**Idaho
National
Engineering
Laboratory**

*Managed
by the U.S.
Department
of Energy*

*Work performed under
DOE Contract
No. DE-AC07-76ID01570*

EGG-EP-9029
May 1990

Received by OSTI

MAY 21 1990

INFORMAL REPORT

FRAC-UNIX THEORY AND USER'S MANUAL

Tom M. Clemo
John D. Miller
Laurence C. Hull
Sven O. Magnuson

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

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

DISCLAIMER

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

DISCLAIMER

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

EGG-EP--9029

DE90 010915

FRAC-UNIX THEORY AND USER'S MANUAL

Tom M. Clemo
John D. Miller
Laurence C. Hull
Swen O. Magnuson

Published May 1990

EG&G Idaho, Inc.
Idaho Falls, Idaho 83415

MASTER

Prepared for the
U.S. Department of Energy
Idaho Operations Office
Under DOE Contract No. DE-AC07-76ID01570

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

ABSTRACT

The FRAC-UNIX computer code provides a two-dimensional simulation of saturated flow and transport in a fractured porous media. The code incorporates a dual permeability approach in which the rock matrix is modeled as rectangular cells and the fractures are represented as discrete elements on the edges and diagonals of the matrix cells. A single head distribution drives otherwise independent flows in the matrix and in the fractures. Steady-state or transient flow of a single-phase fluid may be simulated. Solute or heat transport is simulated by moving imaginary marker particles in the velocity field established by the flow model, under the additional influence of dispersive and diffusive processes. Sparse-matrix techniques are utilized along with a specially developed user interface. The code is installed on a CRAY XMP24 Computer using the UNICOS operating system. The initial version of this code, entitled FRACSL, incorporated the same flow and transport models, but used a commercial software package for the numerics and user interface. This report describes the theoretical basis, approach and implementation incorporated in the code; the mechanics of operating the code; several sample problems; and the integration of code development with physical modeling and field testing. The code is fully functional, for most purposes, as shown by the results of an extensive code verification effort. Work remaining consists of refining and adding capabilities needed for several of the code verification problems; relatively simple modifications to extend its application and improve its ease of use; an improvement in the treatment of fracture junctions and correction of an error in calculating buoyancy and concentration for diagonal fractures on a rectangular grid.

CONTENTS

ABSTRACT	ii
1. INTRODUCTION	1-1
2. THEORETICAL BASIS, APPROACH, AND IMPLEMENTATION	2-1
2.1 Hydraulics	2-1
2.1.1 Conservation of Mass	2-1
2.1.2 Matrix Flow	2-3
2.1.3 Fracture Flow	2-3
2.1.4 Reservoir Description	2-4
2.1.4.1 Model Grid	2-5
2.1.4.2 Matrix Cells	2-6
2.1.4.3 Fractures	2-6
2.1.5 Head and Flow Models	2-8
2.1.6 Computational Modes	2-11
2.1.7 Boundary Conditions	2-12
2.2 Transport	2-16
2.2.1 Advection-Dispersion Equation	2-17
2.2.2 Solution Techniques	2-18
2.2.3 Random-Walk Dispersion	2-20
2.2.4 Molecular Diffusion	2-23
2.2.5 Matrix Transport Model	2-24
2.2.5.1 Advective Motion	2-24
2.2.5.2 Matrix Transport Implementation	2-28
2.2.5.3 Fracture Effects on Matrix Transport	2-30
2.2.5.4 Nodal Boundary Considerations	2-33
2.2.5.5 External Source Approximation	2-34
2.2.6 Fracture Transport Model	2-35
2.2.6.1 Longitudinal Velocity Distribution	2-35
2.2.6.2 Transverse Velocity	2-38
2.2.6.3 Diffusive Considerations	2-39
2.2.7 Fracture-to-Fracture Transfer Model	2-40
2.2.7.1 Routing for Continuous Sequences	2-41
2.2.7.2 Routing for Discontinuous Sequences	2-42
2.2.7.3 Stream Velocity Profiles	2-43
2.2.7.4 Molecular Diffusion and Mixing	2-45
2.2.8 Fracture-to-Matrix Transfer Model	2-46

2.2.9	Matrix-to-Fracture Transfer Model	2-50
2.2.10	Particle Management	2-52
2.2.10.1	Particle Injection	2-53
2.2.10.2	Particle Monitoring	2-54
2.2.10.3	Particle Exit	2-57
2.2.10.4	Advanced Particle Management	2-58
2.2.11	Electrodes	2-58
3.	CODE DESCRIPTION, USE, AND MODIFICATION	3-1
3.1	Code Structure	3-1
3.2	UNIX Operations	3-2
3.3	Model Control	3-2
3.4	Numerical Routine Control	3-3
3.4.1	Steady-State Solution	3-4
3.4.2	Integration	3-5
3.5	Code Modification	3-7
4.	SAMPLE PROBLEMS	4-1
4.1	Infinite Porous Media	4-1
4.1.1	Isotropic Steady-State Flow	4-3
4.1.2	Anisotropic Steady-State Flow	4-3
4.1.3	Solute Transport at Zero Dispersion	4-3
4.1.4	Solute Transport with Dispersion	4-5
4.2	Verification and Benchmarking Problems	4-7
5.	CODE DEVELOPMENT AND APPLICATION HISTORY	5-1
5.1	Development Summary	5-1
5.2	Raft River Field Tests	5-2
5.3	East Mesa Field Tests	5-3
5.4	Physical Model Tests	5-4
5.5	Sensitivity	5-5
5.6	Work Remaining	5-7
6.	MAINTENANCE AND QUALITY ASSURANCE PROGRAM	6-1
7.	REFERENCES	7-1

APPENDIX A--MODEL NOMENCLATURE	A-1
APPENDIX B--INPUT/OUTPUT SUMMARY	B-1
APPENDIX C--NUMERICAL ROUTINE INPUT/OUTPUT	C-1
APPENDIX D--DIRECTORIES AND FILES	D-1
APPENDIX E--SAMPLE PROBLEM INPUT/OUTPUT	E-1
APPENDIX F--UNIX OPERATIONS	F-1

FIGURES

2.1. Sample FRAC-UNIX model	2-60
2.2. Basic concepts of "random-walk" computer program	2-61
2.3. Progress of a slug around the mean flow	2-62
2.4. Temperature dependence of tracer diffusion coefficients	2-63
2.5. Marker particle and matrix cell schematic diagram	2-64
2.6. Fractures affecting particle movement in the matrix	2-65
2.7. Mass balance for transverse velocity	2-66
2.8. Simple four-branch fracture junction	2-67
2.9. Four-branch discontinuous fracture junction	2-68
3.1. Directory and file structure	3-8
4.1. Comparison of FRAC-UNIX and analytic dispersion in radial flow	4-8
5.1. Raft River Well RRGP-5BF near well-bore fracture system	5-8
5.2. Raft River Well RRGP-5BF near well-bore computer model	5-9
5.3. Slug tracer recovery for Test 56-30-3	5-10
5.4. Marker particle positions after Test 56-30-4	5-11
5.5. Slug tracer recovery from Tests 56-30-4 and -5	5-12
5.6. Test 56-19-4 continuous tracer recovery	5-13
5.7. Test 56-19-6 continuous tracer recovery	5-14

5.8.	Laboratory physical model used for tracer testing	5-15
5.9.	Head loss in the physical model as a function of flow rate	5-15
5.10.	Comparison of tracer concentration data with FRACSL simulated concentrations for Electrode 6	5-16
5.11.	Comparison of tracer concentration data with FRACSL simulated concentrations for Electrode 11	5-16
5.12.	Comparison of tracer concentration data with FRACSL simulated concentrations for Electrode 14	5-17
5.13.	Comparison of tracer concentration data with FRACSL simulated concentrations for Electrode 17	5-17
5.14.	Comparison of tracer concentration data with FRACSL simulated concentrations for Electrode 18	5-18
5.15.	Conceptual model of a geothermal reservoir	5-18
5.16.	Steady state pressure distributions for dual-permeability fracture networks	5-19
5.17.	Tracer breakthrough curves for hydraulically equivalent reservoirs	5-20

TABLES

2.1.	Partial head derivatives	2-27
4.1.	Infinite porous media model parameters	4-2
4.2.	Comparison of FRAC-UNIX and analytical steady state drawdowns ..	4-4
4.3.	Transport of a tracer slug	4-6
5.1.	Summary of dual-permeability systems	5-6

1. INTRODUCTION

The FRAC-UNIX computer code has been under development at the Idaho National Engineering Laboratory (INEL) since 1983 to provide a tool for the analysis of fractured geothermal reservoirs under simultaneous injection and production. EG&G Idaho, Inc. has been conducting this work for the Division of Geothermal Technology of the U.S. Department of Energy.

The approach used in the code is based on two fundamental concepts: the dual-permeability representation of fractured geologic media and the marker particle mechanization of the basic transport processes.

In the dual permeability representation, flow occurs in both fractures and porous matrix. The fractures are both discrete elements of a stochastic fracture system and deterministic features which have been specifically identified by geophysical means.

The marker particle transport models provide direct, mechanistic implementations of the fundamental transport equations. This approach is well suited to the dual permeability representation and avoids the numerical dispersion inherent in standard numerical solution techniques. In this approach, imaginary particles are tagged with specified amounts of solute or energy and traced through the fractures and the rock matrix under the influence of advective, dispersive and diffusive processes. Solute or temperature distribution is then determined from particle population statistics.

The code utilizes Darcian flow in the porous matrix. Flow is calculated in uniform, parallel-sided fractures of infinite depth. Laminar or turbulent flow may be calculated. A special computation is provided for laminar flow in finite depth fractures as found in a bounded physical model. A number of head or flow boundary conditions are provided.

Trace concentrations of conservative solute are modeled; the feedback of solute concentration to the flow calculation has therefore been neglected. In the case of heat transport, the flow and transport computations are coupled through density, viscosity and buoyancy effects. The marker particle models used in the code are highly detailed, incorporating second-order variation of local velocity within the matrix and local variation of longitudinal and transverse velocities within the fractures. Random-walk models are used except in simulating transfer between fractures and matrix where higher-order stochastic models are used to reduce execution times.

The code is written in Fortran as a set of algebraic equations, first-order nonlinear differential equations and mechanistic and stochastic particle movement relationships. Sparse-matrix techniques are used to solve the flow relationships for steady-state and transient cases. The initial version of this code, named FRACSL (Miller, Clemo and Hull, 1987), utilized numerics and a user interface from a commercial problem solver code.

The code is installed on the INEL CRAY XMP24 computer using the UNICOS operating system. While the code has been developed for geothermal studies, this report provides the content specified in the NRC guidelines for codes used in support of license applications for high-level waste disposal (Silling, 1983). Included are discussions of the theoretical background and implementation, the details of code operation, several sample problems, the code development history and the code maintenance program. The appendices provide details including a complete description of the important code variables as incorporated in a data base.

The development of this code, as described under the Code Development and Application History, has been enhanced by the integration with physical model studies and field testing. In addition, substantial effort has been expended in providing features for operational requirements, user convenience and optimum usage of computer resources.

An extensive verification effort has been conducted, including comparisons to analytic solutions and numerical solutions generated on other codes. Preliminary results were presented at a technology transfer meeting at Lawrence Berkeley Laboratory on February 1 and 2, 1989. A discussion of the code approach and capabilities and hands-on training in its use were also presented.

Subsequent development effort has been conducted to add an option for units in common use in the petroleum industry and to improve the coupled heat-flow calculation. Completion of the verification effort would follow the addition of the extensions described within this manual; neither of these efforts is currently planned.

2. THEORETICAL BASIS, APPROACH, AND IMPLEMENTATION

This section presents the theoretical basis for the hydraulics and transport models, the rationale and the mechanics of the approach selected and the details of the implementation of the individual models. The implementation discussion is to be used in conjunction with the Appendices and the section on Code Description, Use and Modification.

2.1 Hydraulics

This section describes the bases for the head and flow equations; the reservoir physical description; and the detailed models, boundary conditions and computational modes.

2.1.1 Conservation of Mass

The equation of continuity is developed (Bird et al., 1960) by applying a mass balance to a small stationary element of fixed volume $\Delta x \Delta y \Delta z$ through which a fluid is flowing

$$\left[\begin{array}{c} \text{rate of mass} \\ \text{accumulation} \end{array} \right] = \left[\begin{array}{c} \text{rate of} \\ \text{mass in} \end{array} \right] - \left[\begin{array}{c} \text{rate of} \\ \text{mass out} \end{array} \right]$$

$$\Delta x \Delta y \Delta z \frac{\partial \rho}{\partial t} = \Delta y \Delta z \Delta_x (\rho v_x) + \Delta x \Delta z \Delta_y (\rho v_y) + \Delta x \Delta y \Delta_z (\rho v_z) \quad (2.1)$$

where

$$\rho = \text{density} \quad \left(\frac{M}{L^3} \right)$$

$$t = \text{time} \quad (T)$$

v = velocity $\left(\frac{L}{T}\right)$

$\Delta_x(\rho v_x) =$ change in flux in the x direction over the interval Δx .

Neglecting the change in density across the control volume, the mass balance becomes

$$V \frac{1}{\rho} \frac{\partial \rho}{\partial t} = Q_x + Q_y + Q_z \quad (2.2)$$

where

V = control volume $\Delta x \Delta y \Delta z$ (L^3)

Q_i = net volumetric flow in the i direction into the control volume. $\left(\frac{L^3}{T}\right)$

The compressibility for the mass of saturated rock is

$$\beta = \frac{1}{\rho} \frac{\partial \rho}{\partial H} \quad (2.3)$$

where

β = compressibility $(H^{-1})^a$
 H = head. (H)

The time rate of change of density is then

$$\frac{\partial \rho}{\partial t} = \frac{\partial \rho}{\partial H} \frac{dH}{dt} = \rho \frac{\beta dH}{dt} \quad (2.4)$$

Substituting in Equation (2.2)

a. H is a unit of pressure defined by the height of a column of water.

$$\frac{dH}{dt} = \frac{Q_x + Q_y + Q_z}{V\beta} \quad (2.5)$$

2.1.2 Matrix Flow

The Darcy equation for the steady state flow of a single phase fluid through a porous medium (Bear, 1972) is

$$v_x = - K_x \frac{\partial H}{\partial x} \quad (2.6)$$

where

$$K_x = \text{hydraulic conductivity} \quad \left(\frac{L}{T}\right)$$

2.1.3 Fracture Flow

Flow in fractures of specified hydraulic (effective) aperture and infinite width is computed for either laminar or turbulent flow regimes. The laminar flow relationship is given by the solution of the Navier-Stokes equation for flow between parallel plates (Neuzil and Tracy, 1981).

$$v = \frac{\rho g}{12\mu} b^2 \frac{\Delta H}{L} \quad (2.7)$$

where

ΔH	=	head loss	(H)
L	=	length	(L)
v	=	velocity	$\left(\frac{L}{T}\right)$
g	=	acceleration of gravity	$\left(\frac{L}{T^2}\right)$
ρ	=	density	$\left(\frac{M}{L^3}\right)$

μ = viscosity $\left(\frac{M}{LT}\right)$
 b = fracture aperture (L)

The turbulent flow relationship is derived from the

Darcy equation for liquid flow in a pipe (Crane, 1969)

$$\Delta H = f \frac{L}{D} \frac{v^2}{2g} \quad (2.8)$$

where

D = hydraulic diameter (L)

f = friction factor (-)

Turbulent flow may exist in a rough fracture at a Reynolds number as low as 80 for a relative roughness of 1.0 (Gale et al., 1985). For fully developed turbulent flow, the friction factor is a function solely of the relative roughness. Using a hydraulic diameter equal to twice the aperture, the Darcy equation gives the velocity in a turbulent flow as

$$v = \sqrt{\frac{4bg}{f} \frac{\Delta H}{L}} \quad (2.9)$$

Flow relationships for a closed channel, as in physical models of fracture systems, are given in the section on Longitudinal Velocity Distribution.

2.1.4 Reservoir Description

FRAC-UNIX models flow and transport in a two-dimensional reservoir defined by a rectangular pattern of grid rows and columns. Heads are determined at each of the row-column intersections (nodes).

Figure 2.1 presents an example of a FRAC-UNIX model. The areas defined by the individual rows and columns, called matrix cells, represent the fine fracture network and the rock itself. The dotted line surrounding the central node is an example of the nodal area, i.e., the area to be associated with a node for fluid accounting purposes. The double lines are discrete fractures. Fractures are restricted to connections between adjacent nodes and only one diagonal fracture may exist in a cell.

2.1.4.1 Model Grid. Columns are numbered from left to right using index I. Rows are numbered from bottom to top using index J. Nodes are indexed from the lower left corner proceeding upwards in the first column and then moving from bottom to top in succeeding columns. This sequence is given by the IJ index: $IJ = J + (I-1) * NROW$. DX and DY are the intervening distances between columns and rows, respectively. XCOL and YROW are the absolute positions of the columns and rows, respectively.

The node configuration can be defined in three different ways using the integer variable IGRID.

If IGRID =

- 0 user supplies the XCOL and YROW information in the input section
- 1 grid system is defined with the origin at node 1. The DX(1) and DY(1) variables are supplied by defining DXZ and DYZ in the input section. Further spacing of rows and columns depends on XEXP and YEXP. With XEXP = 1.2 each DX is 20% larger than the previous DX. The YEXP has a similar meaning.
- 2 similar to IGRID = 1 except the origin is at node = NODIN. In this case, the DX's and DY's each expand away from the origin in two directions, i.e., right and left for DX and up and down for DY. NODIN must be the center node in the X direction and the Y direction.

2.1.4.2 Matrix Cells. Matrix cells have the same index as the node at the lower left corner of the cell. Matrix properties are homogeneous within a cell, can be different for each cell and are indexed by a matrix material number, IMMAT. The matrix properties currently used are:

- MTDSYL-longitudinal dispersivity (L)
- MTDSYT-transverse dispersivity (L)
- MTCONX-conductivity in the x direction at 100°C (L)
- MTCONY-conductivity in the y direction at 100°C (L)
- MTSS-specific storage (H^{-1})
- MTPOR-porosity (-)
- MTTCON-thermal conductivity (ML/T³θ)
- MTRHO-density (M/L³)
- MTCP-specific heat (L²/T²θ)

2.1.4.3 Fractures. Fractures are defined by specifying their individual ending nodes, by specifying all fractures in multi-segment collinear strings or by combining synthesized fractures with deterministic fractures. The end nodes are input as pairs in array FREN (2,NF) with the smaller node index (NOD1) given first. Fractures are only valid if they connect adjacent nodes. FRAC-UNIX will not identify an invalid fracture description, therefore the user must ensure that the fracture definition is valid. Fracture aperture (B) is input for each fracture.

Alternatively, all of the fractures in the system may be specified by inputting the number of fracture strings (NSTR), the number of fractures in each string (NFISTR), the angle of the fractures in the string (PSISTR), the smallest node number on the string (STREND) and the aperture of the string (BSTR).

Finally, a stochastic fracture system may be synthesized and combined with specified (deterministic) fractures. A square grid must be used

(IGRID = 1, XEXP = 1., YEXP = 1., DXZ = DYZ). Independent sets of fractures are synthesized in the vertical (set 1), horizontal (set 2) and diagonal (upwards to the right, set 3) directions.

For the vertical fractures, the synthesis region extends from the origin to WSY in the X direction and from the origin to HTSY in the Y direction. A horizontal scan-line extends across the synthesis region at mid-height. For horizontal fractures the synthesis region and scan-line are rotated 90 degrees clockwise. For diagonal fractures the region and scan-line are rotated 45 degrees clockwise and the region dimensions are internally increased by the square root of two.

Fracture centers are distributed uniformly in the dimension along the scan-line. Fracture length is selected from an exponential distribution with mean equal to the product of factor LFAC and SEPAV, the mean separation of scan-line intercepts. Centers are then located in the dimension normal to the scan-line. While the center position in the normal direction may lie in a very large range, only those positions which result in the fracture penetrating the synthesis region are of interest. The position is therefore selected as equally likely in the interval (Andersson et al., 1984):

$$- \frac{HTSY + L}{2}, \frac{HTSY + L}{2} .$$

Diagonal fractures lying completely outside the WSY by HTSY region are excluded.

Fractures are synthesized until the number of scan-line intercepts corresponds to SEPAV plus an analytic modification for the exclusion of fractures shorter than LCUTL. Fractures longer than LCUTH are then excluded. Individual fractures are shifted laterally and adjusted in length to contact the nearest grid node. Apertures are drawn from exponential distributions with mean values BAV.

Deterministic fractures may be included by specifying the apertures, BVD, BHD and BDD, of the vertical, horizontal and diagonal fractures

beginning at each node and extending upwards, to the right or upwards and to the right, respectively.

The net aperture of the combined vertical, combined horizontal and combined diagonal model fracture is found as the cube-root of the sum of the cubes of the individual apertures. This approximates the system flow characteristics because the individual fractures act in parallel.

The routine then determines the end nodes, FREN, for each fracture lying in a model region which is equal in size or smaller than the synthesis region and has an identical cell size. The model region is specified by defining the first and last synthesis columns to be incorporated in the model, ICOL1 and ICOL2, and the first and last synthesis rows to be incorporated in the model, JROW1 and JROW2.

Fracture length, FRL, and angle, PSI, are calculated by the code. Fracture specific storage, FRSS, is input and is indexed by the fracture material index, IFMAT.

2.1.5 Head and Flow Models

The conservation of mass equation is implemented at a node as

$$DH = (QQMSM + QQFSM + QQEXT)/(VOLM * MTSS + VOLF * FRSS) \quad (2.10)$$

where

DH	=	time rate of change of head	(H/T)
QQMSM	=	net matrix flow into the nodal volume, corrected to density at 100 C	(L ³ /T)
QQFSM	=	net fracture flow into the nodal volume, corrected to density at 100 C	(L ³ /T)
QQEXT	=	external flow into node, input at or corrected to density at 100 C	(L ³ /T)

VOLM	=	volume of matrix within nodal volume	(L^3)
MTSS	=	matrix specific storage	(H^{-1})
VOLF	=	fracture volume within nodal volume	(L^3)
FRSS	=	fracture specific storage	(H^{-1})

and the volume terms are computed for the matrix cell quadrants and half-fractures lying within the nodal volume shown by the dotted lines in Figure 2.1.

As subsequently described, the head distribution is found by integrating the head derivatives or by iterating to a steady state. The matrix and fracture flows are then calculated based on the resulting distribution.

Matrix flow across the boundaries of a nodal volume is computed by using the advective marker particle velocity calculation described in the Matrix Transport section to determine a maximum of eight velocities. A velocity is computed at the center of each half boundary as shown in Figure 2.1. The product of the velocity, the length of the half boundary and the thickness of the model is the flow through the half boundary. Summing over the maximum of eight half boundaries yields the net matrix flow into the nodal volume.

Fracture flow is computed for either turbulent or laminar flow regimes in models with variation of longitudinal velocity in one dimension (model typical of infinitely thick reservoir). In addition, laminar flow may be computed in closed channels in simulations of physical models with cover plates.

For laminar flow of water in infinite channels

$$VFLAV = RTOV * CTPF * GRAV * \frac{B^2}{12} * \frac{DH}{DL} \quad (2.11)$$

where

$$VFLAV = \text{Average fracture velocity} \quad \left(\frac{L}{T}\right)$$

$$DH = H1 - H2 + [\cos(PSIND1) * \sin(PSIX) + \sin(PSIND1) * \cos(PSIY)] * DL * (1 - \frac{RRATF}{RRATN}) \quad (H)$$

$$H1, H2 = \text{Head at fracture end with smaller, larger node number} \quad (H)$$

$$PSIND1 = \text{Fracture angle from due east at end with smaller node} \quad (-)$$

$$PSIX = \text{Angle of x unit vector with horizontal} \quad (-)$$

$$PSIY = \text{Angle of y unit vector with vertical} \quad (-)$$

$$DL = \text{Fracture length} \quad (L)$$

$$B = \text{Parallel plate hydraulic aperture} \quad (L)$$

$$RRATF = \text{Ratio of fracture density to density at } 100^\circ\text{C} \quad (-)$$

$$RRATN = \text{Ratio of native fluid density to density at } 100^\circ\text{C} \quad (-)$$

$$RTOV = \text{Ratio of density to viscosity at } 100^\circ\text{C} \quad \left(\frac{T}{L^2}\right)$$

$$CTPF = \frac{\text{Ratio of density to viscosity at fracture temp}}{RTOV} \quad (-)$$

$$GRAV = \text{Acceleration of gravity} \quad \left(\frac{L}{T^2}\right)$$

and the volumetric flow rate is

$$QF = Z * B * VFLAV \quad (2.12)$$

where

$$Z = \text{model thickness} \quad (L)$$

For turbulent flow at an assumed relative roughness of 0.05, the Moody chart (Crane, 1969) gives a friction factor of 0.072. The resulting flow equation is

$$V_{FLAV} = \left[\frac{4 * GRAV * B}{FRFAC} \frac{dH}{dL} \right]^{1/2} \quad (2.13)$$

where the default value of the friction factor FRFAC is 0.072.

The laminar calculation is the default condition (IQCASE = 1). The user may dictate the regime for individual fractures by inputting IQCASE = 3 and laminar flags LAM = true for laminar fractures and false for turbulent fractures. In addition, a limited capability exists for automatic selection of regime according to Reynolds number.

An initial calculation is made with IQCASE = 1 to define, on the basis of laminar flow, the various flow velocities and Reynolds numbers. The code sets flags LAM true for Reynolds numbers below 3000. A second calculation is made, for a user input IQCASE = 2, in which the appropriate regime is selected depending on the individual value of LAM. This capability is only valid for two successive steady state (by iteration) cases. It should be noted that less restrictive schemes permitting free selection of regime are difficult because of the potential for a bistable condition.

The closed channel flow case is enabled by inputting F3D true. The average flow velocity across the cross-section is given in the section on Longitudinal Velocity Distribution.

2.1.6 Computational Modes

Either steady-state or transient flow conditions can be simulated.

A steady-state case occurs under two circumstances. If the reservoir is in equilibrium with equal volumes of fluid entering and leaving the reservoir, the heads will remain constant. Alternatively, a

quasi-steady-state condition exists if all heads rise and fall at the same rate. A confined aquifer or simulation of a small portion of a well-developed drawdown cone are two examples of the latter scenario. A Newton-Raphson iteration is used to solve for the steady-state or quasi steady-state head distribution.

The transient flow capability uses variable order, variable step integration techniques to simulate changing flow conditions. The time increment DTQ is user supplied and essentially represents a reporting interval. The integration algorithm chooses smaller steps within the DTQ interval to satisfy specified tolerances.

Two logical variables are used to control the flow calculation. The steady-state simulation requires DYNMIC to be false. With SOLVE set true, the iterative capability is used to calculate the head profile. -For a transient flow calculation, DYNMIC is set true and SOLVE is set false.

Solute transport is calculated in a steady flow field by moving particles continuously, or in a transient flow field by interrupting the particle move periodically to update the flow field.

Heat transport is calculated in conjunction with a transient flow simulation. As discussed in Section 2.2.10.2, particles are moved to define the temperatures at the end of the reporting interval. The transient flow computation then advances through this interval with its own smaller steps, using temperatures interpolated for the intermediate times.

2.1.7 Boundary Conditions

The FRAC-UNIX boundary conditions are:

- Impermeable boundaries
- Fixed external flow rate at specified nodes
- Transient variation of external flow rate at the inlet node NODIN

- Fixed head at any node
- A remote boundary connected to the model boundary by fracture and matrix flow paths
- A leaky boundary for flow out of the plane of the model
- Infinite porous media approximation.

The impermeable boundary condition is the default selection.

Fixed flows at 100°C may be specified at any node using the QQEXT array.

The external flow at the inlet node NODIN, again at 100°C, may be input as a function of time by using the flag QEXTTR and the tabular function FLUX.

The FREEZE array may be used to fix the head at any node to its initial value by setting QQEXT for that node equal to the negative of the sum of the internal flows at that node.

A steady-state flow calculation which does not refer the head distribution to a remote boundary must fix the head at any one node by setting the FREEZE element true for that node to preclude an otherwise indeterminate solution.

Any node may be connected to a remote boundary by fractures or matrix material, or both. Fractures within the model which reach a model boundary, and do not lie on that boundary, are continued to the remote boundary by specifying:

FRDEB - set to true for fractures which dead-end on one of the above-indexed nodes and are, therefore, not connected to the remote boundary

- HRB - remote boundary head for each connected node

- INRB - an array of size NNRB containing the indices of the
 nodes with connections to the remote boundary

- LFTORB - fracture length from node to remote boundary for each
 indexed node

- LMTORB - matrix distance from each node to the remote boundary

- NNRB - the number of nodes connected to the remote boundary.

The sum of the cubes of the apertures of all fractures which end at an indexed node, do not lie on the boundary and have FRDEB=FALSE is used to calculate the conductivity from that node to the zero head boundary. Additional conductivity is provided by the matrix path at the conductivity for that node.

A leaky aquifer may be simulated by inputting QQTOHL and HL for that node. Flow out of that node is then calculated as the product of QQTOHL and H-HL.

An approximation of an infinite boundary can be applied as a boundary condition by setting PERMBD true. If transport is being simulated, SYMTRY must be set to 4. This condition assumes:

- Radial symmetry about an injection node located at the lower left corner of the model

- Homogeneous matrix with no fractures.

An infinite boundary is approximated by creating an ellipse of zero head through the upper right hand corner of the model. If the matrix conductivities are isotropic, the ellipse would be a circle. The hydraulic conductivity to this locus is calculated for each boundary node. The model

will respond to stress at the injection node as if the boundaries were at an infinite distance. The heads are calculated relative to the zero head ellipse, rather than the true zero at infinity. The shape of a steady-state simulation would be correct, except for the above offset. At early times in a dynamic simulation, the boundary is not "seen" by the head distribution. At longer times, an error in the rate of change of the head values can be calculated.

A flow term is calculated for each matrix quarter cell edge along the two sides of the model furthest from the injection node (see sketch below). The location used to determine the flow is marked with an "x".

- - - Dashed lines represent nodal volume boundaries
- Dots represent node locations

The flows are found from the Theis equation (Theis, 1935).

Steady-state errors are introduced as follows:

Along the edges of the model, nodes are not centered in the nodal volumes. The head calculated for a node is, therefore, offset by $DX/4$ and $DY/4$.

An error is also introduced in interior nodes because the pressure distribution is not linear.

In a transient simulation, the zero head boundary is a constant; the error introduced is:

$$h_e = \frac{Q}{4\pi kT} \ln \left(2.25 \frac{kTt}{r_e^2 S} \right)$$

Where:

Q	= volumetric flow at injection node	(L ³ /T)
k	= hydraulic conductivity	(L/T)
T	= model thickness	(L)
r _e	= eff. distance of upper right hand node from the injection node	(L)
S	= storage coefficient	(-)
t	= time since pumping started	(T)

2.2 Transport

This section describes the physics of the various transport processes incorporated in FRAC-UNIX, along with the approach and implementations used in the code. FRAC-UNIX models solute transport and heat transport in the same way because the underlying physics of these two transport processes are the same.

The transport processes incorporated include advection, dispersion, and diffusion. Nonconservative processes and radioactive decay are not included. Since trace concentrations are treated, the effect of solute

concentration on fluid density and flow is neglected. Energy transport and the resultant temperature distributions do, however, influence the flow process through the dependence of buoyancy and flow on fluid density and viscosity, as discussed in Section 2.1.

The detailed description of the head and velocity determined by the flow model is used as the basis for the transport model. The individual properties of the matrix and discrete fractures are, therefore, carried over to the transport model. The discussions which follow address matrix transport, fracture transport and their interactions with each other and with the model boundaries.

2.2.1 Advection -Dispersion Equation

The solute transport equation, in one dimension, is given by Prickett, et al. (1981), as:

$$\frac{\partial C}{\partial t} = \frac{\partial}{\partial x} \left(\frac{D}{R_d} \frac{\partial C}{\partial x} \right) - \frac{v}{R_d} \frac{\partial C}{\partial x} \quad (2.14a)$$

Similarly, for heat transport,

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial x} \left(\frac{A}{R_d} \frac{\partial T}{\partial x} \right) - \frac{v}{R_d} \frac{\partial T}{\partial x} \quad (2.14b)$$

where

C	= concentration	(M/L ³)
t	= time	(T)
D	= hydrodynamic dispersion coefficient = d _L v + D _M	(L ² /T)
d _L	= longitudinal dispersivity	(L)
D _M	= molecular diffusion coefficient	(L ² /T)
v	= interstitial velocity	(L/T)
R _d	= retardation factor	(-)

T	= temperature	(θ)
A	= thermal dispersion coefficient = $dT_L v + \alpha$	(L^2/T)
dT_L	= longitudinal thermal dispersivity	(L)
α	= thermal diffusion coefficient = $k/\rho C_p$	(L^2/T)
k	= thermal conductivity	($ML/T^3\theta$)
ρ	= fluid density	(M/L^3)
C_p	= fluid specific heat	($L^2/T^2\theta$)

The left side of equation (2.14a) [and (2.14b)] represents accumulation, while the first and second terms on the right-hand side of the equation represent dispersion and convection, respectively. The similarity between equations (2.14a) and (2.14b) allows the use of same solution approach for both solute and heat transport. While subsequent discussions only mention solute transport, they apply equally to heat transport.

2.2.2 Solution Techniques

The classical method of implementing this equation in a computer code consists of discretizing the solution over space and then solving for the temporal variation, as was done for the conservation of mass. This approach was rejected because of numerical dispersion and stability problems (Wang et al., 1982; Thomas et al., 1982) and because of the computer resources needed to separately nodalize matrix and fracture domains. The selected approach is a modification of the random-walk technique of Ahlstrom et al., (1977) and Prickett et al., (1981). Prickett et al. give an excellent comparison of the two particle-based techniques: random-walk and method-of-characteristics (MOC). Portions of their comparison are given as follows:

A popular numerical technique used for solving (the advection-dispersion equation) is the method of characteristics (MOC), or the particle-in-a-cell method. MOC treats the equation in two parts. First, the convective term containing the velocity (V) is solved with

an adaptation of the usual finite difference flow type of model. Then the dispersive term is solved by using another finite difference grid associated with the concentration distribution. A large number of computer-generated particles move about by the velocity vectors which are solved in the flow model and which carry the concentration information between the convection and dispersion terms during the solution of the equation. The description of the MOC is straightforward, but the computer code is highly involved and very expensive to operate and it requires a large computer to effect a solution. Therefore, researchers have been looking for a more efficient and more direct way to solve problems concerning solute transport in groundwater. The conceived "random-walk" method follows.

The random-walk technique is based on the concept that dispersion in porous media is a random process. On a microscopic basis, dispersion may occur as shown in Figure 2.2. As indicated in Figure 2.2c, dispersion can take place in two directions even though the mean flow is in one direction to the right. A particle, representing the mass of a specific chemical constituent contained in a defined volume of water, moves through an aquifer with two types of motion. One motion is with the mean flow (along streamlines determined by finite differences), and the other is random motion, governed by scaled probability curves related to flow length and the longitudinal and transverse dispersion coefficients. Finally, in the computer code, enough particles are included so that their locations and density, as they move through a flow model, are adequate to describe the distribution of the dissolved constituent of interest.

The advantages of this random-walk technique over the MOC or, for that matter, over many other numerical schemes are many:

1. There is no dispersion equation to solve. The dispersion part (of the advection-dispersion equation) is solved in the computer code by the addition of only 11 Fortran statements attached to the solution of the convection part (of the equation).
2. There is only one finite difference grid involved in solving the convective portion (of the equation). The particle movement takes place in continuous space.
3. Concentration distribution needs to be calculated only when it is of interest. In the MOC, after each particle is moved, new concentrations are assigned on the basis of the solution of the dispersion term in the above equation for every time step of the simulation.
4. Computer CPU time is drastically reduced. — — —
5. Particles are needed only where water quality is of interest. These particles are not needed everywhere in the model, as with the MOC.

6. Solutions are additive. If not enough particles are included for adequate definition in one run, a second run can be done and the results accumulated. This is not true of the MOC, where possible sparse areas of particles may occur, causing loss of accuracy.
7. This method is particularly suited to time-sharing systems where velocity fields can be stored and manipulated in conjunction with an on-line particle mover code.
8. In the traditional sense of the words, finite differencing phenomena associated with "over-shoot" and "numerical dispersion" are eliminated.

Although there are numerous advantages to this technique, there are also some disadvantages:

1. As with the MOC, concentrations greater than initial conditions are possible, especially when coarse discretizing is used.
2. A printout of concentrations may not be pleasing to the eye when the number of particles is small. -- -- -- -- --
--
3. The method may take an unusually large number of particles to produce an acceptable solution for some problems. No more than 5,000 particles are used in this (Prickett's) report, however.
4. Engineering judgment is an absolute requirement in arriving at an acceptable solution. This is because of the "lumpy" character of the output. Therefore, experience with this technique is needed before one can apply the code successfully to a field situation.

The FRAC-UNIX approach, as subsequently discussed, follows the random-walk technique except for the use of second order gradients and the use of higher order models to avoid the computational burden imposed by an excessive number of very small steps.

2.2.3 Random-Walk Dispersion

The following discussion, also from Prickett et al., (1981), is pertinent at this point in the discussion.

To illustrate the details of the random-walk technique as it relates to dispersion, consider the progress of a unit slug of tracer-marked fluid, placed initially at $x = 0$, in an infinite column of porous medium with steady flow in the x direction. The advection dispersion equation describes the concentration of the slug as it moves downstream. Bear (1972) gives the solution as

$$C(x,t) = \frac{1}{(4\pi d_L vt)^{1/2}} \exp - \left[\frac{(x - vt)^2}{4d_L vt} \right] \quad (2.15)$$

where

C	=	concentration	(-)
d_L	=	longitudinal dispersivity	(L)
v	=	interstitial velocity	$\left(\frac{L}{T}\right)$
t	=	time	(T)
x	=	distance along the x axis.	(L)

The shapes of the curves $C(x',t)$ are shown in Figure 2.3 where $x' = x - vt$.

Based upon concepts found in comprehensive statistics books (for example, see Mood and Graybill, 1963) a random variable x is said to be normally distributed if its density function, $n(x)$, is given by

$$n(x) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp - \left[\frac{(x - u)^2}{2\sigma^2} \right] \quad (2.16)$$

where

σ	=	standard deviation of the distribution
u	=	mean of the distribution.

Now, let us equate the following terms of Equations (2.15) and (2.16) as

$$\sigma = \sqrt{2d_L vt} \quad (2.17)$$

$$u = vt \quad (2.18)$$

$$n(x) = C(x,t). \quad (2.19)$$

With the identities of Equations (2.17) through (2.19) taken into account, it will be seen that Equations (2.15) and (2.16) are equivalent.

