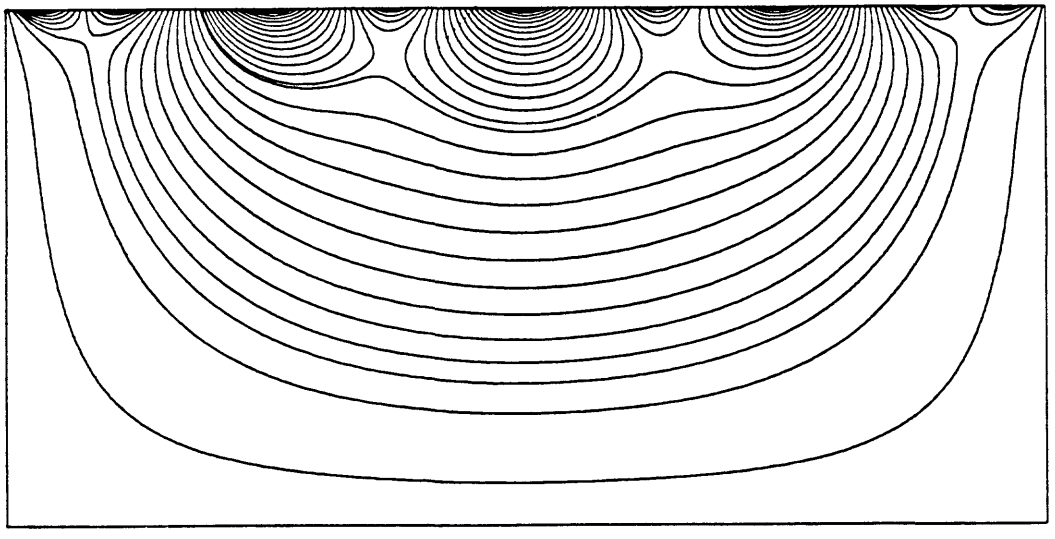
DOCUMENTATION OF COMPUTER PROGRAMS TO COMPUTE AND DISPLAY PATHLINES USING RESULTS FROM THE U. S. GEOLOGICAL SURVEY MODULAR THREE-DIMENSIONAL FINITE-DIFFERENCE GROUND-WATER FLOW MODEL

U. S. GEOLOGICAL SURVEY

Open File Report 89-381



About the Cover

The illustration on the cover of this report shows the local, intermediate, and regional scale flow systems that develop in reponse to sinusoidal variations in water table elevation about an average linear regional gradient. In this figure, a regional ground-water divide exists at the right boundary and the regional discharge area is in the upper left corner of the system. The analysis on which this illustration is based was originally done by Toth (1963) using analytical solutions for the ground-water flow equation. The cover illustration was generated by a particle tracking analysis based on a finite difference solution of case 2f in Toth's paper. The finite-difference solution used 100 cells in the vertical direction and 200 cells in the horizontal direction.

Toth, J., 1963, A theoretical analysis of groundwater flow in small drainage basins, Journal of Geophysical Research, v. 68, no. 16, pp. 4795-4812.

DOCUMENTATION OF COMPUTER PROGRAMS TO COMPUTE AND DISPLAY PATHLINES USING RESULTS FROM THE U. S. GEOLOGICAL SURVEY MODULAR THREE-DIMENSIONAL FINITE-DIFFERENCE GROUND-WATER FLOW MODEL

by David W. Pollock

U. S. GEOLOGICAL SURVEY

Open File Report 89-381

Reston, Virginia
1989

DEPARTMENT OF THE INTERIOR

MANUEL LUJAN, JR., Secretary

U. S. GEOLOGICAL SURVEY

Dallas L. Peck, Director

For additional information write to:

David W. Pollock
U. S. Geological Survey
411 National Center
Reston, VA 22092
(703) 648-5007

This report can be purchased from:

U. S. Geological Survey
Books and Open-File Report Section
Federal Center, Box 25425
Denver, CO 80225
(303)236-7476

PREFACE

This report describes a particle tracking post-processing package for computing three-dimensional path lines using output from steady-state simulations obtained with the U. S. Geological Survey modular three-dimensional finite-difference ground-water flow model. The program is intended for general use and may have to be modified by the user for specific applications. The methodology used in these computer programs is based on specific assumptions and has limitations that must be thoroughly understood to obtain meaningful results. Readers are strongly encouraged to carefully study the sections METHOD and LIMITATIONS before undertaking analyses using these computer programs.

