION NTEM(25),NTEN(25)
      READ (1,*) NFRCT,NFRSUM,NCFSUM,NNODE,NBOND
      NFRJ=0
      NCFJ=0
      DO 20 J = 1, NFRCT
      NFRJP1 = NFRJ + 1
      NCFJP1 = NCFJ + 1
      READ (1,*) J1,APER(J), NFRJ1, (NTEM(J2),J2=1,NFRJ1),
     1                       NCFJ1, (NTEN(J2),J2=1,NCFJ1)
      DO 6 K=NFRJP1,NFRJP1+NFRJ1
 6    NDFR(K) = NTEM(K-NFRJP1+1)
      DO 7 K=NCFJP1,NCFJP1+NCFJ1
 7    NDCF(K) = NTEN(K-NCFJP1+1)
      IF ( J1 .NE. J) THEN
      WRITE (10,100) J, J1
 100  FORMAT (2X,'*** INPUT DATA ERROR IN INBNL LOOP 20 *********',
     1        'J = ',I5,5X,'J1 = ',I5)
      RETURN
      ENDIF
      NFRJ = NFRJ + NFRJ1
      NCFJ = NCFJ + NCFJ1
      NFR(J) = NFRJ
      NCF(J) = NCFJ
 20   CONTINUE
      IF ((NFR(NFRCT) .NE. NFRSUM) .OR. (NCF(NFRCT) .NE. NCFSUM)) THEN
      WRITE (10,110) NFRSUM,NFR(NFRCT),NCFSUM,NCF(NFRCT)
 110  FORMAT (2X,'*** INPUT DATA ERROR IN INBNL LOOP 20 *********',
     1        'NFRS,NFR(S)=',2I5,5X,'NCFS,NCF(S)=',2I5)
      ENDIF
      DO 30 J = 1, NNODE
      READ (1,*) J1, PX(J1), NDID(J1,1),NDID(J1,2), (XYZI(J1,K),K=1,4)
      IF ( J1 .NE. J) THEN
```

```
      WRITE (10,120) J, J1
120   FORMAT (2X,'*** INPUT DATA ERROR IN INBNL LOOP 30 *********',
     1        'J = ',I5,5X,'J1 = ',I5)
      RETURN
      ENDIF
30    CONTINUE
      DO 40 J = 1, NBOND
      READ (1,*) J1, PO(J1), NDXD(J1,1),NDXD(J1,2), (XYZX(J1,K),K=1,4)
      IF ( J1 .NE. J) THEN
      WRITE (10,130) J, J1
130   FORMAT (2X,'*** INPUT DATA ERROR IN INBNL LOOP 40 *********',
     1        'J = ',I5,5X,'J1 = ',I5)
      RETURN
      ENDIF
40    CONTINUE
      RETURN
      END
```

```
C
C
C
      SUBROUTINE FLOWNET (VELOAGE,VOLSUM)
C
C
C   PURPOSE:     THIS SUBROUTINE GENERATES A COMPLETE FLOW NETWORK FOR
C                THE FRACTURE SYSTEM. THE OUTPUT OF THIS PROGRAM IS STORED
C                IN COMMON BLOCKS /FLNT/, /FBND/, /ERTI/ AND /XRTI/.
C
C   ARGUMENTS:
C
C      AVGVEL           AVERAGE VELOCITY OF THE FRACTURE NETWORK.
C
C      VOLSUM           TOTAL VOLUME OF THE FLOWING FLUID IN THE NETWORK.
C
C
C
C
      COMMON /FRCT/ NFRCT,APER(200),NFR(200),NCF(200)
      COMMON /NODE/ NNODE,NDID(300,2),PX(300),XYZI(300,4)
      COMMON /BOND/ NBOND,NDXD(200,2),PO(200),XYZX(200,4)
      COMMON /FLNT/ INET(300),FINND(300)
      COMMON /FBND/ INXT(200),FEXND(200)
      COMMON /ERTI/ ITRE(1800),FTRE(1800),TYMIN(1800)
      COMMON /XRTI/ ITRX( 800),FTRX( 800),TYMEX( 800)
      COMMON /RFDN/ NDFR(700),NDCF(300)
      DIMENSION XØ(4), X1(4), XFLUX(4Ø),JFL(4Ø),XTYM(4Ø)
      DIMENSION XVELO(4Ø)
      VOLSUM=Ø.Ø
      VOLVEL=Ø.Ø
      DO 4Ø J = 1, NNODE
      FINND (J) = Ø.
      JCNT = Ø
      FLUXPOS = Ø.
      FLUXNEG = Ø.
      DO 9 J5 = 1, 4
9     XØ(J5) = XYZI(J,J5)
      DO 3Ø JØ = 1, 2
      JSD = NDID(J,JØ)
      IF (JSD .EQ. 1) THEN
        NFRJM1=Ø
        NCFJM1=Ø
      ELSE
        NFRJM1=NFR(JSD-1)
        NCFJM1=NCF(JSD-1)
      ENDIF
      DO 15 J1 = 1,NFR(JSD)-NFRJM1
      JND = NDFR(NFRJM1+J1)
      IF ( JND .NE. J ) THEN
      JCNT = JCNT + 1
```