The key to solute transport as described in this report is the realization that dispersion in a porous medium can be considered a random process, tending to the normal distribution. There are no new concepts here, as very complete discussions of statistical models of dispersion and the way dispersion affects water quality have been given by others, including Bear (1972) and Fried (1975). The method by which this statistical approach gets into the computer code and is applied to transport problems is new. -- -- -- -- --

This approach is implemented in FRAC-UNIX as follows: During a time increment (Δt) a particle with coordinates xx,yy is first moved from an old to a new position in the aquifer by convection according to its velocity at the old position v_x . Then a random movement in the $+x$ or $-x$ direction is added to represent the effects of dispersion. This random movement is given the magnitude

$$\sqrt{2d_L DX} \cdot \text{ANORM}(0)$$

where

$$\sqrt{2d_L DX} = \sigma = \sqrt{2d_L v_x \Delta t} \quad (2.20)$$

$\text{ANORM}(0)$ = a number between -6 and +6, drawn from a normal distribution of numbers having a standard deviation of 1 and a mean of zero.

The new position of the particle is the old position plus a convective term ($v_x \Delta t$) plus the effect of the dispersion term,

$$\sqrt{2d_L DX} \text{ ANORM}(0) \quad .$$

If the above process is repeated for numerous particles, all having the same initial position and convective term, a map of the new positions of the particles can be created having the discrete density distribution

$$C(x,t) \rightarrow n(x) \rightarrow \frac{N}{d_x} = \frac{N_0}{\sqrt{2\pi} \sqrt{2d_x DX}} \exp - \left[\frac{(x - v_x \Delta t)^2}{4d_x v_x t_x} \right] \quad (2.21)$$

where

d_x = incremental distances over which N particles are found

N_0 = total number of particles in the experiment.

Equations (2.15), (2.16), and (2.21) are equivalent, with the exception that Equations (2.15) and (2.16) are continuous distributions and Equation (2.21) is discrete.

The complete FRAC-UNIX dispersion solution includes a similar random component in the y or transverse direction.

2.2.4 Molecular Diffusion

The random movement of the solute molecules due to their thermal energy is incorporated by adding a displacement

$$\sqrt{2 D_M \Delta t} \text{ ANORM}(0)$$

where

$$D_M = \text{molecular diffusion coefficient} \quad \left(\frac{L^2}{T}\right)$$

(See Figure 2.4).

The longitudinal and transverse components of this movement are determined by multiplying by the cosine and sine of a random number, corresponding to a random direction of motion.

2.2.5 Matrix Transport Model

The following sections describe the FRAC-UNIX implementation of the random-walk approach for matrix transport.

2.2.5.1 Advective Motion. Although FRAC-UNIX is based on the developmental work of Prickett et al., the use of the hydraulic solution to solve the advective term in Equation (2.14) differs greatly from the Prickett et al. code. The Prickett et al. code uses linear interpolation of the flow velocities from the hydraulic finite difference solution. This was deemed inadequate to simulate the flow perturbations caused by fractures.

The curvature of the head distribution is determined from the values at the node nearest the particle and the eight nodes adjacent to it. Figure 2.5 is a schematic of these nine nodes and the particle. The schematic is valid as long as the particle remains in the nodal volume of the central node, which is outlined by the dashed lines. When a particle leaves the nodal volume, a new central node is established. From the nine values a head distribution can be calculated within the nodal volume which is continuous to the second derivative in both the x and y directions. At any point, the head can be calculated using a Taylor series expansion of head variation with location.

$$\begin{aligned}
H(x,y) = & H_0 + \left. \frac{\partial H}{\partial x} \right|_0 dx + \left. \frac{\partial^2 H}{\partial x^2} \right|_0 \frac{dx^2}{2} + \left. \frac{\partial H}{\partial y} \right|_0 dy + \left. \frac{\partial^2 H}{\partial y^2} \right|_0 \frac{dy^2}{2} + \left. \frac{\partial^2 H}{\partial x \partial y} \right|_0 dx dy \\
& + \left. \frac{\partial^3 H}{\partial x^2 \partial y} \right|_0 \frac{dx^2 dy}{2} + \left. \frac{\partial^3 H}{\partial x \partial y^2} \right|_0 \frac{dx dy^2}{2} + \left. \frac{\partial^4 H}{\partial x^2 \partial y^2} \right|_0 \frac{dx^2 dy^2}{4} \\
& + \text{Higher Order Terms}
\end{aligned} \tag{2.22}$$

where

$H(x,y)$ is head at the particle location

H_0 = head at the central node

$\left. \frac{\partial H}{\partial x} \right|_0$ = partial derivative of H with respect to x , evaluated at the central node

$\left. \frac{\partial H}{\partial y} \right|_0$ = partial derivative of H with respect to y , evaluated at the central node

dx = distance of the particle from the central node in the positive x direction

dy = distance of particle from the central node in the positive y direction

When the higher order terms are included, Equation (2.22) is exact. If the head information used is limited to the nine nodes of Figure 2.5, then all of the higher order terms are zero. The equations for calculating the eight partial differentials are listed in Table 2.1.

Assuming laminar flow, the velocity of the particle in a particular direction is directly proportional to the gradient of the head in that direction,

$$\begin{aligned}
 v_x &= -\frac{K_x C_T}{\theta} \left[\frac{\partial H}{\partial X} - \left(1 - \frac{\rho_M}{\rho_N} \right) \sin \phi_x \right] \\
 v_y &= -\frac{K_y C_T}{\theta} \left[\frac{\partial H}{\partial Y} - \left(1 - \frac{\rho_M}{\rho_N} \right) \cos \phi_y \right]
 \end{aligned}
 \tag{2.23}$$

where

v_i	=	velocity in the i th direction	$\left(\frac{L}{T}\right)$
K_i	=	hydraulic conductivity at 100°C in the i th direction	$\left(\frac{L}{T}\right)$
C_T	=	$\frac{\rho_M \mu_R}{\rho_R \mu_M}$	(-)
ρ_M	=	density of water in matrix	$\left(\frac{M}{L^3}\right)$
ρ_R	=	density of water at 100°C	$\left(\frac{M}{L^3}\right)$
ρ_N	=	density of native water	$\left(\frac{M}{L^3}\right)$
μ_M	=	viscosity of water in matrix	$\left(\frac{M}{LT}\right)$
μ_R	=	viscosity of water at 100°C	$\left(\frac{M}{LT}\right)$
θ	=	porosity	(-)
ϕ_x	=	angle of x axis with horizontal	(-)
ϕ_y	=	angle of y axis with vertical	(-)

TABLE 2.1 PARTIAL HEAD DERIVATIVES

$$\frac{\partial H}{\partial x} \Big|_i = \frac{H_{i3} - H_{i1}}{dx_1 + dx_2}$$

$$\frac{\partial H}{\partial y} \Big|_j = \frac{H_{3j} - H_{1j}}{dy_1 + dy_2}$$

$$\frac{\partial^2 H}{\partial y^2} \Big|_j = \left[\left(\frac{H_{3j} - H_{2j}}{dy_2} \right) - \left(\frac{H_{2j} - H_{1j}}{dy_1} \right) \right] \frac{2}{dy_2 + dy_1}$$

$$\frac{\partial^2 H}{\partial x^2} \Big|_i = \left[\left(\frac{H_{i3} - H_{i2}}{dx_2} \right) - \left(\frac{H_{i2} - H_{i1}}{dx_1} \right) \right] \frac{2}{dx_2 + dx_1}$$

$$\frac{\partial^2 H}{\partial x \partial y} \Big|_{22} = \left(\frac{\partial H}{\partial x} \Big|_3 - \frac{\partial H}{\partial x} \Big|_1 \right) \frac{1}{dy_1 + dy_2}$$

$$\frac{\partial^3 H}{\partial x^2 \partial y} \Big|_{22} = \left(\frac{\partial^2 H}{\partial x^2} \Big|_3 - \frac{\partial^2 H}{\partial x^2} \Big|_1 \right) \frac{1}{dy_1 + dy_2}$$

$$\frac{\partial^3 H}{\partial x \partial y^2} \Big|_{22} = \left(\frac{\partial^2 H}{\partial y^2} \Big|_3 - \frac{\partial^2 H}{\partial y^2} \Big|_1 \right) \frac{1}{dx_1 + dx_2}$$

$$\frac{\partial^4 H}{\partial x^2 \partial y^2} \Big|_{22} = \left[\left(\frac{\partial^2 H}{\partial x^2} \Big|_3 - \frac{\partial^2 H}{\partial x^2} \Big|_2 \right) \frac{1}{dy_2} - \left(\frac{\partial^2 H}{\partial x^2} \Big|_2 - \frac{\partial^2 H}{\partial x^2} \Big|_1 \right) \frac{1}{dy_1} \right] \frac{2}{dy_2 + dy_1}$$

2.2.5.2 Matrix Transport Implementation. FRAC-UNIX calculates intermediate head values at the intersections of rows with a vertical line through the particle location and at the intersection of columns with a horizontal line through the particle location. These locations are shown in Figure 2.5 as the x' and y' positions, respectively.

The heads at x' , y' locations are calculated from the appropriate row or column assuming a separable second order pressure profile. The V_x and V_y are then determined from the primed heads using Equation (2.23). This procedure is equivalent to the equations derived from the Taylor expansion. It is used because of the relative ease in adjusting the effects of fractures compared to the Taylor expansion.

The equations for acceleration in the x direction contain terms dependent on movement in the y direction and vice versa. This makes an explicit solution for movement of the particle intractable. Instead, the situation is simplified by approximating the acceleration as constant over a time step. The time step is limited to ensure that the errors induced by the constant acceleration approximation are below a given percentage.

Once the movement duration has been determined the total distance is calculated as the sum of advective, dispersive and diffusive movements:

$$DL = ADL + DISPL + DIFL \quad (2.24)$$

where

$$DL = \text{net distance moved} \quad (L)$$

$$ADL = \text{advective movement} \quad (L)$$

$$DISPL = \text{dispersive movement} \quad (L)$$

$$DIFL = \text{diffusive movement} \quad (L)$$

each term has x and y components:

$$ADL = (ADLX^2 + ADLY^2)^{1/2} \quad (2.25)$$

$$ADLX = VX * DT + ADXDT * DT^2/2 \quad (2.26)$$

$$VX = \text{velocity in x direction} \quad \left(\frac{L}{T}\right)$$

$$ADXDT = \text{acceleration in x direction} \quad \left(\frac{L}{T^2}\right)$$

$$DT = \text{time step} \quad (T)$$

$$DISPLL = \eta_1 * (2 * MTDSYL * ADL)^{1/2} \quad (2.27)$$

$$DISPLL = \text{dispersion in the direction of movement} \quad (L)$$

$$\eta_i = \text{Gaussian random number of zero mean and unit variance} \quad (-)$$

$$MTDSYL = \text{longitudinal dispersivity of matrix cell} \quad (L)$$

$$DISPLT = \eta_2 * (2 * MTDSYT * ADL)^{1/2} \quad (2.28)$$

$$DISPLT = \text{dispersion perpendicular to direction of movement} \quad (L)$$

$$MTDSYT = \text{transverse dispersivity of matrix cell} \quad (L)$$

$$DISPLX = \frac{ADLX}{ADL} * DISPLL - \frac{ADLY}{ADL} * DISPLT \quad (2.29)$$

$$DISPLY = \frac{ADLX}{ADL} * DISPLT + \frac{ADLY}{ADL} * DISPLL$$

$$DIFLX = \cos (PSIDIF) * \eta_3 * (2 * DIFCO * T)^{1/2} \quad (2.30)$$

$$PSIDIF = \text{uniform random angle from x axis} \quad (L)$$

$$DIFCO = \text{diffusion coefficient} \quad \left(\frac{L^2}{T}\right)$$

thus,

$$DLX = ADLX + DISPLX + DIFLX \quad (2.31)$$

$$DLY = ADLY + DISPLY + DIFLY$$

where

$$DLX = \text{net movement in x direction for time step DT} \quad (L)$$

$$DLY = \text{net movement in y direction for time step DT} \quad (L)$$

The movement is repeated until the particle has moved the full time step, left the nodal volume, entered a fracture or entered a new matrix cell. A particle leaves the nodal volume if the central node is no longer the closest to the particle. If it crosses the central row or column, it has entered a new matrix cell. A particle will enter a fracture if the calculated movement would cause it to cross the fracture. If a particle moves to within a distance RWEL of the specified NODOT, it is treated as if it entered the well. If a particle has not moved the full time step, then a new movement calculation is initiated in the particle's new environment with the time available for movement reduced by the amount used to enter the new domain.

2.2.5.3 Fracture Effects on Matrix Transport. The solid lines in Figure 2.6 represent all the possible fracture locations which may affect the movement of a particle in the matrix. Not all of these fractures may

exist concurrently. In particular, only one diagonal fracture may exist in a matrix cell.

There are three primary fracture effects on particle movement.

1. Head varies linearly along a fracture.
2. The first derivative of head need not be continuous across a fracture. Therefore, no head information affects a particle if a fracture exists between the node and the particle.
3. A transient flow component exists near a fracture due to local storage of fluid. This represents fluid which exits the fracture and is stored within the nodal volume. This flow is not represented in the head distribution because the fluid does not cross the edges of the nodal volume.

The treatment of the three effects are given in the same order as follows:

1. The intermediate head values (at x' and y' of Figure 2.5) on a horizontal or vertical fracture are found from a linear interpolation between the fracture end nodes.
- 2a. If a diagonal fracture intersects a line connecting the x 's or the y 's, then the head at the intersection is found as in Item 1 above. The distance between the x 's or y 's is calculated from the intersection location also.
- 2b. If a fracture exists on the central row or column, then the head variation is assumed to be linear between the x 's or y 's on either side of the particle.
3. A velocity term which is normal to the fracture is calculated for each fracture in the nodal volume. The term is found from the head rise in the volume which is due to mass storage. The

velocity is assumed to decrease linearly from the value at the edge of the fracture (see Transverse Velocity section below) to zero at the edge of the nodal volume.

The acceleration term is also affected by the fractures. Again corresponding to the three primary fracture effects:

1. If the x's and y's are on a fracture, the second derivative of head is set to zero.
- 2a. If a diagonal fracture exists in the same matrix cell as the particle then all second derivatives of head are set to zero.
- 2b. If a diagonal fracture exists in a matrix cell different from the particle's cell and does not have the central node as an end, then the second derivatives of the outside row and column intersected by the fracture are set to zero.
- 2c. If a diagonal fracture exists in a different matrix cell and does contain the central node as an end node, then the second derivatives of the central row and column are set to zero as are the second derivatives of the row and column furthest from the particle.

In addition $\frac{\partial^4 H}{\partial x^2 \partial y^2} = 0$

and $\frac{\partial^3 H}{\partial x^2 \partial y}$ and $\frac{\partial^3 H}{\partial x \partial y^2}$ are evaluated between

the central row and column and the row and column on the other side of the particle.

- 2d. If a fracture exists on the central row and column but does not intersect a line connecting the x's or y's, then it is treated similarly to Case 2c.
- 2e. If a fracture exists on a central row or column and intersects a line connecting the x's or y's, then all second derivatives perpendicular to the fracture are set to zero and the first derivatives are evaluated between the central row or column and the opposing row or column.
- 3. A constant second derivative term is added to that calculated by the heads which are consistent with a linear decrease in velocity away from the fracture.

2.2.5.4 Nodal Boundary Considerations. Particle movements are limited by the dashed border in Figure 2.5. Movement is also limited by the possibility of a particle crossing fractures or moving into a new matrix cell with different properties. If the unrestrained movement of a particle would move it past a fracture or into a different type of element, then the particle is restricted to move to the material discontinuity.

The time to reach the boundary is calculated by assuming the dispersive movement is a function of the square root of time. The equation for time is therefore quadratic with respect to the movement limit. This routine is likely to be replaced by a Macro Monte Carlo implementation similar to the special high Peclet number fracture-to-matrix transfer routine.

Subroutine OSHOOT provides the logic to determine whether particle movement will be restricted. The routine allows an overshoot of the dashed boundary because dispersion dominates at short distances. If a particle is within a distance less than the dispersivity coefficient then the movement of the particle is poorly correlated to the direction of advection. In

practice if a particle were constrained to move a trivial distance past a boundary, then on its next move the particle would have nearly a fifty percent chance of recrossing the border because of dispersion. The overall movement of a particle is not affected by numerous recrossings of the boundary but an execution load is put on the computer which severely increases the cost of a simulation. The problem was alleviated by allowing particles to overshoot the boundary. This introduces an error in the movement routine which is minimized by keeping the overshoot small.

2.2.5.5 External Source Approximation. If the central node in Figure 2.5 is either the injection or withdrawal well and no fractures extend from that node, then a radial flow situation is assumed. The special routine solves for the average velocity in a time step and sets the acceleration terms to zero. This routine is based on an analytic solution. For an infinite porous medium, Hoopes and Harleman (1967) give the radial variation of head for an isotropic conductivity as

$$H_{r_1} = H_{r_2} - \frac{Q}{2\pi z K} \ln \frac{r_1}{r_2} \quad (2.32)$$

where

$$H_{r_i} = \text{head at radius } r_i \quad (H)$$

$$Q = \text{volumetric flow rate} \quad \left(\frac{L^3}{T}\right)$$

$$K = \text{hydraulic conductivity} \quad \left(\frac{L}{T}\right)$$

$$z = \text{aquifer thickness} \quad (L)$$

The velocity is

$$v = \frac{Q}{2\pi z} \frac{1}{r} \quad (2.33)$$

The average velocity between r_1 and r_2 is

$$\bar{v} = \frac{1}{r_2 - r_1} \int_{r_1}^{r_2} v(r) dr = \frac{Q}{2\pi z (r_2 - r_1)} \ln \frac{r_1}{r_2} \quad (2.34)$$

The anisotropic case can be derived by transforming coordinates such that $x' = ax$ and $y' = y/a$.

where

$$a = \left(\frac{K_y}{K_x} \right)^{.25} \quad (2.35)$$

in the transformed coordinates, the conductivity is isotropic and Equation (2.32) applies with:

$$r_i = \sqrt{a^2 x_i^2 + y_i^2 / a^2} \quad (2.36)$$

and

$$K = \sqrt{K_x K_y} \quad (2.37)$$

2.2.6 Fracture Transport Model

The equation describing transport in the fractures is the same Equation (2.14) as in the matrix with the exception that the dispersion coefficient D_d is zero. The following sections describe the computation of the local longitudinal and transverse velocities and the diffusive motion.

2.2.6.1 Longitudinal Velocity Distribution. The FRAC-UNIX code treats parallel walled channels of infinite depth in which the longitudinal velocity depends on the transverse position in the fracture. It also treats

closed rectangular channels (F3D = .T.) in which the longitudinal velocity depends on both transverse and vertical positions. The rectangular channel treatment is strictly valid for channels with aspect ratios (depth/width; a/b) greater than five.

The Poiseuille velocity profile for laminar, viscous flow in a channel of infinite depth and width b gives the velocity at a distance y from the edge of the fracture as:

$$v = 6 \bar{v} [y/b - (y/b)^2] \quad (2.38)$$

where $\bar{v}$ is the average velocity across the channel.

A closed, rectangular channel has a two-dimensional distribution of longitudinal velocity. The velocity distribution across the width of the channel resembles the Poiseuille velocity distribution. The vertical distribution, however, is relatively square due to the generally greater distance between the upper and lower boundaries. The exact distribution of velocity can be calculated using (Happel and Brenner, 1965, p 38):

$$v(y,z) = \frac{\rho g (H_2 - H_1)}{2\mu L} \left\{ y(y - b) + \frac{2b^2}{\pi^3} \right. \\ \cdot \sum_{m=1}^{\infty} \left[\frac{[\cos(m\pi) - 1] [\sin(m\pi y/b)]}{m^3 (1 + \exp(-m\pi a/b))} \right. \\ \cdot \left. \left. \left[\exp(-m\pi z/b) + \exp(-m\pi (a - z)/b) \right] \right] \right\} \quad (2.39)$$

where

$$\begin{aligned} v &= \text{longitudinal velocity at } y, z & \left(\frac{L}{T}\right) \\ y &= \text{transverse position in fracture} & (L) \\ z &= \text{vertical position in fracture} & (L) \end{aligned}$$

a = depth of fracture (L)

b = width of fracture (L)

L = fracture length (L)

Because the calculation of longitudinal velocity using Equation (2.39) is computationally intensive, a simplified approach has been implemented in the FRAC-UNIX code. The technique breaks the velocity at a point in the fracture into three components. This breakdown is valid for fractures with aspect ratios (a/b) greater than five.

For a fracture of infinite depth the peak velocity at the center of the fracture ($y/b = 0.5$) is $1.5 \bar{v}$. For rectangular channels the peak velocity is greater than $1.5 \bar{v}$. Equation (2.39) was solved for $y/b = 0.5$ and $z/a = 0.5$ for a range of aspect ratios. The resulting data were fit using a least squares technique to provide a simplified means of calculating the peak velocity.

$$v(\max) = 1.5 + 1.1664 (a/b)^{-1.0557} \quad (2.40)$$

Since this calculation is dependent only on fracture aspect ratio it need only be done once for each fracture at the beginning of the run. Therefore, for fractures with aspect ratios less than 5, Equation (2.39) is used to calculate $v(\max)$. The value of $v(\max)$ is normalized to $\bar{v}$.

The transverse and vertical velocity profiles are independent and of essentially constant shape for channels with aspect ratios greater than five. Both profiles are normalized to $v(\max)$. The transverse velocity profile is identical to the Poiseuille profile:

$$v(y) = 4 [y/b - (y/b)^2] \quad (2.41)$$

The vertical velocity profile was obtained by fitting a polynomial to data generated using Equation (2.39); where

$$v(z) = 15.56 (z/a) - 97.7 (z/a)^2 + 308 (z/a)^3 - 513 (z/a)^4 + 431 (z/a)^5 - 144 (z/a)^6 \quad (2.42)$$

Because $v(y)$ and $v(z)$ must be calculated for every particle every time step, Equations (2.41) and (2.42) are used for fractures of any aspect ratio.

For infinite parallel plates ($F3D = .F.$), $v(\max)$ is set equal to 1.5 and $v(z)$ is set equal to 1.0. Equation (2.41) therefore, gives an exact longitudinal velocity for a Poiseuille velocity profile. The actual longitudinal velocity at any point y, z within a fracture is then calculated by:

$$v(y, z) = \bar{v} \cdot v(\max) \cdot v(y) \cdot v(z) \quad (2.43)$$

2.2.6.2 Transverse Velocity. Matrix permeability, and hence fracture wall permeability, is an important feature in the systems FRAC-UNIX has been designed to simulate and is included in the current code. It is assumed that the transverse flows into the matrix are small enough that the longitudinal velocity previously described is unaffected.

Transverse fracture velocities are not explicitly driven by the head distribution. Instead there may be a transfer of fluid between fractures and adjacent matrix material to preserve the mass balance within the node. This fluid transfer is calculated as the net difference in inflow, outflow and fluid storage in the node between each two fractures dissecting the node. In the example shown in Figure 2.7, there are two fractures dissecting the node (fractures 2 and 3). The equations for flow from the fractures to the matrix for the system shown in Figure 2.7 are:

$$Q_{f2-m}(\text{top}) = Q_{f3-m}(\text{top}) = (Q_3 + Q_4 - Q_5 - 3/8 \frac{dH}{dt} SV)/2 \quad (2.44)$$

$$Q_{f2-m}(\text{bot}) = Q_{f3-m}(\text{bot}) = (Q_1 + Q_2 - Q_6 - Q_7 - Q_8 - 5/8 \frac{dH}{dt} SV)/2 \quad (2.45)$$

where

S = specific storage (H^{-1})

V = volume. (L^3)

The sum of the flows through the fracture walls are apportioned equally to the two fracture segments.

The transverse velocity at any point in the fracture results from the fluid transfer between the fracture and the matrix. Linear interpolation between the two edges of the fracture provides the value at any point in the fracture.

2.2.6.3 Diffusive Considerations. The mechanics of moving a particle in a fracture depend on the relative dominance of diffusion or turbulence compared to laminar advection. If transport in the fracture is diffusion dominated (fracture Peclet number, PE, >200) an advanced Monte Carlo routine is used as discussed in the section on Fracture-to-Matrix Transfer. While usage of the Monte Carlo routine is appropriate for turbulent flow conditions described earlier, the logic to implement the routine is not yet in place.

The low Peclet case, PE <200, uses a two step predictor-corrector algorithm. Equation (2.38) is used to determine the longitudinal advective velocity and the transverse velocity is computed as described above. Diffusion is estimated by a random walk simulation and superimposed on the advective movement. From the predicted final positions in the fracture new velocities are calculated based on constant velocity in the y direction. These new velocities are used in conjunction with the diffusive term to calculate the final position. The average longitudinal velocity for a move from y_1 to y_2 is

$$\bar{v} = 4[(y_1 + y_2)/2 - (y_1^2 + y_2^2)/3.] \quad (2.46)$$

If the particle is approaching the end of a fracture which does not join with another fracture, the velocity is adjusted for fluid loss out the sides of the fracture, which reduces the longitudinal velocity to zero at the dead-end.

2.2.7 Fracture-to-Fracture Transfer Model

The final model in the fracture travel regime is the transfer across a junction into another fracture. This model is not applied to dead-ended fractures or to fracture junctions with an external flow QQEXT. Particle transfer from dead-end fractures is included in the treatment given in the section on Fracture to Matrix Transfer Model. Particles entering a node with a negative QQEXT are removed from the system. Particles entering a node with a positive QQEXT (inflow) are returned to the fracture from whence they came.

Longitudinal and lateral velocities in the fracture, combined with random diffusive movements, establish the particle lateral position as it enters a junction. While most real fracture junctions have 3 or 4 branches, the model treats up to the 8 branch maximum permitted in FRAC-UNIX fracture layouts. The transfer model establishes, for given geometry and branch flows, the routing of the individual streams. Individual particle paths are then calculated on appropriate streamlines through the junction. Tracer transfer between streamlines due to diffusive movement can be included in the calculation for junctions of more than 2 branches.

Flow in the junction area is described as a number of discrete streams each originating at a fully developed region of an inlet fracture, moving through a transition, the junction itself, a second transition and terminating at a fully developed region of an exit fracture. Simulations with the SALE (Simplified Arbitrary Lagrangian-Eulerian) fluid dynamics code (Amsden et al., 1980) have shown that the Poiseuille profile is almost completely established within two channel widths from the junction, in

either inlet or exit branches. Fracture apertures are a very small fraction of their length as shown, for example, by Rouleau and Gale (1984). The travel in a total transition length equal to four apertures is therefore neglected and particles are transferred directly from an established lateral position at the end of an inlet fracture to a lateral position at the beginning of the appropriate exit fracture.

The sequence of branches around the junction may incorporate a continuous series of inlets or may be interrupted by an exit branch, in which case it is identified as discontinuous.

A fracture junction of three branches is continuous for any reservoir condition since any two inlet branches must be adjacent. Discontinuous sequences may exist for junctions of four or more branches in reservoirs with multiple production or injection wells or with injection into a reservoir with a natural hydraulic gradient. The model treats both types of junctions.

2.2.7.1 Routing for Continuous Sequences. Figure 2.8 shows a simple four-branch fracture junction which illustrates the approach and the format used throughout this discussion. As shown at the top of the figure, volumetric flows of six and four units enter from the right and top, respectively. Flows of seven and three units leave from the left and bottom, respectively. The sequence of inlet branches is continuous, i.e., not interrupted by an exit fracture.

The center sketch shows the actual streamlines. Streamline A is at the inlet (and exit) origin of the flow scale and, as indicated in the center figure, moves from the edge of Branch 3 to the edge of Branch 2. The other streamlines show the overall flow pattern: Branch 3 dominates, filling Branch 2 and part of Branch 1. Two basic properties are illustrated: no streamline crosses another, and streamlines from adjacent branches, e.g., B and C, are in the same direction for most of their length. Shown around the outside of the middle sketch are the flow scales: counterclockwise (CCW) from due east for the inlet flow and clockwise (CW) from due east for the exit flow.

The sketch at the bottom of the figure is a schematic of the transfer function used in the model. The vertical scale is cumulative flow running from zero at the top of the figure to the total inlet flow (or exit flow) at the bottom of the figure. The left side of the sketch is the junction inlet scale and the right side the exit scale. The inlet scale begins with the first inlet streamline starting at due east and proceeding CCW around the junction. The branches encountered in this sweep are shown on the figure; first No. 3 and then No. 4. The exit scale on the right side of the figure represents the exit streamlines in CW sequence around the junction from the first inlet streamline. Branches 2 and 1 are encountered in that order. Streamlines are indicated on this figure by the horizontal arrows labeled A through F.

Also shown in the lower sketch are the particle location extremes in each branch. Lateral particle (fractional) position, as represented in FRAC-UNIX, ranges from 0 at the left edge of vertical or diagonal fractures and the bottom of horizontal fractures to 1.0 at the opposite edge. The zero edge is shown by an arrow from a small circle and the 1.0 edge by an arrow to a line.

2.2.7.2 Routing for Discontinuous Sequences. Figure 2.9 shows a streamline schematic for a discontinuous four branch sequence. Note that Branch 4 interrupts the sequence of inlet branches, 3 and 1. The scheme shown satisfies continuity, i.e., it provides seven flow units in from the right, three in from the left, seven out the top and three out the bottom. In addition, there are no crossing streamlines. The flow distribution satisfies the additional criteria that flow at all points on adjacent streamlines travel in the same direction. This requires that each exit channel receive water from the two inlet channels in the ratio of their flows. The existence of a stable flow condition satisfying this criteria was demonstrated in Plexiglas models by Hull and Koslow (1986).

The essential features of the arrangement in this figure are the following: the exit scale is started with the first exit fracture (No. 2) CW of the first inlet fracture. This is the convention adopted for the continuous case: the inlet scale proceeds CCW and the exit scale proceeds

CW from the same point on the circumference of the junction. The next exit fracture (No. 4) spans the break between inlet fractures (3 and 1). The exit scale is completed by the initial exit fracture which, therefore, spans the other break between the two inlet fractures.

The circular layout of the exit scale on this diagram incorporate two complete revolutions of the figure plus a small additional amount. Note that the exit flow in Branch 4 between Streamlines C and D is placed on the cumulative flow scale before the first (CW) part of the fracture between streamlines E and F.

The strategy used in the current program centers the initial exit group on the beginning--end of the exit flow scale. The entire sequence is shifted as necessary to assure that each group of exit fractures spans a break between inlet fracture groups. The algorithm will be modified to incorporate the criteria determined experimentally.

Studies on a discontinuous junction of five branches have shown that the strategy may be applied by grouping adjacent inlet (or exit) branches. Tests on a larger number of branches have not been conducted. The selected strategy should, however, be equally valid.

2.2.7.3 Stream Velocity Profiles. Individual streamlines are traced through a fracture junction by matching the cumulative flow from the origin of the exit flow scale to the same cumulative flow (from the same origin) on the inlet flow scale. For example, in Figure 2.8 streamline E enters the junction approximately 6.5 units from the origin of the inlet scale and leaves the junction 6.5 units from the origin of the exit scale. The position of a streamline in a particular fracture and the flow between that streamline and the edge of that fracture are related by the properties of the Poiseuille velocity profile. The local velocity at any point in this profile was given in the Longitudinal Velocity Distribution section.

The profile is continuous over the succession of streams comprising the combined flow. With h as the fractional distance from the edge of a channel, the flow between a streamline at fractional position h_1 and the edge of the fracture is found as

$$Q_1 = \int_0^{\eta_1} v b d\eta \quad . \quad (2.47)$$

Substituting for v from Equation (2.38) and integrating:

$$Q_1 = 6\bar{v}b \int_0^{\eta_1} (\eta - \eta^2) d\eta = 6\bar{v}b \left[\frac{\eta_1^2}{2} - \frac{\eta_1^3}{3} \right] \quad . \quad (2.48)$$

After choosing the correct root the cubic equation above is solved for η_1 as:

$$\eta_1 = \cos \left[\frac{1}{3} \cos^{-1} \left[1 - \frac{2Q_1}{\bar{v}b} \right] + \frac{4\pi}{3} \right] + \frac{1}{2} \quad . \quad (2.49)$$

These relationships are used to trace streamlines through the fracture junction in a two-step procedure. In the first step, the steam routing described in the previous section is determined and the cumulative flow rates at the edges of each of the inlet and exit fractures are determined as illustrated in the center sketch in Figure 2.8. These flow rates define the transfer structure which is maintained until the system flow rates are changed.

In the second step streamlines are determined for each of the individual particles. A cumulative inlet flow rate, Q_{ABS} , is determined for an entering particle as the sum of its flow increment from the fracture edge (Equation (2.48)) plus the cumulative flow at that edge. For streamline E in Figure 2.8, this is equal to $0.5 + 6.0$. Moving to the exit side, the streamline is located in the appropriate exit fracture (1) and the flow from the edge of that fracture found ($3.5 = 6.5 - 3.0$). Equation (2.49) is then used to determine the position in that fracture. Lateral motion due to diffusion, for certain junctions, is determined by a preliminary step described in the following section.

2.2.7.4 Molecular Diffusion and Mixing. Diffusive movement results in lateral transfer of tracer between streamlines and some mixing between streams in a fracture junction. The transfer is implemented by modifying the position of the solute particle on the inlet flow scale before moving it across the junction. This process is not applied to junctions with only 2 branches.

The limiting case of complete homogenization of tracer will only occur at very low flow rates and is provided for comparison of results to other transport codes currently available which use the complete mixing approach. This option is implemented by randomizing the position of the particle on the cumulative inflow scale. The particle can then exit the junction by any exit fracture, with a probability determined by the relative flow through the exit fractures. This process is also not applied to junctions of only 2 branches.

Diffusive motion is based on a residence time in the fracture junction. Residence time is approximated as the distance a particle travels across the junction divided by the velocity of the particle as it enters the junction. Distance is assumed to be linearly related to the position of the particle on the cumulative inlet scale ranging from 0 at the two ends of the cumulative inlet scale to the junction diagonal at the center. Referring to Figure 2.8, particles in streamlines A and F travel very short distances, while particles in streamlines C and D travel much further. This configuration is strictly valid only for a junction of four branches where the two inlet and two exit fractures all have equal flows.

Velocity is based on the average of the particle's inlet velocity and exit velocity as calculated from Equation (2.46). The transverse movement of a particle (Δy) due to diffusion is calculated from:

$$\Delta y = (2D_m x/v)^{1/2} ANORM \quad . \quad (2.50)$$

This Δy must be transformed from a linear distance scale to a flow scale before modifying the inlet position of a particle. This is done using the derivative of Equation (2.48).

$$\Delta Q = 6\bar{v} [y/b - (y/b)^2] \Delta y \quad . \quad (2.51)$$

The ΔQ calculated from Equation (2.51) is used to modify the particle position on the inlet flow scale. Particles displaced beyond the range of the cumulative inlet scale are reflected back into the range by the amount they fell outside the range. The outlet position is then calculated by the junction transfer function in the usual manner.

2.2.8 Fracture-to-Matrix Transfer Model

A particle moving down a fracture which strikes the wall of the fracture has a chance of transferring to the matrix adjoining the fracture. Whether the particle does transfer into the matrix depends on why it reached the fracture wall and if it struck a pore or rock. A particle can reach the wall by advection or diffusion or both. If the particle advects to the fracture wall then it is transferred to the matrix. If the particle strikes the wall due to diffusion then the probability of it transferring to the matrix is proportional to the matrix porosity. Usually both advection and diffusion contribute to the particles movement. In this case the particle is assigned a condition of advection or diffusion with probabilities given by the relative contribution of each to the net transverse movement.

When the diffusive movement of a particle in a fracture is large relative to the fracture aperture, then the particle will strike the fracture wall many times during transit through the fracture. This requires breaking the time step down into many smaller steps each time the particle strikes a fracture wall, resulting in inefficient execution. A special Monte Carlo routine was written to deal with solute transport in fractures under hydrodynamic conditions where diffusive movement is significant. Significant is defined to be when the fracture Peclet number [Hull et al., 1987] is greater than 200.

$$Pe = \frac{210 D_m L}{vb^2} \quad (2.52)$$

where

D_m	=	molecular diffusion coefficient	$(\frac{L^2}{T})$
L	=	length	(L)
v	=	velocity	$(\frac{L}{T})$
b	=	aperture	(L)

At large fracture Peclet numbers (low flow rates), transverse diffusion in the fracture will cause a particle to sample many locations in the velocity profile. As a first approximation, therefore, a particle travels at the mean velocity of the fluid in the fracture. The effects of dispersion in the fracture are then added later. Concentration equations for the fracture and matrix can be written [Carslaw and Jaeger, 1959; Neretnieks, 1980]:

$$\frac{\partial C_f}{\partial t} = - \frac{\partial}{\partial x} v C_f + \delta_{m-f} \quad (2.53a)$$

$$\frac{\partial C_m}{\partial t} = \frac{\partial}{\partial y} D_m \frac{\partial C_m}{\partial y} \quad (2.53b)$$

$$\delta_{m-f} = \left. \frac{D_m \theta}{b} \frac{\partial C_m}{\partial y} \right|_{y=0} \quad (2.53c)$$

where

$$\theta = \text{porosity} \quad (-)$$

Equation (2.53a) describes the longitudinal concentration in the fracture, which is uniform across the fracture. Equation (2.53b) describes the concentration in the matrix with a boundary condition that $C_m = C_f$ at $y = 0$. Equation (2.53c) describes the diffusion of tracer across the boundary between the fracture and the matrix. Solutions to

Equations (2.53a) through (2.53c) for a unit pulse at time 0 and a fracture of length L at time t are:

$$C_f = \frac{\sqrt{D_m} \theta x}{4 \sqrt{\pi} v^2 b^2 a (t - x/v)^{3/2}} \exp - \left[\frac{D_m \theta^2 x^2}{4 v^2 b^2 (t - x/v)} \right] \quad (2.54a)$$

$$C_m = \left[\frac{\sqrt{D_m} \theta x}{4 \sqrt{\pi} v^2 b^2 a} + \frac{y}{4 \sqrt{\pi} v b a \sqrt{D_m}} \right] (t - x/v)^{-3/2} \exp - \left[\left(\frac{\sqrt{D_m} \theta x}{2 v b} + \frac{y}{\sqrt{D_m}} \right)^2 \frac{1}{4 (t - x/v)} \right] \quad (2.54b)$$

$$C_{ex} = C_f \Big|_{x=L} = \frac{\sqrt{D_m} \theta L}{4 \sqrt{\pi} v^2 b^2 a (t - L/v)^{3/2}} \exp - \left[\frac{D_m \theta^2 L^2}{4 v^2 b^2 (t - L/v)} \right] \quad (2.54c)$$

where C_{ex} is the concentration at the exit of the fracture

Equations (2.54a) through (2.54c) represent the concentration for a unit input pulse of tracer and so also describe the probability distribution of tracer at time t. Integrating C_f and C_m in space and C_m from 0 to t yields the probability of whether a tracer particle will be in the fracture, the matrix, or have exited the fracture during the time step. These probabilities are:

$$P_f = \exp \left[\frac{D_m \theta^2}{b^2} t \right] \left[\operatorname{erfc} \left[\frac{\sqrt{D_m} \theta}{b} t^{1/2} \right] - \operatorname{erfc} \left[\frac{\sqrt{D_m} \theta}{b} \frac{(t - x/v)}{(t - x/v)^{1/2}} \right] \right] \quad (2.55a)$$

$$P_m = 1 - \exp \left[\frac{D_m \theta^2}{b^2} t \right] \left[\operatorname{erfc} \left[\frac{\sqrt{D_m} \theta}{b} t^{1/2} \right] - \operatorname{erfc} \left[\frac{\sqrt{D_m} \theta}{b} \frac{(t - x/2v)}{(t - x/v)^{1/2}} \right] - \operatorname{erfc} \left[\frac{\sqrt{D_m} \theta x}{2vb} \frac{1}{(t - x/v)^{1/2}} \right] \right] \quad (2.55b)$$

$$P_{ex} = \operatorname{erfc} \left[\frac{\sqrt{D_m} \theta x}{2vb} (t - x/v)^{-1/2} \right] \quad (2.55c)$$

The sum of P_f , P_m , and P_{ex} equals 1.