The user is requested to notify the originating office of any errors found in this report or in the computer programs. Updates may occasionally be made to both the report and the computer program. Users who wish to be added to the mailing list to receive updates, if any, may send a request to the following address:

MODPATH
U. S. Geological Survey
411 National Center
Reston, VA 22092

The computer program is available at cost of processing from:

U. S. Geological Survey
WATSTORE Program Office
437 National Center
Reston, VA 22092
Telephone: (703) 648-5686

CONTENTS

Abstract....1

Introduction....2

Method....3

Theory....3

Modifications for special cases....12

Non-rectangular vertical discretization....12

Water table layers....14

Quasi three-dimensional representation of confining layers....14

Boundary conditions and discharge points....15

Backward tracking....18

Limitations....19

Limitations due to underlying assumptions of the method....19

Limitations due to discretization effects....19

Limitations due to uncertainty in parameters and boundary conditions....20

Modpath....22

Organization and structure....22

Flow system files....22

Main data file....22

Stress package data files....24

Cell-by-cell budget file....28

Head file....28

Opening flow system files....28

Output summary of flow field data....29

Interactive input....29

Name of file containing flow system files....29

Output mode....30

Starting location data....31

Direction of tracking computation....34

Criteria for terminating path lines....34

Mass balance check....35

Cell-by-cell data summary....35

Output....35

Execution without interactive input....36

Modpath-plot....37
Organization and structure....37
Flow system files....37
Interactive input and program options....37
Name of file containing flow system files....37
Title....37
Device code....39
Grid....39
Plot type....39
Plot boundaries and scale....40
Zone codes....41
Color plots....41
Border....42
Symbols for stress package features....42
River cells....42
Wells, drains, and general head boundaries....42
Sample problem....43
References....67
Appendix I -- Input for main data file....68
Appendix II -- Input for array utility subroutines....71
Appendix III -- Input for stress package data files....72
Appendix IV -- Output files....76
Appendix V -- Sample problem data files....79
Appendix VI -- Listing of fortran computer codes....82

ILLUSTRATIONS

Page

Figure 1. Finite-difference cell showing definitions of x-y-z and i-j-k coordinate systems and face flow terms. 4

2. Schematic illustration of the computation of exit point and time of travel for a particle in a two-dimensional cell. 8

3. Flow chart for the particle tracking algorithm. 10

4. Additional combinations of velocities at cell-face pairs. 11

5. Schematic illustration of a finite-difference representation of an inclined aquifer with variable thickness. 13

6 Definition of local vertical coordinates for the cases of true three dimensional and quasi-three-dimensional systems 16

7. Flow chart for MODPATH main program and DRIVER subroutine. 23

8. Definition of IFACE for a finite-difference cell 25

9. Example of how boundary fluxes are assigned to cell faces. 26

10. Possible arrangements of particles on a cell face using the automatic generation option. 33

11. Flow chart for MODPATH-PLOT main program and DRIVER subroutine 38

12. Block diagram showing hydrogeology, geometry, and boundary conditions for sample problem. 44

13. Cross-section along row 14 showing path lines and time of travel points for backward tracking analysis 50

14. Map view of recharge locations at the water table for an array of particles tracked backwards from cell (14, 14, 4). 54

15. Map view of recharge locations for particles starting at the water table that discharge to cell (14, 14, 4) 60

16. Map view showing locations of particles after 0 , 30 , and 60 years, based on a forward tracking time series analysis 66

TABLES

Page

Table 1. Interactive dialogue from MODPATH for cross-sectional path line analysis..... 45

2. Interactive dialogue from MODPATH-PLOT for the cross-section plot..... 48

3. Interactive dialogue from MODPATH for the backward endpoint analysis..... 51

4. Interactive dialogue from MODPATH-PLOT for the backward endpoint analysis..... 53

5. Interactive dalogue from MODPATH for the forward endpoint analysis..... 56

6. Interactive dialogue from MODPATH-PLOT for the forward endpoint analysis..... 58

7. Interactive dialogue from MODPATH for time series analysis..... 62

8. Interactive dialogue from MODPATH-PLOT for time series analysis..... 64

CONVERSION FACTORS

The numerical examples in this report use units of feet and days. The following factors are provided for conversion to some other commonly used units of measure:

Multiply	By	To obtain
foot (ft)	0.3048	meter (m)
day (d)	86,400	second (s)
day (d)	1,440	minute (min)
cubic foot (ft^3)	7.48	gallon (g)

ABSTRACT

A particle tracking post-processing package was developed to compute three-dimensional path lines based on output from steady-state simulations obtained with the U. S. Geological Survey modular three-dimensional finite-difference ground-water flow model. The package consists of two FORTRAN 77 computer programs: (1) MODPATH, which calculates pathlines, and (2) MODPATH-PLOT, which presents results graphically.

MODPATH uses a semi-analytical particle tracking scheme. The method is based on the assumption that each directional velocity component varies linearly within a grid cell in its own coordinate direction. This assumption allows an analytical expression to be obtained describing the flow path within a grid cell. Given the initial position of a particle anywhere in a cell, the coordinates of any other point along its path line within the cell, and the time of travel between them, can be computed directly.

Data is input to MODPATH and MODPATH-PLOT through a combination of files and interactive dialogue. Examples of how to use MODPATH and MODPATH-PLOT are provided for a sample problem. Listings of the computer codes and detailed descriptions of input data format and program options are also presented.