```
      DO 12 J5 = 1, 4
      X1(J5) = XYZI(JND,J5)
12    CONTINUE
      CALL FLXCAL( X0, PX(J), X1, PX(JND), APER(JSD), XFLUX(JCNT),
     1            XVELO(JCNT), XTYM(JCNT), VOLM, FLUXPOS, FLUXNEG )
      VOLSUM = VOLSUM + VOLM
      VOLVEL = VOLVEL + XVELO(JCNT)*VOLM
      JFL (JCNT) = JND
      ENDIF
15    CONTINUE
      DO 20 J1 = 1,NCF(JSD)-NCFJM1
      JXD = NDCF(NCFJM1+J1)
      JCNT = JCNT + 1
      DO 17 J5 = 1, 4
      X1(J5) = XYZX(JXD,J5)
17    CONTINUE
      CALL FLXCAL( X0, PX(J), X1, PO(JXD), APER(JSD), XFLUX(JCNT),
     1            XVELO(JCNT), XTYM(JCNT), VOLM, FLUXPOS, FLUXNEG )
      VOLSUM = VOLSUM + VOLM
      VOLVEL = VOLVEL + XVELO(JCNT)*VOLM
      JFL (JCNT) = -JXD
20    CONTINUE
30    CONTINUE
      FINND(J) = FLUXPOS
      IF ( ABS(FLUXPOS+FLUXNEG) .GT. 1.E-4 ) THEN
      WRITE (10,100) J, FLUXPOS, FLUXNEG
100   FORMAT(5X,'*** WARNING ON SUBROUTINE *FLOWNET*  ',/
     1        5X,'INODE = ',I3,5X,'FLUXPOS = ',F8.4,5X,
     2                           'FLUXNEG = ',F8.4)
      ENDIF
      IF (J .EQ. 1) THEN
        JJM1= 0
      ELSE
        JJM1= INET(J-1)
      ENDIF
      IF (FLUXPOS .LT. 1.E-5) THEN
        JA = 0
      ELSE
        JA = 0
        DO 35 J5 = 1, JCNT
        IF (XFLUX(J5) .GT. 1.E-6) THEN
          FTREJJA = XFLUX(J5)/FLUXPOS
          IF (FTREJJA .GT. 1.E-5) THEN
            JA = JA + 1
            JJM2 = JJM1 + JA
            FTRE (JJM2) = FTREJJA
            TYMIN(JJM2) = XTYM (J5)
            ITRE (JJM2) = JFL  (J5)
          ENDIF
        ENDIF
```

```
35        CONTINUE
        ENDIF
        INET(J) = JJM1 + JA
40      CONTINUE
        DO 70 J = 1, NBOND
        FEXND (J) = Ø.
        JCNT = Ø
        FLUXPOS = Ø.
        FLUXNEG = Ø.
        DO 45 J5 = 1, 4
45      XØ(J5) = XYZX(J,J5)
        JSD = NDXD(J,1)
        IF (JSD .EQ. 1) THEN
          NFRJM1=Ø
          NCFJM1=Ø
        ELSE
          NFRJM1=NFR(JSD-1)
          NCFJM1=NCF(JSD-1)
        ENDIF
        DO 55 J1 = 1,NFR(JSD)-NFRJM1
        JND = NDFR(NFRJM1+J1)
        JCNT = JCNT + 1
        DO 52 J5 = 1, 4
        X1(J5) = XYZI(JND,J5)
52      CONTINUE
        CALL FLXCAL( XØ, PO(J), X1, PX(JND), APER(JSD), XFLUX(JCNT),
     1              XVELO(JCNT), XTYM(JCNT), VOLM, FLUXPOS, FLUXNEG )
        VOLSUM = VOLSUM + VOLM
        VOLVEL = VOLVEL + XVELO(JCNT)*VOLM
        JFL(JCNT) = JND
55      CONTINUE
        DO 6Ø J1 = 1,NCF(JSD)-NCFJM1
        JXD = NDCF(NCFJM1+J1)
        IF ( JXD .NE. J) THEN
        JCNT = JCNT + 1
        DO 57 J5 = 1, 4
        X1(J5) = XYZX(JXD,J5)
57      CONTINUE
        CALL FLXCAL( XØ, PO(J), X1, PO(JXD), APER(JSD), XFLUX(JCNT),
     1              XVELO(JCNT), XTYM(JCNT), VOLM, FLUXPOS, FLUXNEG )
        VOLSUM = VOLSUM + VOLM
        VOLVEL = VOLVEL + XVELO(JCNT)*VOLM
        JFL(JCNT) = -JXD
        ENDIF
6Ø      CONTINUE
        FLUXNET = FLUXPOS + FLUXNEG
        FEXND(J) = FLUXNET
        IF (J .EQ. 1) THEN
          JJM1= Ø
        ELSE
```