These probability distributions are used in conjunction with a Monte Carlo sampling routine to determine the final position of the particle. Based on the value of a uniformly distributed random number (N) between 0 and 1, the particle is identified as being in one of the three probability distributions (fracture, matrix, exited). If the particle has left the fracture, ($N < P_{ex}$) Equation (2.55c) is solved for the time of exit with N substituted for P_{ex} . If the particle remains in the fracture, ($N - P_{ex} < P_f$) Equation (2.55a) is solved for the position, x , within the fracture with $N - P_{ex}$ substituted for P_f . If the particle has transferred to the matrix, ($N - P_{ex} > P_f$) the longitudinal position of exit along the fracture, x , is determined, and then the depth of penetration into the matrix, y , calculated. Equation (2.55b) is solved iteratively for x , with $N - P_{ex} - P_f$ substituted for P_m . With x defined, the cumulative probability of C_m from 0 to y is:

$$P_{mix} = 1 - \exp - \left[\left(\frac{\theta^2}{v^2 b^2} xy + \frac{y^2}{D_m} \right) \frac{1}{4 (t - x/v)} \right] \quad (2.56)$$

A new uniformly distributed random number is substituted for C_{mix} and Equation (2.56) solved for y .

The Monte Carlo routine solves for the position of the marker particle at the end of the time step. The only limits on this time step are for particles diffusing into the matrix. The time step must be limited so that the distance the particle enters the matrix, y , has a low probability of being more than one-half the width of the matrix cell. This time limit is calculated by:

$$t_{lim} = \frac{-y_{lim}^2}{4D_m} \ln(0.01) \quad (2.57)$$

The Monte Carlo routine assumes small transverse advective flow in the fracture and no longitudinal dispersion in the fracture. Transverse advection is treated prior to entering the Monte Carlo routine so particles will advect out of the fracture in preference to diffusing out of the fracture. The Monte Carlo routine is an addition which post dates Hull et al., (1987). Longitudinal dispersion within the fracture is calculated using the Peclet number to determine the dispersion coefficient [Hull et al., 1987]. The dispersive movement is used to adjust the exit time for particles exiting the end of the fracture from the Monte Carlo routine.

2.2.9 Matrix-to-Fracture Transfer Model

If the movement of a particle intersects a fracture, then the particle is transferred into the fracture at that location. The particle is placed near the appropriate edge of the fracture. If the particle in the matrix is adjacent to a fracture, then a special diffusion dominated movement routine is used. A particle is considered to be adjacent to a fracture if its expected diffusive distance during the time step is at least half of the distance to the fracture and the effective diffusive velocity during the step is greater than the advective velocity.

If the particle is adjacent to a fracture, then advection is ignored. The fracture is treated as an absorbing surface. In other words, any movement that crosses the fracture removes the particle from the matrix.

This situation could be solved by taking many small random-walk steps which is inefficient. A similar problem has been solved by Chandrasekhar (1943). The solution has been implemented in FRAC-UNIX in a single step calculation. The calculation determines if and when a particle enters the fracture or its final position in the matrix. The calculation is based on a Monte Carlo sampling technique.

If the particle enters the fracture the time of entry is given by:

$$t = x^2 / 2 D_m [\text{erfc}^{-1}(R)]^2 \quad (2.58)$$

where

t = time of entry (T)

x = closest distance to fracture (L)

D_m = diffusion coefficient in matrix $(\frac{L^2}{T})$

erfc^{-1} = inverse of the complimentary error function

R = Monte Carlo sampling variable

in FRAC-UNIX this becomes

$$DTUSED = DL2 ** 2 / (2 * DIFCO * GEOFAC * CERFINV ** 2) \quad (2.59)$$

where

CERFINV is provided by IMSL routine MERFCI, the inverse error function solver.

If the particle remains in the matrix its position is defined by

$$1-R = 1/2 \left[\text{erf}[(y-x)/(2 D_m t_f)^{1/2}] - \text{erf}[(y+x)/(2 D_m t_f)^{1/2}] \right] \quad (2.60)$$

where

- R = Monte Carlo sampling variable
- y = final distance of the particle from the fracture (L)
- x = initial distance of the particle from the fracture (L)
- t_f = length of the movement time (T)
- D_m = matrix diffusion coefficient $(\frac{L^2}{T})$
- erf = error function

The first error function term represents the cumulative distribution from negative infinity of final positions of a source at position x. The second error function term represents a negative source at -x. For positive value of y these terms duplicate the existence of the fracture at y = 0. In FRAC-UNIX the equation is solved using Newton-Raphson iteration on y.

2.2.10 Particle Management

Particles are identified by their position index (IP) in the particle data arrays. A particle location is defined by the index of its host fracture or matrix element (PPHOST) and the position of the particle within its host (PPXU, PPYU, PPZU). PPHOST contains the fracture index or negative of the element index of the host feature. PPXU, PPYU, and PPZU are all normalized by the dimension of the feature. These, therefore, vary from 0 to 1. These arrays, along with the number of particles in the group (PPZNP), the concentration (energy per unit volume for heat transport) of the injected fluid (PPZCON) and the amount of solute or energy represented by each particle (PPZPM) are set up to be moved out to a mass storage device as a group. This would greatly reduce the required storage for a simulation which requires a great number of particles. The swapping is handled by subroutine EXCHANG which is not currently functional.

2.2.10.1 Particle Injection. Particles are introduced into the model through an injection node NODIN or at a horizontal or vertical line source parallel to a total of NNLS nodes given in the INLS array. The line source is offset to the right or upwards from the INLS nodes by a fraction OLSU of the distance to the next upwards row or rightwards column.

The particles are distributed from the source according to a flow specified for the injection node or calculated for the line source. Constant temperature or constant concentration boundaries cannot be specified, but must be approximated by specifying the appropriate flux.

The injection of solute (or energy) is described by the concentration of solute (or energy per unit volume) in the injected fluid and the volumetric flow rate. The injection rate at a single node can vary with time by using the QEXTTR option. The net volume of fluid introduced into the node during the injection period, TSREL to TEREL, (TINIT to TFINAL) is determined. The solute (energy) represented by an individual particle is then fixed by:

$$PPZPM = PPZCON * Q * (TEREL - TSREL) / ([NPMX - 1] * [NPGMX]) \text{ solute (2.61)}$$

$$PPZPM = PPZCON * Q * (TINIT - TFINAL) / ([NPMX - 1] * [NPGMX]) \text{ heat}$$

where

Q = volumetric flow rate at 100°C into injection node (L³) or through the line source. If the injection T node flow is varied with time using QEXTTR, the user must input QQINJ, the flow rate to be used in the PPZPM calculation.

NPMX-1 = number of particles per group, less one reserved for mass calculations.

NPGMX = number of particle groups.

PPZCON = solute concentration (energy per unit volume) at injection well.

Particles are released randomly over the injection interval TSREL to TEREL for solute or TINIT to TFINAL for heat. This is accomplished by determining the amount of solute released during a particle movement time step. The mass of tracer being injected during DTSTP is calculated from:

$$PL = PPZCON * Q * DTSTP/NPGMX + OLDPL \quad (2.62)$$

DTSTP is a movement and reporting interval which is either the increment DTP input by the user or the smaller increment to an event: the start of release, the end of release or the end of the particle movement time.

OLDPL is a variable that carries over any unreleased mass from one time step to the next. RELEASE is a loop in which PL is decremented by PPZPM and a particle is added to the system. The loop terminates when:

$$(URN() .GT.PL/PPZPM)$$

and OLDPL is set equal to the remaining PL

Thus, OLDPL will sometimes be positive and sometimes negative. The particle is placed at a random position in a fracture or matrix element according to the flow distribution. RELEASE then calls MARKER to move the particle. The release time is passed to MARKER from RELEASE, and is a random time between 0.0 and DTSTP. This gives a continuous, random release of particles over the entire interval.

Note that a change in Q from the value used to establish PPZPM will cause a proportionate change in the rate of particle release but the desired concentration will be maintained. A reduction in flow will cause fewer than NPMX-1 particles to be released. An increase in flow will require more than NPMX-1 particles. This condition can be accommodated by the advanced particle management scheme discussed subsequently.

2.2.10.2 Particle Monitoring. Particle progress through the system may be monitored using the MVTST or IPRINT flags. The MVTST flag reports

code development output for all particles. Excessive output may be avoided by setting MVTST true for specific particle numbers at the beginning of subroutine MARKER, as shown by the commented statement in that location. The IPRINT flag reports release, model exit and transfer information for a specified particle.

In a heat transport computation, the temperature of a matrix element is found from its average energy (relative to native temperature TPN) over the DTSTP particle movement interval. This is found by assigning each particle a time at beginning of residence, TINP. This is either the time at the beginning of the DTSTP interval or the time of entry into a new host feature. A contribution to the integral with time in the current interval of the element energy DEM is computed at the time of exiting the element or at the end of the DTSTP interval. The TINP value for the particle is updated at those times. The element temperature at the end of the DTSTP interval is computed as follows:

$$TPM = 2 * (TPN + \frac{DEM}{CPI * DTSTP}) - TPMPV \quad (2.63)$$

where

CPI	=	$[MTPOR * RHO * CPW + MTRHO * MTCP * (1-MTPOR)] * VOL$	
MTPOR	=	matrix porosity	(-)
MTRHO	=	rock density	(M/L ³)
MTCP	=	rock specific heat	(L ² /T ² θ)
RHO	=	water density	(M/L ³)
CPW	=	water specific heat	(L ² /T ² θ)
VOL	=	element volume	(L ³)
TPMPV	=	element temperature at beginning of interval.	(θ)

At each flow computation point within the DTSTP interval, the matrix temperature TPMNW is found by linearly interpolating between TPMPV and TPM.

A different scheme was selected for determining fracture temperatures because of the difficulty of maintaining a stable flow computation in the presence of rapid changes in temperature as particles pass through the fractures. A fracture energy integral FRE is computed in the same manner as for the matrix elements. This integral is accumulated over NLAG particle movement intervals DTSTP. After NLAG particle movement intervals have been accumulated, a linear fraction corresponding to the new DTSTP interval is discarded to make room for the energy accumulated in the new interval. The end of increment temperature is then computed as follows:

$$TPF = TPN + \frac{FRE}{CPI * TLAG} \quad (2.64)$$

where

$$CPI = RHO * CPW * FRL * B$$

$$FRL = \text{fracture length} \quad (L)$$

$$B = \text{fracture aperture} \quad (L)$$

$$TLAG = \text{total time in NLAG intervals} \quad (T)$$

Fracture temperatures at intermediate times are again computed by linear interpolation over the DTSTP interval.

While this calculation does introduce a lag behind the true temperature, the effect can be minimized by judicious selection of particle move interval DTP and the number of intervals NLAG.

A location plot of all particles in the system at the end of this move time is generated by setting PLOCPT true. Concentration is computed at nodes given in the IOUT array by computing the total solute masses leaving

these nodes in the current time interval, MPWEL, and the corresponding concentration, CONWEL. For a heat transport calculation, MPWEL is the total energy (above native temperature TPN) leaving the node in the current time interval and CONWEL is the temperature of the fluid leaving the node. In addition, the mean and standard deviation of distance from the origin is computed for a model composed of porous matrix only.

2.2.10.3 Particle Exit. Particles may exit the model under a number of user-specified conditions. If a particle reaches the end of a fracture with an outflow from the model (negative QQEXT), the particle leaves the model. If a particle moves in the matrix to within a distance RWEL of a node, which is both identified as an outflow node in the NODOT array and has an outflow, the particle is removed. Finally, if a matrix particle crosses a boundary, it is: (1) transferred to a fracture on that boundary or, if no fracture exists, it (2) leaves a boundary with an outflow, or (3) diffuses across a boundary with zero flow if EXTBND is true.

A quick summary of particle exit characteristics is computed for the NOUT nodes listed in array IOUT. This feature is intended to monitor all exit nodes, both at a boundary and a production well. In addition to specifying NOUT and IOUT, the user must dimension the appropriate arrays NOUT (see Table A-6). The code computes the total number of particles leaving the system at those nodes, NPOUT. In addition, a rough accounting is made of those particles leaving the node between the first (not necessarily the earliest) computed exit time for that node, TSMALL, and 1.5 TSMALL. The total number leaving the node in that interval, NPOSML, and the average exit time for those particles, TRESAV, are computed.

A residence time histogram is computed for each of the NDOT nodes (typically the production wells) listed in array NODOT. In addition to specifying NDOT and NODOT, the user must dimension the appropriate arrays NDOT (see Table A-6).

A 100-interval histogram is established between user-specified TRESMN and TRESMX for each node NODOT. Satisfactory results have been obtained for fractured-porous media, with a small number of particles held up in

the matrix, by selecting TRESMX such that the bulk of the particles fall in the first 25 intervals. The resulting output are the 100 TRES, NTINT point coordinate pairs for each of the NDOT exit nodes.

2.2.10.4 Advanced Particle Management. A pulse release of particles with a large subsequent movement interval and all particles leaving at a common exit will be well managed by the previously-discussed approach. However, multiple paths or a distributed release, such as is necessary for a transient heat injection, will lead to many of the allocated number of particles leaving the system before the end of the run. More efficient usage of the 4 arrays dimensioned MNPMX can be achieved in these situations by setting CMPRES true and setting the MNPMX array dimension smaller than NPMX. Each location emptied by a leaving particle is filled by the last (the PPZNPth) particle. An overflow beyond MNPMX-1, preferably by a small number of particles, is prevented by replacing a particle chosen, at random, with particle number MNPMX.

The current inventory of solute or energy in the model is maintained by increasing the amount, PPZPM, in each particle corresponding to a redistribution of the amount in the deleted particle. In other words, MNPMX-1 particles are now carrying the same total solute as MNPMX did before the deletion. The PPZPM increase occurs at the end of the particle move interval DTSTP. Since a given particle number may represent, over time, more than one particle, the CMPRES variable is reset to false if MVTST is set true so that meaningful particle tracking can be provided.

2.2.11 Electrodes

In addition to values in an exit stream, concentrations, CELC, may be calculated at interior volumes. These may be used for comparison to electrode data for a physical model or to provide information on solute distribution within any model. A total of NELC "electrodes" may be placed in fractures or in matrix cells and should be located at least a distance, RELC, from a node or cell boundary. Particles within a distance, RELC, from the electrode are counted, divided by the volume of water in the zone

of influence specified by RELC, and normalized by the injection concentration. Each electrode is specified by supplying the host index, PELHST, which is either the fracture number or the negative of the cell number. The exact location within the host is given by PELXYU (2,NELC). For a matrix cell, the values required are the fractional x position from the left edge of the cell and the fractional y position from the bottom of the cell. For a fracture location, the fractional distance from NOD1 (smaller node index) to NOD2 is required. Data output is made at TELCNX intervals. Since subroutine ELCTROD is called after RELEASE and MARKER, which are called at DTP intervals, TELCNX should be an integral number of DTP intervals.

Figure 2.1 Sample FRAC-UNIX model.

Figure 2.2 Basic concepts of "random-walk" computer program.

Figure 2.3 Progress of a slug around the mean flow.

Figure 2.4 Temperature dependence of tracer diffusion coefficients (boundaries represent extremes of ions other than H^+ , T^+ , OH^-).

- ⊙ Particle at location y, x
- Node location
- Location of intermediate head

Figure 2.5 Marker particle and matrix cell schematic diagram.

FIGURE 2.6 Fractures Affecting Particle Movement in the Matrix

7-1612

Figure 2.7 Mass balance for transverse velocity.

Figure 2.8 Simple four-branch fracture junction.

Flow schematic

Streamline map

Transfer function

- Streamline
- Location of zero lateral position in fracture (left edge of diagonal or vertical fractures, bottom edge of horizontal fractures)
- ⬇ Location of maximum lateral position

7-10166

Figure 2.9 Four-branch discontinuous fracture junction.

3. CODE DESCRIPTION, USE, AND MODIFICATION

This section, together with the appendices, provides a description of the code structure, detailed information on using the model and background material useful in modifying the code.

User information includes a primer on UNIX operations (Section 3.2 and Appendix F) and material (Sections 3.2 and 3.3 and Appendices A, B, C, and E) at three different levels on code input, output and operation. Information useful in modifying the code is given in Section 3.5 and Appendices A and D.

3.1 Code Structure

The code consists of Fortran flow and transport models, sparse matrix techniques (Marwil) for transient and direct steady state calculations and a user interface (Albano, 1987) programmed in the C language. The user interface lexical analyzer uses the Lex preprocessor (Lesk, 1975) and the command line interpreter uses the Yacc preprocessor (Johnson, 1978). The code operates on CRAY computers using the UNICOS 4.0 operating system and the IMSL 10.0 mathematical library.

Figure 3.1 shows the directory and file structure established for a multi-developer, multi-user environment. The reference configuration is maintained using the UNICOS Source Code Control System (SCCS). The reference configuration and certain executable commands are maintained by the administrator, designated jdm in Figure 3.1. The administrator's directories and files are shown to the left and below the dotted line on the figure. Code modification under this system is discussed in Section 3.5.

A user, designated jdm in Figure 3.1, creates directories and files shown to the right of the dotted line in this figure. The user establishes (see Section 3.2) administrative procedures in the •login, •logout, and •cshrc files in the home directory jdm/. Object coding for the numerical routines is established by creating the gears directory and making the

statement `get__gears`. This creates the `Gears.revs` file (revision summary) and all of the source files for the numerical routines. The `make` statement creates the corresponding object files.

The flow and transport files are then established by creating the code directory and making the `get__fru` and `make` statements. The linked object code and the executable `fru` are also created by the `make` statement.

The information pertinent to a particular problem is created within the `manf` directory by inputting or copying the `GEARS`, `ZERO`, `PRINTM` and `frac.inpt` files. The statement beginning with `fru` executes the problem, creating the output files listed below the statement.

A second user may use the code and gears directories belonging to `jdm` and would only need a unique working directory for his particular problem.

3.2 UNIX Operations

Appendix F summarizes the large majority of the UNIX commands required to operate and modify the FRAC-UNIX Code. Material is included on logging in, directory and file manipulation, miscellaneous monitoring and control operations, file creation and editing using the `vi` editor, modifying compiler options and debugging after a system fault.

Additional material on code modification is discussed in Section 3.5.

3.3 Model Control

This section describes all of the FRAC-UNIX input and output except that given in Section 3.4 for the steady state and transient numerical routines.

Guidance is provided in this section at three different levels. At the simplest level a user may modify the input listing given in Appendix E for one of three different problem sets.

At the second level the user may refer to a comprehensive I/O summary in Appendix B to create a model from scratch. Five different types of runs are described: (1) steady state flow solution at a specified fixed temperature, (2) transient flow solution at a specified fixed temperature, (3) solute transport in a fixed isothermal flow field, (4) solute transport in a transient, isothermal flow field, and (5) heat transport in a transient flow field. In addition the "deriv" command generates the head derivative vector at the current head distribution and the "continue" command advances a transient from a previous integration when the end time, TEND, has been advanced. Cases of these various types may be stacked in a single run with the provision that the head distribution for a Type (3) case must be specified by input or a previous flow calculation.

At the third level of description the user may refer to Appendix A for a detailed description of a particular variable. Background discussions are given in Section 2 of this report.

It should be noted that the user must always check that array dimensions are large enough for the specified model, yet not so large that excessive core space is used. Table A-6 lists the current parameter values used for array dimensioning. Note also that bounds checking (see Appendix F--Compiler Options) may be used with a substantial increase in run time.

Modification of the dimensioning parameters and the addition of a new "debug" output variable to the namtab file will require recompiling the program using the make command (see Section 3.1).

3.4 Numerical Routine Control

This section, together with Appendix C, describes the approach and the usage of the numerical routines for solving for a steady state solution and for calculating a flow transient.

3.4.1 Steady-State Solution

A general purpose library routine developed by Marwil is used to determine the steady-state flow solution. The driver subroutine, dsgpdr.F, and the supporting subroutines are included in the gears directory and are described in Appendix D. Inputs and outputs are summarized in Appendix C.

The routine finds a zero of a system of nonlinear functions with a sparse Jacobian. The functions, calculated by subroutine zfcn.F, are the time derivatives of the heads at the grid nodes. The basic solution method, as developed by Powell (1970), combines a steepest-descent approach for initial values with the Newton method as the final solution is approached. The implementation of J. J. More' et al. (1980), has been modified to use a sparse matrix solver, rather than full matrix routines.

Iteration is performed until the rms value of the changes in individual head derivatives from the previous values, divided by the rms value of the current derivatives, is less than the user supplied XTOL = ZRPARM(1). The solution is also affected by the following parameters.

EPSFCN = ZRPARM(2) is used in determining the step from the current head to the next trial value. For a nonzero H, the step-size is $\text{SQRT}(\text{EPSFCN}) * H$. For zero H, the step is $\text{SQRT}(\text{EPSFCN})$. A value of 1.0 is recommended. Smaller values should be used if convergence results at a local minima.

FACTOR = ZRPARM(3) determines the initial step bound to control the actual step taken in the steepest descent or Newton directions or combinations thereof. Recommended values are in the range of .1 to 100. A large value aggressively seeks a quick solution but could lead to an undesirable solution when multiple solutions are possible.

THRESH = ZRPARM (4) is the non-negative threshold value used for determining sparseness in the Jacobian elements. At times,

it is advisable to use a THRESH other than the default value of 0. Small entries in the Jacobian tend to create large entries in the inverted Jacobian. These large entries cause stability problems which adversely affect the convergence of the derivatives to steady-state solutions.

The final output includes the estimate of the heads and head derivatives. Additional output given at each iteration is the index of the node with the greatest absolute derivative, relative to nonzero heads, and the index of the node with the greatest absolute derivative. Both these outputs are only given for nodes that do not have a prescribed head. Additionally, upon convergence to steady-state, a table is output of the most active nodes in both categories.

Other output and error conditions are summarized in Appendix C.

3.4.2 Integration

A flow transient is computed by the Gears stiff-system technique which is ideally suited to the wide range of eigenvalues found in a fractured-porous media flow system. Sparse matrix techniques are used to reduce core requirements. The computation uses a general-purpose library routine developed by Marwil. The driver routine dsgpdr.F and the supporting subroutines are included in the gears directory and described in Appendix D. Inputs and outputs are summarized in Appendix C. The routine performs one step of the integration of an initial value problem for a system of first-order, ordinary differential equations defined, for the flow problem, by subroutine fcn. The solution has been developed from the Hindmarsh (1974) approach.

Following a step of size h at order q , the GEAR package, at intervals of $q+2$ steps, attempts to choose a larger step size by estimating the local truncation errors at orders $q-1$, q , and $q+1$. The largest value h' of the three step sizes obtained is then chosen, and the order reset accordingly.

Also, the Nordsieck array must be rescaled by powers of h'/h . The data used to take the subsequent steps of size h' is, in effect, obtained by interpolating with the data at a spacing of h .

User-specified timing is processed in subroutine `doderiv`. The initial head distribution, `HZ`, must be specified, along with the following controls on the integration process:

`RTOL` maximum relative error in an elementary integration interval, default = `1.E-5`

`ATOL` maximum absolute error in an elementary integration interval, default = `1.E-5`

A sufficiently-accurate step is made when:

$$\left[\sum \left(\frac{\Delta H_i}{RTOL * |H_i| + ATOL} \right)^2 \right]^{1/2} \leq 1.0$$

where ΔH_i is the predicted step error in head H_i .

`MOSS` determines the method for calculating sparseness structure. In `GEARS`, it is set to 3, which causes the `FRAC-UNIX` routine `SPARSE` to determine sparseness. For stiff problems, `MOSS=2` was found to perform more satisfactorily. With `MOSS=2`, the sparseness is determined by the first $n+1$ calls to routine `FCN`, which calculates derivatives.

`IMXER` = index of node with maximum absolute weighted error at every tenth time step for the first 100 steps and every fiftieth after that. The purpose is to allow the user to see how the accuracy of the solution behaves with time. Additionally, a cumulative table is output showing the ten nodes which have the greatest occurrence of the maximum-weighted error and the number of times the maximum weighted error occurred at those nodes.

Error conditions are output, as summarized in Appendix C.

3.5 Code Modification

Appendices A and D provide background material on variable usage and subroutine flow, respectively, which is useful in modifying the code. Code modification may be done on a temporary basis, or using the UNICOS SCCS system, by modifying the reference configuration. As shown in Figure 3.1, directories jdm/fru/sccs/cray and jdm/fru/sccs/gears contain the reference versions of the flow and transport models and the numerical routines, respectively. Various utilities are also maintained in the cray directory. In this system any user may modify the reference configuration. Each modification is recorded and a current version of the reference configuration, or any earlier version, may be constructed from the individual modifications. Appendix F provides detailed commands for use in modifying the reference configuration using the SCCS software.

FIGURE 3.1 DIRECTORY AND FILE STRUCTURE

4. SAMPLE PROBLEMS

An extensive code verification and benchmarking effort was initiated to assure that FRAC-UNIX is fully operational and ready for release to the technical community. The approach and problem set for this study has been given by Arnett and Baca (1989) and is summarized in Section 4.2. Included are analytic and fully complex problems in flow, solute transport and heat transport. FRAC-UNIX solutions to these problems are compared to analytic solutions and those from other codes, principally MAGNUM-2D (England et al., 1985) and CHAINT (Kline et al., 1985).

Preliminary results were presented in a technology transfer and technical review at Lawrence Berkeley Laboratory on February 1-2, 1989 along with a discussion and hands-on training session. Final computation and preparation of a summary report have not been completed. Input and output files for several of these problems are given in Appendix E.

Also included in Appendix E are the input and output files for one of the sample problems used earlier in verifying the code; flow and transport in an infinite porous media. The following discussion gives a comparison of FRAC-UNIX and analytic results for this problem.

4.1 Infinite Porous Media

Simulations have been compared to analytic results for a well flowing into an infinite layer of homogeneous porous material. These comparisons were made for steady-state pressure response with isotropic and anisotropic conductivities and solute transport with isotropic conductivity, steady-state flow and with and without dispersion. The FRAC-UNIX model described in Table 4.1 is a square representing one quarter of the reservoir with the well at the origin.

TABLE 4.1. INFINITE POROUS MEDIA MODEL PARAMETERS

Model size:	51.917 ft x 51.917 ft 11 rows x 11 columns expanded by 1.2 from initial x and y intervals of 2.0 ft.
Hydraulic conductivity:	7.9 ft/day or 23.7 in y direction 2.63 in x direction
Storage coefficient:	1×10^{-5}
Porosity:	0.25
Thickness:	1 ft
Flow rate:	115.5 ft ³ /day into quarter section model

4.1.1 Isotropic Steady-State Flow

The first comparison was for a steady-state flow in an isotropic aquifer. The analytic expression for head was given in Equation (2.32).

A FRAC-UNIX simulation was performed using the parameters shown in Table 4.1. Table 4.2 shows the excellent agreement between these results and the analytic solution.

4.1.2 Anisotropic Steady State Flow

Table 4.2 shows a comparison between simulation results and the analytic solution [Equations (2.32) and (2.35) through (2.37)] for a conductivity 9 times larger in the y direction than in the x direction. The two are in close agreement, particularly at locations removed from the origin where the nodding is not so critical.

4.1.3 Solute Transport at Zero Dispersion

Solute transport was simulated for steady-state, isotropic flow conditions. In the first of 2 cases, the dispersivities were set to zero and a brief pulse of solute was injected. The FRAC-UNIX simulation therefore approximates an expanding circle of tracer particles. The theoretical radial position of the particles is given by

$$\bar{r} = \sqrt{\frac{t\theta}{b\pi r} + r_{\text{well}}^2} \quad (4.1)$$

where

t = time since start of injection (T)

θ = porosity (-)

r_{well} = radius of the well (L)

TABLE 4.2. COMPARISON OF FRAC-UNIX AND ANALYTIC STEADY STATE DRAWDOWNS

Isotropic Case			
<u>r</u> <u>ft</u>	<u>Analytic</u> <u>Drawdown</u> <u>ft</u>	<u>FRAC-UNIX</u> <u>Drawdown</u> <u>ft</u>	<u>Error</u> <u>%</u>
2.0	33.536	34.539	2.99
7.28	21.511	21.550	0.18
19.86	12.170	12.156	-0.12
51.917	3.225	3.173	-1.61
73.422	0.	0.195	--

Anisotropic Case					
<u>x</u> <u>ft</u>	<u>y</u> <u>ft</u>	<u>r</u> <u>ft</u>	<u>Analytic</u> <u>Drawdown</u> <u>ft</u>	<u>FRAC-UNIX</u> <u>Drawdown</u> <u>ft</u>	<u>Error</u> <u>ft</u>
2.0	0.0	3.4652	30.819	30.659	0.160
0.0	2.0	1.154	41.057	42.397	-1.340
7.28	0.0	12.613	18.786	18.681	0.105
0.0	7.28	4.202	29.024	29.677	-0.653
19.860	0.0	34.409	9.440	9.379	0.061
0.0	19.860	11.463	19.678	19.774	-0.096
51.917	0.0	89.951	0.490	0.471	0.019
0.0	51.917	29.965	10.728	10.676	0.052
51.917	51.917	94.811	0.000	.071	-0.071

Table 4.3 compares the results of these simulations to analytic results from Equation (4.1).

The uncertainty given in the FRAC-UNIX results is the standard deviation of the particle position about the given radius. The particle deviations are preferential with greater errors along the axes and diagonals of the grid. The small deviation at 0.05 days is due to special handling of the region near the injection well. The delay and recovery of the particles at 0.25 days and 0.55 days, respectively, is due to the coarse nodalization.

4.1.4 Solute Transport With Dispersion

A second transport test was run with dispersion and continuous particle injection. The longitudinal dispersivity was 0.6 ft and the transverse dispersivity was 0.2 ft. Bear (1972) defines the dependence of concentration on radial distance traveled by:

$$C/C_0 = 1/2 \operatorname{erfc} \left(\frac{r - \bar{r}}{4/3 D_L \bar{r}} \right) \quad (4.2)$$

where

C = tracer concentration at radius r (-)

C_0 = tracer injection concentration (-)

erfc = complimentary error function

D_L = coefficient of longitudinal dispersion $\left(\frac{L^2}{T} \right)$

$\bar{r} = \bar{r}$ from Equation 4.1.

Figure 4.1 shows C/C_0 as a function of distance from the well and shows an excellent match to the theoretical results. The results indicate that the

TABLE 4.3. TRANSPORT OF A TRACER SLUG

<u>Time (days)</u>	<u>Analytic Radius (ft)</u>	<u>FRAC-UNIX Radius (ft)</u>	<u>Error (%)</u>
0.0	0.25	.256 $\pm$.003	2.4
0.05	5.429	5.647 $\pm$.043	4.0
0.25	12.129	12.338 $\pm$.032	1.72
0.55	17.989	18.188 $\pm$.034	1.11
1.00	24.255	24.452 $\pm$.037	0.81
2.00	34.301	34.497 $\pm$.044	0.57

deviations about the nominal radius apparent in Table 4.3 are small compared to the dispersive term. In Figure 4.1 the variability of the concentration near the well is a property of dispersion for the limited number of tracer particles used in the simulation.

4.2 Verification and Benchmarking Problems

The following problems (Arnett and Baca, 1989) were used to further evaluate the FRAC-UNIX code:

- FF-1 Transient Flow in a Porous Reservoir with a Planar Fracture--Flow proceeds from a fully penetrating well into a two-dimensional aquifer which is bounded at the top by a confining layer and at the bottom by a fracture and another confining layer.
- FF-2 Transient Response to Pumping of a Confined Aquifer--A two-dimensional porous aquifer is bounded above and below by confining layers and on the one side by a fluid reservoir. Water is pumped from a fully penetrating well.
- FF-3 Steady Flow in a Network of Discrete Fractures--A rectangular region contains a system of orthogonal fractures with impermeable boundaries. Flow is induced by a uniform gradient and a fixed head at one point on the boundary.
- FF-4 Flow in a Porous Medium with Discrete Fractures--A porous medium is overlain with an irregular system of orthogonal fractures. Flow is introduced at one well and withdrawn from two others.
- FF-5 Flow in a Porous Media with Fracture Sets--Constant head boundary conditions drive flow through a more complex fractured porous media.
- MT-1 Transport in a Single Fracture--Solute is driven through an infinite fracture with impermeable walls.

Figure 4.1. Comparison of FRAC-UNIX and analytic dispersion in radial flow.

- MT-2 Transport in a Two-Dimensional Porous Medium--Solute is driven into porous media from a line source at the edge of the media.
- MT-3 Transport in a Porous Medium with a Single Fracture--Solute travels down a fracture and diffuses into the bounding porous media.
- MT-4 Transport in Porous Medium with Discrete Fractures--Solute transport in the FF-4 flow system.
- MT-5 Transport in a Porous Medium with Fracture Sets--Solute transport in the FF-5 flow system.
- HT-1 Transient Heat Conduction in a Semi-Infinite Domain--Heat conduction version of the FF-2 problem.
- HT-2 Energy Transport by Advection and Dispersion in a Nondirectional Flow Field--Energy transport version of MT-2.
- HT-3 Heat Sweep in a Geothermal Reservoir--Coupled flow and heat transport are calculated between two wells in a fractured-porous media.

5. CODE DEVELOPMENT AND APPLICATION HISTORY

5.1 Development Summary

FRAC-UNIX has been under continuous development since 1983 as part of a Department of Energy, Division of Geothermal Technology, study on the injection of spent geothermal fluids. Injection is of concern to geothermal developers because of the need to dispose of large volumes of fluid without damaging the environment, to offset reservoir depletion and maintain reservoir pressures, and to avoid thermal breakthrough to production wells and undesirable chemical interactions.

The research study at INEL incorporates field testing and laboratory studies on physical models in addition to numerical simulation.

Field studies have been conducted at the Raft River, Idaho and East Mesa, California field test sites in cooperation with the University of Utah Research Institute and Republic Geothermal Inc.

FRAC-UNIX, and its predecessor FRACSL, were developed to model fractured geothermal reservoirs. This requirement dictated the use of a discrete representation of the fractures along with the conventional distributed representation of the rock itself as a homogeneous porous media. The marker particle transport model was selected because it was well-suited to the dual-permeability structure and because it avoided the numerical dispersion inherent in classical solutions of the advection-dispersion equation.

Initial development (Miller, 1983) was directed at developing a suite of models for marker motion and was conducted in conjunction with a Navier-Stokes analysis of flow in a fracture junction using the SALE fluid-dynamics code. (Amsden et al., 1980.) Head distribution and breakthrough curves were given for a sample fractured reservoir.

Subsequent development has provided additional capabilities to meet specific requirements in correlating field and lab test results and has refined and added geometric detail to the individual mass balance and particle models. Substantial effort has been expended in developing higher level Monte Carlo models to reduce the execution time burden of very small particle movements at the fracture-matrix interface.

5.2 Raft River Field Tests

Raft River well RRGP-5BF was tested in September through November 1982 as the initial exploration of the use of single well injection-backflow tests to characterize the reservoir near a well. The well was completed in fractured metasedimentary quartzite and schist at a depth of 4911 ft. The completion included hydraulic stimulation. Testing included flow and acoustic televiewer borehole logging and drawdown and injection-backflow tracer tests. (Large, 1983)

Two FRACSL models were developed (Miller et al., 1984). The smaller model (Figures 5.1 and 5.2) is a vertical section centered on the hydrofracture shown on the acoustic televiewer log and extending in an approximate E-W direction. The model is 189 ft high by 77 ft wide and includes the production interval and the major aquifer observed in leg A of the well. Figures 5.1 and 5.2 show the conceptual model and the computer approximation, respectively. Simulation was conducted in an attempt to determine the range of effective apertures in the fracture system. Each fracture was assigned the observed location and orientation and assumed to extend to the boundaries of the model. The apertures of the hydrofracture and the upper aquifer fracture were set at .0025 ft and the apertures of the other fractures adjusted to match the flow distribution given by the spinner log. The resulting apertures, moving down from the bottom of the well liner are .00095, .00095, .0018, .0018 and .0025 ft. Steady-state simulations were conducted with a 1000 ft distance to a zero head remote boundary. The resulting injection head was 34.6 ft as compared to a measured value of 49.0 ft. The apertures given above are believed accurate to within an order of magnitude.

Transient simulations were correlated with observed responses over a range of flow rates by a specific storage of $1.E-7$ per foot which is reasonable in view of information in the literature.

A second model incorporated a fracture system synthesized within a 500 ft by 500 ft vertical section using the procedure described in Section 2.1.4.3. While successful steady-state flow and tracer dispersion runs were made, the analysis was not completed.

5.3 East Mesa Field Tests

The FRACSL code was used to correlate the results of flow and tracer tests conducted in the Imperial Valley at East Mesa, California in July-August 1983 and January-February 1984 (Freiburger, 1984). The two wells tested are 4500 and 7500 ft deep and are completed with a slotted liner in the production interval. The wells are located in a stratigraphic sequence of siltstones and mudstones.

Horizontal plane models (Clemo and Miller, 1984) were developed to represent 100 and 139 ft thick production zones at the two wells. The models consisted of matrix cells only, with cell properties which were anisotropic but uniform over the various cells.

Drawdown testing of both wells showed a sharp but inconsistent transition between early and late time which was not satisfactorily correlated by FRACSL simulation. The inconsistency is believed to be due to a data problem.

Dispersion simulations were made using velocity fields established for quasi-static conditions and assuming an impermeable model boundary. Figure 5.3 shows the measured and predicted slug tracer responses for test 56-30-3 in which tracer was injected and recovered at the same well. The excellent agreement shown was accomplished by adjusting longitudinal and transverse dispersivities while maintaining the longitudinal-to-transverse ratio at the 3.0 value common in the literature.

The tracer injected in test 56-30-4 and recovered six months later in test 56-30-5 showed a delayed peak consistent with a natural drift. Results were correlated by varying the ratio of conductivity in a direction transverse to the drift with that along the drift. A theoretical relationship was used between the dispersivities (maintained at a 3/1 ratio) and the ratio of conductivities. Tracer location for a conductivity ratio of 9 is shown in Figure 5.4. The ellipse centered on the origin (the well) is the predicted location at the end of injection. The ellipse at the right shows the drifted position and the distortion due to dispersion. Figure 5.5 shows the comparison of measured and predicted reduced concentration (return/injected) versus reduced volume (return/injected). The results in Figures 5.3-5.5 are for identical anisotropic conductivity and dispersivity. Figures 5.6 and 5.7 show a comparison of predicted and measured concentration for continuous injection (and subsequent recovery) at the second well. Note that in Figures 5.5 and 5.7 the test data is shown as an x and the curves shown are a polynomial fit to the FRACSL output.

5.4 Physical Model Tests

FRACSL predictions have been compared to measurements of head loss and solute transport in the laboratory fracture network shown in Figure 5.8 (Hull et al., 1987). The model consists of 2 orthogonal sets of fractures milled into a 172 cm by 58 cm sheet of Plexiglas. The tops of the fractures were sealed using a thin sheet of Plexiglas, forming closed channels with aperture of 0.3175 cm and height of 1.905 cm.

During flow testing the pressure drop was measured between two piezometers separated by 21 fractures. Figure 5.9 shows the excellent agreement between measured results, the analytic expression for flow in a closed channel incorporated in FRACSL and the FRACSL simulation itself. The agreement supports the conclusion that the head loss in the fracture junctions is negligible.

Tests were conducted to study the effects of diffusion on tracer movement. A slug of tracer was injected into the reservoir through a port on the side of the model while a background flow was passing through the

model. Figures 5.10 through 5.14 show correlations between FRACSL prediction and tracer concentrations measured at 5 different electrodes. The predictions were made without empirical fits, using only the diffusion coefficient of $1.09 \times 10^{-3} \text{ cm}^2/\text{min}$ for KCl (injected tracer) and CsCl_4 (background flow) (Miller, 1982). The results indicate that diffusion between streamlines is a critical factor in dispersion in fracture networks.

5.5 Sensitivity

The effects of using a continuum to represent a fractured matrix were explored by modeling a fracture system plus modifications where matrix permeability was substituted for some of the fractures (Hull and Clemon, 1987). Steady state pressure distributions and breakthrough curves were calculated for each of the models.

Figure 5.15 shows the full fracture system incorporated in a vertical section 90 m high by 1200 m long. Injection and production wells lie on the left and right edges of the model, respectively. Table 5.1 summarizes the 6 models studied.

Figure 5.16 shows similar steady state pressure distributions for models 2 and 5. The results suggest that only the most significant fractures are needed to provide a reasonable estimate of the pressure distribution.

Figure 5.17 shows tracer breakthrough for systems 1, 5 and 6. For model 6 (single fracture plus matrix) the breakthrough is essentially defined by the fracture. For model 1, (matrix only) the entire volume is swept, the arrival is delayed and somewhat distributed. Model 5 (full network plus matrix) provides an intermediate mean arrival time with a wide distribution.

TABLE 5.1. SUMMARY OF DUAL-PERMEABILITY SYSTEMS

<u>System</u>	<u>Description</u>	<u>Number of Fracture Strings</u>	<u>Well-to-Well Connecting Paths^a</u>	<u>Matrix Hydraulic Conductivity (m/day)</u>	<u>Matrix Intrinsic Permeability (darcy)</u>
1	Matrix only	0	0	7.9	1.62
2	No fracture connection between wells	5	0	0.78	0.16
3	One cross connection between wells	10	4	0.42	0.086
4	Several connections between wells	14	12	0.18	0.037
5	Full network	22	21	0.009	0.002
6	Single fracture	1	1	0.009	0.002

a. Possible routes between wells.

5.6 Work Remaining

Further work is necessary on code verification problem ht-3, heat transport in a fractured porous media. Results to date using five particle move intervals (NLAG) in the moving average fracture temperature, a one degree injection-native temperature difference and a 10 day injection period have yielded stable, although slow, computations. The next step consists of studying the effect on the computation of various integration algorithm parameters. Additional particle movement coding is necessary to provide an advective-diffusive source for two of the code verification problems. In addition, simultaneous computation of solute and heat transport, using one group of particles for each, is required to study realistic injection tests. Relatively simple modifications are required for these capabilities.

A system of defaults, tests, writes and stops would provide immediate and specific responses to a failure to input one of the quantitative model properties.

The model for particle transfer through a fracture junction should be modified to (1) better treat a particle which diffuses into an inlet branch (currently removed from system), (2) change the bias for a discontinuous junction to correspond to the laboratory solution (Hull and Koslow, 1986), and (3) improve the diffusion model to better correspond to various junction geometries.

A problem exists in that the angle PSIND1 is used in buoyancy calculations and in calculating concentrations in subroutines CONC and POST as if it were a continuously varying parameter. However it is calculated in increments of $\pi/4$, and is therefore incorrect for diagonal fractures on a rectangular grid.

Figure 5.1. Raft River Well RRG-5BF near wellbore fracture system.

Figure 5.2. Raft River Well RRG-5BF near wellbore computer model.

FIGURE 5.3 Slug tracer recovery for test 56-30-3

FIGURE 5.4 Marker particle positions after test 56-30-4

FIGURE 5-5 Slug tracer recovery from tests 56-30-4 and 5

TEST 56-19-4
CONTINUOUS INJECTION

FIGURE 5.6 Test 56-19-4 continuous tracer recovery

FIGURE 5.7 Test 56-19-6 continuous tracer recovery

FIGURE 5.8 Laboratory physical model used for tracer testing.

FIGURE 5.9 Head loss in the physical model as a function of flow rate.

FIGURE 5.10 Comparison of tracer concentration data from the laboratory physical model with FRACSL simulated concentrations for Electrode 6.

FIGURE 5.11 Comparison of tracer concentration data from the laboratory physical model with FRACSL simulated concentrations for Electrode 11.

FIGURE 5.12 Comparison of tracer concentration data from the laboratory physical model with FRACSL simulated concentrations for Electrode 14.

FIGURE 5.13 Comparison of tracer concentration data from the laboratory physical model with FRACSL simulated concentrations for Electrode 17.

FIGURE 5.14 Comparison of tracer concentration data from the laboratory physical model with FRACSL simulated concentrations for Electrode 18.

7-1619

FIGURE 5.15 Conceptual model of a geothermal reservoir.

7-1613

MODEL 2

7-1614

MODEL 5

FIGURE 5.16 Steady state pressure distributions for dual-permeability fracture network.

FIGURE 5.17 Tracer breakthrough curves for hydraulically equivalent reservoirs.

6. MAINTENANCE AND QUALITY ASSURANCE PROGRAM

The code described in this document, in addition to the disc files and tapes maintained by the developers, is maintained in the EG&G Idaho Archival Storage. The maintained files include the code listing, the input and output results for each of the sample problems, and the dBase III data base describing the nomenclature.

Temporary changes in the code are documented by the developer including derivations, rough backup, logbook of changes and effects, check calculations and a listing of published results. Changes are incorporated in the reference code using the SCCS system as described in Section 3.5 and Appendix F.

7. REFERENCES

- Ahlstrom, S. W., et al., Multi-component Mass Transport Model: Theory and Numerical Implementation. BNWL-2127 Pacific Northwest Laboratories, Richland, WA, 1977.
- Albano, J. S., FRAC-UNIX User Interface External Documentation, EG&G Internal Report, May 1987.
- Amsden, A. A., Ruppel, H. M., Hirt, C. W., SALE: A Simplified ALE Computer Program for Fluid Flow at All Speeds, LA-8095, Los Alamos National Laboratory, 1980.
- Andersson, J., Shapiro, A. M. and Bear, J., A Stochastic Model of a Fractured Rock Conditioned by Measured Information, Water Resources Research, 20, 1, January 1984, pp. 79-88.
- Arnett, R. C. and Baca, R. G., Verification and Benchmark Test Plan for the FRAC-UNIX Computer Code, EG&G Idaho Draft Report, January 1989.
- Bear, J., Dynamics of Fluids in Porous Media, American Elsevier Publishing Co., Inc., 1972.
- Bird, R. B., Stewart, W. E., and Lightfoot, E. N., Transport Phenomena, John Wiley & Sons, New York, 1960.
- Carslaw, H. S., and Jaeger, J. C., Conduction of Heat in Solids, Oxford University Press, 1959.
- Chandrasekhar, S., Stochastic Problems in Physics and Astronomy, Reviews of Modern Physics, 15, No. 1, 1943.
- Clemo, T. M. and Miller, J. D., Simulation of Flow and Dispersion in the East Mesa Reservoir, EG&G Internal Report SE-PB-84-056, 1984.
- Crane Co., Flow of Fluids Through Valves, Fittings and Pipe, Technical Paper No. 410, 1969.
- England, R. L., et al., MAGNUM-2D Computer Code: User's Guide, RHO-BW-CR-43, Rockwell Hanford Operations, 1985.
- Freiburger, R. M., Hydrothermal Injection Program East Mesa 1983-84 Test Data, EG&G Idaho Report SE-PB-84-044, September 1984.
- Fried, J. J., Groundwater Pollution, Elsevier Scientific Publishing Company, Amsterdam, 1975.
- Gale, J. E., Rouleau, A., and Atkinson, L. C., Hydraulic Properties of Fractures, IAH Symposium on Hydrogeology of Rocks of Low-Permeability, Tucson, January 1985, pp. 1-16.

- Happel, J. and Brenner, H., Low Reynolds Number Hydrodynamics With Special Applications to Particulate Media, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1965.
- Hindmarsh, A. C., GEAR: Ordinary Differential Equation System Solver, UCID-3001, Rev. 3, Lawrence Livermore National Laboratory, CA, 1974.
- Hoopes, J. A. and Harleman, D. R. F., Dispersion in Radial Flow from a Recharge Well, Journal of Geophysical Research, 72, No. 14, pp. 3595-3607, July 15, 1967.
- Hull, L. C. and Clemo, T. M., Sensitivity Analysis of Dual-Permeability Systems, Proceedings: Twelfth Workshop on Geothermal Reservoir Engineering, Stanford University, 1987.
- Hull, L. C., and Koslow, K. N., Streamline Routing Through Fracture Junctions, Water Resources Research, 22, 12, pp. 1731-1734, 1986.
- Hull, L. C., Miller, J. D. and Clemo, T. M., Laboratory and Simulation Studies of Solute Transport in Fracture Networks, Water Resources Research, 23, 8, pp. 1505-1513, 1987.
- IMSL, 7500 Bellaire Blvd, Houston, TX 77036.
- Johnson, S. C., Yacc - Yet Another Compiler - Compiler, Bell Laboratories Computing Science Technical Report #32, July 1978.
- Kline, N. W., England, R. L., and Baca, R. G., CHAI NT Computer Code: Users Guide, RHO-BW-CR-144P, Rockwell Hanford Operations, December 1985.
- Large, R. M., Hydrothermal Injection Program Phase 1 Test Data Index, EG&G Idaho Report REPB-83-015, July 1983.
- Lesk, M. E., Lex - A Lexical Analyzer Generator, Bell Laboratories Computing Science Technical Report, 1975.
- Marwil, E. S., MINERVA Mathematical Library, EG&G Idaho, Inc.
- Miller, D. G., Estimation of Tracer Diffusion Coefficients of Ions in Aqueous Solution, UCRL-53319, 1982.
- Miller, J. D., A Fundamental Approach to the Simulation of Flow and Dispersion in Fractured Media, Proceedings: Ninth Workshop on Geothermal Reservoir Engineering, Stanford University, SGP-TR-74, Stanford, CA, pp. 373-380, 1983.
- Miller, J. D., Allman, D. W. and Wulf, D., Raft River Fracture Characterization and Reservoir Simulation, Idaho National Engineering Laboratory SE-PB-84-058, 1984.
- Miller, J. D., Clemo, T. M., and Hull, L. C., FRACSL User's Manual, Idaho National Engineering Laboratory ST-ES-06-87, November 1987.

- Mood, A. M. and Graybill, F. A., Introduction to the Theory of Statistics, McGraw-Hill, New York, 1963.
- More', J. J., Garbow, B. S., and Hillstrom, K. E., User Guide for MINPACK-1, ANL-80-74, Argonne National Laboratory, 1980.
- Neretniecks, I., Diffusion into the Rock Matrix; An Important Factor in Radionuclide Retardation, Journal of Geophysical Research, 85, B8, pp. 4379-4397, 1980.
- Neuzil, C. E., and Tracy, J. V., Flow Through Fractures, Water Resources Research, Vol. 17, No. 1, pp. 191-199, 1981.
- Powell, M. J. D., A Hybrid Method for Nonlinear Equations, in Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, ed., Gordon and Breach, 1970.
- Prickett, T. A., Naymik, T. G., Lonquist, C. G., A "Random-Walk" Solute Transport Model for Selected Groundwater Quality Evaluations. Illinois Department of Energy and Natural Resources, 1981.
- Rouleau, A. and Gale, J. E., Statistical Characterization and Numerical Simulation of a Fracture System for Hydrogeological Purposes, International Groundwater Symposium on Groundwater Resources Utilization and Contaminant Hydrogeology, Montreal, Canada, 1984, pp. 188-196.
- Silling, S. A., Technical Position on Documentation of Models, NUREG-0856, 1983.
- Theis, C. V., The Relation Between the Lowering of the Piezometric Surface and the Rate and Duration of Discharge of a Well Using Ground Water Storage, Trans Am. Geophys. Union, 16, pp. 514-524, 1935.
- Thomas, S. D., Ross, B., and Mercer, J. W., A Summary of Repository Siting Models, NUREG/CR-2782, 1982.
- Wang, J. S. Y., Tsang, C. F., and Sterbentz, R. A., The State-of-the-Art of Numerical Modeling of Thermohydrologic Flow in Fracture Rock Masses, LBL-10524, May 1982.

APPENDIX A
MODEL NOMENCLATURE

CONTENTS

Table A-1.	Nomenclature Components	A-4
Table A-2.	Variable Summary	A-7
Table A-3.	Data base fields	A-36
Table A-4.	Subroutines, functions, common blocks, include files	A-37
Table A-5.	Tasks	A-41
Table A-6.	Dimensioning	A-42
Table A-7.	Deltas	A-43

APPENDIX A

MODEL NOMENCLATURE

This appendix summarizes the nomenclature of the flow and transport modeling portion of FRAC-UNIX. Appendix B presents a summary of input and output for this portion of the code and Appendix C provides a description of the input and output for the library-derived numerical routines.

The FRAC-UNIX variable names are synthesized from the variable, prefix and initial and final subscript symbols described in Table A-1. Variable names, as far as possible, conform to common usage such as Q for flow. A limited number of prefixes are used, principally to designate input data for fracture (FR), matrix (MT) and particle (PR). The initial subscript designates the substance (F, M, P) for computed fracture, matrix, and particle variables or the location or type of variable. A final subscript may be used to designate a mathematical qualification such as minimum, MN. The true value of a flag always causes the action indicated by the flag name. The number zero is never used in a variable name.

All global variables, some local variables, all variables used elsewhere in this appendix to dimension other variables and all variables transferred in argument lists have been entered into a data base using the dBase III (Ashton-Tate 1984) program on an IBM personal computer. A complete display of the data base is shown in Table A-2. The data base fields, as described in Table A-3, give the name, the Fortran type, the array size, the availability of the variable, the default value, the defining subroutines, the using subroutines, the tasks which use the variable and finally a description of the variable.

Output from the data base may be made based on any logical statement selecting recognizable text. Output alphabetized by variable name may be generated for all variables, for specific defining or using (or both) subroutines or for specific tasks. Outputs organized by the various array sizes and for the complete set of input variables is presented in this appendix.

TABLE A-1. NOMENCLATURE COMPONENTS

Variable

A	area
B	aperture
C	constant, concentration
CAP	capacitance
CON	conductivity
CP	specific heat
DIF	diffusion
DSN	dispersion
DSY	dispersivity
E	enthalpy
G	normal distance
H	head
I	index, column number
IF	fracture number
IJ	node number (also NOD)
IM	matrix block number (matrix block with node IM at lower left corner)
IP	particle number
J	row number
L	length, distance
N	total number
NOD	node number (also IJ)
PE	Peclet number
PI	3.14159265
POR	porosity
PSI	fracture angle at current node, CCW from due east, radians
PSIND1	fracture angle at node 1, CCW from due east, radians
Q	volumetric flow
QQ	volumetric flow corrected to density at 100°C
R	radius
RE	Reynolds number

TABLE A-1. (continued)

RHO	density
RT2	square root of 2.0
SS	specific storage
T	time
TP	temperature
V	velocity
VISC	viscosity
VOL	volume
W	weight
X	position coordinate
Y	position coordinate (Y or transverse)
Z	position coordinate, model thickness

Prefix

D	temporal derivative, delta
FR	fracture
MT	matrix (used only for data input or other fixed properties)
PR	particle (used only for data input or other fixed properties)

Initial Subscript

1	temporary
B	boundary, bottom
DIAG	diagonal
DN	down
EL, ELC	electrode
EXT	external
F	fracture (computed value), final
FIRST	first
IN, I	inlet
J	junction
L	longitudinal, left

TABLE A-1. (continued)

LAST	last
M	matrix (computed value)
N	node; a parameter (see rest of subscript) evaluated at a node
NEXT	next
OT,0	outlet
P	particle (computed value)
PREV	previous
R	right
SY	synthesized fracture
T	transverse, top
TO	used in distance to something, ratio to something
V	vertical
WEL	well
X	in the X direction
Y	in the Y direction
Z	in the Z direction, or initial

Final Subscript

A	absolute
AV	average
CM	cumulative (global)
GM	geometric mean
LIM	limit
MN	minimum
MX	maximum
NAT	native
none	algebraic value implied
SM	sum
SN	sign
STR	string
TR	transient
U	normalized

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE	
ADFG	L	A				MAR	MTF		set .T. if particle advected out of a fracture during previous move. then prevents diffusion dominated transport.	
ADVECT	R	L	1			MAR		MM	net advective movement in matrix, (VPX**2+ VPY**2)**.5*DT	
AOE	I	G	1			MAR	SET	PST	OU	no of particles advecting out end of fracture in explicit high Peclet moves
AOS	I	G	1			MAR	SET	PST	OU	no of particles advecting out the side of fracture in explicit high Peclet moves
ATOL	R	G				INP				see Appendix C
B	R	G	NF		0.	INP BDA	SET	REL ADV BOR FRV FTF JUN MAR PRP MAC CON	AL	fracture aperture, default is for B(1) only.
BAV	R	G	3			INP		SET	SY	mean aperture of synthesized set. Vertical, horizontal, diagonal upwards to the right
BD	R	G	NNSY			SET			SY	net aperture, diagonal
BDD	R	G	NNSY			INP		SET	SY	aperture, diagonal, deterministic
BH	R	G	NNSY			SET			SY	net aperture, horizontal
BHD	R	G	NNSY			INP		SET	SY	aperture, horizontal, deterministic
BIAS	R	L	1			JUN		JUN	JU	shift (when pos) to larger QOT from QOTINT (NGRPOT) for discontinuous junction

TABLE A-2. VARIABLE SUMMARY
Page No. 2
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
BSTR	R	G	NSTR			INP	SET	AL	aperture of all fractures in a string
BV	R	G	NNSY			SET		SY	net aperture, vertical
BVD	R	G	NNSY			INP	SET	SY	aperture, vertical, deterministic
CAPN	R	G	NN			SET	MAS	DH	capacitance at a node
CELC	R	G	NELC			ELC	OUT	EL	particle concentration at electrode locations
CMPRES	L	G	1		T	INP MAR	BDA ADV REL DER	RE	replace particle which has left system with last particle in array
CONCPT	L	G	1		F	INP	BDA DER		
CONGM	R	L	1			MAS		DH	conductivity of matrix outside of model, geometric mean of x and y. used only for permeable boundaries
CONWEL	R	G	NOUT			DER		OU	concentration or temperature at node IOUT
CONXY	R	L	1			MAS	MTV	DH VM	fourth root of ratio of conductivities, used to calculate velocities around well node
CPW	R	G	1			INP	PRP SET DER	HT	heat capacity of water
CQRBF	R	G	NNRB			SET	MAS	DH	Constant, flow, remote boundary, matrix
CQRBM	R	G	NNRB			SET	MAS	DH	Constant, flow, remote boundary, fracture
CROCK	R	G	NMTM			SET	PRP	HT	rock portion of R*CP
CROCK2	R	G	NMTM			SET	MAR MTV	HT	rock thermal property
CTPF	R	G	NF			QVS	FRV	HT	density over viscosity, frac-to-ref

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
CTPM	R	G	NN			QVS	DIA MAS MTV	HT	density over viscosity, matrix-to-ref
CTPRBF	R	G	NNRB			QVS	MAS	HT	density over viscosity, remote boundary, fracture-to-ref
DEL	R	L	1			JUN		JU	flow to END(ISECT) minus flow avail in current exit frac
DELT12	P	G	1			DEL			Cray 1.E-12, other 1.E-5
DELT14	P	G	1			DEL			Cray 1.E-14, other 1.E-5
DELT15	P	G	1			DEL			Cray 1.E-15, other 1.E-5
DELT30	P	G	1			DEL			Cray 1.E-30, other 1.E-30
DELT50	P	G	1			DEL			Cray 1.E-50, other 1.E-30
DELT90	P	G	1			DEL			Cray 1.E-90, other 1.E-30
DELTA4	P	G	1			DEL			Cray 1.E-4, other 1.E-5
DELTA5	P	G	1			DEL			Cray 1.E-5, other 1.E-5
DELTA6	P	G	1			DEL			Cray 1.E-6, other 1.E-5
DELTA7	P	G	1			DEL			Cray 1.E-7, other 1.E-5
DELTA8	P	G	1			DEL			Cray 1.E-8, other 1.E-5
DELTA9	P	G	1			DEL			Cray 1.E-9, other 1.E-5
DEM	R	G	NN			ENT MTF MAR REL SET	PRP	HT	time integral of energy in matrix node
DH	R	G	NN			MAS	FCN	DH	temporal derivative of head
DH12	R	L	1			FRV		TR	head at node 1 minus head at node 2
DHSS	R	G	1			SET	MAS	DH	steady state head derivative for the entire model due to user specified QQEXT. Not applicable to flow transients, remote boundaries or infinite porous media solution.

TABLE A-2. VARIABLE SUMMARY
Page No. 4
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
DHXDY	R	L	3			MTV		MM	derivative of head in the y direction. 3 values along vert line through particle
DHYDX	R	L	3			MTV		MM	derivative of head in the x direction. 3 values along hor line through particle
DIAGFR	I	A	1			NER MAR	MTF	MM	1 = rising frac in host cell, 2 = declining, 0 = otherwise
DIFCO	R	G	1		1.E-9 0	ADV	MAR FTF BOR MAC	MM	diffusion coefficient. equal to TRDIF for solute, THDIF for heat
DL1	R	A				NER	BOR MAR	MM	distance to the border
DLMIN	R	G				SET	BOR	MM	smallest DX or DY to limit matrix diffusion
DLN	R	A				BOR	MAR	MM	distance moved normal to the border
DLU	R	L	1			MTF		MF	fraction of length along fracture newly entered from matrix
DQFN	R	L	4			JUN		JU	excess of flow at end of ex grp over QIBINT
DQFNAB	R	L	1			JUN		JU	ABS (DQFNMN)
DQFNMN	R	L	1			JUN		JU	min, over all ex groups, of DQFN
DQST	R	L	4			JUN		JU	excess of QIBINT over flow at start of ex grp
DQSTAB	R	L	1			JUN		JU	ABS (DQSTMN)
DQSTMN	R	L	1			JUN		JU	min, over all ex groups, of DQST
DTP	R	G	1		1.	BDA INP DER	BOR	MF MM	max particle move time step. Must be equal to DTQ for heat transport
DTPEXP	R	G	1		1.	BDA INP DER	DER	MF MM	expands DTP after each DTP

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
DTQ	R	G	1		1.	BDA INP DER	INT	DH	max flow transient time step
DTQEXP	R	G	1		1.	BDA INP DER	DER	DH	expands DTQ after each DTQ
DTREM	R	G	1			MAR ADV ENT MTF REL MAC	BOR FTF MTV	MF MM	portion of particle movement time step remaining
DTREM1	R	A				MAC	ROO	HP	temporary value of DTREM
DTSTP	R	G	1		1.	DER BDA	MAR ELC ADV ENT FTF MTF REL	CO	particle movement time step
DTTOX	R	L	1			MAR		MM	time needed to reach the approaching x element boundary
DTTOY	R	L	1			MAR		MM	time needed to reach the approaching y element boundary
DTUSED	R	A	1			BOR	MAR MTF		time used so far by particle
DVDX	R	G	1			MTV	MAR	MM	matrix velocity gradient in x direction
DVDY	R	G	1			MTV	MAR	MM	matrix velocity gradient in y direction
DVXDY	R	G	1			MTV	MAR	MM	change in particle velocity in the X direction wrt movement in the Y direction
DVYDX	R	G	1			MTV	MAR	MM	change in particle velocity in Y direction wrt movement in X direction
DX	R	G	NCOL			GRD	ALL	XY	distance between node columns
DX1	R	L	2			MTV		XY	temporary storage of DX
DXF	R	A				DIA	MTV	MM	distance in the X direction to the nearest fracture
DXL	R	A				MAR	MAC	HP	distance to end of fracture in the direction of particle movement

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
DXZ	R	G	1			INP	GRD	XY	distance between the two left most columns
DY	R	G	NROW			GRD	ALL	XY	distance between node rows
DY1	R	L	2			MTV	MAS	XY	temporary storage of DY
DYF	R	A				DIA	MTV	MM	distance in the Y direction to the nearest fracture
DYMC	R	A	1			MAC	MAR	HP	y distance (MAC) or normalized y distance (MAR) moved into matrix by subroutine MAC
DYNMIC	L	G	1		T	INP BDA SET	REL DER MAS FCN	CO	enables derivative calculations for transient flow calculation
DYZ	R	G	1			INP	GRD	XY	distance between lowest two rows
ELENAM	A	G	1		EL	BDA	MAR REL ENT MTF	OU	alphanumeric for particle location output
END	R	L	8			JUN		JU	max cumulative exit flow in a section
ERR	R	A				BOR	NRP	MM	minimization function to find particle diffusion distance
EXTBND	L	G			F	BDA INP	MAR	MM	.T. - matrix particles may diffuse out of model through a no-flow boundary, .F. - particle may not diffuse out of model
F3D	L	G	1		F	INP BDA	MAR SET REL FRV	3D	flag to allow three dimensional fracture representations.
FLUX	R	G	30			INP	DER	BC	trans var of flow at NODIN (with QEX TTR = .T.) up to 10 x-y point pairs.
FRCNAM	A	G	1		FR	BDA	MAR REL ENT MTF	OU	alphanumeric for particle location output
FRDEB	L	G	NF			INP	SET PRP	BC	fractures dead ended at boundary
FRE	R	G	NF			SET ENT FTF MAR MTF REL	PRP	HT	time integral of energy in fracture IF

TABLE A-2. VARIABLE SUMMARY
Page No. 7
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
FREEZE	L	G	NN		.F.	INP BDA	MAS SET	BC	flag to freeze head at node by setting QQEXT=-internal flow
FREND	I	G	2	NF		INP SET	FRV FRT JUN MAR MAS MTV PST REL CON	XY	beginning (smaller of the two) and ending (larger) node indices for each fracture
FREZFG	L	G	1			SET		BC	true if any node is frozen
FRFAC	R	G	1		.072	BDA INP	FRV	TR	Darcy friction factor
FRL	R	G	NF			SET	ALL	XY	fracture length
FRLX	R	L	NF			SET		XY	x component of fracture length
FRLY	R	L	NF			SET		XY	y component of fracture length
FRSS	R	G	NFM			INP	SET	DH	fracture specific storage
FRSTFG	L	G	1		T	BDA SET	DER	CO	flag to run MASS, FTVEL and JUNFUN once to initialize a constant flow transport case run from input heads
FVEC	R	G					MXD		
GEOFAC	R	G	NMTM			ADV	MAR BOR	MM	Solute- (TRGF) factor to adjust diffusion to account for tortuosity. Heat- (THGF) correction of diffusion to simulate conduction
GRAV	R	G	2		*	BDA	FRV SET	DH	gravitational constant 7.323E10, 2.404E11
GRP	I	L	8			JUN		JU	exit group the isect section belongs to
RPIN	I	L	8			JUN		JU	inlet group the nth fracture at node belongs to
RPOT	I	L	8			JUN		JU	exit group the nth fracture at node belongs to
	R	G	NN			FCN INT STA DSG ZSG	DIA FRV MAS MTV PST	DH	head

TABLE A-2. VARIABLE SUMMARY
Page No. 8
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
H1	R	L	3	3		MTV		DH	head at 9 nodes closest to particle
HB	R	L	1			DIA		VM	head at element boundary. used to determine head gradient away from a diagonal fracture
HEADPT	L	G	1		F	INP BDA	PST DER STA	OU	flag to create plots of head distribution
HEAT	L	G	1		F	SET BDA	DER	HT	a particle group is for heat
HEATP	L	G	NPGM X		F	INP BDA	ADV ENT FTF MAR MTF MTV REL SET DER	CO	indicates type of particle group true-> heat particle false-> solute
HL	R	G	NN		0.	INP BDA	MAS	BC	head at parallel aquifer to which the model is leaking
HPET	R	G	NN			SET	DER STA		internally stored head for petroleum units option
HRB	R	G	NN		0.	INP BDA	MAS	BC	head at remote boundary
HTSY	R	G	1			INP	SET	SY	height of synthesis region. Must be greater than NROWSY * DYZ
HX	R	L	3			MTV		MM	head values on a hor line thru particle
HXF	R	A				DIA	MTV	MM	head in a fracture at the elevation of the particle
HY	R	L	3			MTV		MM	head values on a vert line through particle
HYF	R	A				DIA	MTV	MM	head in a fracture at the x position of the particle
HZ	R	G	NN			STA	FCN INT	DH	initial head
I	I	L	1			ALL	ALL	AL	column index
II	I	L	1			DIA BOR MTV SET		VD	temporary value of column index

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
2	I	L	1			MTV		XY	index of column at node 2 of fracture
IA	I	G				SPA			see Appendix C
ICOL1	I	G	1			INP	SET	SY	first synthesis column used as a model column
ICOL2	I	G	1			INP	SET	SY	last synthesis column used as a model column
IER	L	A				NRP BOR		MM	identifies Newton-Raphson failure
IER	I	A				ROO	MAC	HP	error flag to identify failure of root to converge
IF	I	A	1			FTF NER MAR	BOR MTF	AL	fracture index
IF1	I	L	1			FTF FTV MTV	BOR	MM	temporary storage of fracture index
IFMAT	I	G	NF			INP	SET	VF	material index for the fracture
IFOT	I	G	8	NN		JUN	FTF		index (NF array) of frac containing lth exit stream at node ij
IFRST	I	G	1		0.	MXD MXE BDA SET			0. if first time through
IGRID	I	G	1		1	INP BDA	GRD	XY	defines grid type (0-input: XCOL&YROW. 1-expand from origin at node1, input DXZ,DYZ,XEXP,YEXP-2- expand from origin at NODIN, inp:DXZ, DYZ, XEXP, YEXP)
IGRP	I	L	1			JUN		JU	group index
IJ	I	A	1			MAR MAS	NER FTV OSH MTF	AL	element index (index of node at lower left corner)
IJ1	I	A	1			FTV MAR MAS	MTV	MM	IJT in FTVEL, IJ in other calling subroutines

TABLE A-2. VARIABLE SUMMARY
Page No. 10
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
IJ2	I	A	1			MTV	DIA	DH	temp value of IJ, upper or lower element adjacent to node.
IJ22	I	G	1			MTV	OSH	MM	number of node nearest the particle (central node)
IJP	I	L	1			MAR		MM	node at upper left of element
IMIX	I	G	1		0	INP BDA	FTF	JU	integer flag to control mixing in junctions with more than 2 branches. 0 = no mixing, 1= diffusion, 2= complete mixing
IMMAT	I	G	NN			INP	DIA MAR MAS MTV OSH PRP CON BOR SET	MM	matrix material number--must be input for all nodes even though some border nodes are not at lower left of a cell.
IMMAT1	I	L	1			MAR MAS MTV PRP SET		MM	temporary value of matrix material number
INFO	I	G	1				DER		
INFSQ	I	G	8	NN		SET	DIA FTF FTV JUN MAR MTV NER OSH REL PRP	XY	indices of fracs at each node stored by fracture angle in CCW order, location one is -PI/4 from the positive x coordinate vector
INLS	I	G	NNLS			INP	REL SET		indices of nodes on line source
INRB	I	G	NNRB			INP	SET MAS PRP	BC	indices of boundary nodes connected to remote boundary.
IOUT	I	G	NOUT			INP	DER ENT	OU	particle exit nodes
IP	I	A				ADV REL	MAR OSH NER MTF ENT FTF MTV	RE VD MF MM	particle index
IPARM	I	G				BDA	MXE	CO	see Appendix C
IPG	I	G	1			ADV PST SET	ALL	PT	index of current particle group (local variable)

TABLE A-2. VARIABLE SUMMARY
Page No. 11
6/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
PJ	I	L	1			MAR		MM	node at lower right of element
PJP	I	L	1			MAR		MM	node at upper right of element
POT	I	G	1			ENT DER		OU	Cumulative total of particles which have left the model
PRINT	I	G	2		0,0	BDA INP	ENT FTF MAR REL MTF	OU	IPRINT(1) print particle transfers (0 none, 2 release and exit model, 3 all fracture/element transfers. IPRINT(2) index of particle to be reported.
QCASE	I	G	1		1	INP FRV BDA		TR	Two-dimensional fracture flow regime control. See text (laminar or turbulent).
SECT	I	L	1			JUN		JU	section index
SIDE	I	L	1			MAR FTV		MM	element boundary crossed (1-horizontal, 2-vertical)
STYP	I	L	8			JUN		JU	sec typ (1 1st of ext, 2 1st of int, 3 2nd of int, 2nd of ex)
	I	L	1			ALL	ALL	AL	node row index (local variable)
I	I	L	1			SET		XY	index of row at node 1 of fracture
A	I	G				SPA			see Appendix C
J	I	L	1			JUN		JU	seg no (CCW of due e) of exit frac
OB	I	G				BDA INT			see Appendix C
PRINT	I	G	1		0	INP BDA	JUN	OU	IF 1 print junction details
ROW1	I	G	1			INP	SET	SY	first synthesis row used as a model row
ROW2	I	G	1			INP	SET	SY	last synthesis row used as a model row
OT	I	L	1			JUN		JU	index of exit fractures in current group

TABLE A-2. VARIABLE SUMMARY
Page No. 12
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
L	R	L	1		MAS			DH	runs sum of LXWEL and LYWEL
L	I	L	1		JUN			JU	exit stream counter
LAM	L	G	NF		FRV INP			TR	laminar fracture flow flag
LASTOT	L	L	1		JUN			JU	true if last branch was an outlet
LCUTH	R	G	3		INP	SET		SY	upper limit of synthesized fracture length: vertical, horizontal, diag. upwards to right
LCUTL	R	G	3		INP	SET		SY	lower limit of synthesized fracture length: vertical, horizontal, diag. upwards to right
LEAKY	L	G	1	F	BDA SET			BC	flag to allow simulation of a leaky aquifer.
LFAC	R	G	3		INP	SET		SY	ratio of synthesized fracture mean length to mean separation: vertical, horizontal, diag. upwards to right
LFTORB	R	G	NNRB		INP	SET		BC	dist in frac to remote boundary from a boundary node.
LINTST	L	G	1	F	BDA DER			BC	flag to call LINTS one time only for FLUX
LMTORB	R	G	NNRB		INP	SET		BC	distance in matrix to remote boundary from a boundary node.
LWK	I	G	1		BDA				see Appendix C
MAD	I	G	1		MAR SET	PST		OU	no of all matrix moves except diffusive transfers to matrix
MAY	I	G	1		MAR SET	PST		OU	no of particles remaining in matrix

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
MBF	I	G	1			MAR SET	PST	OU	no of Monte Carlo transfers remaining in matrix
MBI	I	G	1			MAR SET	PST	OU	no of matrix Monte Carlo particle moves
MBS	I	G	1			MAR SET	PST	OU	no of Monte Carlo diffusive transfers to fracture
MCF	I	G	1			MAR SET	PST	OU	no of low Peclet particle moves in fracture
MCT	I	G	1			MAR SET	PST	OU	no of high Peclet (pe > 200) particle moves in fracture
MIF	I	G	1			MAR SET	PST	OU	no of particles staying in fracture in Monte Carlo high Peclet moves
MNCOL	P	G	1			COM			dimension, 15
MNDOT	P	G	1			COM			dimension, 1
MNELC	P	G	1			COM			dimension, 10
MNF	P	G	1			COM			dimension, 120
MNFRM	P	G	1			COM			dimension, 10
MNFSET	P	G	1			COM			dimension, 3
MNMTM	P	G	1			COM			dimension, 10
MNN	P	G	1			COM			dimension, MNROW * MNCOL
MNNLS	P	G	1			COM			dimension, MNROW
MNNRB	P	G	1			COM			dimension, 40
MNNSY	P	G	1			COM			dimension, 144
MNOUT	P	G	1			COM			dimension, 11
MNPGMX	P	G	1			COM			dimension, 1
MNPMX	P	G	1			COM	DER		dimension, 1000

TABLE A-2. VARIABLE SUMMARY
Page No. 14
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
MNROW	P	G	1			COM			dimension, 12
MNSTR	P	G	1			COM			dimension, 40
MOE	I	G	1			MAR SET	PST	OU	no of particles moving out end of fracture in Monte Carlo high Peclet moves
MOW	I	G	1			MAR SET	PST	OU	no of particles out well into matrix in Monte Carlo high-Peclet moves
MPWEL	R	G	NOUT			ENT DER		OU	cumulative amount of solute or energy leaving exit node during the communication interval
MSTST	L	G	1		F	INP SET BDA	MAS MTV FTV	OU	sets debug output on during mass balance calculations
MTCONX	R	G	NMTM			INP	DIA MAS MTV SET	VM	matrix conductivity in x direction at 100 degrees C
MTCONY	R	G	NMTM			INP	DIA MAS MTV SET	VM	matrix conductivity in y direction at 100 degrees C
MTCP	R	G	NMTM			INP	SET	HT	matrix-specific heat
MTDSYL	R	G	NMTM			INP	MAR	MM	longitudinal matrix dispersivity
MTDSYT	R	G	NMTM			INP	MAR	MM	transverse matrix dispersivity
MTPOR	R	G	NMTM			INP	SET PRP MTV MAR CON	VM	matrix porosity
MTRHO	R	G	NMTM			INP	SET	HT	matrix density
MTSS	R	G	NMTM			INP	SET	DH	matrix specific storage
MTTCON	R	G	NMTM			INP	SET	HT	matrix thermal conductivity
MVTST	L	G	1		F	INP BDA	MTV MAR FTF MTF MAC	MM OU	print MATVEL calc., turns on code development debug output during particle movement calculations
NBKPT	I	G	1			INP	DER	BC	number of breakpoints in tabular input of transient QQEXT (NODIN)