```
    JJM1= INXT(J-1)
  ENDIF
  IF (FLUXNET .LT. 1.E-5) THEN
    JA=0
  ELSE
    JA=0
    DO 65 J5 = 1, JCNT
    FTRXJA = XFLUX(J5) / FLUXPOS
    IF (FTRXJA .GT. 1.0E-5) THEN
      JA = JA + 1
      JJM2 = JJM1 + JA
      FTRX (JJM2) = FTRXJA
      TYMEX(JJM2) = XTYM(J5)
      ITRX (JJM2) = JFL(J5)
    ENDIF
65    CONTINUE
  ENDIF
  INXT(J) = JJM1 + JA
70  CONTINUE
  VELOAGE = VOLVEL / VOLSUM
  RETURN
  END
```

```
C
C
C
      SUBROUTINE TRACER (JTRDE,TYMRØ,TYMR1,NTYM,TEMC)
C
C
C   PURPOSE:    THIS SUBROUTINE DETERMINES A BREAKTHROUGH CURVE AT A
C               SPECIFIED NODAL POINT.  THIS ROUTINE HAS AN INFINITE
C               RESOLUTION IN TIME.   DUE TO THE LIMITATION OF COMPUTER
C               STORAGE, THE BT CURVE IS STORED IN DISCRETE FORM.
C
C   ARGUMENTS:
C
C       JTRDE           THE NODAL NUMBER OF WHICH A BT CURVE IS DESIRED.
C                         IF JTRDE .GT. Ø, THEN JTRDE IS AN INTERNAL NODE;
C                         IF JTRDE .LT. Ø, THEN JTRDE IS AN EXTERNAL NODE.
C
C       TYMRØ,TYMR1     THE BEGINNING (TYMRØ) AND ENDING (TYMR1) TIME
C                         INTERVAL OF A BT CURVE.  TIME Ø IS DEFINED AS THE
C                         BEGINNING OF TRACER INJECTION AT THE SOURCE.
C
C       NTYM            NUMBER OF TIME DISCRETIZATIONS BETWEEN TYMRØ AND
C                        TYMR1.
C
C       TEMC(I)         OUTPUT VECTOR OF LENGTH NTYM+2 CONTAINING THE CON-
C                       CENTRATION (C/CØ) HISTRORY AT THE END OF TIME STEP
C                       I (OR TIME UNIT  TYMRØ + I*(TYMR1-TYMRØ)/NSTP ).
C
C
C
      COMMON /FRCT/ NFRCT,APER(2ØØ),NFR(2ØØ),NCF(2ØØ)
      COMMON /NODE/ NNODE,NDID(3ØØ,2),PX(3ØØ),XYZI(3ØØ,4)
      COMMON /BOND/ NBOND,NDXD(2ØØ,2),PO(2ØØ),XYZX(2ØØ,4)
      COMMON /FLNT/ INET(3ØØ),FINND(3ØØ)
      COMMON /FBND/ INXT(2ØØ),FEXND(2ØØ)
      COMMON /ERTI/ ITRE(18ØØ),FTRE(18ØØ),TYMIN(18ØØ)
      COMMON /XRTI/ ITRX( 8ØØ),FTRX( 8ØØ),TYMEX( 8ØØ)
      COMMON /IJET/ NIJET, NJET(12)
      DIMENSION WRKCG(2Ø5Ø), TEMC(2Ø5Ø)
      DIMENSION KWRK1(1ØØ),KWRK2(1ØØ),KMTHR(1ØØ)
      DOUBLE PRECISION TTX
      TYMST = (TYMR1-TYMRØ) / REAL(NTYM)
      TRAVLT = Ø.Ø
      DO 5 JØ=1,NTYM+5
5     WRKCG(JØ)=Ø.Ø
      DO 15 JØ = 1,1ØØ
15    KWRK2(JØ)=1
      KCONT = Ø
      TTØ   = Ø.Ø
      TTX   = 1.ØDØ
```