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
NCMPRS	I	G	1			DER REL		RE	number of particles lost due to compression in current DTSTP interval
NCOL	I	G	1			INP	ALL	XY	number of nodal columns
NCOLM1	I	G	1			SET	PRP GRD	XY	NCOL-1
NCROSS	I	G	3			SET	OUT	SY	number of scan-line crossings in set: vertical, horizontal, diag. upwards to right
NDOT	I	G	1		0	INP BDA	DER ENT MAR MTV OSH SET	OU	number of particle exit wells, detailed particle data processing will be performed
NELC	I	G	1		0	INP BDA	ELC SET ADV REL	OU	number of electrodes
NF	I	G	1		0	INP BDA SET	ALL	XY	number of fractures (each complete in an element)
NFISTR	I	G	NSTR			INP	SET	XY	number of fractures in a given fracture string
NFRM	I	G	1		*	BDA INP			number of fracture materials * MNFRM
NFX	I	G	1			MAR SET	PST	OU	no of nondiffusive-fracture transfer moves
NGRPOT	I	L	1			JUN		JU	number of continuous groups of exit (and inlet) fractures
NIN	I	G	NN			JUN		JU	number of inlet fractures at each node
NIN1	I	L	1			JUN		JU	index num, in order of PSISQ, of 1st inlet ccw of due east or cw inlet of inlet grp straddling e
ILAG	I	G	1			INP	DER	HT	number of particle move intervals DTSTP to be included in calculation of moving average fracture temperatures
IMTM	I	G	1			INP	ADV SET		number of matrix materials

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
NMX	I	G	1			MAR SET	PST	OU	no of matrix explicit (non-Monte Carlo) particle moves
NN	I	G	1			INP SPA	ALL	XY	number of nodes
NNF	I	G	NN			SET	FTF MTV JUN	XY	number of fractures at each node
NNLS	I	G	1		0	INP BDA	SET REL	RE	number of nodes on line source
NNRB	I	G	1		0	INP BDA	SET MAS PRP	BC	number of boundary nodes connected to remote boundary.
NNSOT	I	G	NN			JUN	FTF	JU	number of streams with flow exiting the node
NNSY	I	D				SET		SY	number of nodes in synthesis region
NOD1	D		1					VF	fracture end with smaller node index
NOD2	D		1					VF	fracture end with larger node index
NODIN	I	G	1		9999	INP BDA	GRD MAS MTV OSH REL SET DER	AL	mass and particle injection node
NODOT	I	G	NDOT			INP	ENT MAR MTV OSH	OU	particle and flow exit wells, must be given if exit is to be permitted and if QQEXT is specified at node
NODPE	I	A	1			MAR	FTF ENT	MM MF	temporary value of node destination of part in fractures
NODPI	I	A				MAR	FTF		temporary value of node source of part in fracture
NOT1	I	G	NN			JUN		JU	1st exit (cw of 1st inlet)
NOUT	I	G	1		0	INP BDA	ENT DER	OU	no of particles exit nodes (wells and boundary nodes)
NPGMX	I	G	1		0	INP BDA	ADV ELC EXC PST REL SET DER	PT	total number of particle groups must be GT.0 for a particle move

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
IPMX	I	G	1		1	INP BDA	SET EXC	RE	maximum number of particles in a group. An input value of 1, along with NPGMX.GT.0 will cause a stop and write
IPOSML	I	G	NOUT			DER ENT		OU	no of particles exiting node at time between TSMALL and 1.5 TSMALL
IPOSS	I	L	8			JUN		JU	seq no(ccw of due e)of fracs in parent grp & poss this sec
IPOUT	I	G	NOUT			DER ENT		OU	no of particles exited at each exit node
IROW	I	G	1			INP	ALL	XY	number of nodal rows
IROWM1	I	G	1			SET	PRP GRD	XY	number of nodal rows minus one
SECT	I	L	1			JUN		JU	no of exit flow sections at node
SEED	R	G	1		*	BDA INP	SET	RE	seed for random number generation, positive 8-digit real number. * Initial value is 87654321.
SET	I	G	3			SET		SY	number of synthesized fracture between LCUTL and LCUTH: vertical, horizontal, diag. upwards to right
SNCHG	I	G	NN			JUN		JU	no of sign(flow dir)chngs in ccw sweep of fracs at a node
STR	I	G	1		0	INP BDA	SET	XY	number of fracture strings, used to generate fracture segments from input descriptions of whole fractures
SY	D		1					SY	number of synthesized fractures
TINT	I	G	100	NDOT		SET ENT	DER	OU	number of particles in interval
UMCOL	I	G	1		5	INP BDA		OU	number of columns in debug output
LDPL	R	G	1			REL SET		RE	amount of particle mass to be released in time DTSTP

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
OLSU	R	G	1		.001	INP BDA	REL SET	RE	offset, line source, normalized
PE	R	G	NF			ADV	MAR BOR	MF	Peclet number in fracture. If PE > 200. use macro monte carlo otherwise use random walk in fracture
PELHST	I	G	NELC			INP	ELC SET	EL	fracture (+) or element (-) location of electrodes
PELXYU	R	G	2	NELC		INP	ELC	EL	normalized x-y cordimates of electrodes within an element
PERMBD	L	G	1		F	INP BDA	MAS SET	BC	enables flow from boundaries for infinite media solutions
PETU	L	G	1		F	SET BDA	DER STA	CO	flag to convert heads to p.s.i. for output
PFLAG	L	A				FTV MAR MAS	MTV DIA	CO	identifies call to MATVEL or DIAGVEL as for particle move (.T.) or mass balance
PI	R	G	1			BDA	ALL	XY	3.14159265
PLCORE	R							RE	dimension = NPGMX (4NPMX+3)
PLOCPT	L	G	1		F	BDA INP	DER PST	OU	call POST to set up particle location plots
POR	R	A				MAR	MAC	HP	average porosity over the matrix adjacent to the two sides of a fracture
PPHOST	I	G	NPMX			MAR REL MTF FTF ENT ADV INT	PST ELC DER CON	RE MM MF	partic loc(+ fr index,o-left from bdry or well,- el index)
PPXU	R	G	NPMX			MTF MAR REL ADV FTF FTV MAS SET	ELC NER MTV PST OSH MAC DER	RE MF MM	part pos,frac of el wth from 1,frac of frac lngth fr NOD1
PPYU	R	G	NPMX			MTF MAR REL ADV FTF FTV MAS SET	ELC NER MTV PST OSH MAC DER	JN RE MF MM	part pos,frac of el hgt fr bot,frac of frac width fr left for vertical or diagonal frac, from bottom for horizontal

TABLE A-2. VARIABLE SUMMARY
Page No. 19
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
PPZCON	R	G	1			SET	REL	HT OU	concentration of solute or energy in injected fluid, may be negative if energy
PPZNP	I	G	1		0	REL SET ADV INT BDA CON	ELC PST CON DER	RE	number of particles released in a group (if CMPRES, reduced by number which have left model)
PPZPM	R	G	1		1.	BDA REL SET DER	ELC ENT FTF MTF MAR CON	HT OU	amount of solute or energy carried by each particle, may be negative if energy
PPZU	R	G	NPMX			REL MAR ADV	FTF MTF	MF	normalized z location of particle
PRPST	L	G	1		F	BDA QVS		HT	true if LINTS has been called for RHOTB, VISTB
PSI	R	L				FTV REL		JU	frac "angle" from due E at any node. possible val in CCW seq: 0. to 2 pi in pi/4 increments
PSI1	R	A				NER	MAR	MM	angle of fracture nearest to particle
PSIND1	R	G	NF		0.	SET BDA	FRV MAR MTV MTF CON PST	XY	frac "angle" from due E at NOD1. possible val in CCW seq: 0., pi/4, pi/2, - pi/4, default for PSIND1(1) only
PSISTR	I	G	NSTR			INP	SET	XY	angle of fracture string, 1=vertical, 2=rising diagonally, 3=horizontal, 4=dipping diagonally
PSIX	R	G	1		0.	INP BDA	FRV DIA MTV	HT	angle of X unit vector wrt horizontal, radians
PSIY	R	G	1		*	INP BDA	FRV DIA MTV	HT	angle of Y unit vector wrt vertical, radians, * PI/2
EXTTTR	L	G	1		F	INP BDA	DER SET	BC	flag enabling transient variation of QQEXT (FLUX) at NODIN
PF	R	G	NF			FRV	MAS	OU	flow in frac at nodal volume edge, positive from NOD1 to NOD2

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
QFATN	R	G	2	NF		JUN	FTF	JU	flow in fracture at (NOD1,NOD2). positive entering node
QFNSQ	R	L	8			JUN		JU	sequenced (INFSQ order) algeb frac flow at node
QIBINT	R	L	4			JUN		JU	cum inlet flow at interruption by a group of exits
QIN	R	L	1			MAS		DH	flow into node from quadrant
QINCM	R	G	2	NF		JUN	FTF	JU	cumulative inlet flow (CCW from CW edge of NIN1) where PPYU = 0
QINSN	R	G	2	NF		JUN	FTF	JU	+1. QINCM increasing with PPYU, -1. otherwise
QLS	R	G	***			REL SET	REL SET	RE	flow through half element length of line source. *** = 2*NNLS
QLSS	R	G	1			REL SET	REL SET	RE	total absolute flow through line source
QMSM	R	L	1			MAS		DH	temp value of net algebraic flow into node from all source
QNFOT	R	G	NN			JUN	FTF	JU	total fracture flow leaving a junction
QOT	R	L	1			JUN		JU	running cum exit flow in test of QOTINT spanng corr QIBINT
QOT1	R	L	1			JUN		JU	running cum exit flow in sweeping streams at a junction
QOTBIA	R	G	8	NN		JUN	FTF	JU	abs flow between PPYU=0. and ccw edge of exit stream
QOTCM	R	G	8	NN		JUN	FTF	JU	minimum cumulative exit flow CW from CCW edge of NEX1 at node and ith exit stream
QOTFN	R	L	1			JUN		JU	value of QOT after adding a QOTINT
QOTINT	R	L	4			JUN		JU	tot flow in a continuous group of exits

ARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
OTOI	R	G	NN			JUN	FTF	JU	ratio of total outlet QFATN to total inlet QFATN at node
OTSN	R	L	1			JUN	FTF	JU	+1. for QOTCM increasing with YPU, -1. otherwise
OTST	R	L	1			JUN		JU	value of QOT before adding a QOTINT
QCUM	R	A				MAS	FTV	DH	total volumetric inflow to node
QEXSM	R	G	1			SET MAS		DH	sum, over all nodes, of external flow, pos into model
QEXT	R	G	NN		0.	INP MAS DER BDA FCN	MAR REL SET FTF MTV JUN	BC	ext flux at node at 100 degrees C, pos into node QQEXT (NODIN) may not be smaller than a QQEXT at the other end of an isolated fracture
QFI	R	G	NN			MAS	MAR	DH	total fracture flow entering nodal volume
QFO	R	G	NN			MAS	MAR REL	DH	total fracture flow exiting nodal volume
QINJ	R	G	1		1.E+1 5	INP BDA SET		RE	flow at 100 degrees C to be used in calculating PPZPM if QEXTTR is true
QOIN	R	G	4			MAS	REL	RE	matrix flows leaving NODIN
QOSUM	R	G	1			MAS	REL	RE	sum of those matrix flows leaving NODIN nodal volume, positive value only
QTOHL	R	G	NN		0.	INP BDA	MAS SET	BC	flow-to-head ratio at local temperature at nodes leaking to a parallel aquifer
S	R	A	8			MAS	FTV	VF	matrix flow into half of a side of a model volume
SAVED	R	L	1			JUN		JU	flow reserved for second section of an interior exit group
SGN	R	L	1			JUN		JU	magnitude =1., sign is that of QFNSQ

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
QSGNP	R	L	1			JUN		JU	prev val of QSGN,used to determine if flow direction chang
QTIME	R	G	1			DER	FCN		time at end of flow time interval
QTOGO	R	L	1			JUN		JU	exit flow increment to end of section
QUADF	R	L	1			MAR		MM	+1. particle finished above diag fract, -1. below
RANDOM	R	L	1			REL BOR MAR		RE	random num determ. initial lat pos and host for particle
RANUM	R	A				MAC	ROO PMT	HP	random number between 0. and 1., less the probabilities of entering junction or staying in fracture
RE	R	G	NF			FRV		TR	Reynolds number in frac based on apertures and av long vel
RELC	R	G	NELC			INP	ELC SET	EL	radius of influence of electrode
RHO	R	G	2		*	BDA	DER PRP SET	DH	water density at 100 degrees celsius. * 958.07, 59.812
RHOTB	R	G	30			BDA	QVS		density table
RPARM	R	G	INT			INT			see Appendix C
RPAV	R	G	1			DER		OU	average particle radius for cases without fractures
RPSD	R	G	1			DER		OU	standard dev-of particle radius
RRATF	F	G	NF			QVS	FRV MAS REL		density ratio, fracture-to-ref
RRATIN	R	G	1			QVS			density ratio, inlet-to-ref
RRATM	R	G	NN			QVS	DIA MAS MTV		density ratio, matrix-to-ref
RRATN	R	G	1			QVS	DIA FRV MTV		density ratio, native-to-ref

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
RATRB	R	G	NNRB			QVS	MAS	HT	density ratio, fracture path from node to remote boundary-to-ref
T2	R	G	1			SET	MTF FTV MTV	AL	square root of 2.0
TOL	R	G				INP	DSG		see Appendix C
TOV	R	G	2		*	BDA	FRV SET	HT	density to viscosity at 100 degrees celsius. * 36.465, 3.387
WEL	R	G	1		.25	BDA	MAR REL	RE	particle release and recovery radius of the well, absolute
AVEJJ	I	L	1			JUN		JU	jj value of frac to begin ISTYP four sect
EPAV	R	G	3			INP	SET	SY	mean separation of synthesized fracture set: vertical, horizontal, diag. upwards to right
SETRUN	L	G	1		F	BDA INP	SET	CO	enables (with NPGMX.GT.0) initializing particle calculations first time through SETUP. SETRUN = .F. must be input to run a second (new) particle move case from the same velocity field.
KIP	L	L	1			ADV		CO	controls call of marker from advan to ensure time corresponsence of hydraulics and particle movement
SOLVE	L	G	1			INP DER STA	SET MTV MAS FCN	CO	enables derivative calculation for steady state (by iteration)
SOLVEG	I	G	1			SET	MAS	DH	zeroes DH calculation if SOLVE or DYNMIC are not true
SS	R	A				NRP	BOR	MM	SS = 0 indicates Newton-Raphson has converged
TREND	I	G	NSTR			INP	SET	XY	the smallest or beginning nodal index of the fracture string

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
SYMTRY	R	G	1		4.	INP BDA	MAT	MM	ratio of complete wellbore periphery to portion included in model for particle calcs. 4. is only valid value for an infinite porous media problem with a single well. Also used in general case for all well nodes not on fractures.
SYNTH	L	G	1		.F.	INP BDA	SET	SY	flag, enabling synthesis of fracture system
SYSEED	R	G			*	INP BDA	SET	SY	eight-digit positive real number seed for fracture synthesis. * Initial value 87654321.
T	R	G	1		0.	INT BDA	ALL	CO	problem time--end of increment
TCPCUR	R	G				DER		CO	current CP time, decimal seconds
TCPMX	R	G	1		600.	INP BDA	DER	CO	terminate transient 15 CP sec before this CP time
TCRIT	R	G	1			DER	DSG		equiv. RPARAM(1), end of increment time
TCUR	R	A	1			DSG	FCN MXE		current integration time
TELCNX	R	G	1		1.E6	DER BDA	ELC	EL	electrode reporting interval, should be an integer number of DTP
TEND	R	G	1		0.	INP BDA	DER PST MXE	CO	time to end transient
TEREL	R	G	1		0.	INP BDA	ADV SET DER	RE	time at end of solute particle release
TFINAL	R	G	1		0.	BDA	SET ADV DER	HT	end of heat flow
THDIF	R	G	1		.01	INP BDA	ADV SET	HT	diffusion coefficient for heat particles
THGF	R	G	1		10.	INP BDA SET	ADV	HT	matrix GEOFAC for heat with a single well is 4. Also used in the general case for all well nodes not on fractures

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
INIT	R	G	1		0.	BDA	SET ADV DER	HT	start of heat flow
MAR	R	G	1			ADV SET		RE	next time MARKER is to be called
N	R	G	1			DER			
NOW	R	L				FTF MAR MTF REL	ENT	OU	time of reported particle event
OTPOS	I	L	1			JUN		JU	tot # of frags in parent group & possibly this section
PF	R	G	NF			DER SET		HT	fracture temperature
PFNW	R	G	NF			FCN SET		HT	current fracture temp
PFPV	R	G	NF			DER SET		HT	fracture temp at beginning of DTQ
PIN	R	G	1			INP	SET	HT	temperature of water injected at a node
PLS	R	G	1			INP	SET	HT	temperature of water injected at a line source
PM	R	G	NN			DER SET		HT	matrix temperature
PMNW	R	G	NN			FCN SET		HT	current matrix temp
PMPV	R	G	NN			DER SET		HT	matrix temp at beginning of DTQ
PN	R	G	1			INP	DER PRP SET	HT	native fluid temperature
PV	R	G	1			DER		HT	time at beginning of DTQ
RDIF	R	G	1		1.E-5	INP BDA	ADV	PM	diffusion coefficient for solute
RES	R	G	100	NDOT		SET	DER	OU	time at midpoint of class interval
RESAV	R	G	NOUT			DER ENT		OU	avg. exit time for all particles leaving this node between TSMALL and 1.5 TSMALL
RESDL	R	G	NDOT			SET	ENT	OU	class interval for particle exit histogram

TABLE A-2. VARIABLE SUMMARY
Page No. 26
06/05/89

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
TRESMN	R	G	NDOT			INP	ENT SET	OU	min time for particle exit histogram
TRESMX	R	G	NDOT			INP	SET	OU	max time for particle exit histogram
TRESSM	R	G	NOUT			DER ENT		OU	sum of exit times for this node
TRGF	R	G	NMTM		.1	INP BDA	ADV	PM	GEOFAC for solute transport
TSMALL	R	G	NOUT			DER ENT		OU	time of earliest exit at this node
TSP	R	A	1			MTV	FTV MAR MAS	MM	particle time step limit for head gradient rapidly changing with distance from injection or pumping well
TSREL	R	G	1		0.	INP BDA	SET DER ADV	RE	time at start of solute particle release sequence
TWRIT	R	G	1		1.E6	BDA SET ELC	ADV	EL	time of electrode output
TZ	R	G	1		0.	INP BDA	INT MXE	CO	initial condition of time, set prior to CONTINUE command
UNITS	I	G	1			INP	DER FRV PRP QVS SET		1-M, K, DAY, degrees Celsius, 2-FT, LB, DAY, degrees Fahrenheit, 3-specialized reservoir units
USED	R	L	1			JUN		JU	amount of flow in frac not available for current section
USEDJJ	R	L	1			JUN		JU	amount of frac exit flow not available for sect 4
V	R	A				MAC	ROO	HP	average speed in fracture
VFLAV	R	G	NF		0.	FRV BDA	JUN MAR ADV REL MAC	VF	average longitudinal velocity at fracture midpoint, pos from NOD1 to NOD2. Default value for VFLAV(1) only
VFSTOR	R	G	2	NF		FTV	MTV	VF	normal velocity in half of fracture (NOD1,NOD2) due to local matrix storage, pos out

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
FTB	R	G	2	NF		FTV MAS	JUN MAR	VF	normal velocity in half fracture (NOD1,NOD2) at PPYU = 0, positive in direction of increasing PPYU
FTT	R	G	2	NF		FTV MAS	JUN MAR	VF	normal velocity in half fracture (NOD1,NOD2) at PPYU = 1, positive in direction of increasing PPYU
ISTB	R	G	30			BDA	QVS	HT	viscosity table
MAX	R	G	NN			SET	MAR	VF	centerline fracture longitudinal velocity over mean
OLELC	R	G	NELC			SET	ELC	EL	volume measured by an electrode
PX	R	G	1			MTV	MAR FTV MAS REL SET	MM	advective velocity of matrix particle in x direction
PX2	R	G	NROW			MAS		DH	temp storage of velocity across nodal volume boundary, quadrant 2
PX3	R	G	NROW			MAS		DH	temp storage of velocity across nodal volume boundary, quadrant 3
PXDIA	R	G	1			DIA MAR MTV		MM	VPX modified to account for head in diag frac
PY	R	G	1			MTV	MAR FTV MAS REL SET	MM	advective velocity of matrix particle in y direction
PYDIA	R	G	1			DIA MAR MTV		MM	VPY modified to account for head in diagonal fracture
HERE	R	A				MAC	MAR	HP	media at end of subroutine macro particle move --1 junction, 0 fracture, 1 matrix
SY	R	G	1			INP	SET	SY	width of synthesis region
1	R	A				MAC	ROO	HP	lower bound of distance moved in fracture

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
X2	R	A				MAC	ROO	HP	upper bound of distance moved in fracture
XCOL	R	G	NCOL			INP GRD	MAS PST SET CON DER	XY	node column positions on x axis
XEND	R	G	1			GRD	MAS	XY	width of model (x direction)
XEXP	R	G	1	1.		INP BDA	GRD	XY	ratio of width of an element to previous width
XFLG	I	A	1			MAR	OSH	MM	particle location in matrix--1 if PPXU.LE..5, otherwise 3
XP	R	A	1			MTV	DIA	MM	x position in element
XPU1	R	A	1			MAR	MTF	MF MM	temporary storage of PPXU on entering an element
XTOL	R	A				MAC	ROO	HP	accuracy requirement for ROOT
XUOV	R	A				OSH	MAR	MM	allowable cell boundary overshoot in x direction
XV	R	A	1			MAR NER	MTF OSH	MF	end of frac app or crossed:-1. node 1, +1. node 2
Y	R	A				NRP	BOR	MM	trial diffusion distance into matrix
YEND	R	G	1			GRD	MAS	XY	height of model (y direction)
YEXP	R	G	1	1.		INP BDA	GRD	XY	ratio of height of an element to previous height
YFLG	I	A				MAR	OSH	MM	particle location in matrix--1 if PPYU.LE..5, otherwise 3
YP	R	A	1			MTV	DIA	M	y position in element.
YPU1	R	A	1			MTF MAR		MF MM	temporary storage of PPYU on entering an element or a fract
YROW	R	G	NROW			INP GRD	MAS PST SET CON DER	XY	node row positions on y axis

VARIABLE	TYPE	DOM AIN	SUB 1	SUB 2	INIT VALUE	DEFINED IN	USED IN	TASK	DESCRIPTION OF VARIABLE
YUOV	R	A				OSH	MAR	MM	allowable cell boundary overshoot in y direction
YV	R	A	1			NER MAR	MTF OSH	MF	side of frac app or crossed-1. PPYU+0., +1. PPYU=1.
Z	R	G	1		1.0	INP BDA	SET FRV MAR CON	3D	model thickness - unity for field models actual thickness for physical models
ZCONC	R	G	1		1.	BDA INP	SET	RE	tracer concentration in injection well
ZERO	R	G			0.	BDA	MAC BOR MXE MXD	HP	
ZIPARM	I	G				BDA			see Appendix C
ZJOB	I	G				BDA			see Appendix C
ZRPARM	R	G				BDA	ZSG		see Appendix C

TABLE A-3. DATA BASE FIELDS

Label	Description
VARIABLE	Name, 6 characters maximum. If an argument, name in called routine is given
TYPE	Real (R), integer (I), logical (L), alphanumeric (A), dimension or information only, variable does not exist (D), parameter (P)
DOMAIN	Global (G), argument(A), local (L)
SUB1	Numerical or parameter dimension
SUB2	Numerical or parameter dimension
INIT VALUE	Value preset in bdata1.F or commons.i or deltas.i at compilation
DEFINED IN	Subroutine, function, common block or input defining the variable. Acronyms are given in Table A-4. A maximum of 8 entries are made. If more entries exist the word ALL is entered
USED IN	Other subroutines (Table A-4) using the variable. Use by the defining subroutine is assumed. Maximum of 8 or ALL
TASK	A task such as junction routing (JU) may be addressed in multiple subroutines. Acronyms are given in Table A-5 Maximum of 8 or ALL.
DESCRIPTION OF VARIABLE	Lower case letters except for names of variables or subroutines. Semicolon cannot be used since it causes a jump to a new line.

TABLE A-4. SUBROUTINES, FUNCTIONS, COMMON BLOCKS, INCLUDE FILES
(Files in Cray Directory, except as noted)

Abbreviation	File Title	Functional Title	Type ^a	Common	Function
ADV	advan.F	ADVAN	S	X	Driver for particle movement
ALL		ALL	L	X	Used in many subroutines
BDA	bdata1.F	BDATA1	B	X	Block data--initial values which should not be changed
BOR	border.F	BORDER	S	X	Decides on element-to-element border crossing
BOU	bound.F	BOUND	F		Limit arg 3 between arg 1 and arg 2
COM	commons.i	COM	I		Dimensioning parameters, type, and data block assignments
CON	conc.F	CONC	S	X	Concentration in matrix and fractures
DEL	deltas.i	DEL	I		Small increments--see Table A-7
DER	deriv.F	DODERIV	S	X	Driver for transport and flow transient
DIA	diagvel.F	DIAGVEL	S	X	Head or matrix velocity near a diagonal fracture
DOF	dofcn.F	DOFCN	S	X	Calls FCN
DSG	dsgpdr.F	DSGPDR	S		Integration driver (gears directory)
ELC	elctrod.F	ELCTROD	S	X	Concentration at electrodes
ENT	entwel.F	ENTWEL	S	X	Bookkeeping for particle entering well
EXC	exchang.F	EXCHANG	S	X	Moves particle data to large core (not used)
		EXCHBLK	B		Large core storage (not used)
EXP	experfc.F	EXPERFC	F		Exp (CO** 2) * erfc (C1)
FCN	fcn.F	FCN	S	X	Driver for flow model

TABLE A-4. (continued)

Abbreviation	File Title	Functional Title	Type ^a	Common	Function
FRV	fracvel.F	FRACVEL	S	X	Fracture velocity, flow and Reynolds no
FTF	frtofr.F	FRTOFR	S	X	Fracture-to-fracture transfer
FTV	ftvel.F	FTVEL	S	X	Transverse velocity at fracture edges: storage and pressure terms
GRD	grid.F	GRID	S	X	Row and column locations
INP		INPUT	L		Designates data input at execution
INT	initial.F	DOINITL	S	X	Initialize for first case or restart
JAC	jac.F	JAC	S		Place holder only
JUN	junfun.F	JUNFUN	S	X	Particle routing through fracture junctions
LNT	lint.F	LINT	S		Linear interpolation and extrapolation
LTS	lints.F	LINTS	S		Transforms point-pair data for tabular function from point-pair to slope-intercept
MAC	macro.F	MACRO	S	X	High Peclet particle movement in fractures
MAI	main.F	FRACSL	P		Driver program
MAR	marker.F	MARKER	S	X	Particle movement
MAS	mass.F	MASS	S	X	Conservation of mass
MTF	mtofr.F	MTOFR	S	X	Matrix-to-fracture transfer
MTV	matvel.F	MATVEL	S	X	Particle velocity in matrix
MXD	mxdrv.F	MXDRV	S	X	Reports performance of steady-state integration
MXE	mxerr.F	MXERR	S	X	Reports performance of integration routine

TABLE A-4. (continued)

Abbreviation	File Title	Functional Title	Type ^a	Common	Function
NAM	namtab.F	NAMTAB	S	X	List input and output variables
NER	nerfrc.F	NERFRC	S	X	Calculates distance to nearest fracture
NOR	anorm.F	ANORM	F		Normal distribution
NRP	nraph.F	NRAPH	S		Newton-Raphson iteration, available for local iteration
OSH	oshoot.F	OSHOOT	S	X	Cell boundary overshoot
OUT		OUTPUT	L		For output only
PMT	probmt.F	PROBMT	F		Erfc (D) + Exp (A) * (Erfc[b]-Erfc[c])
PRP	props.F	PROPS	S	X	Thermal properties
PST	post.F	POST	S	X	Particle move statistics, particle location, and pressure plot
QVS	qvstp.F	QVSTP	S	X	Density and viscosity table lookup
REL	release.F	RELEASE	S	X	Particle release into model
ROO	root.F	ROOT	F		Bounded Newton-Raphson for high Peclet
SET	setup.F	SETUP	S	X	Preliminary calculations
SPA	sparse.F	SPARSE	F		Initialize IA and JA to Jacobian sparseness structure
STA	static.F	DOSTATIC	S	X	Driver for steady-state solution
URN	urn.F	URN	F		Uniform random number generator for particle movement
UR2	urn2.F	URN2	F		Uniform random number generator for fracture synthesis

TABLE A-4. (continued)

<u>Abbreviation</u>	<u>File Title</u>	<u>Functional Title</u>	<u>Type</u> ^a	<u>Common</u>	<u>Function</u>
ZFC	zfcn.F	ZFCN	S	X	Reorders arguments and calls FM
ZSG	zsgpdr.F	ZSGPDR	S		Steady-state driver (gears directory)

a. Block data (B), Function (F), Include file (I), Label only (L), Program (P), Subroutine (S).

TABLE A-5. TASKS (TAS)

Task	Description	Manual Section
3D	Three dimensional effects due to finite height (Z) of fractures in physical models	Fracture Transport Model
BC	Boundary Conditions	Boundary Conditions
CO	Control of overall program	Computational Modes
DH	Derivative of head, conservation of mass	Head and Flow Models
EL	Electrode	Electrodes
HP	High Peclet particle movement	Fracture-to-Matrix Transfer Model
HT	Heat	
JU	Junction, establish marker routing through fracture junctions	Fracture-to-Fracture Transfer Model
MF	Movement of particles in fractures	Fracture Transport Model
MM	Movement of particles in matrix	Matrix Transport Model
OU	Data output	
PT	Particle transfers between matrix and fractures	Fracture-to-Matrix Transfer Model Matrix-to-Fracture Transfer Model
RE	Release of particles	Particle Tracking
SY	Synthesis of independent sets of vertical, horizontal and diagonal fractures	Reservoir Description
TR	Turbulent (or laminar) flow regime	Head and Flow Models
VD	Diagonal fracture, matrix velocities	Matrix Transport Model
VF	Fracture velocities, longitudinal and transverse	Fracture Transport Model
VM	Matrix velocities	Matrix Transport Model
XY	Map of fracture and matrix grids	Reservoir Description

TABLE A-6. DIMENSIONING

Arrays are redimensioned by changing the parameter statement in file commons.i and recompiling using the make command.

<u>Parameter Name</u>	<u>Current Value</u>	<u>Array Size Required</u>	<u>Significance</u>
MNCOL	18	NCOL	Columns
MNROW	12	NROW+1	Rows
MNN	MNCOL*MNROW	NN	Nodes
MNF	120	NF	Fractures
MNFRM	10	NFRM	Fracture Materials
MNMTM	10	NMTM	Matrix Materials
MNPMX	1000	NPMX	Particles per Group
MNPGMX	1	NPGMX	Particle Groups (only 1 at present)
MNNRB	40	NNRB	Nodes Connected to Remote Boundaries
MNSTR	40	NSTR	Fracture Strings
MNELC	10	NELC	Electrodes
MNNSY	144	NNSY	Fracture Synthesis Nodes
MNOUT	11	NOUT	Particle Exit Nodes
MNDOT	1	NDOT	Particle Exit Histograms
MNFSET	3	3	Synthesized Fracture Sets
MNNLS	MNROW	NNLS	Line Source Nodes

TABLE A-7. DELTAS

<u>Routine</u>	<u>Delta</u>	<u>Line No.</u>	<u>Type</u>	<u>Comments</u>
ADVAN	Delta6	19	1	
	Delt90	34	4	
BORDER	Delta7	115	2	
DERIV	Delta9	94	1	
		167	1	
ELCTROD	Delta6	42	1	
EXPERFC	Delt15	37	3	This could be changed to a parameter.
FRACVEL	Delta6	49	3	
FRTOFR	Delta8	60	2	
		61	2	
FTVEL	Delta6	171	2	Put psuedo particle just outside fracture. 1×10^{-4} in other places.
	Delta5	185	5	Test for zero+Delta6.
MACRO	Delt14	110	5	If Kappa.LT.Delt14 MERFCI will fail. ERFCI may also fail.
	Delta5	147	3	Set convergence to Delta5 * length of fracture.
MARKER	Delt12	113	4	If ADVECT/ (LXDIF+DISPX).LT. Delta4, make DTTOX more accurate.
	Delta4	168	5	
		178		
	Delta8	235,236	2	Transfer if to close to edge.
		409	2	
		606	2	
	Delta6	280,282	2	Could change to Delta8.
		289,290		
		345,350		
		356,357		
		567	2	Transfer if to close to end.
		607		

TABLE A-7. (continued)

<u>Routine</u>	<u>Delta</u>	<u>Line No.</u>	<u>Type</u>	<u>Comments</u>
	Delta7	414,486	2	Particle just inside FRAC.
		415	4	
	Delta9	600	2	
MATVEL	Delta30	356,357	4	
	Delta8	364,365	4	
	Delta9	390	4	
MTOFR	Delta6	23,24,30	2	
		31,32,33		
NERFRC	Delta5	87	2	
NRAPH	Delta6	49	3	
OSHOOT	Delta7	70,80	2	
		138,148		
RELEASE	Delta8	80,81	2	

Type Code:

1. Deltas introduced to avoid possible timing problems due to round off errors.
2. Deltas move particles away from boundaries between blocks or blocks and fractures. Concern is about both round off errors and possible strange behavior as the distance to a boundary goes to zero.
3. Convergence criteria
4. Avoids divide by zero failures.
5. Other; see comment.

APPENDIX B
INPUT/OUTPUT SUMMARY

CONTENTS

Table B-1. Conventions	B-4
Table B-2. Input Matrix	B-5
Table B-3. Input area details	B-6
Table B-4. Output details	B-10
Table B-5. Error messages	B-12

APPENDIX B
INPUT/OUTPUT SUMMARY

Options not used need not be considered--the default flag or integer will apply.

The segloader option which presets common to indefinite is used to guarantee that a missed input cannot be ignored.

Unit systems are:

- UNITS = 1 meter, kg, day, °C
= 2 ft, lb, day, °F
= 3 A subset of UNITS=2 in which the user is allowed to enter several specific inputs in reservoir units. The specific variables are:

<u>Variable</u>	<u>Reservoir Units</u>	<u>Description</u>
MTCONX MTCONY	Millidarcy	Permeability
MTSS	psi/lbm	System compressibility
QQEXT QQINJ FLUX	lbm/hr	Mass flux at 212°F
HZ HRB HL	psi	Prescribed pressure
QQT0HL	lbm/hr/psi	Flow-to-head ratio at nodes leaking to a parallel aquifer.

All other inputs must be the same as for UNITS = 2. The output heads will be in psi. All other outputs are in the UNITS = 2 system.

Except as described above for UNITS=3, the user must maintain the dimension system selected by the UNITS value throughout the simulation.

TABLE B-1. CONVENTIONS

PPHOST positive when particle is in a fracture
negative when particle is in a matrix cell

TABLE B-2. INPUT MATRIX.

Thermal Conditions Flow Transport	1 Isothermal Steady-State	2 Isothermal Transient	3 Isothermal Existing s.s. Solute	4 Isothermal Transient +Solute	5 Non-Isothermal Transient +Heat
number processes	1	1	1	2	2
coupled	-	-	-	no	yes
dimensions--Table A-6	x	x	x	x	x
grid*	x	x	x	x	x
matrix materials*	x	x	x	x	x
matrix flow props*	x	x	x	x	x
matrix/water trans props*			x	x	x
fractures*	opt	opt	opt	opt	opt
flow boundary cond*	x	x	x	x	x
initial conditions*	guess	uniform, 1,2	1,2	uniform, 1,2	1
particles*			x	x	x
units = 1	x	x	x	x	x
tpn = 200.	x	x	x	x	x
dtq = .1		x		x	x
dtqexp = 1.0		x		x	x
tend = 4		x	x	x	x
tcpmx= 300.		x	x	x	x
numcol = 5	x	x	x	x	x
plocpt =			true/false	true/false	true/false
headpt =	true/false	true/false		true/false	true/false
concpt =			true/false	true/false	
debug**	x	x	x	x	x
sourced data	@ZERO***	@GEARS****		@GEARS	@GEARS
solve =	true	false	false	false	false
dynmic =	false	true	false	true	true
static	x				
start		x	x	x	x
final output**	@PRINTM	@PRINTM	@PRINTM	@PRINTM	@PRINTM

* See Table B-3.

** See Table B-4.

*** See Table C-1.

**** See Table C-2.

TABLE B-3. INPUT AREA DETAILS.

```

#grid
nrow = 3
ncol = 3
psix = 0.          # x axis angle with horizontal
psiy = 1.5708      # y axis angle with vertical

nn = nrow * ncol

a.  igrid = 0.      # input xcol and yrow, rectangular or square cells
    xcol(1) = 0.
    xcol(2) = 5.
    xcol(3) = 10.
    yrow(1) = 0.
    yrow(2) = 5.
    yrow(3) = 10.

b.  igrid = 1.      # expand in x and y from origin
    dxz = 5.
    dyz = 5.
    xexp = 1.
    yexp = 1.

c.  igrid = 2.      # expand in x & y, in both directions, from origin at NODIN
    nodin = 5       # must be central node
    dxz = 5.
    dyz = 5.
    xexp = 1.
    yexp = 1

```

```

# matrix materials
nmtm = 2
immat(1:3) = 1      # input required for all nodes
immat(4:9) = 2

```

```

# matrix flow properties
mtconx(1:2) = .864e-5
mtcony(1:2) = .0864
mtss(1:2) = 1.E-5
mtpor(1:2) = .001

```

```

# matrix/water transport properties

```

<u>both</u>	<u>solute</u>	<u>heat</u>	<u>description</u>
mtdsyt(1:2) = 5.			
mtdsynt(1) = 1.			
mtdsynt(2) = 2.			
	trdif = 1.E-6	thdif = .01274	# diffusion
		cpw = 4186.	# specific heat
	trgf = .1		# tortuosity
		mtcp(1:2) = 1100.	# rock specific heat
		mtrho(1:2) = 2000.	# rock density
		mttcon(1:2) = .2E+6	# rock thermal cond.

TABLE B-3. (continued)

fractures

```

frss = 0.
f3d = false
frfac = .072
iqcase = 1
# field model
# turbulent flow
# laminar

```

a. nf = 4 # specify discrete fractures

```

frend(1,1) = 1
frend(2,1) = 2
frend(1,2) = 2
frend(2,2) = 3
frend(1,3) = 5
frend(2,3) = 8
frend(1,4) = 1
frend(2,4) = 5
b(1:4) = .002

```

b. nf = 4 # specify fracture strings

```

nstr = 3
nfistr(1) = 2
nfistr(2:3) = 1
psistr(1) = 1
psistr(2) = 3
psistr(3) = 2
strend(1) = 1
strend(2) = 5
strend(3) = 1
bstr(1:3) = .002
# vertical
# horizontal
# rising to right

```

c. synth = true # synthesize fractures (square grid only)

```

syseed = 55555555.
htsy = 20.
wsy = 20.
icol1 = 1
icol2 = 3
jrow1 = 1
jrow2 = 3
lcut1(1:3) = 2.5
lcuth(1:3) = 20.
sepav(1) = 2.
sepav(2) = 5.
sepav(3) = 10.
lfac(1:3) = 4.
bav(1:3) = .002
bhd(1:25) = 0.
bvd(1:25) = 0.
bvd(1:2) = .020
bvd(11:12) = .020
bdd(1:25) = 0.

```

synthesis region

model region

set properties

vertical fractures

horizontal fractures

diagonal upwards to the right fractures

deterministic fracture in synthesis region at node1

```

20. . . . .
15. . . . .
10. . . . .
5. . . . .
0. . . . .
0. 5. 10. 15. 20.

```

TABLE B-3. (continued)

# flow boundary conditions	
a.	# impermeable, no input required
b.	# fixed flow at specified nodes
nodin = 3	# may not be combined, at a node, with d or e or f
qqext(1:9) = 0.	# injection node, a second injection node must be listed in nodot (see matvel)
qqext(3) = 5.	
rwei = .25	# injection-radius, absolute
c.	# transient variation at nodin
nodin = 3	# may not be combined, at a node, with d or e or f
flux(1) = -1.	# injection node, a second inj. node must be in nodot
flux(2) = 0.	# values are data pairs (time, rate)
flux(3) = 0.	
flux(4) = 0.	
flux(5) = .1	
flux(6) = 5.	
flux(7) = 1.	
flux(8) = 5.	
nbkpt = 4	# number of point pairs
qexttr = true	
qqinj = 5.	# required for transport simulation
rwei = .25	# injection radius, absolute
d. freeze(1:8) = false	# fixed head (steady state or transient simulation)
freeze(9) = true	# may not be combined with g or, at a node, with b or c or e or f
	# must be specified at at least one node in steady state if e or f or g is not used
e.	# remote boundary
nnrb = 2	# may not be combined with g or, at a node, with b or c or d
inrb(1) = 4	
inrb(2) = 6	
lftorb(1:2) = 100.	
lmtorb(1:2) = 100.	
hrb(1:2) = 0.	
frdeb(1:4) = true	
f.	# leaky boundary
qtohl(1:9) = 2.	# may not be combined with g, or at a node, with b or c or d
hl(1:9) = 0.	
g.	# infinite porous media
permbd = true	# required for transport simulation
symtry = 4	# must specify b or c at node 1

# initial conditions	
hz(1:9) = 10.	
tz = 0.	

TABLE B-3. (continued)

particles

	<u>both</u>	<u>solute</u>	<u>heat</u>	<u>description</u>
	compres = true			# compress arrays
	npgmx = 1			# no. groups
	npmx = 1000			# no. particles
	nseed = 87654321.			
#a.	point source			
	nodin = 5			
	rwel = .01			
		zconc = 1.	tpin = 80.	# absolute distance
				# inlet
#b.	line source			
	nnls = 2			# no nodes on line source
	inls (1) = 2			# nodes at ll corner
	inls (2) = 5			# hor or vert only
	olsu = .99			# offset, normalized positive only
		zconc = 1.	tpls = 80.	# inlet
# exits, node concentration, temperature, npout, tresav				
	nout = 3			
	iout (1) = 1			
	iout (2) = 8			
	iout (3) = 7			
	extbnd = false			# particle may not diffuse out of model
# production wells, residence time histogram				
	rwel = .25			# capture radius, absolute
	ndot = 2			
	nodot (1) = 1			# must be mentioned to allow particle exit
	tresmn (1:2) = .01			
	tresmx (1:2) = 4.0			
# internal monitoring				
	nelc = 2			# no of electrodes
	pelhst (1) = 2			# fracture 2
	pelhst (2) = -3			# element 3
	relc (1:2) = 2.5			# radius, absolute
	pelxyu (1,1:2) = .5			# x position
	pelxyu (2,2) = .5			# y position
# timing				
	telcnx = .05			# reporting interval
		tsrel = 0.	tinit = 0.	
		terel = .2	tfinal = 4.	
	dtp = .1			# time step
	dtpexp = 1.			# time step expansion
# type		heatp = false	heatp = true	
# tracking, cmpres set false internally				
	iprint (1) = 3			# release, exit, transfer
	iprint (2) = 68			# particle no
# mixing				
	imix = 1			# diffusion in junctions
# temperature smoothing			nlag	# no. dtstp in frac. temp

TABLE B-4. OUTPUT DETAILS

1. Input echo

The right hand side of all replacement statements in the input file is printed for documentation of run inputs.

2. Integration performance

Details on timing and critical nodes is printed for all flow transients.

3. Steady state iteration

A summary of the iteration process is printed for steady state flow solutions.

4. Particle move summary

A labeled summary of all particle moves is printed for any run in which particles have been released.

5. Error messages

Messages are described in Table B-5.

Optional Discrete Output on the Standard Output File

1. debug t h dh qf
undebug qf

Add to or remove global variables (in commons.i) from the debug list which will be printed at dtq intervals for a flow transient or dtp intervals for a particle move. The debug list will also be printed after a steady state calculation. Variables must be identified in namtab.F by a call to symbol. Any addition to namtab.F will require a make (recompile). The complete array must be printed.

2. @ PRINTM

echo h; h(1:nn)

Sourcing the echo statements in PRINTM causes a listing at the point in the job stream where PRINTM is sourced. Repetitive output is avoided and portions of an array can be printed. The variable must again be listed in namtab.F. A sample PRINTM file is given as follows:

Jan 29 16:34 1989 PRINTM Page 1

```
1  echo T; t(1:1)      # Show time--ranges are always printed.
2  echo H; h(1:nn)     # Show active H values.
3  echo DH; dh(1:nn)   # Show derivative.
4  # @stdin           # Recursively source stdin--look around.
```

TABLE B-4. (continued)

3. `iprint(1) = 3`
`iprint(2) = 35`
Traces the history of particle 35. `iprint(1) = 2` gives only release and exit information. `iprint(1) = 3` adds all transfers between various fractures and matrix elements. The `cmpres` variable is set false to retain the particle identity.
4. `jprint = 1`
Provides details of the fracture-to-fracture transfer function.
5. `mvtst = true`
Prints code development output on particle movement. If input at execution time will print data on all particles. Commented statements in `marker.F` may be modified to execute on specified particles.
6. `mstst = true`
Prints code development output on matrix velocities except when `solve` is true.
7. Fortran debug
Following a numerical failure, a detailed listing of local and global variables may be obtained using the Fortran debug statement described in Appendix F.

Optional Plots

1. `plocpt = true`
Prints the end of run x, y location of every particle in the system on plot.7 in a format suitable for post-processing.
 2. `headpt = true`
Prints the end-of-run head at all the node locations on plot.7 in a format suitable for post-processing.
 3. `concpt = true`
Prints end-of-run concentration, in individual matrix elements and fractures on plot.7 in a format suitable for post processing.
 4. `nelc not equal 0`
Writes the electrode concentration-time histories to plot.10.
 5. `ndot not equal 0`
Writes a residence time histogram for each exit node `ndot` in arrays `tres (100,ndot)`, `ntint (100,ndot)`.
 6. Particle recovery, `ipot` and `conwel` histories are generated by subroutine `spare` in an obsolete form.
-

TABLE B-5. ERROR MESSAGES

Message:

NRAPH FAILED IN BORDER

Meaning:

NRAPH failed to converge in solving for particle location in matrix.

Solution:

First: Try problem again with new NSEED.

Second: Adjust the initial guess of y by making a perturbation of constant in Line 10.

Message:

BORDER FAIL WITH NEG Y2

Meaning:

Non-physical solution given by border.

Solution:

First: Try problem again with new NSEED.

Second: Change 1.11 in Line 10 to 1.101.
Change 2.01 in next line to 2.001.

Message:

ERROR IN LINE 508 FRTOFR

Meaning:

No exit fracture was found for the particle node location. QOTCM is given in the output file just before last DEBUG dump. Probable cause is a poor convergence of hydraulic solution.

Solution:

Check QOTCM for viable outlet. Check for convergence failure in hydraulic solution. Compare heads against expected values. It is possible that in a transient simulation, a node is nonconservative. If so, try adjusting hydraulic error controls, gears_atol and gears_xtol.

Message:

FAILURE IN MACRO

Meaning:

Particle move upstream in the Macro Monte Carlo Routine. This indicates a lack of precision in the EXPERFC routine or in IMSL MERFCI routine.

Solution:

Increase accuracy of EXPERFC by changing tolerances identified in the routine. If MERFCI is the problem, one can change the error from a fatal error to a repeat of the calculation with a new random number.

TABLE B-5. (continued)

Message:

ROOT SOLVER FAILED IN MACRO

Meaning:

An unacceptable level of failures of the root solver have occurred.

Solution:

Failures of root are normally due to limiting the distance moved to $0.999 \cdot DTREM \cdot V$, which eliminates a singularity in the solution. It is expected that 0.1% of the particles will fail. A variable FAILTOL controls allowed failures. Currently, FAILTOL is set to 3. The program should fail 10^{-9} per particle move, a small but not impossible likelihood. One can reduce the likelihood of failure by changing 0.999 to be larger but still less than 1, or increasing FAILTOL.

Message:

DTUSED = 0 IN FRAC MOVE

Meaning:

The calculated time taken to move a particle is zero and hence the particle will not move. Without the stop, the program would land in an infinite loop.

Solution:

Calculated times to reach x, y, z boundaries are inserted into the output stream. The diffusion controlled time limit is also printed along with the particle's location and identity. Redo problem using MVTST and particle identity.

Message:

FLOW DIRECTION ERROR AT LINE 19 OF RELEASE

Meaning:

Particle is being injected into a fracture that is flowing towards the node.

Solution:

Condition is caused by either improper input specifications or oscillations in the hydraulic response. Hydraulic response may be improved by either smoothing the boundary condition perturbations or reducing the error criteria.

TABLE B-5. (continued)

Message:

NUMBER OF FRACTURES •NE• NF

Meaning:

Number of fractures entered via fracture strings does not agree with specified number of fractures.

Solution:

Check input.

Message:

ILLEGAL FRACTURE DEFINITION

Meaning:

The ends of a fracture do not connect adjoining nodes.

Solution:

The offending definition is printed in the output stream.

Message:

IMPROPER FRACTURE STRING DEFINITION

Solution:

Check definition of offending string as printed in the output string.

APPENDIX C
NUMERICAL ROUTINE INPUT/OUTPUT

CONTENTS

Table C-1. Steady State Solution Input/Output	C-4
Table C-2. Integration Input/Output	C-7

APPENDIX C
NUMERICAL ROUTINE INPUT/OUTPUT

This appendix describes the inputs and the available outputs for the steady-state and integration routines. The information is presented as listings of the sourced files ZERO and GEARS. Each is the header of a driver subroutine, with input values inserted immediately after their definitions. Sourcing (insertion) into the input data file is accomplished by giving the statement @ZERO or the statement @GEARS.

TABLE C-1. STEADY-STATE SOLUTION INPUT/OUTPUT

Jan 29 16:34 1989 ZERO Page 1

```

1
2 # ZSGPDR parameter definitions.
3 #
4 # echo "Sourcing file ZERO."
5 #
6 # ...1... FUNCTION / PURPOSE .....
7 #
8 #     CALLS THE SUBROUTINES REQUIRED TO FIND A ZERO OF A SYSTEM OF
9 #     N NONLINEAR FUNCTIONS IN N VARIABLES WITH A SPARSE JACOBIAN
10 #
11 #             FVEC(X) = 0
12 #
13 # ...2... ARGUMENTS / PARAMETERS / CONSTANTS .....
14 #
15 #     ..A.. INPUT ARGUMENTS .....
16 #
17 #     FCN [E] - USER SUPPLIED SUBROUTINE WHICH CALCULATES THE
18 #     FUNCTION VECTOR FVEC; MUST BE DECLARED IN AN EXTERNAL
19 #     STATEMENT IN THE USER CALLING PROGRAM
20 #
21 #     N [I:1] - NUMBER OF FUNCTIONS AND VARIABLES
22 #
23 #     X [R:N] - INITIAL ESTIMATE OF THE SOLUTION VECTOR
24 #
25 #     JAC [E] - USER SUPPLIED SUBROUTINE WHICH CALCULATES THE
26 #     JACOBIAN COLUMNWISE; MUST BE DECLARED EXTERNAL
27 #     IN THE USER CALLING PROGRAM. JAC IS CALLED ONLY IF
28 #     IJAC = 1; OTHERWISE A DUMMY ROUTINE MAY BE SUBSTITUTED
29 #
30 #     IA [I:N+1] - POINTERS TO DELIMIT COLS IN JA, I.E. IA(J)
31 #     POINTS TO THE START OF THE JTH COLUMN IN JA; REQUIRED
32 #     ONLY IF MOSS=3, OTHERWISE UNUSED
33 #
34 #     JA [I:NZA] - ROW INDICES OF NONZERO ELEMENTS OF THE JACOBIAN;
35 #     NZA = IA(N+1)-1; REQUIRED IF MOSS=3, OTHERWISE UNUSED
36 #
37 #     RPARM [R:4] - REAL PARAMETERS
38 #         (1)/XTOL - X CONVERGENCE PARAMETER; TERMINATION OCCURS
39 #         WHEN THE ALGORITHM ESTIMATES THAT THE RELATIVE
40 #         ERROR BETWEEN X AND THE SOLUTION IS AT MOST XTOL;
41 #         DEFAULT IS MIN (SQRT(URound), 1.E-5)
42 zero_xtol = 0
43 #         (2)/EPSFCN - USED IN DETERMINING A SUITABLE STEP LENGTH
44 #         LENGTH FOR THE FORWARD-DIFFERENCE JACOBIAN APPROX-
45 #         IMATION. RELATIVE ERRORS IN THE FUNCTIONS ARE
46 #         ASSUMED TO BE OF THE ORDER MAX (EPSFCN, UROUND).
47 #         USED ONLY IF IJAC = 2; DEFAULT IS ZERO.
48 zero_epsfcn = 1.
49 #         (3)/FACTOR - USED IN DETERMINING THE INITIAL STEP BOUND;
50 #         MUST BE .GT. ZERO. IN MOST CASES FACTOR SHOULD LIE
51 #         IN THE INTERVAL (.1,100.); DEFAULT IS 100.
52 zero_factor = 0
53 #         (4)/THRESH - NONNEGATIVE THRESHOLD VALUE FOR JACOBIAN
54 #         ELEMENTS
55 zero_thresh = 1.e-15
56 #

```

Jan 29 16:34 1989 ZERO Page 2

```

57 # IPARM [I:8] - INTEGER PARAMETERS
58 # (1)/LWK - LENGTH OF THE WORK ARRAY WK; MUST BE AT LEAST
59 # 5*N + 11*N + 4*NZA, IF IJAC=1
60 # + 13*N + 4*NZA, IF IJAC=2
61 # THIS LENGTH CANNOT BE PREDICTED IN GENERAL. IT
62 # DEPENDS ON THE SPARSITY STRUCTURE OF THE PROBLEM.
63 # SOME EXPERIMENTATION MAY BE REQUIRED.
64 # (3)/MAXFEV - MAXIMUM NUMBER OF FUNCTION EVALUATIONS
65 # ALLOWED; DEFAULT IS 100*(N+1)
66 zero_maxfev = 0
67 #
68 # WK [R:LWK] - WORKING STORAGE;
69 # (WK AND IWK SHOULD BE EQUIVALENCED)
70 #
71 # IWK [R:LRATIO*LWK] - WORKING STORAGE
72 # (IWK AND WK SHOULD BE EQUIVALENCED)
73 #
74 # JOB [I:3] - JOB PATH PARAMETERS
75 # (1)/IJAC - JACOBIAN APPROXIMATION PARAMETER
76 # = 1, JACOBIAN IS EVALUATED IN THE USER SUPPLIED
77 # SUBROUTINE JAC
78 # = 2, JACOBIAN IS EVALUATED USING A FINITE
79 # DIFFERENCE ALGORITHM. JAC IS NOT REFERENCED
80 zero_ijac = 2
81 # (2)/MODE - TYPE OF SCALING
82 # = 1, NO VARIABLE SCALING
83 # = 2, USER SUPPLIED DIAGONAL SCALING MATRIX IN THE
84 # FIRST N LOCATIONS OF WK
85 # = 3, INTERNAL VARIABLE SCALING
86 zero_mode = 3
87 # (3)/MOSS - METHOD OF OBTAINING SPARSENESS STRUCTURE
88 # = 1, USER SUPPLIED JAC; STRUCTURE WILL BE OBTAINED
89 # FROM N INITIAL CALLS TO JAC
90 # = 2, STRUCTURE WILL BE OBTAINED FROM N+1 INITIAL
91 # CALL TO FCN
92 # = 3, USER SUPPLIED IA AND JA
93 zero_moss = 2
94 #
95 # ..B.. OUTPUT ARGUMENTS .....
96 #
97 # X - FINAL ESTIMATE OF THE SOLUTION VECTOR
98 #
99 # FVEC [R:N] - FUNCTION VALUES AT THE OUTPUT VALUE OF X
100 #
101 # IPARM - INTEGER PARAMETERS
102 # (2)/LREQ - LENGTH REQUIRED FOR WK
103 # (4)/NFEV - NUMBER OF FCN EVALUATIONS (NOT INCLUDING
104 # THOSE TO COMPUTE THE FINITE DIFFERENCE JACOBIAN,
105 # IF REQUIRED)
106 # (5)/NJEV - NUMBER OF JACOBIAN APPROXIMATIONS REQUIRED,
107 # EITHER USER SUPPLIED OR FORWARD DIFFERENCING
108 # (6)/ITER - NUMBER OF ITERATIONS REQUIRED
109 # (7)/NZA - NUMBER OF NONZEROS IN THE JACOBIAN
110 # (8)/NGP - NUMBER OF GROUPS FOR F.D. JACOBIAN
111 #
112 # INFO [I:1] - ERROR CONDITION / MESSAGE CODE

```

TABLE C-1. (continued)

Jan 29 16:34 1989 ZERO Page 3

```

113 #
114 # =0, NORMAL COMPLETION
115 #
116 # << TYPE 3 - TERMINAL >>
117 #
118 # =31, MAXIMUM NUMBER OF FUNCTION EVALUATIONS HAS
119 # BEEN REACHED
120 # =32, XTOL IS TOO SMALL. NO FURTHER IMPROVEMENT IN
121 # THE APPROXIMATE SOLUTION X IS POSSIBLE
122 # =33, JACOBIAN IS SINGULAR AT THE CURRENT VALUE OF X
123 # =34, ITERATION IS NOT MAKING GOOD PROGRESS, AS
124 # MEASURED BY THE IMPROVEMENT FROM THE LAST
125 # FIVE JACOBIAN EVALUATIONS
126 # =35, ITERATION IS NOT MAKING GOOD PROGRESS, AS
127 # MEASURED BY THE IMPROVEMENT FROM THE LAST
128 # TEN ITERATIONS
129 #
130 # << TYPE 4 - FATAL >>
131 #
132 # =40, ILLEGAL INPUT, (I1)/N
133 # =41, (R1)/RPARM((I1)) IS INCONSISTENT
134 # =42, (I2)/IPARM((I1)) IS INCONSISTENT
135 # =43, (I2)/JOB((I1)) IS ILLEGAL
136 # =44, (R1)/DIAG((I1)) IS ILLEGAL
137 # =46, ERROR FROM SUBSIDIARY ROUTINE
138 # I1=1, GIORDR: I2=ERROR FROM GIORDR, INSUFFICIENT
139 # WORKSPACE
140 # I1=2, LSGPDR (SYMBOLIC FACTORIZATION): I2=ERROR
141 # I1=3, LSGPDR (NUMERIC FACTORIZATION THROUGH ZSGPSL)
142 # I2=ERROR
143 #
144 # ..D.. REFERENCES .....
145 #
146 # (1) J.J. MORE, B.S. GARROW AND K.E. HILLSTROM, USER GUIDE
147 # FOR MINPACK-1, ANL-80-74, ARGONNE NATIONAL LABORATORY,
148 # 1980
149 #
150 # (2) J.E. DENNIS, JR. AND E.S. MARWIL, "DIRECT SECANT UPDATES
151 # OF MATRIX FACTORIZATIONS", TO APPEAR IN MATH. COMP.
152 #
153 # (3) E.S. MARWIL, "SOME NUMERICAL RESULTS USING DIRECT SECANT
154 # UPDATES OF MATRIX FACTORIZATIONS", SUBMITTED TO ACM TRANS.
155 # MATH. SOFTW.
156 #
157 # ..E.. HISTORY .....
158 #
159 # ORIGIN - ZSGEDR - MINERVA
160 # MODIFIED - DATA STRUCTURE - E.S. MARWIL, 28 MAY 82
161 #
162 # Comments copied to fracsl script file Aug 25, 1987 by
163 # J S Albano and T M Clemon.
164 #
165 # echo "Done with file ZERO."

```

TABLE C-2. INTEGRATION INPUT/OUTPUT

Jan 29 16:34 1989 GEARS Page 1

```

1
2 # Header information copied from MINERVA library DSGPDR routine
3 # header by J S Albano on Aug 17, 1987.
4 #
5 # echo "Sourcing file GEARS."
6 #
7 #         RTOL [R:1/N] - RELATIVE TOLERANCE PARAMETER(S); ALLOWS
8 #         INDIVIDUAL ERROR CONTROL OF THE COMPONENTS OF Y
9 #         (SEE ITOL)
10 gears_rtol (1:nn) = 1e-3
11 #
12 #         ATOL [R:1/N] - ABSOLUTE TOLERANCE PARAMETER(S); ALLOWS
13 #         INDIVIDUAL ERROR CONTROL OF THE COMPONENTS OF Y
14 #         (SEE ITOL)
15 gears_atol (1:nn) = 1e-5
16 #
17 #         RPARM [R:10] - REAL PARAMETERS
18 #         (1)/TCRIT - CRITICAL VALUE OF T WHICH THE SOLVER IS
19 #         NOT TO OVERSHOOT (REQUIRED IF ITSK = 4 OR 5)
20 gears_tcrit = 0.0
21 #         * (2)/H0 - INITIAL STEP SIZE IN T, DEFAULT IS COMPUTED
22 #         INTERNALLY
23 gears_h0 = 0.0
24 #         (3) - (UNUSED)
25 #         * (4)/HMAX - MAXIMUM STEP SIZE ALLOWED, DEFAULT IS INFINITE
26 gears_hmax = 0.0
27 #         * (5)/HMIN - MINIMUM STEP SIZE ALLOWED, DEFAULT IS ZERO
28 gears_hmin = 0.0
29 #
30 #         IPARM [I:14] - INTEGER PARAMETERS
31 #         (1)/IY - MAXIMUM NUMBER OF COLUMNS TO BE USED IN THE
32 #         NORDSEICK ARRAY, DEFAULT IS 13 IF METH = 1, AND
33 #         6 IF METH = 2 ON THE FIRST CALL (USED FOR INPUT
34 #         ONLY ON THE FIRST CALL)
35 gears_iy = 0
36 # gears_iy = 2*NROW + 3
37 #         (2)/LWK - LENGTH OF THE WORK ARRAY WK; MUST BE AT LEAST
38 #         (IY+3)*N + 11*N + 4*NZA, IF MITR = 1
39 #         + 13*N + 4*NZA, IF MITR = 2
40 #         + N, IF MITR = 3
41 #         + 0, IF MITR = 4
42 #         FOR MITR = 1 OR 2, THIS LENGTH CANNOT BE PREDICTED
43 #         IN GENERAL. IT DEPENDS ON THE SPARSITY STRUCTURE OF
44 #         THE PROBLEM. SOME EXPERIMENTATION MAY BE REQUIRED.
45 gears_lwk = lwk
46 #         * (4)/MXSTEP - MAXIMUM NUMBER OF (INTERNALLY DEFINED) STEPS
47 #         ALLOWED DURING ONE CALL TO THE SOLVER, DEFAULT IS 500
48 gears_mxstep = 0
49 #         * (5)/MXHNIL - MAXIMUM NUMBER OF MESSAGE CODE #11 TO BE
50 #         PRINTED, DEFAULT IS 10
51 gears_mxhnsl = 10
52 #         * (6)/MXORD - MAXIMUM ORDER TO BE ALLOWED, DEFAULT IS IY-1
53 gears_mxord = 2*NROW + 3
54 #
55 #         JOB [I:6] - JOB PATH PARAMETERS
56 #         (1)/ITSK - TASK INDICATOR

```

TABLE C-2. (continued)

Jan 29 16:34 1989 GEARs Page 2

```

57 #           = 1, NORMAL COMPUTATION OF OUTPUT VALUES OF Y(T)
58 #           AT T = TN, BY OVERSHOOTING AND INTERPOLATING
59 #           = 2, TAKE SINGLE STEP AND RETURN (TN IGNORED)
60 #           = 3, PROCEED TO THE FIRST INTERNAL MESH POINT AT OR
61 #           BEYOND T = TN AND RETURN
62 #           = 4, NORMAL COMPUTATION OF OUTPUT VALUES OF Y(T)
63 #           AT T = TN, BUT WITHOUT OVERSHOOTING T = TCRIT
64 #           = 5, TAKE SINGLE STEP AND RETURN, BUT WITHOUT
65 #           OVERSHOOTING T = TCRIT (TN IGNORED)
66 gears_itsk = 4
67 #           (2)/INDX - START/RESTART INDICATOR
68 #           (NEED NOT BE RESET FOR NORMAL CONTINUATION)
69 #           = 1, FIRST CALL: NEW PROBLEM
70 #           = 2, SUBSEQUENT CALL: CONTINUE THE INTEGRATION
71 #           = 3, SUBSEQUENT CALL: CHANGE OF N, METH, MITR, MOSS,
72 #           IA, JA, TOLERANCES OR PARAMETERS
73 #           = 4, SUBSEQUENT CALL: CHANGE OF H ONLY
74 gears_indx = 1
75 #           (3)/MITR - ITERATION METHOD INDICATOR (USED FOR INPUT
76 #           ONLY ON FIRST CALL UNLESS INDX = 3)
77 #           = 1, CHORD METHOD: ANALYTIC JACOBIAN
78 #           = 2, CHORD METHOD: FINITE DIFFERENCE JACOBIAN
79 #           = 3, CHORD METHOD: DIAGONAL JACOBIAN APPROXIMATION
80 #           (BASED ON A DIRECTIONAL DERIVATIVE)
81 #           = 4, FUNCTIONAL ITERATION
82 gears_mitr = 2
83 #           (4)/METH - BASIC METHOD INDICATOR (USED FOR INPUT
84 #           ONLY ON FIRST CALL UNLESS INDX = 3)
85 #           = 1, ADAMS METHODS
86 #           = 2, STIFF METHODS OF GEAR OR
87 #           BACKWARD DIFFERENTIATION FORMULAS (BDF)
88 gears_meth = 2
89 #           (5)/ITOL - TYPE OF ERROR CONTROL FOR SINGLE STEP ERROR
90 #           ESTIMATES IN THE WEIGHTED ROOT-MEAN-SQUARE NORM;
91 #           THE WEIGHTING VECTOR MUST HAVE POSITIVE COMPONENTS
92 #           = 1, EWT(I) = RTOL * ABS(Y(I)) + ATOL
93 #           = 2, EWT(I) = RTOL * ABS(Y(I)) + ATOL(I)
94 #           = 3, EWT(I) = RTOL(I) * ABS(Y(I)) + ATOL
95 #           = 4, EWT(I) = RTOL(I) * ABS(Y(I)) + ATOL(I)
96 gears_itol = 1
97 #           (6)/MOSS - METHOD OF OBTAINING SPARSENESS STRUCTURE
98 #           = 1, USER SUPPLIED JAC; STRUCTURE WILL BE OBTAINED
99 #           FROM N INITIAL CALLS TO JAC
100 #           = 2, STRUCTURE WILL BE OBTAINED FROM N+1 INITIAL
101 #           CALLS TO FCN
102 #           = 3, USER SUPPLIED IA AND JA
103 gears_moss = 3
104 #
105 #           ..A..  USAGE  .....
106 #
107 #           (1) CALL ONCE FOR EACH OUTPUT VALUE OF T.  AFTER THE INITIAL
108 #           CALL, IF A NORMAL RETURN OCCURRED (INFO=0) AND A NORMAL CON-
109 #           TINUATION IS DESIRED, SIMPLY RESET TN AND CALL AGAIN.  ALL
110 #           OTHER PARAMETERS WILL BE READY FOR THE NEXT CALL.  A CHANGE
111 #           OF PARAMETERS WITH INDX = 3 CAN BE MADE AFTER EITHER A
112 #           SUCCESSFUL OR AN UNSUCCESSFUL RETURN.

```


Jan 29 16:34 1989 GEARs Page 3

```

113 #
114 # (2) IF GLOBAL ERRORS ARE TO BE ESTIMATED BY MAKING A REPEATED
115 # RUN ON THE SAME PROBLEM WITH SMALLER TOLERANCES, THEN ALL
116 # COMPONENTS OF RTOL AND ATOL SHOULD BE SCALED DOWN UNIFORMLY.
117 #
118 # (3) STIFFNESS OCCURS WHEN THE JACOBIAN MATRIX HAS AN EIGEN-
119 # VALUE WHOSE REAL PART IS NEGATIVE AND LARGE IN MAGNITUDE
120 # COMPARED TO THE RECIPROCAL OF THE T-SPAN OF INTEREST
121 #
122 # (4) FORM OF FCN AND JAC
123 #
124 # SUBROUTINE FCN (YDOT, N, Y, T)
125 # INTEGER N
126 # REAL YDOT(N), Y(N), T
127 # -----
128 # CALCULATE THE FUNCTION AT Y AND T
129 # RETURN THIS VECTOR IN YDOT
130 # -----
131 # RETURN
132 # END
133 #
134 # SUBROUTINE JAC (COL, J, N, Y, T)
135 # INTEGER J, N
136 # REAL COL(N), Y(N), T
137 # -----
138 # CALCULATE THE JTH COLUMN OF THE JACOBIAN AT Y AND T
139 # RETURN THIS VECTOR IN COL
140 # COL(I) IS THE PARTIAL OF YDOT(I) WITH RESPECT TO Y(J)
141 # -----
142 # RETURN
143 # END
144 #
145 # * (5) FOR THOSE PARAMETERS WITH DEFAULTS, INPUT A ZERO TO ALLOW
146 # THE DEFAULT TO TAKE EFFECT
147 #
148 # (6) WHEN N IS DECREASED, THE EFFECT IS TO DROP THE LAST
149 # EQUATION(S) FROM THE SYSTEM AND TO INTEGRATE THE RESULTING
150 # SMALLER SYSTEM WITH THE LAST VALUE(S) OF Y(I) HELD CONSTANT
151 #
152 # (7) STANDARD CHOICES FOR METH, MITR, AND MOSS ARE:
153 # (A) FOR A NONSTIFF PROBLEM: METH=1, MITR=4, AND MOSS=N/A
154 # (B) FOR A STIFF PROBLEM WITH IA/JA SUPPLIED:
155 # METH=2, MITR=1 OR 2, AND MOSS=3
156 # (C) FOR A STIFF PROBLEM WITH JAC SUPPLIED:
157 # METH=2, MITR=1, AND MOSS=1
158 # (D) FOR A STIFF PROBLEM WITH NEITHER IA/JA NOR JAC SUPPLIED:
159 # METH=2, MITR=2, AND MOSS=2
160 #
161 # ..B.. ALGORITHM .....
162 #
163 # ..C.. PORTABILITY .....
164 #
165 # ANSI 66 - NONSTANDARD SUBSCRIPTS
166 # ANSI 66 - NONSTANDARD CHARACTER CONSTANT IN ULEMSG CALLS
167 #
168 # ..D.. REFERENCES .....

```

Jan 29 16:34 1989 GEARS Page 4

```

169 #
170 # (1) A.C. HINDMARSH, GEAR.. ORDINARY DIFFERENTIAL EQUATION
171 # SYSTEM SOLVER, UCID-30001, REV 3, LAWRENCE LIVERMORE
172 # LABORATORY, 1974.
173 #
174 # (2) S.C. EISENSTAT, M.C. GURSKY, M.H. SCHULTZ, A.H. SHERMAN,
175 # YALE SPARSE MATRIX PACKAGE I, THE SYMMETRIC CODES,
176 # RESEARCH REPORT NO. 112, 1977.
177 #
178 # (3) S.C. EISENSTAT, M.C. GURSKY, M.H. SCHULTZ, A.H. SHERMAN,
179 # YALE SPARSE MATRIX PACKAGE II, THE NONSYMMETRIC CODES,
180 # RESEARCH REPORT NO. 114, 1977.
181 #
182 # ..E.. HISTORY .....
183 #
184 # ORIGIN - DRIVES AND PREP, A.C. HINDMARSH AND A.H. SHERMAN,
185 # GEARS, 9/5/79
186 # MODIFIED - HEADER, STURCTURE, E.S. MARWIL, 4 MAY 82
187 # echo "Done with file GEARS."

```

APPENDIX D
DIRECTORIES AND FILES

CONTENTS

Directories	D-4
Files	D-5
cray - Fortran	D-5
Include	D-24
Makefile	D-25
User interface	D-26
gears - Fortran	D-32

APPENDIX D
DIRECTORIES AND FILES

This appendix describes the directories and files for all components of FRAC-UNIX (i.e., the models, user interface, and numerical routines).

DIRECTORIES

DIRECTORY jdm/code

This directory holds the FORTRAN model files, minor FORTRAN supporting numerics, the include files (commons.i and deltas.i), the user interface files, and Makefile. This is the working directory and includes both source and object files.

DIRECTORY jdm/gears

This working directory holds source and object versions of the routines which perform integration or iteration to steady state.

DIRECTORY jdm/fru/sccs/cray

This directory holds the Source Code Control System (SCCS) files for the code working directory.

DIRECTORY jdm/fru/sccs/gears

This directory holds the SCCS files for the gears working directory.

DIRECTORY jdm/fru/bin

This directory holds the text for various executable statements

DIRECTORY jdm/manf

This directory holds input and output files for a sample problem.

FILES

DIRECTORY Cray: Fortran Files

FILE advan.F

SUBROUTINE ADVAN

ADVAN is the driver routine for particle movement. Properties are defined for the particle group, the particle arrays are compressed, MARKER is called to move existing particles, RELEASE is called to release new particles and ELCTROD is called.

called by: DODERIV

calls: ELCTROD, MARKER, RELEASE, EXCHANG

FILE anorm.F

FUNCTION ANORM

ANORM produces a normally-distributed random number between -6 and 6, with a mean of zero and a standard deviation of one. ANORM sums 12 uniform random numbers between 0 and 1 and subtracts 6 from the total. The result is gaussian, as predicted by the central limit theorem.

called by: FRTOFR, MARKER

calls: URN

FILE bdata1.F

BLOCK DATA BDATA1

Initialization of variables -BDATA1 should not be changed by the user.

DIRECTORY Cray: Fortran Files (cont'd)

FILE border.F

SUBROUTINE BORDER

Particles which are likely to diffuse into a fracture during the current time step are moved by BORDER. The special situation of a nearby material discontinuity is treated explicitly by a Monte Carlo routine. The probability density function of entering the fracture is sampled by the routine. If a particle enters the fracture, the time of entry is determined. If it remains in the matrix, then its position normal to the fracture is calculated from a probability distribution. Transverse diffusion and advection are treated subsequently upon return to MARKER. BORDER is not called if advection is significant compared to diffusion or if a particle has just advected out of a fracture, as identified by ADFG.

called by: MARKER

calls: DODEBUG, NRAPH, URN, EXPERFC

FILE bound.F

FUNCTION BOUND

Bound limits a variable to explicit upper and lower bounds.

called by: FRTOFR, MARKER, MTOFR

calls: None

FILE conc.F

SUBROUTINE CONC

CONC calculates and outputs matrix and fracture concentrations based on particle positions at the end of a particle move. CONC is only called if CONCPT is set true. Both matrix and fracture concentrations are calculated based on mass lumping of all particles in them. The number of particles in

DIRECTORY Cray: Fortran Files (cont'd)

a matrix or fracture element is multiplied by the mass per particle to determine the mass of solute in the element. The mass of solute is divided by the volume of water to determine the concentration.

called by: DODERIV

calls: None

FILE deriv.F

SUBROUTINE DODERIV

Computes particle moves, flow transients, or both simultaneously. Subroutine DOFCN is called to initialize the derivative vector. Particle calculations are called for discrete intervals or a flow transient is executed by discrete intervals or the two are integrated on a common interval.

In the particle move, subroutine JUNFUN may be called to set up particle transfers through junctions. Subroutine ADVAN is then called to execute the particle move for the required time increment. At each increment, well concentration and radial spreading are calculated and plot data is calculated for IPOT and CONWEL.

A flow transient is calculated by calling Subroutine DSGPDR.

Incremental digital output on the debug list is output by calling Subroutine DODEBUG.