```
      IF (JTRDE .LT. Ø ) THEN
        KWRK1(1) = NNTRCX (-JTRDE,INXT)
        KMTHR(1) = JTRDE
      ELSE
        KWRK1(1) = NNTRCX (JTRDE,INET)
        KMTHR(1) = JTRDE
      ENDIF
      IF (KWRK1(1) .EQ. Ø) GO TO 3Ø
        KWRK2(1) = 1
      JLEV=1
2Ø    CONTINUE
      IF (JLEV .GT. 1) THEN
        IF (KMTHR(JLEV-1) .LT. Ø ) THEN
          NTEM = NNTRCR(-KMTHR(JLEV-1),INXT) + KWRK2(JLEV-1)
          KMTHR(JLEV) = ITRX (NTEM)
        ELSE
          NTEM = NNTRCR( KMTHR(JLEV-1),INET) + KWRK2(JLEV-1)
          KMTHR(JLEV) = ITRE (NTEM)
        ENDIF
        IF (KMTHR(JLEV) .LT. Ø ) THEN
          KWRK1(JLEV) = NNTRCX(-KMTHR(JLEV),INXT)
        ELSE
          KWRK1(JLEV) = NNTRCX( KMTHR(JLEV),INET)
        ENDIF
      ENDIF
      IF (KWRK2(JLEV) .LE. KWRK1(JLEV))  THEN
        IF (KMTHR(JLEV) .LT. Ø) THEN
          NTEM = NNTRCR(-KMTHR(JLEV),INXT) + KWRK2(JLEV)
          TTTT = TYMEX (NTEM)
          FFFF = FTRX  (NTEM)
          IIII = ITRX  (NTEM)
        ELSE
          NTEM = NNTRCR( KMTHR(JLEV),INET) + KWRK2(JLEV)
          TTTT = TYMIN (NTEM)
          FFFF = FTRE  (NTEM)
          IIII = ITRE  (NTEM)
        ENDIF
        IF (FFFF .LT. 1.E-6) FFFF = 1.E-6
        TTØ = TTØ + TTTT
        TTX = TTX * DBLE(FFFF)
        IF ( (TTX .LT. 1.D-6) .OR. (TTØ .GT. TYMR1) ) THEN
          TTØ = TTØ - TTTT
          TTX = TTX / DBLE( FFFF )
          KWRK2(JLEV)=KWRK2(JLEV)+1
          JLEV=JLEV-1
        ELSE
        IF ( IIII .LT. Ø ) THEN
          KCONT=KCONT + 1
          KKMM=NDXD(-IIII,2)
          CFATOR = Ø.Ø
```

```
         DO 22 KLMM = 1, NIJET
         IF (KKMM .EQ. NJET(KLMM) ) CFATOR = 1.0
22       CONTINUE
         IDX = INT( (TT0-TYMR0)/ TYMST ) +2
         IF (TT0 .LE. TYMR0)  IDX = 1
         IF (TT0 .GT. TYMR1) IDX = NTYM +2
         WRKCG (IDX) = WRKCG (IDX) + CFATOR * REAL(TTX)
         TRAVLT = TRAVLT + CFATOR * REAL(TTX) * TT0
         TT0 = TT0 - TTTT
         TTX = TTX / DBLE( FFFF )
         KWRK2(JLEV)=KWRK2(JLEV)+1
         JLEV=JLEV-1
       ENDIF
       ENDIF
       JLEV=JLEV+1
     ELSE
       KWRK2(JLEV)=1
       JLEV=JLEV-1
       IF (KMTHR(JLEV) .LT. 0) THEN
         NTEM = NNTRCR(-KMTHR(JLEV),INXT) + KWRK2(JLEV)
         TTTT = TYMEX  (NTEM)
         FFFF = FTRX   (NTEM)
       ELSE
         NTEM = NNTRCR( KMTHR(JLEV),INET) + KWRK2(JLEV)
         TTTT = TYMIN  (NTEM)
         FFFF = FTRE   (NTEM)
       ENDIF
       IF (FFFF .LT. 1.E-6) FFFF = 1.E-6
       TT0 = TT0 - TTTT
       TTX = TTX / DBLE(FFFF)
       KWRK2(JLEV)=KWRK2(JLEV)+1
     ENDIF
     IF (KWRK2(1) .LE. KWRK1(1)) GO TO 20
30   CONTINUE
     TEMC(1)=WRKCG(1)
     J1=1
     DO 50 J0 = 2,NTYM+1
     TEMC(J0) = TEMC(J0-1) + WRKCG(J0)
50   CONTINUE
     RETURN
     END
```

```
C
C
C
      FUNCTION NNTRCR(N,NR)
C
C   REMARKS:     NUCLEAS CALLED BY VARIOUS SUBPROGRAMS TO UNPACK A
C                ONE-DIMNSIONAL ARRAY WHERE DATA OF TWO-DIMENSIONAL
C                NATURE HAVE BEEN STORED.
C
C
      DIMENSION NR(300)
      IF (N .EQ. 1) THEN
      NNTRCR=0
      ELSE
      NNTRCR=NR(N-1)
      ENDIF
      RETURN
      END
C
C
C
      FUNCTION NNTRCX(N,NR)
C
C   REMARKS:     NUCLEAS CALLED BY VARIOUS SUBPROGRAMS TO UNPACK A
C                ONE-DIMNSIONAL ARRAY WHERE DATA OF TWO-DIMENSIONAL
C                NATURE HAVE BEEN STORED.
C
C
      DIMENSION NR(300)
      IF (N .EQ. 1) THEN
      NNTRCX=NR(1)
      ELSE
      NNTRCX=NR(N)-NR(N-1)
      ENDIF
      RETURN
      END
```

```
C
C
C
      SUBROUTINE FLXCAL (XØ,PØ,X1,P1,AP,FLX,VEL,TYM,VOL,FLXP,FLXN)
C
C   REMARKS:    NUCLEAS CALLED BY SUBROUTINE *FLOWNET* TO CALCULATE
C               THE FLUX BETWEEN TWO NODAL POINTS.
C
C
C
      REAL XØ(4), X1(4)
      DATA RHO,UN,UNITCON / 981Ø., Ø.ØØ1, Ø.Ø1 /
      VOL = Ø.
      T2  = Ø.
      DO 1Ø J5 = 1, 3
      T2 = T2 + (X1(J5)-XØ(J5))**2.
1Ø    CONTINUE
      XØX1 = SQRT (T2)
      XX   = (X1(4) + XØ(4))/2.
      COEF = XX * AP**3. * RHO /
     1         ( 12. * UN * XØX1 )
      FLX  = COEF * (P1+X1(3) - (PØ+XØ(3)))
      VEL  = FLX / ( XX * AP )
      IF ( FLX .EQ. Ø.) THEN
        TYM = 1.ØE+12
      ELSE
        TYM = XØX1 / VEL
      ENDIF
      IF ( FLX .GT. Ø.) THEN
        FLXP = FLXP + FLX
        VOL  = XØX1 * XX * AP
      ELSE
        FLXN = FLXN + FLX
      ENDIF
      RETURN
      END
```

```
C
C
      SUBROUTINE OUTFLNT
C
C   PURPOSE:     THIS SUBROUTINE PRINTS OUT THE FLOW NETWORK OF THE
C                FRACTURE SYSTEM.
C
C
      COMMON /FRCT/ NFRCT,APER(200),NFR(200),NCF(200)
      COMMON /NODE/ NNODE,NDID(300,2),PX(300),XYZI(300,4)
      COMMON /BOND/ NBOND,NDXD(200,2),PO(200),XYZX(200,4)
      COMMON /FLNT/ INET(300),FINND(300)
      COMMON /FBND/ INXT(200),FEXND(200)
      COMMON /RFDN/ NDFR(700),NDCF(300)
      COMMON /ERTI/ ITRE(1800),FTRE(1800),TYMIN(1800)
      COMMON /XRTI/ ITRX( 800),FTRX( 800),TYMEX( 800)
      DO 10 J = 1, 10, 1
      IF (J .EQ. 1) THEN
        JJM1= 0
      ELSE
        JJM1= INET(J-1)
      ENDIF
      WRITE (10,110) J, INET(J)-JJM1, FINND(J)
      WRITE (10,111) (ITRE(J0),  J0=JJM1+1,INET(J))
      WRITE (10,112) (FTRE(J0),  J0=JJM1+1,INET(J))
      WRITE (10,113) (TYMIN(J0), J0=JJM1+1,INET(J))
110   FORMAT(//2X,'**  INODE= ',I3,5X,'# OF INFLOWS= ',I2,
     1       5X,'INFLOW FLUX = ',F10.4)
111   FORMAT (2X,'INFLOW NODE : ',5I10,/
     1        (2X,'               ',5I10))
112   FORMAT (2X,'REL. FLUX   : ',5F10.4,/
     1        (2X,'               ',5F10.4))
113   FORMAT (2X,'TRAVEL TIME : ',5F10.2,/
     1        (2X,'               ',5F10.2))
10    CONTINUE
      DO 20 J = 1, 10, 1
      IF (J .EQ. 1) THEN
        JJM1= 0
      ELSE
        JJM1= INXT(J-1)
      ENDIF
      WRITE (10,120) J, INXT(J)-JJM1, FEXND(J)
      WRITE (10,111) (ITRX(J0),  J0=JJM1+1,INXT(J))
      WRITE (10,112) (FTRX(J0),  J0=JJM1+1,INXT(J))
      WRITE (10,113) (TYMEX(J0), J0=JJM1+1,INXT(J))
120   FORMAT(//2X,'**  XNODE= ',I3,5X,'# OF INFLOWS= ',I2,
     1       5X,'INFLOW FLUX = ',F10.4)
20    CONTINUE
      RETURN
      END
```