Subroutine POST is called to print particle move statistics, to generate a particle location plot or a head plot. Residence time histograms are printed out.

called by: Y.TAB.C

calls: DOFCN, JUNFUN, ADVAN, DSGPDR, DODEBUG, POST, SPARSE, LINT, LINTS, CONC

DIRECTORY Cray: Fortran Files (cont'd)

FILE diagvel.F

SUBROUTINE DIAGVEL

If a particle exists in a quadrant that contains a fracture, the second-order pressure profile calculation breaks down. Under this circumstance, a linear pressure profile is calculated by DIAGVEL. The pressure drop along a fracture is assumed to be linear.

called by: MATVEL

calls: None

FILE dofcn.F

SUBROUTINE DOFCN

Computes the derivative for the current head values. It is activated from the command line interpreter with the "deriv" command.

called by: Y.TAB.C, DODERIV

calls: DODEBUG, FCN

FILE elctrod.F

SUBROUTINE ELCTROD

ELCTROD calculates tracer concentration at the electrodes.

called by: ADVAN

calls: None

DIRECTORY Cray: Fortran Files (cont'd)

FILE entwel.F

SUBROUTINE ENTWEL

Particles which are next to the withdrawal well are removed from the model. If IPRINT is greater than 2, then a report is made to output. If the particle is a heat particle, then the heat storage of its resident element is decremented.

called by: MARKER, FRTOFR

calls: None

FILE exchang.F

SUBROUTINE EXCHANG (inactive)

EXCHANG handles the swapping of particle group information in and out of large core memory. Warning--EXCHANG contains local array PEXT which must be dimensioned large enough to contain all the particle information.

called by: None

calls: None

FILE experfc.F

FUNCTION EXPERFC

EXPERFC calculates the product of $\text{EXP}(\text{CO}^{**2})$ and $\text{ERFC}(\text{cl})$. If the arguments to FORTRAN functions EXP and ERFC are within the proper limits, the multiplication is performed using these functions. If the arguments are too large for these functions, then a series solution is used which is accurate to the tenth decimal place.

called by: MACRO, PROBMT, BORDER

calls: None

DIRECTORY Cray: Fortran Files (cont'd)

FILE fcn.F

SUBROUTINE FCN

Computes the derivative for the current head values.

called by: DSGPDR and DSGPCI (GEARS DIRECTORY), DOFCN, ZFCN

calls: PROPS, FRACVEL, MASS, DODEBUG, LINT

FILE fracvel.F

SUBROUTINE FRACVEL

FRACVEL supplies the fluid velocities in the fractures. If turbulent flow is to be simulated, then the Reynolds number of each fracture is calculated. The velocity is then calculated based on whether the flow is laminar (LAMFG true) or turbulent. If rectangular fractures are modelled instead of parallel plate fractures, then flow is always modeled as laminar and the velocity calculation is modified for the rectangular aperture. External flows from the model are also determined in FRACVEL if the flag LEAKY has been set.

called by: FCN

calls: None

FILE frtofr.F

SUBROUTINE FRTOFR

FRTOFR determines the exit fracture for a particle which has entered a fracture intersection node. If the node has an external outflow, then the particle is removed via a call to ENTWEL. The particle is placed back into its original fracture if there is no exit flow from the node. If the node is a dead end node, then the particle is placed back in the fracture as if it sustained a constant deceleration from its initial position to a zero velocity state at the node.

DIRECTORY Cray: Fortran Files (cont'd)

called by: MARKER

calls: ENTWEL, ANORM, BOUND, URN

FILE ftvel.F

SUBROUTINE FTVEL

FTVEL solves for the normal velocity components at the top and bottom edges of the fractures at locations one quarter and three quarters down the length of the fracture. Transverse flows are computed for each half fracture edge except for edges facing a diagonal fracture. The fracture flows are identical for both halves of the fracture if adjacent to a diagonal fracture. The flow from fractures which contributes to fluid storage in an element is also calculated for each edge, excepting external fracture faces, of all half fractures at each node.

called by: MASS

calls: MATVEL

FILE grid.F

SUBROUTINE GRID

GRID generates the nodalization grid pattern. Grid patterns are user supplied, expanding or uniform centered at the well, or expanding or uniform centered at location zero, zero.

called by: SETUP

calls: None

DIRECTORY Cray: Fortran Files (cont'd)

FILE initial.F

SUBROUTINE INITIAL

Initializes for first case or restart. Sets T = TZ and H = HZ. Flags DSGPDR to restart.

called by: LEX.YY.C

calls: None

FILE jac.F

SUBROUTINE JAC

This is a place holder. It could be replaced by a module which explicitly calculates the head Jacobian, as described in the integration package dsgpdr.F documentation header. A new module, zjac.F could be created to reorder parameters, as described in the steady-state package zsgpdr.F documentation header. Finally, module static.F could be modified to supply zsgpdr.F with zjac.F, rather than jac.F.

called by: None

calls: None

FILE junfun.F

SUBROUTINE JUNFUN

JUNFUN creates the data for the fracture intersection transfer routine for all nodes with 2 or more fractures, except those with an external flow, QQEXT. It also calculates the fracture flow at every fracture end, QFATN.

called by: DODERIV

calls: None

DIRECTORY Cray: Fortran Files (cont'd)

FILES lint.F and lints.F

SUBROUTINES LINT AND LINTS

These two subroutines furnish a rapid interpolation and extrapolation capability by transforming a function tabulated in point-pair form into slope-intercept form where the index of the last segment used is retained. LINTS must only be called once for each function because the original point-pair data is overwritten. The data array must be overdimensioned by 50%. The fourth argument in the call to LINT must be uniquely named for each function and multiple uses of the same function.

called by: QVSTP, DODERIV, FCN

calls: None

FILE macro.F

SUBROUTINE MACRO

Special routine for movement down a fracture-matrix system. MACRO is called if the particle is in a fracture with a fracture Peclet number (PE) greater than 200. MACRO first picks the resident domain of the particle after the movement and then calculates the final position based on cumulative probability distributions.

called by: MARKER

calls: EXPERFC, ROOT, PROBMT, URN

FILE main.F

Program FRACSL is the process entry routine. Since it is a FORTRAN module, stdin, stdout, and stderr are initialized to FORTRAN logical units 5, 6, and 0, respectively. Logical unit 7 is opened as a temporary file. Logical units 12, 14, and 15 are opened, as indicated by the command line

DIRECTORY Cray: Fortran Files (cont'd)

arguments 1, 2, and 3 (respectively) or are defaulted to plot 12, plot 14, and plot 15, where their command line argument is not supplied.

called by: None

calls: CMAIN

FILE marker.F

SUBROUTINE MARKER

This is the main particle movement routine. MARKER moves a single particle the duration of a particle time step, DTSTP or the time remaining after release. If a particle is in the matrix, the nearest fracture boundary to the particle is found. If this fracture is so close that diffusion is likely to move the particle into the fracture, the subroutine BORDER is called. Otherwise, particles are advected in the matrix. MARKER checks to see if fractures are crossed and adjusts the time of the move accordingly. If a particle enters a fracture, then MTOFR is called to place the particle in the fracture. The routine to move a particle in a fracture is determined by the fracture Peclet number. If the Peclet number is, greater than 200, then the special Monte Carlo routine MACRO is called. If the Peclet number is less than 200, then the particle is advected down the fracture under the Poiseuille velocity profile. Diffusion is added as a random gaussian process. Transverse advection is considered in each case. If a particle crosses the side of the fracture, then the particle is checked to see if it advected or diffused. If diffusion dominated, then the particle may strike the wall and be placed back into the fracture. Otherwise, it is transferred to the matrix.

called by: ADVAN, RELEASE

calls: ANORM, BORDER, ENTWEL, FRTOFR, MACRO, MATVEL, MTOFR, NERFRC, OSHOOT, DODEBUG, BOUND, URN

DIRECTORY Cray: Fortran Files (cont'd)

FILE mass.F

SUBROUTINE MASS

MASS solves for the derivatives of the nodal pressures. For each element first the net flow from fractures is determined. The exit flow from the outer boundaries of each quadrant of the element is found by placing a fictitious particle just inside the boundary. MATVEL is called to find the bulk velocity at this point. If the quadrant boundary is a model exterior boundary and PERMBD is true, the flow out of the model is estimated based on the infinite porous media approximation.

called by: FCN

calls: FTVEL, MATVEL

FILE matvel.F

SUBROUTINE MATVEL

Particle velocities and accelerations are found from a second order pressure profile based on the nine closest nodes surrounding a particle. First a velocity term due to local storage is developed. The routine differs depending on the existence and type of diagonal fracture in the cell. In MATVEL, array H1 contains the pressure data from the surrounding nine nodes. These values and even the locations of the pressures are adjusted for the existence of fractures near the particle. Then a second order pressure profile is determined in the vertical and horizontal directions. From these profiles velocity and acceleration terms are computed. These velocities are bulk velocities. If the particles are real and not just for the pressure dynamic calculations, then the velocities are divided by the porosity to obtain local velocities. A special routine is supplied in MATVEL to calculate velocities next to the injection well.

DIRECTORY Cray: Fortran Files (cont'd)

called by: MASS, MARKER, FTVEL, RELEASE, SETUP

calls: DIAGVEL

FILE mtofr.F

SUBROUTINE MTOFR

MTOFR stands for matrix to fracture transfer. This routine places the particle into a fracture if the movement of the particle caused it to intersect a fracture. MTOFR does the required bookkeeping tasks for the transfer.

called by: MARKER

calls: BOUND

FILE mxdrv.F

SUBROUTINE MXDRV

MXDRV is a bookkeeping routine to track the performance of the steady-state solver. At each iteration of the solver, the maximum absolute and relative head derivative and the associated nodes are output. Additionally, upon convergence of the head solution, a table is output which indicates the ten nodes having the highest occurrence of the maximum and absolute head derivatives for all the iterations. MXDRV includes the dimensioning file, commons.i; otherwise, it would belong in the jdm/gears directory.

called by: ZSGPDR, ZSGPSL (both in gears directory)

calls: none

DIRECTORY Cray: Fortran Files (cont'd)

FILE mxerr.F

SUBROUTINE MXERR

MXERR is a bookkeeping routine to track the performance of the transient integration routine, DSGPPR. During the transient, at certain intervals the integration step number, the current problem time, the time step, the node with the maximum absolute weighted error, and the value of the maximum absolute weighted error are output. The output intervals are chosen, as follows:

- (1) for the first 100 integration steps, the previous values are output every 10 steps
- (2) up to 500 steps, output every 50 steps
- (3) up to 5,000 steps, output every 250 steps
- (4) up to 50,000 steps, output every 1,250 steps, and so on

Upon completion of the transient, a table is output of the ten nodes having the highest occurrence of the maximum weighted error over all the integration steps. Additionally, the average time step for the transient interval and the number of function evaluations required are output.

called by: VSWAMI (gears directory)

calls: None

FILE namtab.F

SUBROUTINE NAMTAB

Initializes the symbol table that interfaces the command line interpreter with the pre-existing FORTRAN model. Any variable to be input or output must be listed in this subroutine.

called by: CMAIN.C

calls: SYMBOL

DIRECTORY Cray: Fortran Files (cont'd)

FILE nerfrc.F

SUBROUTINE NERFRC

NERFRC calculates the distance and direction to the nearest fracture from a particle. The distance is limited to the boundaries of the cell.

called by: MARKER

calls: None

FILE nraph.F

SUBROUTINE NRAPH

Solves for the root of a function using the Newton-Raphson iterative procedure.

called by: BORDER

calls: None

FILE oshoot.F

SUBROUTINE OSHOOT

Subroutine OSHOOT makes the movement of particles in the matrix more efficient. A major problem with the movement routine had occurred when a particle crossed a matrix element boundary. After crossing a boundary, a particle was positioned a trivial distance away from the boundary. The likelihood of a particle diffusing back across the boundary is great even with a strong advective flux. This problem caused the code to spend an inordinate amount of time crossing and recrossing boundaries due to diffusion. This time did not affect the accuracy of the results but severely limited the number of particles which could move in the matrix.

DIRECTORY Cray: Fortran Files (cont'd)

OSHOOT allows particles to overshoot the matrix element boundaries if such an overshoot would not degrade the accuracy of the calculations. This places the particles away from the element boundaries and free from artificial diffusion dominance. The routine also allows an overshoot when a particle crosses a boundary which is not between the quadrants. This introduces an error in the movement routine. If the overshoot is kept small, the error will also be small. The overshoot can be justified by considering the source of the changes in velocity. Once the particle crosses the quadrant boundary new nodes contribute to the velocity calculations. These nodes and the nodes lost to the calculations are all far away from the particle and hence contribute little to the velocities. The gain in efficiency outweighs the loss in accuracy.

called by: MARKER

calls: URN

FILE post.F

SUBROUTINE POST

POST prints out a particle move summary and writes particle locations and pressure values to the output plot files. Post is called if a particle move is performed or if either HEADPT or PLOCPT is set TRUE.

called by: DODERIV, STATIC

calls: None

FILE probmt.F

FUNCTION PROBMT

Calculates $\text{ERFC}(D) + \text{EXP}(A) (\text{ERFC}[B] - \text{ERFC}[C])$

DIRECTORY Cray: Fortran Files (cont'd)

called by: MACRO

calls: EXPERFC

FILE props.F

SUBROUTINE PROPS

PROPS is used for heat particle modeling. PROPS calculates the temperature, the density-to-viscosity ratio divided by the same ratio at reference temperature, and the ratio of density to that at reference temperature for each matrix element, fracture, and fracture path-to-remote boundary.

called by: FCN

calls: QVSTP

FILE qvstp.F

SUBROUTINE QVSTP

Calls LINTS to transform density and viscosity tables to slope-intercept form. Calls LINT to calculate ratios of density and viscosity to those at reference temperature. Calculates the density-to-viscosity ratio divided by the same ratio at reference temperature.

called by: PROPS, SETUP

calls: LINT, LINTS

FILE release.F

SUBROUTINE RELEASE

DIRECTORY Cray: Fortran Files (cont'd)

RELEASE injects particles into the model along a line source or in locations surrounding the well node (NODIN). First the number of particles to be injected (PL) during the time step is determined. Particles are distributed in the matrix along the length of a horizontal or vertical line source according to the flow distribution. Particles are distributed into fractures or matrix at a well node according to its flow distribution. The probability of being placed in an exit fracture of the well node is the flux in the fracture divided by the total flux out of the well. If a particle is not placed in a fracture, then it is located in a random angle a distance equal to the well radius away from the well. If the last particle injected takes the last storage location in the particle data arrays, then a random particle is removed from the system and the new particle is stuffed into the freed storage location.

called by: ADVAN

calls: MARKER, MATVEL, URN

FILE root.F

FUNCTION ROOT

Newton-Raphson root solver using bounded input. The first argument to root is the function to be solved. Next are the upper and lower bounds on the possible solutions. The fourth argument is the tolerance of the solution with respect to accuracy of the solution, not the closeness of the function to zero. Arguments 5-8 are used by function PROBMT which is called by ROOT. The final argument IER is an error code to identify that the solution range does not contain a unique solution.

called by: MACRO

calls: None

FILE setup.F

SUBROUTINE SETUP

The following calculations are performed in SETUP:

- Fracture synthesis
- particle release concentrations
- electrode volumes
- fractures positions from fracture string data or checks FRENDD data for non-physical inputs
- calls GRID
- calculates PSIND1 from FRENDD
- nodal capacitances for pressure dynamics
- initializes VMAX for rectangular fracture channels
- total width of boundary fractures
- conductance to fixed head boundary if model is leaky
- forces pressure derivatives to zero if SOLVE is FALSE
- set DHSS to zero for DYNMIC and PERMBD runs

called by: YTAB.C

calls: QVSTP, MATVEL, DODEBUG, URN2, URN, GRID

FILE sparse.F

FUNCTION SPARSE

Fills IA and JA vectors with the Jacobian sparseness structure description in the format required by the steady state and integration packages. It assumes a regular rectangular structure to the grid that is characterized by current values of NROW and NCOL.

called by: DODERIV, STATIC

calls: None

DIRECTORY Cray: Fortran Files (cont'd)

FILE static.F

SUBROUTINE DOSTATIC

This module uses the steady-state package to find a steady-state head distribution. Calls POST to output head plot.

called by: Y.TAB.C

calls: SPARSE, ZSGPDR, DODEBUG, POST

FILE urn.F

FUNCTION URN

URN(a,b) returns uniform random numbers on the interval [a,b] where a = 0. and b = 1.0. A new sequence of random numbers is initialized in SETUP by passing URN a negative 8-digit real number as the second argument. In this case, the first argument, a, is a logical unit for error messages.

called by: ANORM, BORDER, FRTOFR, MACRO, MARKER, OSHOOT, RELEASE,
SETUP

calls: None

FILE urn2.F

FUNCTION URN2

URN2 functions as URN but is used exclusively for fracture synthesis. It is also initialized in SETUP but with a different seed. This maintains the ability to resynthesize the same fractures while obtaining a different realization of particle transport.

called by: SETUP

calls: None

DIRECTORY Cray: Fortran Files (cont'd)

FILE zfcn.F

SUBROUTINE ZFCN

Reorders arguments in call to FCN for steady-state calculation.

called by: ZSGPDR, ZSGSMS, and ZSGPSL (all in gears directory)

calls: FCN

DIRECTORY Cray: Makefile

FILE Makefile

This file tells the UNIX make utility which files depend on which other files. In general, object files depend on their source and any included header files and executable files depend on each of their object files. If a file is dependent on a file that is newer than itself, it is rebuilt. The make utility supplies default rebuild actions for common file types. These can be overridden by the Makefile See the UNIX make utility documentation.

DIRECTORY Cray: User Interface Files

FILE asgn.c

Copies a value into a rectangular region of an array. It is used in region assignment.

called by:

calls:

FILE cmain.C

Creates and initializes symbol table, initializes the plotter files, executes the command line interpreter, and then terminates plotter files.

called by:

calls:

FILE debug.c

Prints out the debug list.

called by: BORDER, DODERIV, DOFCN, FCN, MARKER, SETUP, STATIC

calls:

FILE gethash.c

Performs hash table lookup.

called by:

calls:

DIRECTORY Cray: User Interface Files (cont'd)

FILE getplace.c

Copies a hash table array value to the parser stack.

called by:

calls:

FILE hash.h

Defines hash table-related data structures.

called by:

calls:

FILE index.c

Performs hash table lookup.

FILE input.l

Describes the lexical analyzer. The UNIX lex utility preprocesses this file to generate lex.yy.c, which recognizes keywords, special characters, numbers, comments, strings, and illegal characters.

called by:

calls:

FILE input.y

Describes the syntactic analyzer. The UNIX yacc utility preprocesses this file to generate y.tab.c and y.tab.h, which recognize commands and

DIRECTORY Cray: User Interface Files (cont'd)

define token identifier values.

called by:

calls:

FILE lex.yy.c

Generated from input.l by the UNIX lex utility. It recognizes keywords, special characters, numbers, comments, strings, and illegal characters from the parser input stream.

called by:

calls:

FILE newhash.c

Dynamically allocates and initializes a hash table.

called by:

calls:

FILE place.c

Validates and indexes a FORTRAN global variable, using information available in the symbol table.

called by:

calls:

DIRECTORY Cray: User Interface Files (cont'd)

FILE prnt.c

Prints out a region of an array.

called by:

calls:

FILE puthash.c

called by:

calls:

FILE remhash.c

Removes hash table associations. It is not currently used.

called by:

calls:

FILE strhash.c

Computes a hash value from a null terminated string.

called by:

calls:

FILE symbol.c

Initializes and enters a FORTRAN global variable description into the

DIRECTORY Cray: User Interface Files (cont'd)

symbol table. Parameters include the symbol name, the variable location, the row and column dimensions, and the variable type.

called by:

calls:

FILE symbols.h

Header file describing symbol table data structures.

called by:

calls:

FILE types.h

Header file describing abstract types and some special constants.

called by:

calls:

FILE where.c

Provides source file traceback in the event of errors occurring in a sourced file.

called by:

calls:

DIRECTORY Cray: User Interface Files (cont'd)

FILE y.tab.c

Generated by the UNIX yacc utility from input.y. It recognizes and interprets commands.

called by:

calls:

FILE y.tab.h

Header file generated by the UNIX yacc utility from input.y. It defines token identifier values and the parser stack data structure.

called by:

calls:

FILE yyerror.c

Prints a syntax error message.

called by:

calls:

FILE yywrap.c

Closes sourced files and restores the previous input stream.

called by:

calls:

DIRECTORY gears: FORTRAN Files

FILE dsgeia.F

SUBROUTINE DSGEIA

Calculates the initial step size and adjusts, as necessary.

FILE dsgein.F

SUBROUTINE DSGEIN

Computes interpolated values of the Kth derivative of the dependent variable vector Y, and stores it in DKY.

FILE dsgenm.F

SUBROUTINE DSGENM

Manipulates the NORDSEICK array.

FILE dsgeos.F

SUBROUTINE DSGEOS

Computes step size ratios for potential change of order.

FILE dsgeosc.F

SUBROUTINE DSGESC

Sets the coefficients for the core integrator. These are set for all orders for the current method.

FILE dsgewt.F

SUBROUTINE DSGEWT

Initializes or updates the weighting vector for use in the root-mean-square norm.

DIRECTORY gears: FORTRAN Files (cont'd)

FILE dsqpci.F

SUBROUTINE DSGPCI

Performs one step of the integration of an initial value problem for a system of first-order, ordinary differential equations.

FILE dsqpdr.F

SUBROUTINE DSGPDR

Calls the subroutines required to solve the initial value problem for a system of first-order ordinary differential equations with a sparse Jacobian.

FILE dsqppc.F

SUBROUTINE DSGPPC

Performs the corrector iteration.

FILE dsqsms.F

SUBROUTINE DSGSMS

Computes sparseness structure of the matrix used in the Newton iteration (performing the corrector). This structure is the same as that of the Jacobian, plus diagonal elements.

FILE gimddr.F

SUBROUTINE GIMDDR

Finds a minimum degree ordering of the rows and columns of a symmetric matrix A stored in (IA,JA,AS) format.

FILE gimdfe.F

SUBROUTINE GIMDFE

Forms element from uneliminated neighbors of VK.

DIRECTORY gears: FORTRAN Files (cont'd)

FILE gimdin.F

SUBROUTINE GIMDIN

Initializes work vectors and lists for sparse matrix routines.

FILE gimdpe.F

SUBROUTINE GIMDPE

Purges inactive elements and does mass elimination.

FILE gimdud.F

SUBROUTINE GIMDUD

Updates degrees of uneliminated vertices in EK.

FILE giordr.F

SUBROUTINE GIORDR

Driver for sparse symmetric matrix ordering routines.

FILE lsqpax.F

SUBROUTINE LSGPAX

Forms the real vector $X = A * B$ or $X = \text{Trans}(A) * B$ using the LDU-factorization for sparse A (compressed pointer storage).

FILE lsgpdr.F

SUBROUTINE LSGPDR

Driver for solving sparse nonsymmetric systems of linear equations (compressed pointer storage).

DIRECTORY gears: FORTRAN Files (cont'd)

FILE lsgpnf.F

SUBROUTINE LSGPNF

Performs numerical LDU-factorization of sparse nonsymmetric matrix and solution of system of linear equations (compressed pointer storage).

FILE lsgpro.F

SUBROUTINE LSGPRO

Reorders rows of AS, leaving row order unchanged.

FILE lsgpsf.F

SUBROUTINE LSGPSF

Performs symbolic factorization to determine where fill in will occur during numeric factorization.

FILE lsgpsl.F

SUBROUTINE LSGPSL

Numerical solution of sparse nonsymmetric system of linear equations given LDU-factorization (compressed pointer storage).

FILE ullvl.F

SUBROUTINE ULELVL

Specifies error condition/message code level.

FILE ulemsg.F

SUBROUTINE ULEMSG

Write error condition/message code information to the 6th logical unit.

DIRECTORY gears: FORTRAN Files (cont'd)

FILE ulunit.F

SUBROUTINE ULUNIT

Specifies six logical units or devices.

FILE usgsgc.F

SUBROUTINE USGSGC

Groups column indices of the Jacobian matrix for use in the numerical evaluation of the Jacobian by finite differences.

FILE usmach.F

REAL FUNCTION USMACH

Defines machine-dependent constants for the Sun computer.

FILE vicopy.F

SUBROUTINE VICOPY

Copies an integer vector to an integer vector.

FILE vifill.F

SUBROUTINE VIFILL

Fills an integer vector with a constant integer.

FILE vsaxpy.F

SUBROUTINE VSAXPY

Real constant, times a real vector, plus a real vector.

DIRECTORY gears: FORTRAN Files (cont'd)

FILE vsaxty.F

SUBROUTINE VSAXTY

Replaces each component of a real vector with the product of a real scalar, times the vector component, times the corresponding component of another real vector.

FILE vscopy.F

SUBROUTINE VSCOPY

Copies a real vector to a real vector.

FILE vsfill.F

SUBROUTINE VSFILL

Fills a real vector with a constant value.

FILE vsnrm2.F

REAL FUNCTION VSNRM2

Euclidean norm of a real vector.

FILE vswami.F

INTEGER FUNCTION VSWAMI

Finds index of the first element having maximum absolute weighted value of a real vector.

FILE vswrms.F

REAL FUNCTION VSWRMS

Weighted Euclidean norm of a real vector.

DIRECTORY gears: FORTRAN Files (cont'd)

FILE vsxtsa.F

SUBROUTINE VSXTSA

Scales a real vector by a real constant.

FILE zsgpcs.F

SUBROUTINE ZSGPCS

Determines the Dogleg step X by the convex combination of the Gauss-Newton and the scaled gradient directions that minimizes $(A*X - B)$ in the least squares sense, subject to the restriction that the Euclidean norm of $\text{diag}*X$ be at most δ , where A is given by its LU factors.

FILE zsgpdr.F

SUBROUTINE ZSGPDR

Calls the subroutines required to find a zero of a system of N nonlinear functions in N variables with a sparse Jacobian.

FILE zsgpsl.F

SUBROUTINE ZSGPSL

Finds a zero of a system of N nonlinear functions in N variables.

FILE zsgpud.F

SUBROUTINE ZSGPUD

Updates the upper triangular factor of a factored approximate Jacobian matrix for use in the solution of nonlinear systems.

FILE zsgsms.F

SUBROUTINE ZSGSMS

Computes sparseness structure of the matrix used in the Newton-like iteration.

APPENDIX E
SAMPLE PROBLEM INPUT/OUTPUT

APPENDIX E
SAMPLE PROBLEM INPUT/OUTPUT

mt4 - I/O for test problems ff-5 and mt-4: steady flow and mass transport in a porous medium with discrete fractures.

mt4.inp	input	E-3
mt4.rpt	output	E-6
plot.7	concentrations	E-10
plot.10	electrode concentrations	E-13

ht2 - I/O for test problem ht-2: heat transport in a unidirectional flow field.

ht2.inp	input	E-14
ht2.rpt	output	E-16

Sample problem - Infinite porous media

frac.inp	input	E-22
frac.rpt	output	E-24

```

# frac-unix input for test problems ff-5 and mt-4
# steady flow in a porous medium with discrete fractures
#
# UNITS: meters, kilograms, days
#
# grid
#
nrow=9
ncol=17
psix=0.
psiy=1.5708
nn=nrow*ncol
igrid=1
dxz=0.2
dyz=0.2
xexp=1.
yexp=1.
#
# matrix materials
#
nmtm=1
immat(1:nn)=1
#
# matrix flow properties
#
mtconx=8.64      #meters/day
mtcony=8.64      #meters/day
mtss=1.e-5
mtpor=0.25
#
# matrix/water transport properties
#
mtdsyl=0.1        #meters
mtdsynt=0.02      #meters
trdif=2.7397e-6   # m**2/day => 1.e-3 meters**2/year
#
# fracture properties and geometry
#
nf=46
ifmat(1:nf)=1
frss=0.
nstr=16
nfistr(1)=3 ; psistr(1)=2 ; strend(1)=2 ; bstr(1)=5.0e-4;
nfistr(2)=2 ; psistr(2)=2 ; strend(2)=6 ; bstr(2)=5.0e-4;
nfistr(3)=2 ; psistr(3)=4 ; strend(3)=6 ; bstr(3)=5.0e-4;
nfistr(4)=3 ; psistr(4)=4 ; strend(4)=24 ; bstr(4)=5.0e-4;
nfistr(5)=5 ; psistr(5)=2 ; strend(5)=28 ; bstr(5)=5.0e-4;
nfistr(6)=3 ; psistr(6)=4 ; strend(6)=36 ; bstr(6)=5.0e-4;
nfistr(7)=4 ; psistr(7)=4 ; strend(7)=50 ; bstr(7)=5.0e-4;
nfistr(8)=3 ; psistr(8)=2 ; strend(8)=74 ; bstr(8)=5.0e-4;
nfistr(9)=3 ; psistr(9)=4 ; strend(9)=80 ; bstr(9)=5.0e-4;
nfistr(10)=4 ; psistr(10)=4 ; strend(10)=86 ; bstr(10)=5.0e-4;
nfistr(11)=3 ; psistr(11)=2 ; strend(11)=96 ; bstr(11)=5.0e-4;
nfistr(12)=4 ; psistr(12)=2 ; strend(12)=102 ; bstr(12)=5.0e-4;
nfistr(13)=1 ; psistr(13)=4 ; strend(13)=116 ; bstr(13)=5.0e-4;
nfistr(14)=3 ; psistr(14)=2 ; strend(14)=50 ; bstr(14)=5.0e-4;
nfistr(15)=2 ; psistr(15)=4 ; strend(15)=134 ; bstr(15)=5.0e-4;
nfistr(16)=1 ; psistr(16)=2 ; strend(16)=124 ; bstr(16)=5.0e-4;
#
# flow boundary conditions
#
freeze(6)=true
freeze(92)=true

```

```

freeze(150)=true
#
# initial conditions
#
hz(1:nn)=1.0
hz(6)=1.1
hz(92)=1.0
hz(150)=1.0
#
# miscellaneous
#
units=1          #meters,kilograms,days
tpn=25.          #temperature of fluid
tcpmx=400.       #maximum CPU seconds allowed for run
numcol=5         #number of columns in output
#
# outputs desired
#
headpt=false     #output heads at all nodes
concpt=false     #output concentrations for matrix and fractures
plocpt=false     #output particle position at end of run
#
# sourced data
#
@ZERO
zero_moss=2      #overwrite the ZERO value for zero_moss
zero_thresh=0.
#
# add variables to the debug list for outputting
#
debug zero_xtol zero_epsfcn zero_thresh zero_moss zero_nza
#
# define solution variables
#
solve=true
dynmic=false
#
# begin the steady state flow solution
static
echo "... steady state analysis complete"
#
# source the output statements in PRINTM
@PRINTM
#
# remove variables from the debug list
undebg zero_xtol zero_epsfcn zero_thresh zero_moss zero_nza
#
# STACK THE PARTICLE MOVE SO IT USES THE STEADY FLOW SOLUTION
#
# particles
#
npgmx=1
npxm=2000
nseed=87654321.
# point source
nodin=6
rwel=0.01
# exits, code concentration, temperature
nout=2
iout(1)=92
iout(2)=150
extbnd=false
# internal monitoring
nelc=6
pelhst(1)=7;    relc(1)=0.14142    #sqrt(0.08)/2
pelhst(2)=-67;  relc(2)=0.1

```

```

pelhst(3)=38;    relc(3)=0.14142
pelhst(4)=10;    relc(4)=0.14142
pelhst(5)=12;    relc(5)=0.14142
pelhst(6)=13;    relc(6)=0.14142
pelxyu(1,1:nelc)=0.5; pelxyu(2,2)=0.5

```

```

#
# outputs desired
#
plocpt=false      #output particle locations at end of run
concpt=true       #output matrix and element concentrations at end
headpt=false
#
# time control
#
tsrel=0.          #time to begin the particle release
terel=0.0347222   #time to end particle release
tend=0.0347222    #time to end problem
dtp=0.001         #time step for particle move
telcnx=0.001      #reporting interval for electrodes
#
# define solution variables
#
solve=false
dynmic=false
#
# begin the particle move, "start" initiates subroutine DODERIV
#
start
echo "... particle moves finished"
#
# terminate FRAC-UNIX
stop

```

```

0, 1 >> 0, 2 >> 0, 3 >> 0, 4 >> 0, 5 >> 0, 6 >> 0, 7 >> 0, 8 >> 9;
0, 9 >> 17;
0, 10 >> 0;
0, 11 >> 1.5708;
0, 12 >> 153;
0, 13 >> 1;
0, 14 >> 0.2;
0, 15 >> 0.2;
0, 16 >> 1;
0, 17 >> 1;
0, 18 >> 0, 19 >> 0, 20 >> 0, 21 >> 1;
0, 22 >> 1;
0, 23 >> 0, 24 >> 0, 25 >> 0, 26 >> 8.64;
0, 27 >> 8.64;
0, 28 >> 1e-05;
0, 29 >> 0.25;
0, 30 >> 0, 31 >> 0, 32 >> 0, 33 >> 0.1;
0, 34 >> 0.02;
0, 35 >> 2.7397e-06;
0, 36 >> 0, 37 >> 0, 38 >> 0, 39 >> 46;
0, 40 >> 1;
0, 41 >> 0;
0, 42 >> 16;
0, 43 >> 3; 2; 2; 0.0005;
0, 44 >> 2; 2; 6; 0.0005;
0, 45 >> 2; 4; 6; 0.0005;
0, 46 >> 3; 4; 24; 0.0005;
0, 47 >> 5; 2; 28; 0.0005;
0, 48 >> 3; 4; 36; 0.0005;
0, 49 >> 4; 4; 50; 0.0005;
0, 50 >> 3; 2; 74; 0.0005;
0, 51 >> 3; 4; 80; 0.0005;
0, 52 >> 4; 4; 86; 0.0005;
0, 53 >> 3; 2; 96; 0.0005;
0, 54 >> 4; 2; 102; 0.0005;
0, 55 >> 1; 4; 116; 0.0005;
0, 56 >> 3; 2; 50; 0.0005;
0, 57 >> 2; 4; 134; 0.0005;
0, 58 >> 1; 2; 124; 0.0005;
0, 59 >> 0, 60 >> 0, 61 >> 0, 62 >> TRUE;
0, 63 >> TRUE;
0, 64 >> TRUE;
0, 65 >> 0, 66 >> 0, 67 >> 0, 68 >> 1;
0, 69 >> 1.1;
0, 70 >> 1;
0, 71 >> 1;
0, 72 >> 0, 73 >> 0, 74 >> 0, 75 >> 1;
0, 76 >> 25;
0, 77 >> 400;
0, 78 >> 5;
0, 79 >> 0, 80 >> 0, 81 >> 0, 82 >> FALSE;
0, 83 >> FALSE;
0, 84 >> FALSE;
0, 85 >> 0, 86 >> 0, 87 >> 0, 88 >> 0, 89 >> 2;
0, 90 >> 0;
0, 91 >> 0, 92 >> 0, 93 >> 0, 94 >> 0, 95 >> 0, 96 >> 0, 97 >> 0, 98 >> TRUE;
0, 99 >> FALSE;
0, 100 >> 0, 101 >> 0, 102 >>

```

	ITER	NODE	MAX REL DERIV	NODE	MAX ABS DERIV
	1	14	0.8342E+07	14	0.8342E+07
	2	48	0.2271E-05	48	0.2403E-05

OCCURENCE OF MAXIMUM REL DERIVS (REL TO NONZERO HEADS)

	NODE	NTIMES
	14	48
	1	1

OCCURENCE OF MAXIMUM ABSOLUTE DERIVATIVES

	NODE	NTIMES
	14	48
	1	1

ZERO_XTOL
1e-07;

ZERO_EPSFCN
1;

ZERO_THRESH
0;

ZERO_MOSS
2;

ZERO_NZA
1485;

0, 103 >> . . . steady state analysis complete

0, 104 >> 0, 105 >> 0, 106 >> T

0;

H

1.07731;	1.0802;	1.08165;	1.08538;	1.0914;
1.1;	1.09587;	1.09299;	1.09184;	1.07528;
1.07691;	1.08049;	1.08427;	1.09006;	1.09064;
1.09512;	1.09219;	1.0906;	1.06993;	1.07177;
1.07552;	1.08112;	1.07963;	1.07564;	1.08339;
1.0919;	1.08222;	1.05989;	1.0652;	1.06922;
1.07274;	1.07389;	1.06992;	1.07096;	1.06928;
1.05741;	1.05816;	1.05916;	1.06303;	1.0656;
1.06258;	1.06178;	1.06075;	1.05549;	1.05496;
1.05456;	1.05528;	1.05797;	1.05536;	1.05024;
1.05364;	1.05246;	1.05161;	1.05158;	1.04957;
1.04989;	1.05056;	1.0498;	1.04911;	1.04913;
1.04827;	1.04797;	1.04782;	1.04387;	1.04396;
1.04346;	1.04481;	1.04747;	1.04657;	1.04452;
1.04397;	1.0438;	1.03843;	1.03686;	1.03737;
1.03862;	1.0407;	1.0459;	1.04143;	1.03969;
1.03937;	1.03542;	1.0272;	1.03209;	1.03243;
1.02982;	1.03507;	1.03466;	1.03441;	1.03467;
1.01453;	1;	1.02184;	1.02834;	1.02901;
1.02955;	1.02965;	1.03004;	1.03013;	1.01601;
1.01662;	1.02354;	1.02508;	1.02848;	1.02624;
1.02549;	1.02572;	1.02589;	1.02062;	1.02245;
1.02148;	1.02043;	1.02164;	1.02183;	1.02176;
1.0213;	1.02241;	1.02212;	1.02059;	1.01952;
1.01823;	1.01687;	1.01725;	1.0173;	1.01812;
1.0209;	1.01949;	1.01896;	1.01781;	1.01621;
1.01435;	1.01296;	1.01311;	1.01328;	1.01465;
1.01821;	1.01779;	1.01659;	1.01457;	1.01155;
1.00876;	1.00889;	1.01066;	1.01163;	1.0178;
1.01739;	1.01614;	1.01391;	1.01006;	1;
1.00651;	1.00984;	1.01071;		

DH

-1.06026e-07;	1.86851e-06;	5.55112e-08;	1.08108e-07;	1.01585e-07;
0;	-2.76168e-08;	-5.20417e-09;	-3.9968e-08;	7.95197e-08;
6.66134e-08;	-1.19488e-07;	2.72005e-08;	1.01252e-06;	-7.54952e-08;
1.23124e-06;	6.59195e-08;	-3.09475e-08;	1.42664e-07;	-1.04777e-07;
1.72362e-07;	3.33067e-07;	-4.86416e-08;	1.55431e-08;	5.32907e-08;
4.60743e-07;	6.43929e-08;	1.07081e-06;	-4.26048e-08;	-1.08247e-08;
1.50435e-07;	2.05613e-06;	-7.63278e-09;	6.52256e-08;	-3.33067e-08;
1.39222e-06;	-9.21485e-08;	7.99361e-07;	5.16254e-08;	-3.41949e-07;
1.06026e-07;	-4.38538e-08;	2.41474e-08;	1.4333e-06;	-9.67282e-08;
1.60844e-07;	1.29063e-08;	2.40252e-06;	8.61811e-08;	1.38223e-06;
8.46545e-08;	2.05169e-06;	-2.34535e-08;	-1.63758e-08;	-1.25733e-07;
1.13937e-07;	-1.11022e-07;	4.26326e-07;	1.41553e-07;	2.35367e-07;

```

1.50435e-07; 5.02376e-08; 6.13398e-08; 2.26208e-07; -1.26843e-07;
8.9706e-07; 1.21847e-07; 2.44249e-07; 1.62925e-07; 9.19265e-07;
-2.13024e-08; -4.52416e-08; -6.32827e-08; 3.35287e-07; 1.66464e-07;
-7.49401e-09; -6.63358e-08; 4.26326e-07; -2.35922e-09; 1.36779e-06;
2.28983e-08; 9.17044e-07; 5.85643e-08; 9.08162e-07; -5.53724e-08;
2.37033e-07; -3.9968e-08; 5.9952e-08; 1.09218e-07; -4.82947e-08;
7.71605e-08; 0; 9.15934e-08; 1.58762e-06; -1.00614e-08;
1.87184e-06; 9.70751e-08; 3.40006e-08; -4.27436e-08; 2.19269e-08;
1.9984e-08; 3.9746e-07; 1.14214e-07; -2.59515e-07; 8.09769e-08;
3.68594e-07; -4.4062e-08; 1.249e-08; -9.21485e-08; 7.80626e-07;
4.61436e-08; 4.84057e-07; -9.45077e-08; 4.69069e-08; -1.04292e-07;
8.28226e-07; 1.11022e-09; 7.19702e-07; 1.23478e-07; 6.79665e-08;
6.67175e-08; -1.13243e-07; 8.63892e-08; -3.70814e-07; 1.17129e-07;
-2.35922e-08; -1.00267e-07; 1.26635e-08; -2.328e-08; -3.53884e-08;
1.59942e-08; -4.30767e-07; 1.8173e-07; -4.70735e-07; 6.32827e-08;
-2.89352e-08; -2.85015e-08; -3.42434e-08; 2.33841e-08; 3.23352e-08;
1.01724e-07; 1.37668e-06; -6.31439e-08; 8.43769e-08; 3.5319e-08;
7.30319e-08; 9.45077e-08; 3.45557e-08; -6.78624e-08; 0;
1.55431e-08; -8.16708e-08; 7.11237e-08;

```

```

0, 107 >> 0, 108 >> 0, 109 >> 0, 110 >> 0, 111 >> 0, 112 >> 0, 113 >> 0, 114 >> 0, 115 >

```

```

0, 116 >> 2000;

```

```

0, 117 >> 8.76543e+07;

```

```

0, 118 >> 0, 119 >> 6;

```

```

0, 120 >> 0.01;

```

```

0, 121 >> 0, 122 >> 2;

```

```

0, 123 >> 92;

```

```

0, 124 >> 150;

```

```

0, 125 >> FALSE;

```

```

0, 126 >> 0, 127 >> 6;

```

```

0, 128 >> 7; 0.14142;

```

```

0, 129 >> -67; 0.1;

```

```

0, 130 >> 38; 0.14142;

```

```

0, 131 >> 10; 0.14142;

```

```

0, 132 >> 12; 0.14142;

```

```

0, 133 >> 13; 0.14142;

```

```

0, 134 >> 0.5; 0.5;

```

```

0, 135 >> 0, 136 >> 0, 137 >> 0, 138 >> FALSE;

```

```

0, 139 >> TRUE;

```

```

0, 140 >> FALSE;

```

```

0, 141 >> 0, 142 >> 0, 143 >> 0, 144 >> 0;

```

```

0, 145 >> 0.0347222;

```

```

0, 146 >> 0.0347222;

```

```

0, 147 >> 0.001;

```

```

0, 148 >> 0.001;

```

```

0, 149 >> 0, 150 >> 0, 151 >> 0, 152 >> FALSE;

```

```

0, 153 >> FALSE;

```

```

0, 154 >> 0, 155 >> 0, 156 >> 0, 157 >> NOTE - PARTICLE 727 REMOVED BEFORE DIFFUSING;

```

```

NOTE - PARTICLE 144 REMOVED BEFORE DIFFUSINGFROMFRAC25 TO NODE 94

```

PARTICLE MOVEMENT SUMMARY FOR FRACTURES

```

HIGH PECLET MOVES... 0 = 0.0 %

```

```

LOW PECLET MOVES...57278 = 100.0 %

```

```

TOTAL MOVES...57278

```

NO ADVECTION OUT OF FRACTURE IN HI PE MOVES

PARTICLE MOVEMENT SUMMARY FOR MATRIX

```

MONTE CARLO MOVES... 0 = 0.0 %

```

```

EXPLICIT MOVES...37102 = 100.0 %

```

```

ALL MOVES EXC DIFFUS TRANS TO FRACS37102 = 100.0 %

```

```

FRACTURE TRANSFER... 7457 = 20.1 %

```

```

REMAIN IN MATRIX...29645 = 79.9 %

```


EXIT FROM MODEL..... 0 = 0.0 ..