```
*
*
*
*       SAMPLE OUTPUT FROM RUN 2 OF EXPERIMENT 2
*
*


 ******  GLOBAL VOLUME AVERAGED VELOCITY =          4.0684
 ******  GLOBAL VOLUME                   =      11840.6511

*
*       FLOW NETWORK STRUCTURE DEPICTED FROM INTERNAL NODAL POINTS.
*
*

 **  INODE=   1     # OF INFLOWS=  1     INFLOW FLUX =       .6409
 INFLOW NODE :          3
 REL. FLUX   :     1.0000
 TRAVEL TIME :       1.24

 **  INODE=   2     # OF INFLOWS=  3     INFLOW FLUX =       .4145
 INFLOW NODE :          1         3         4
 REL. FLUX   :      .3439     .5477     .1083
 TRAVEL TIME :      10.32      7.20     57.81


*
*       DATA CONTINUES ...
*


 **  INODE=   9     # OF INFLOWS=  4     INFLOW FLUX =       .0069
 INFLOW NODE :          5         6        10        85
 REL. FLUX   :      .0935     .1530     .0138     .7397
 TRAVEL TIME :     633.74   3006.22  19512.33    115.27

 **  INODE=  10     # OF INFLOWS=  2     INFLOW FLUX =       .0033
 INFLOW NODE :          5         6
 REL. FLUX   :      .0478     .9522
 TRAVEL TIME :    8943.19    291.83

*
*       DATA CONTINUES  UNTIL  INNODE = NNODE.
*
```

```
*
*       FLOW NETWORK STRUCTURE DEPICTED FROM EXTERNAL NODAL POINTS.
*
*

 ** XNODE=   1      # OF INFLOWS=  6      INFLOW FLUX =        .8679
 INFLOW NODE :          22         62         63         64         65
                        66
 REL. FLUX   :       .2127      .0637      .3035      .0774      .1936
                     .1491
 TRAVEL TIME :       64.93     114.79      43.36      97.03      44.76
                     51.32

 ** XNODE=   2      # OF INFLOWS=  3      INFLOW FLUX =        .1445
 INFLOW NODE :          87         88         89
 REL. FLUX   :       .3924      .2881      .3195
 TRAVEL TIME :       75.20      92.18      20.07

*
*       DATA CONTINUES ...
*

 ** XNODE=   5      # OF INFLOWS=  7      INFLOW FLUX =        .1627
 INFLOW NODE :         124        125        126        127        128
                       129        130
 REL. FLUX   :       .0983      .2615      .0731      .0467      .1328
                     .1894      .1982
 TRAVEL TIME :      101.39      45.15     284.64     305.93      42.94
                    139.51     127.05

*
*       DATA CONTINUES UNTIL  XNODE = NBOND.
*
```

```
*
*
*       CHARACTERISTIC TIME SCALES ASSOCIATED WITH EACH INDIVIDUAL
*         BREAKTHROUGH (BT) CURVE DETERMINED ST SPECIFIC NODE.
*       NEGATIVE NODAL INDEX IMPLIES AN EXTERNAL NODE IS BEING EXAMINED.
*       BT-TEND IS THE DEGREE OF BT (C/CO) AT THE END OF A SPECIFIED
*         TIME INTERVAL, T-END.
*       TØ IS THE TIME WHEN THE FIRST TRACE OF THE INJECTED MASS IS
*         DETECTED AT THE OUTFLOW BOUNDARY.  T5Ø IS THE TIME WHEN
*         C/CØ = Ø.5.
*
*

NODE, T-END, BT-TEND:    -1      2ØØØ.      .698Ø
T Ø =   66.Ø     T1Ø =   76.1     T25 =   96.1
T5Ø =  129.9

NODE, T-END, BT-TEND:    -2      2ØØØ.      .954Ø
T Ø =  137.Ø     T1Ø =  148.8     T25 =  23Ø.6
T5Ø =  334.1     T75 =  516.2     T9Ø =  934.6

NODE, T-END, BT-TEND:    -3      2ØØØ.      .9534
T Ø =  334.Ø     T1Ø =  332.5     T25 =  346.8
T5Ø =  433.Ø     T75 =  68Ø.9     T9Ø =  854.7

*
*       DATA CONTINUES UNTIL THE DESIRED NUMBER OF BT CURVES ARE
*         REACHED.
*
```

```
C
C
C
C         A.7  SAMPLE INPUT/OUTPUT OF EXPERIMENT 2

          SAMPLE INPUT DATA SETUP FOR EXPERIMENT 2
C
C
#EOR                                                  (END OF RECORD       )
200, 200.. 200., 200.                                 (NFRCT, FRX, FRY, FRZ)
50.,  10.                                             (FRCTLTH, FRCAPTR    )
342., 1468., 325., 6406.                              (DS1, DS2, DS3, DS4  )
100., 100., 100., 0.,  0., 180., 180., 180.           (SAMPLE INFORMATION  )
100., 100., 100., 0.,  0.,  20.,  20.,  20.           (INJECTION ZONE INFO )
#EOR                                                  (END OF RECORD       )
#EOF                                                  (END OF FILE         )
```

```
*
*
*
*       SAMPLE OUTPUT FROM RUN 2 OF EXPERIMENT 2
*
*


 ******  GLOBAL VOLUME AVERAGED VELOCITY =         4.0684
 ******  GLOBAL VOLUME                   =     11840.6511

*
*       FLOW NETWORK STRUCTURE DEPICTED FROM INTERNAL NODAL POINTS.
*
*

 **  INODE=   1     # OF INFLOWS=  1     INFLOW FLUX =       .6409
 INFLOW NODE :           3
 REL. FLUX   :      1.0000
 TRAVEL TIME :        1.24


 **  INODE=   2     # OF INFLOWS=  3     INFLOW FLUX =       .4145
 INFLOW NODE :           1         3          4
 REL. FLUX   :       .3439     .5477      .1083
 TRAVEL TIME :       10.32      7.20      57.81


*
*       DATA CONTINUES ...
*


 **  INODE=   9     # OF INFLOWS=  4     INFLOW FLUX =       .0069
 INFLOW NODE :           5         6         10         85
 REL. FLUX   :       .0935     .1530      .0138      .7397
 TRAVEL TIME :      633.74   3006.22   19512.33     115.27

 **  INODE=  10     # OF INFLOWS=  2     INFLOW FLUX =       .0033
 INFLOW NODE :           5         6
 REL. FLUX   :       .0478     .9522
 TRAVEL TIME :     8943.19    291.83

*
*       DATA CONTINUES  UNTIL  INNODE = NNODE.
*
```

```
*
*        FLOW NETWORK STRUCTURE DEPICTED FROM EXTERNAL NODAL POINTS.
*
*

 **  XNODE=   1      # OF INFLOWS=  6      INFLOW FLUX =        .8679
 INFLOW NODE :          22        62          63          64        65
                        66
 REL. FLUX   :        .2127     .0637       .3035       .0774     .1936
                      .1491
 TRAVEL TIME :        64.93    114.79       43.36       97.03     44.76
                      51.32

 **  XNODE=   2      # OF INFLOWS=  3      INFLOW FLUX =        .1445
 INFLOW NODE :          87        88          89
 REL. FLUX   :        .3924     .2881       .3195
 TRAVEL TIME :        75.20     92.18       20.07

*
*        DATA CONTINUES ...
*

 **  XNODE=   5      # OF INFLOWS=  7      INFLOW FLUX =        .1627
 INFLOW NODE :         124       125         126         127       128
                       129       130
 REL. FLUX   :        .0983     .2615       .0731       .0467     .1328
                      .1894     .1982
 TRAVEL TIME :       101.39     45.15      284.64      305.93     42.94
                     139.51    127.05

*
*        DATA CONTINUES UNTIL  XNODE = NBOND.
*
```

```
*
*
*       CHARACTERISTIC TIME SCALES ASSOCIATED WITH EACH INDIVIDUAL
*         BREAKTHROUGH (BT) CURVE DETERMINED ST SPECIFIC NODE.
*       NEGATIVE NODAL INDEX IMPLIES AN EXTERNAL NODE IS BEING EXAMINED.
*       BT-TEND IS THE DEGREE OF BT (C/CO) AT THE END OF A SPECIFIED
*         TIME INTERVAL, T-END.
*       TØ IS THE TIME WHEN THE FIRST TRACE OF THE INJECTED MASS IS
*         DETECTED AT THE OUTFLOW BOUNDARY.  T5Ø IS THE TIME WHEN
*         C/CØ = Ø.5.
*
*