0, 158 >> . . . particle moves finished

0, 159 >> 0, 160 >> 0, 161 >> *** FRACSL done.

TIME	0.03472	0.03472	
matrix element concentrations			
	x	y	conc
MCON	0.1000E+00	0.1000E+00	0.0000E+00
MCON	0.1000E+00	0.3000E+00	0.0000E+00
MCON	0.1000E+00	0.5000E+00	0.0000E+00
MCON	0.1000E+00	0.7000E+00	0.7638E-02
MCON	0.1000E+00	0.9000E+00	0.1671E+00
MCON	0.1000E+00	0.1100E+01	0.1671E+00
MCON	0.1000E+00	0.1300E+01	0.0000E+00
MCON	0.1000E+00	0.1500E+01	0.0000E+00
MCON	0.3000E+00	0.1000E+00	0.0000E+00
MCON	0.3000E+00	0.3000E+00	0.0000E+00
MCON	0.3000E+00	0.5000E+00	0.6301E-01
MCON	0.3000E+00	0.7000E+00	0.6397E-01
MCON	0.3000E+00	0.9000E+00	0.9547E-02
MCON	0.3000E+00	0.1100E+01	0.1146E-01
MCON	0.3000E+00	0.1300E+01	0.3724E+00
MCON	0.3000E+00	0.1500E+01	0.1050E-01
MCON	0.5000E+00	0.1000E+00	0.0000E+00
MCON	0.5000E+00	0.3000E+00	0.0000E+00
MCON	0.5000E+00	0.5000E+00	0.8593E-02
MCON	0.5000E+00	0.7000E+00	0.5251E-01
MCON	0.5000E+00	0.9000E+00	0.9547E-03
MCON	0.5000E+00	0.1100E+01	0.9547E-03
MCON	0.5000E+00	0.1300E+01	0.5633E-01
MCON	0.5000E+00	0.1500E+01	0.6683E-02
MCON	0.7000E+00	0.1000E+00	0.0000E+00
MCON	0.7000E+00	0.3000E+00	0.0000E+00
MCON	0.7000E+00	0.5000E+00	0.9547E-03
MCON	0.7000E+00	0.7000E+00	0.1623E-01
MCON	0.7000E+00	0.9000E+00	0.9547E-03
MCON	0.7000E+00	0.1100E+01	0.0000E+00
MCON	0.7000E+00	0.1300E+01	0.0000E+00
MCON	0.7000E+00	0.1500E+01	0.0000E+00
MCON	0.9000E+00	0.1000E+00	0.0000E+00
MCON	0.9000E+00	0.3000E+00	0.9547E-03
MCON	0.9000E+00	0.5000E+00	0.1909E-01
MCON	0.9000E+00	0.7000E+00	0.9547E-03
MCON	0.9000E+00	0.9000E+00	0.0000E+00
MCON	0.9000E+00	0.1100E+01	0.0000E+00
MCON	0.9000E+00	0.1300E+01	0.0000E+00
MCON	0.9000E+00	0.1500E+01	0.0000E+00
MCON	0.1100E+01	0.1000E+00	0.0000E+00
MCON	0.1100E+01	0.3000E+00	0.1909E-02
MCON	0.1100E+01	0.5000E+00	0.3819E-02
MCON	0.1100E+01	0.7000E+00	0.0000E+00
MCON	0.1100E+01	0.9000E+00	0.0000E+00
MCON	0.1100E+01	0.1100E+01	0.0000E+00
MCON	0.1100E+01	0.1300E+01	0.0000E+00
MCON	0.1100E+01	0.1500E+01	0.0000E+00
MCON	0.1300E+01	0.1000E+00	0.0000E+00
MCON	0.1300E+01	0.3000E+00	0.9547E-03
MCON	0.1300E+01	0.5000E+00	0.2291E-01
MCON	0.1300E+01	0.7000E+00	0.3724E-01
MCON	0.1300E+01	0.9000E+00	0.9547E-03
MCON	0.1300E+01	0.1100E+01	0.0000E+00
MCON	0.1300E+01	0.1300E+01	0.0000E+00
MCON	0.1300E+01	0.1500E+01	0.0000E+00
MCON	0.1500E+01	0.1000E+00	0.0000E+00
MCON	0.1500E+01	0.3000E+00	0.8593E-02
MCON	0.1500E+01	0.5000E+00	0.2864E-02
MCON	0.1500E+01	0.7000E+00	0.0000E+00
MCON	0.1500E+01	0.9000E+00	0.6206E-01
MCON	0.1500E+01	0.1100E+01	0.0000E+00
MCON	0.1500E+01	0.1300E+01	0.0000E+00

MCON	0.1500E+01	0.1000E+01	0.0000E+00
MCON	0.1700E+01	0.1000E+00	0.4105E-01
MCON	0.1700E+01	0.3000E+00	0.4965E-01
MCON	0.1700E+01	0.5000E+00	0.9547E-03
MCON	0.1700E+01	0.7000E+00	0.0000E+00
MCON	0.1700E+01	0.9000E+00	0.9547E-03
MCON	0.1700E+01	0.1100E+01	0.0000E+00
MCON	0.1700E+01	0.1300E+01	0.0000E+00
MCON	0.1700E+01	0.1500E+01	0.0000E+00
MCON	0.1900E+01	0.1000E+00	0.0000E+00
MCON	0.1900E+01	0.3000E+00	0.8593E-02
MCON	0.1900E+01	0.5000E+00	0.8497E-01
MCON	0.1900E+01	0.7000E+00	0.9547E-03
MCON	0.1900E+01	0.9000E+00	0.0000E+00
MCON	0.1900E+01	0.1100E+01	0.0000E+00
MCON	0.1900E+01	0.1300E+01	0.0000E+00
MCON	0.1900E+01	0.1500E+01	0.0000E+00
MCON	0.2100E+01	0.1000E+00	0.0000E+00
MCON	0.2100E+01	0.3000E+00	0.1909E-02
MCON	0.2100E+01	0.5000E+00	0.1909E-01
MCON	0.2100E+01	0.7000E+00	0.1909E-02
MCON	0.2100E+01	0.9000E+00	0.0000E+00
MCON	0.2100E+01	0.1100E+01	0.0000E+00
MCON	0.2100E+01	0.1300E+01	0.0000E+00
MCON	0.2100E+01	0.1500E+01	0.0000E+00
MCON	0.2300E+01	0.1000E+00	0.9547E-03
MCON	0.2300E+01	0.3000E+00	0.6206E-01
MCON	0.2300E+01	0.5000E+00	0.6683E-02
MCON	0.2300E+01	0.7000E+00	0.0000E+00
MCON	0.2300E+01	0.9000E+00	0.0000E+00
MCON	0.2300E+01	0.1100E+01	0.0000E+00
MCON	0.2300E+01	0.1300E+01	0.0000E+00
MCON	0.2300E+01	0.1500E+01	0.0000E+00
MCON	0.2500E+01	0.1000E+00	0.4583E-01
MCON	0.2500E+01	0.3000E+00	0.0000E+00
MCON	0.2500E+01	0.5000E+00	0.9547E-03
MCON	0.2500E+01	0.7000E+00	0.2864E-02
MCON	0.2500E+01	0.9000E+00	0.0000E+00
MCON	0.2500E+01	0.1100E+01	0.0000E+00
MCON	0.2500E+01	0.1300E+01	0.0000E+00
MCON	0.2500E+01	0.1500E+01	0.0000E+00
MCON	0.2700E+01	0.1000E+00	0.0000E+00
MCON	0.2700E+01	0.3000E+00	0.0000E+00
MCON	0.2700E+01	0.5000E+00	0.0000E+00
MCON	0.2700E+01	0.7000E+00	0.0000E+00
MCON	0.2700E+01	0.9000E+00	0.2864E-02
MCON	0.2700E+01	0.1100E+01	0.0000E+00
MCON	0.2700E+01	0.1300E+01	0.0000E+00
MCON	0.2700E+01	0.1500E+01	0.0000E+00
MCON	0.2900E+01	0.1000E+00	0.0000E+00
MCON	0.2900E+01	0.3000E+00	0.0000E+00
MCON	0.2900E+01	0.5000E+00	0.0000E+00
MCON	0.2900E+01	0.7000E+00	0.0000E+00
MCON	0.2900E+01	0.9000E+00	0.0000E+00
MCON	0.2900E+01	0.1100E+01	0.4774E-02
MCON	0.2900E+01	0.1300E+01	0.0000E+00
MCON	0.2900E+01	0.1500E+01	0.0000E+00
MCON	0.3100E+01	0.1000E+00	0.0000E+00
MCON	0.3100E+01	0.3000E+00	0.0000E+00
MCON	0.3100E+01	0.5000E+00	0.0000E+00
MCON	0.3100E+01	0.7000E+00	0.0000E+00
MCON	0.3100E+01	0.9000E+00	0.0000E+00
MCON	0.3100E+01	0.1100E+01	0.1909E-02
MCON	0.3100E+01	0.1300E+01	0.0000E+00
MCON	0.3100E+01	0.1500E+01	0.0000E+00

RETURN

fracture concentrations

	x	y	conc
FCON	0.1000E+00	0.3000E+00	0.6751E-01
FCON	0.3000E+00	0.5000E+00	0.5401E+00
FCON	0.5000E+00	0.7000E+00	0.1080E+01
FCON	0.1000E+00	0.1100E+01	0.1350E+01
FCON	0.3000E+00	0.1300E+01	0.1755E+01
FCON	0.1000E+00	0.9000E+00	0.1215E+01
FCON	0.3000E+00	0.7000E+00	0.8776E+00
FCON	0.5000E+00	0.9000E+00	0.0000E+00
FCON	0.7000E+00	0.7000E+00	0.1013E+01
FCON	0.9000E+00	0.5000E+00	0.4726E+00
FCON	0.7000E+00	0.1000E+00	0.0000E+00
FCON	0.9000E+00	0.3000E+00	0.0000E+00
FCON	0.1100E+01	0.5000E+00	0.4726E+00
FCON	0.1300E+01	0.7000E+00	0.5401E+00
FCON	0.1500E+01	0.9000E+00	0.4051E+00
FCON	0.7000E+00	0.1500E+01	0.0000E+00
FCON	0.9000E+00	0.1300E+01	0.0000E+00
FCON	0.1100E+01	0.1100E+01	0.0000E+00
FCON	0.1100E+01	0.7000E+00	0.0000E+00
FCON	0.1300E+01	0.5000E+00	0.5401E+00
FCON	0.1500E+01	0.3000E+00	0.8776E+00
FCON	0.1700E+01	0.1000E+00	0.1350E+00
FCON	0.1700E+01	0.3000E+00	0.1418E+01
FCON	0.1900E+01	0.5000E+00	0.1688E+01
FCON	0.2100E+01	0.7000E+00	0.0000E+00
FCON	0.1700E+01	0.1300E+01	0.0000E+00
FCON	0.1900E+01	0.1100E+01	0.0000E+00
FCON	0.2100E+01	0.9000E+00	0.0000E+00
FCON	0.1900E+01	0.7000E+00	0.0000E+00
FCON	0.2100E+01	0.5000E+00	0.9452E+00
FCON	0.2300E+01	0.3000E+00	0.1958E+01
FCON	0.2500E+01	0.1000E+00	0.3241E+01
FCON	0.2100E+01	0.1100E+01	0.0000E+00
FCON	0.2300E+01	0.1300E+01	0.0000E+00
FCON	0.2500E+01	0.1500E+01	0.0000E+00
FCON	0.2300E+01	0.5000E+00	0.6076E+00
FCON	0.2500E+01	0.7000E+00	0.4051E+00
FCON	0.2700E+01	0.9000E+00	0.4726E+00
FCON	0.2900E+01	0.1100E+01	0.1350E+00
FCON	0.2500E+01	0.1300E+01	0.0000E+00
FCON	0.1100E+01	0.9000E+00	0.0000E+00
FCON	0.1300E+01	0.1100E+01	0.0000E+00
FCON	0.1500E+01	0.1300E+01	0.0000E+00
FCON	0.2900E+01	0.1300E+01	0.0000E+00
FCON	0.3100E+01	0.1100E+01	0.2700E+00
FCON	0.2700E+01	0.1300E+01	0.0000E+00

RETURN

ELEC	0.100E-02	0.608E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
ELEC	0.200E-02	0.108E+01	0.000E+00	0.000E+00	0.608E+00	0.000E+00	0.135E+00
ELEC	0.300E-02	0.101E+01	0.000E+00	0.000E+00	0.101E+01	0.000E+00	0.675E+00
ELEC	0.400E-02	0.608E+00	0.000E+00	0.000E+00	0.540E+00	0.000E+00	0.810E+00
ELEC	0.500E-02	0.810E+00	0.000E+00	0.000E+00	0.122E+01	0.000E+00	0.608E+00
ELEC	0.600E-02	0.878E+00	0.000E+00	0.000E+00	0.675E+00	0.000E+00	0.675E+00
ELEC	0.700E-02	0.128E+01	0.000E+00	0.000E+00	0.608E+00	0.000E+00	0.540E+00
ELEC	0.800E-02	0.135E+01	0.000E+00	0.000E+00	0.878E+00	0.000E+00	0.101E+01
ELEC	0.900E-02	0.878E+00	0.000E+00	0.135E+00	0.878E+00	0.000E+00	0.743E+00
ELEC	0.100E-01	0.128E+01	0.000E+00	0.675E-01	0.743E+00	0.000E+00	0.743E+00
ELEC	0.110E-01	0.135E+01	0.955E-03	0.203E+00	0.142E+01	0.000E+00	0.473E+00
ELEC	0.120E-01	0.101E+01	0.955E-03	0.135E+00	0.108E+01	0.000E+00	0.338E+00
ELEC	0.130E-01	0.108E+01	0.955E-03	0.270E+00	0.115E+01	0.000E+00	0.608E+00
ELEC	0.140E-01	0.101E+01	0.955E-03	0.135E+00	0.540E+00	0.000E+00	0.540E+00
ELEC	0.150E-01	0.128E+01	0.955E-03	0.270E+00	0.115E+01	0.000E+00	0.810E+00
ELEC	0.160E-01	0.675E+00	0.955E-03	0.135E+00	0.608E+00	0.000E+00	0.743E+00
ELEC	0.170E-01	0.945E+00	0.955E-03	0.135E+00	0.405E+00	0.000E+00	0.142E+01
ELEC	0.180E-01	0.149E+01	0.955E-03	0.270E+00	0.878E+00	0.000E+00	0.608E+00
ELEC	0.190E-01	0.101E+00	0.955E-03	0.338E+00	0.608E+00	0.000E+00	0.608E+00
ELEC	0.200E-01	0.142E+01	0.191E-02	0.675E+00	0.878E+00	0.000E+00	0.608E+00
ELEC	0.210E-01	0.149E+00	0.955E-03	0.405E+00	0.101E+01	0.000E+00	0.878E+00
ELEC	0.220E-01	0.101E+01	0.191E-02	0.135E+00	0.608E+00	0.000E+00	0.743E+00
ELEC	0.230E-01	0.115E+01	0.955E-03	0.270E+00	0.108E+01	0.000E+00	0.608E+00
ELEC	0.240E-01	0.149E+01	0.191E-02	0.203E+00	0.810E+00	0.000E+00	0.743E+00
ELEC	0.250E-01	0.878E+00	0.286E-02	0.338E+00	0.810E+00	0.000E+00	0.540E+00
ELEC	0.260E-01	0.101E+01	0.191E-02	0.405E+00	0.473E+00	0.000E+00	0.108E+01
ELEC	0.270E-01	0.128E+01	0.955E-03	0.338E+00	0.128E+01	0.000E+00	0.675E+00
ELEC	0.280E-01	0.101E+01	0.955E-03	0.473E+00	0.108E+01	0.000E+00	0.675E+00
ELEC	0.290E-01	0.743E+00	0.955E-03	0.338E+00	0.743E+00	0.000E+00	0.540E+00
ELEC	0.300E-01	0.945E+00	0.955E-03	0.270E+00	0.810E+00	0.000E+00	0.878E+00
ELEC	0.310E-01	0.115E+01	0.191E-02	0.608E+00	0.878E+00	0.000E+00	0.945E+00
ELEC	0.320E-01	0.878E+00	0.191E-02	0.675E-01	0.878E+00	0.000E+00	0.945E+00
ELEC	0.330E-01	0.945E+00	0.191E-02	0.405E+00	0.945E+00	0.000E+00	0.878E+00
ELEC	0.340E-01	0.101E+01	0.286E-02	0.338E+00	0.338E+00	0.000E+00	0.810E+00
ELEC	0.347E-01	0.878E+00	0.000E+00	0.473E+00	0.473E+00	0.000E+00	0.473E+00

RETURN

```

# frac-unix input for test problem ht-2
# advection and dispersion in a unidirectional flow field
#
# grid geometry
#
  nrow=5
  ncol=11
  nn=nrow*ncol
  igrd=1
  dyz=10.
  dxz=10.
#
# matrix properties
#
# matrix materials
  nmtm=1
  immat(1:nn)=1
# matrix flow properties
  mtconx=0.35795
  mtcony=1.e-15
  mtss=1.e-5
  mtpor=0.001
# matrix/water transport properties
  mtdsyl=4.0          #meters          longitudinal dispersivity
  mtdsyf=0.0          #meters          transvers dispersivity
  thdif=0.01274       #meters**2/day    diffusion
  cpw=4185.           #J/(kg*C)         fluid specific heat
  mtcpx=850.          #J/(kg*C)         rock specific heat
  mtrho=2780.         #kg/meter**3      rock density
  mttcon=172800.      #J/(m*day*C)      rock thermal conductivity
#
# flow boundary conditions
#
  freeze(1:5)=true
  freeze(51:55)=true
#
# initial conditions
#
  hz(1:nn)=0.0
  hz(1:5)=10.0        #grad*K=darcy velocity of 8.21918E-05 m/day
#
# miscellaneous
#
  units=1
  tpx=10.
  tpmx=300
  numcol=5
#
# outputs desired
#
  headpt=true
  plocpt=false
  concpt=false
#
# sourced data
#
@ZERO
#
# define solution variables
  solve=true
  heatp=false
  dynmic=false
#
# begin the steady state flow solution
#
  static

```

```

echo ". . . steady state analysis complete"
#
# source the output statements in PRINTM
@PRINTM
#
# STACK THE PARTICLE MOVE SO IT USES THE STEADY STATE FLOW SOLUTION
#
# particles
npgmx=1
npmx=5000
nseed=87654321.
# line source
nnls=5
inls(1)=1;   inls(2)=2;   inls(3)=3;   inls(4)=4;   inls(5)=5;
olsu=0.001
tpls=30.
#
# exits, code concentration, temperature
#
extbnd=false
#
# outputs desired
#
plocpt=true
headpt=false
concpt=false
#
# time control
#
tinit=0.                # time to begin particle release
tfinal=500.             # time to end particle release
tend=500.                # time to end problem
dtp=50.                  # time step for particle move
#
# define solution variables
#
solve=false
dynmic=false
heatp=true
#
# add variables to debug list for outputting
#
debug t ipot ppznp tpm
#
# begin the particle move, "start" initiates subroutine DODERIV
#
start
echo ". . . integration finished"
#
# terminate FRAC-UNIX
stop

```

```

0, 1 >> 0, 2 >> 0, 3 >> 0, 4 >> 0, 5 >> 0, 6 >> 5;
0, 7 >> 11;
0, 8 >> 55;
0, 9 >> 1;
0, 10 >> 10;
0, 11 >> 10;
0, 12 >> 0, 13 >> 0, 14 >> 0, 15 >> 0, 16 >> 1;
0, 17 >> 1;
0, 18 >> 0, 19 >> 0.35795;
0, 20 >> 1e-15;
0, 21 >> 1e-05;
0, 22 >> 0.001;
0, 23 >> 0, 24 >> 4;
0, 25 >> 0;
0, 26 >> 0.01274;
0, 27 >> 4185;
0, 28 >> 850;
0, 29 >> 2780;
0, 30 >> 172800;
0, 31 >> 0, 32 >> 0, 33 >> 0, 34 >> TRUE;
0, 35 >> TRUE;
0, 36 >> 0, 37 >> 0, 38 >> 0, 39 >> 0;
0, 40 >> 10;
0, 41 >> 0, 42 >> 0, 43 >> 0, 44 >> 1;
0, 45 >> 10;
0, 46 >> 300;
0, 47 >> 5;
0, 48 >> 0, 49 >> 0, 50 >> 0, 51 >> TRUE;
0, 52 >> FALSE;
0, 53 >> FALSE;
0, 54 >> 0, 55 >> 0, 56 >> 0, 57 >> 0, 58 >> 0, 59 >> 0, 60 >> TRUE;
0, 61 >> FALSE;
0, 62 >> FALSE;
0, 63 >> 0, 64 >> 0, 65 >> 0, 66 >>
ITER = 1 0 0.0000E+00 6 0.9002E+03
ITER = 2 9 -0.8142E-11 9 -0.7327E-10
NODE MAX REL DERIV NODE MAX ABS DERI
OCCURENCE OF MAXIMUM REL DERIVS (REL TO NONZERO HEADS)
NODE 9 0
NTIMES 1 0
OCCURENCE OF MAXIMUM ABSOLUTE DERIVATIVES
NODE 6 9
NTIMES 1 1
0, 67 >> . . . steady state analysis complete
0, 68 >> 0, 69 >> 0, 70 >> T

```

0;

H

10;	10;	10;	10;	10;
9;	9;	9;	9;	9;
8;	8;	8;	8;	8;
7;	7;	7;	7;	7;
6;	6;	6;	6;	6;
5;	5;	5;	5;	5;
4;	4;	4;	4;	4;
3;	3;	3;	3;	3;
2;	2;	2;	2;	2;
1;	1;	1;	1;	1;
0;	0;	0;	0;	0;

DH

0;	0;	0;	0;	0;
-1.11022e-11;	-7.21645e-11;	-5.4845e-11;	-7.32747e-11;	-4.88498e-12;
-4.52971e-11;	-2.19824e-11;	-2.9754e-11;	-3.77476e-11;	-4.9738e-11;
7.99361e-12;	7.10543e-12;	1.44329e-11;	1.86517e-11;	9.76996e-12;
-1.33227e-12;	7.77156e-12;	2.66454e-12;	5.32907e-12;	4.44089e-12;


```

4.88498e-12; 1.62093e-11; 1.62093e-11; 1.66533e-11; 2.66454e-12;
2.22045e-12; 2.88658e-12; 1.55431e-12; 6.43929e-12; 1.77636e-12;
4.44089e-12; 5.32907e-12; 4.66294e-12; 7.32747e-12; 4.44089e-12;
4.44089e-13; 2.44249e-12; 6.66134e-13; 2.44249e-12; 4.44089e-13;
-2.66454e-12; 8.88178e-13; -1.55431e-12; -6.66134e-13; -1.33227e-12;
0; 0; 0; 0; 0;
0, 71 >> 0, 72 >> 0, 73 >> 0, 74 >> 0, 75 >> 1;
0, 76 >> 5000;
0, 77 >> 8.76543e+07;
0, 78 >> 0, 79 >> 5;
0, 80 >> 1; 2; 3; 4; 5;
0, 81 >> 0.001;
0, 82 >> 30;
0, 83 >> 0, 84 >> 0, 85 >> 0, 86 >> FALSE;
0, 87 >> 0, 88 >> 0, 89 >> 0, 90 >> TRUE;
0, 91 >> FALSE;
0, 92 >> FALSE;
0, 93 >> 0, 94 >> 0, 95 >> 0, 96 >> 0;
0, 97 >> 500;
0, 98 >> 500;
0, 99 >> 50;
0, 100 >> 0, 101 >> 0, 102 >> 0, 103 >> FALSE;
0, 104 >> FALSE;
0, 105 >> TRUE;
0, 106 >> 0, 107 >> 0, 108 >> 0, 109 >> 0, 110 >> 0, 111 >> 0, 112 >> 0, 113 >>
T
0;

```

IPOT

PPZNP

0;

TPM

10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;

T

50;

IPOT

0;

PPZNP

500;

TPM

11.1905;	11.5656;	11.7539;	11.3533;	10;
10;	10.0059;	10;	10.0068;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;

100;

0;

999;

[illegible]

150;

0;

1500;

[illegible]

200;

0;

2000;

[illegible]

250;

IPOT

0;

PPZNP

2500;

TPM

[illegible]

T

300;

IPOT

0;

PPZNP

2999;

TPM

[illegible]

T

350;

IPOT

0;

PPZNP

3499;

TPM

15.7326;	17.1032;	15.6578;	15.6651;	10;
12.6661;	12.9504;	13.7221;	13.6672;	10;
10.7606;	10.8838;	11.0029;	10.944;	10;
10.069;	10.1625;	10.1295;	10.0785;	10;
10.0212;	10.0579;	10;	10.0046;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;
10;	10;	10;	10;	10;

	10;	10;	10;	10;	10;
T	400;				
IPOT	0;				
PPZNP	3999;				
TPM	15.9805;	15.8739;	16.4726;	16.1861;	10;
	13.6785;	14.6999;	13.9229;	13.6142;	10;
	11.0091;	11.2509;	11.1538;	11.3712;	10;
	10.1854;	10.1405;	10.3398;	10.3021;	10;
	9.99837;	9.994;	10.0439;	10.0458;	10;
	10;	10;	10.0044;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;

T	450;				
IPOT	0;				
PPZNP	4499;				
TPM	16.3211;	17.7128;	16.34;	16.4197;	10;
	13.7723;	14.6111;	14.448;	15.0602;	10;
	11.5466;	11.777;	11.9788;	11.6134;	10;
	10.2837;	10.4413;	10.3393;	10.4734;	10;
	10.0857;	10.1135;	10.0673;	10.0123;	10;
	10;	10.0234;	10.0191;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;

T	500;				
IPOT	0;				
PPZNP	4999;				
TPM	16.7963;	15.697;	16.9195;	15.8453;	10;
	14.7527;	15.9553;	15.1963;	15.3144;	10;
	11.8964;	12.3531;	11.909;	12.3068;	10;
	10.3951;	10.546;	10.7607;	10.7669;	10;
	10.0961;	10.0587;	10.0892;	10.2039;	10;
	10.0052;	10.018;	10.0339;	10.0014;	10;
	10;	10;	10.0206;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;
	10;	10;	10;	10;	10;

10;

10;

10;

10;

10;

PARTICLE MOVEMENT SUMMARY FOR MATRIX

MONTE CARLO MOVES... 0 = 0.0 %

EXPLICIT MOVES...72916 = 100.0 %

ALL MOVES EXC DIFFUS TRANS TO FRACS72916 = 100.0 %

FRACTURE TRANSFER... 0 = 0.0 %

REMAIN IN MATRIX....72916 = 100.0 %

EXIT FROM MODEL..... 0 = 0.0 %

0, 114 >> . . . integration finished

0, 115 >> 0, 116 >> 0, 117 >> *** FRACSL done.

```

# input file for FRAC-UNIX Manual problem case 1
# infinite porous media
#
# geometry
#
nrow=ncol=11
nn=nrow*ncol
igrid=1
dxz=dyz=2.
xexp=yexp=1.2
#
# matrix properties
#
# matrix materials
nmtm=1
immat(1:nn)=1
# matrix flow properties
mtconx=mtcony=7.9
mtss=1.e-5; mtpor=0.25
# matrix/water transport properties
mtdsyl=0.6
mtdsynt=0.
trdif=0.          # no diffusion
#
# fractures
#
nf=0
#
# flow boundary conditions
nodin=1
qqext(2:nn)=0.0
qqext(1)=115.5
qexttr=false
# infinite porous media
permdb=true
symtry=4.
#
# initial conditions
#
hz(1)=100.
hz(2:nn)=8.
#
# miscellaneous
#
units=1
tpn=300.
numcol=5
#
# outputs desired
#
plocpt=false
headpt=true
concpt=false
#
# sourced data
#
@ZERO
#
# define solution variables
#
dynmic=false
solve=true

```

```

#
# begin the steady state flow solution
#
# echo "Starting static analysis of head ..."
# static
# echo ". . . Static analysis complete"
#
# source the output statements in PRINTM
#
# @PRINTM
#
# STACK THE PARTICLE MOVE SO IT USES THE STEADY STATE FLOW SOLUTION
#
# particles
#
#   npgmx=1
#   npmx=1000
#   nseed=87654321.
#
# point source
#
#   nodin=1
#   rwel=0.25
#
# outputs desired
#
#   plocpt=true
#   headpt=false
#   concpt=true
#
# time control
#
#   tsrel=0.
#   terel=.00001
#   tend=0.2
#   dtp=.01
#
# define solution variables
#
#   solve=false
#   dynmic=false
#
# begin the particle move, 'start' initiaes subroutine DODERIV
#
# start
# echo "...move finished"
#
# terminate FRAC-UNIX
# stop

```

```

0, 1 >> 0, 2 >> 0, 3 >> 0, 4 >> 0, 5 >> 0, 6 >> 0, 7 >> 11;
0, 8 >> 121;
0, 9 >> 1;
0, 10 >> 2;
0, 11 >> 1.2;
0, 12 >> 0, 13 >> 0, 14 >> 0, 15 >> 0, 16 >> 1;
0, 17 >> 1;
0, 18 >> 0, 19 >> 7.9;
0, 20 >> 1e-05; 0.25;
0, 21 >> 0, 22 >> 0.6;
0, 23 >> 0;
0, 24 >> 0;
0, 25 >> 0, 26 >> 0, 27 >> 0, 28 >> 0;
0, 29 >> 0, 30 >> 0, 31 >> 1;
0, 32 >> 0;
0, 33 >> 115.5;
0, 34 >> FALSE;
0, 35 >> 0, 36 >> TRUE;
0, 37 >> 4;
0, 38 >> 0, 39 >> 0, 40 >> 0, 41 >> 100;
0, 42 >> 8;
0, 43 >> 0, 44 >> 0, 45 >> 0, 46 >> 1;
0, 47 >> 300;
0, 48 >> 5;
0, 49 >> 0, 50 >> 0, 51 >> 0, 52 >> FALSE;
0, 53 >> TRUE;
0, 54 >> FALSE;
0, 55 >> 0, 56 >> 0, 57 >> 0, 58 >> 0, 59 >> 0, 60 >> 0, 61 >> 0, 62 >> FALSE;
0, 63 >> TRUE;
0, 64 >> 0, 65 >> 0, 66 >> 0, 67 >> Starting static analysis of head ...
0, 68 >>

```


APPENDIX F
UNIX OPERATIONS

CONTENTS

Conventions	F-4
Login and set terminal	F-4
Directories and files	F-4
File creation and editing (vi editor)	F-5
Miscellaneous	F-6
Modify reference code (SCCS system)	F-6
Enter a new subroutine	F-7
Compile options	F-7
Debugging after a system fault	F-8

APPENDIX F
UNIX OPERATIONS

While the enclosed material is relatively limited, it represents the large majority of UNIX commands used in actual operation of FRAC-UNIX. This material is to be supported by frequent reference to the UNIX operating system manual.

Conventions

0	zero
o	letter
<key>	
<key><key>	sequential
<key>+<key>	simultaneous

Login and set terminal

<CR>
<CR>
jdm
password
set term=vt240 or other as appropriate

Directories and files

pwd	print working directory
ls	list directories and files except .files
ls -la	list all directories and files

d	directory
r	user - read
w	write
x	execute
r	group - read
w	write
x	execute
r	other - read
w	write
x	execute

chmod u+w *	add user write permission, all files
chmod o+x *	add other execute (copy) permission
mkdir manf	makes directory under current
cd manf	change directory to subdirectory
pwd	
ls	

cd ..	change directory to parent
cd /u3/jdm/manf	change directory to
ls	
cd ~/manf	change directory to your manf
cp /u3/jdm/manf/* .	copy all jdm/manf files to your manf
chmod u+w *	
ls	
rm plot*	remove plot files

File creation and editing (vi editor)

vi frac.inp	get editor and file
<ctrl>+f	one page forward
<ctrl>+b	one page back
G	end of file
:<zero><CR>	top of file
<pre> k ↑ h ← + → l ↓ j </pre>	cursor, one line at a time
/text/	cursor, forward to text (case sensitive)
?text?	cursor, backward to text (case sensitive)
n	continue search to next occurrence
dd	delete line
5dd	delete 5 lines beginning with current
x	delete character above cursor
i <esc>	insert before cursor
A <esc>	append to end of line
o <esc>	insert a new line under cursor
R <esc>	replace, beginning with cursor
:6,7co9	copy lines 6 and 7 and put after line 9
:6,7m9	move lines 6 and 7 after line 9
:q!	quit without saving changes
:w	write current file - saves changes so far
:wq	write current file and quit

Miscellaneous

grep -i -n qquest *.F *.i | more

<cr>

<ctrl>+c

cpr[]filename

pr -n *.F *.i | cpr []

man pr

kill -9 -1

ps

kill processno

ja -s

ja -c

h

logout

get regular expression; print line number,
case independent, expression; in any .F or .
file, pipe results to more

next page

end more

print specified file

[print options on your system]

pipe a concatenated sequence to print

display manual section on pr

from a second terminal, kill the session

display process numbers

kill a process

summary cost report

cost of each command

history of commands

Modify reference code (SCCS system)

rm namtab.F

remove existing version in working
directory

get -e /u3/jdm/fru/sccs/cray/s.namtab.F

get an editable version of namtab.F

chmod u+w *

vi namtab.F

bring namtab.F into editor

make changes

:wq

write and quit

delta /u3/jdm/fru/sccs/cray/s.namtab.F

Replace existing version with modified
version. You are prompted for
description of change: 7 lines maximum
use \<CR> to advance to next line. End
description with <CR>. Modified file
no longer exists in working directory.

rm del -2.6 /u3/jdm/fru/sccs/cray/s.namtab.F remove last delta - only by author of
delta

Enter a new subroutine (by administrator only)

cd ~/fru/sccs/cray	move to sccs directory
cp ./code/new.F .	copy new subroutine from code
admin -inew.F -s.new.F	create an s file
rm new.F	remove copy
cd ~/code	
rm Makefile	
get -e ~/fru/sccs/cray/s.Makefile	
vi Makefile	add new.o to object list and COMMONS and DELTAS as appropriate
:wq	
delta ~/fru/sccs/cray/s.Makefile	

Compiler Options

Makefile line 6

FFLAGS = -o off -dLp -eD -m0 -V

-o optimization

off no optimization

-d disable options on list below

-e enable options on list below

L	all listings
c	list common block names and lengths
g	list generated code
s	list source code
x	cross reference listing
D	generate symbol table for debugger
f	generates flow-trace for error tracing
o	bounds checking - expensive
p	allows double precision

-m message level

0 all messages

Debugging after a system fault (indefinite, divide by zero)

Makefile must have D enabled

debug -B -d 50 -s ../code/fru >debug.out

writes a summary of all local variables in the last subroutine called, the next previous, etc., and the common blocks to file debug.out

maximum first array dimension is 50

../code/fru is the executable which ran the job