 NODE, T-END, BT-TEND:    -1      2ØØØ.      .698Ø
 T Ø =   66.Ø    T1Ø =   76.1    T25 =   96.1
 T5Ø =  129.9

 NODE, T-END, BT-TEND:    -2      2ØØØ.      .954Ø
 T Ø =  137.Ø    T1Ø =  148.8    T25 =  23Ø.6
 T5Ø =  334.1    T75 =  516.2    T9Ø =  934.6

 NODE, T-END, BT-TEND:    -3      2ØØC.      .9534
 T Ø =  334.Ø    T1Ø =  332.5    T25 =  346.8
 T5Ø =  433.Ø    T75 =  68Ø.9    T9Ø =  854.7

*
*       DATA CONTINUES UNTIL THE DESIRED NUMBER OF BT CURVES ARE
*         REACHED.
*
```

NRC FORM 335 (2-84) NRCM 1102, 3201, 3202

U.S. NUCLEAR REGULATORY COMMISSION

BIBLIOGRAPHIC DATA SHEET

SEE INSTRUCTIONS ON THE REVERSE

1. REPORT NUMBER (Assigned by TIDC, add Vol. No., if any): NUREG/CR-4042

2. TITLE AND SUBTITLE

A 3-Dimensional Computer Model to Simulate Fluid Flow and Contaminant Transport Through a Rock Fracture System

3. LEAVE BLANK

4. DATE REPORT COMPLETED

MONTH	YEAR
October	1984

5. AUTHOR(S)

Chi-hua Huang and D. D. Evans

6. DATE REPORT ISSUED

MONTH	YEAR
January	1985

7. PERFORMING ORGANIZATION NAME AND MAILING ADDRESS (Include Zip Code)

Department of Hydrology and Water Resources
University of Arizona
Tucson, Arizona 85721

8. PROJECT/TASK/WORK UNIT NUMBER

9. FIN OR GRANT NUMBER

FIN B7291

10. SPONSORING ORGANIZATION NAME AND MAILING ADDRESS (Include Zip Code)

Division of Radiation Programs and Earth Sciences
Office of Nuclear Regulatory Research
U. S. Nuclear Regulatory Commission
Washington, D. C. 20555

11a. TYPE OF REPORT

Technical

b. PERIOD COVERED (Inclusive dates)

September 1983- October 1984

12. SUPPLEMENTARY NOTES

13. ABSTRACT (200 words or less)

A 3-dimensional fracture generating scheme is presented which can be used to simulate water flow and contaminant (solute) transport through fracture system of a rock. It is presently limited to water saturated conditions, zero permeability for the rock matrix, and steady state water flow, but allows for transient solute transport. The scheme creates finite planar plates of uniform thickness which represent fractures in 3-dimensional space. A given fracture (plate) has the following descriptors: center location, orientation, shape, areal extent and aperture. Each parameter can be described by an appropriate probability distribution. Individual fractures are generated to form an assemblage of a certain fracture density. All fracture intersections and boundary/fracture intersections are determined and deadend fractures are eliminated. Flow through the fracture assemblage is considered laminar and described by Poiseuille's law. The principle of mass conservation at each intersection is used to develop the global matrix equation, which is solved subject to specified boundary conditions to yield the head and flow distribution at each intersection. Solute transport is considered to be advective between intersections with complete mixing at each intersection. Solutes added to the flow system can be explicitly followed and concentration vs. time relationships can be determined anywhere in the system. Some examples are included.

14. DOCUMENT ANALYSIS - a. KEYWORDS/DESCRIPTORS

rock, fracture, water flow, solute transport computer model

b. IDENTIFIERS/OPEN ENDED TERMS

15. AVAILABILITY STATEMENT

UNLIMITED

16. SECURITY CLASSIFICATION

(This page) UNCLASSIFIED

(This report) UNCLASSIFIED

17. NUMBER OF PAGES

18. PRICE

UNITED STATES
NUCLEAR REGULATORY COMMISSION
WASHINGTON, D.C. 20555

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE, $300

FOURTH CLASS MAIL
POSTAGE & FEES PAID
USNRC
WASH. D.C.
PERMIT No. G-67

120555078877 1 1AN1RW
US NRC
ADM-DIV OF TIDC
POLICY & PUB MGT BR-PDR NUREG
W-501
WASHINGTON DC 20555

CONTAMINANT TRANSPORT THROUGH A ROCK FRACTURE SYSTEM