INTRODUCTION

The use of particle tracking techniques to generate path lines and time-of-travel information from the results of numerical models can be extremely helpful in analyzing complex two- and three-dimensional ground-water flow systems. Particle tracking schemes have been incorporated directly into solute transport models to account for the advective component of transport (Konikow and Bredehoeft, 1978; Prickett and others, 1981). Particle tracking also has been used in "advection " models to generate path lines and thereby provide a simple means of evaluating the advective transport characteristics of ground-water systems (Garabedian and Konikow, 1983; Mandle and Kontis, 1986; Shafer, 1987). Advection models cannot be used to compute solute concentrations in ground water because they do not account for the effect of mixing by dispersion. However, advection models represent a valuable intermediate step between ground-water flow models and advection-dispersion solute transport models. The model described in this report is an advection model that is designed to compute path lines in three-dimensional, steady state ground-water flow systems.

The computer program MODPATH is a three-dimensional particle tracking post-processing program designed for use with output from steady-state flow simulations obtained using the USGS modular three-dimensional finite-difference ground-water flow model (McDonald and Harbaugh, 1988). MODPATH can be used to compute three-dimensional path lines and the position of particles at specified points in time. MODPATH also computes discharge point coordinates and the total time of travel for each particle. MODPATH-PLOT generates graphical output using results from MODPATH. Both programs are written in FORTRAN 77. MODPATH references only standard FORTRAN 77 library routines and should require little or no modification to run on any system with a FORTRAN 77 compiler. In contrast, MODPATH-PLOT uses the DISSPLA[1] graphics library subroutines (Computer Associates, 1981) and can run only on systems that have access to the DISSPLA subroutine library.

This report describes the organization, structure, and use of the computer programs MODPATH and MODPATH-PLOT. It also describes the theory and implementation of the particle tracking algorithm. Additional discussion of the particle tracking algorithm is presented in Pollock (1988). A sample problem is included to help illustrate input, output, and the basic ways in which the model can be applied. As is the case with any method of analysis, the particle tracking procedure implemented in MODPATH is based on a specific set of assumptions that must be understood and obeyed in order to obtain meaningful results. Readers are strongly encouraged to carefully study the sections *METHOD* and *LIMITATIONS* before undertaking analyses using MODPATH and MODPATH-PLOT.

---

[1] The use of trade and firm names in this report is for identification purposes only and does not constitute endorsement by the U. S. Geological Survey.

METHOD

THEORY

The partial differential equation describing conservation of mass in a steady-state, three-dimensional ground-water flow system can be expressed as,

$$\partial(n\, v_x)/\partial x + \partial(n\, v_y)/\partial y + \partial(n\, v_z)/\partial z = W \qquad (1)$$

where v_x, v_y, and v_z are the principal components of the average linear ground-water velocity vector, n is porosity, and W is the volume rate of water created or consumed by internal sources and sinks per unit volume of aquifer. Equation 1 expresses conservation of mass for an infinitesimally small volume of aquifer. The finite difference approximation of equation 1 can be thought of as a mass balance equation for a finite-sized cell of aquifer that accounts for water flowing into and out of the cell, and for water generated or consumed within the cell. Figure 1 shows a finite-sized cell of aquifer and the components of inflow and outflow across its six faces.

In the discussion that follows, the six cell faces are referred to as x_1, x_2, y_1, y_2, z_1, and z_2. Face x_1 is the face perpendicular to the x direction at $x = x_1$. Similar definitions hold for the other five faces. The average linear velocity component across each face in cell (i,j,k) is obtained by dividing the volume flow rate across the face by the cross sectional area of the face and the porosity of the material in the cell,

$$v_{x_1} = Q_{x_1}/(n\, \Delta y\, \Delta z), \qquad v_{x_2} = Q_{x_2}/(n\, \Delta y\, \Delta z) \qquad (2a, 2b$$

$$v_{y_1} = Q_{y_1}/(n\, \Delta x\, \Delta z), \qquad v_{y_2} = Q_{y_2}/(n\, \Delta x\, \Delta z) \qquad (2c,2d)$$

$$v_{z_1} = Q_{z_1}/(n\, \Delta x\, \Delta y), \qquad v_{z_2} = Q_{z_2}/(n\, \Delta x\, \Delta y) \qquad (2e,2f)$$

where Q is a volume flow rate across a cell face, and Δx, Δy, and Δz are the dimensions of the cell in the respective coordinate directions. If flow to internal sources or sinks within the cell is specified as Q_S, the following mass balance equation can be written for the cell,

$$(n\, v_{x_2} - n\, v_{x_1})/\Delta x + (n\, v_{y_2} - n\, v_{y_1})/\Delta y + (n\, v_{z_2} - n\, v_{z_1})/\Delta z = Q_S/\Delta x\, \Delta y\, \Delta z \qquad (3)$$

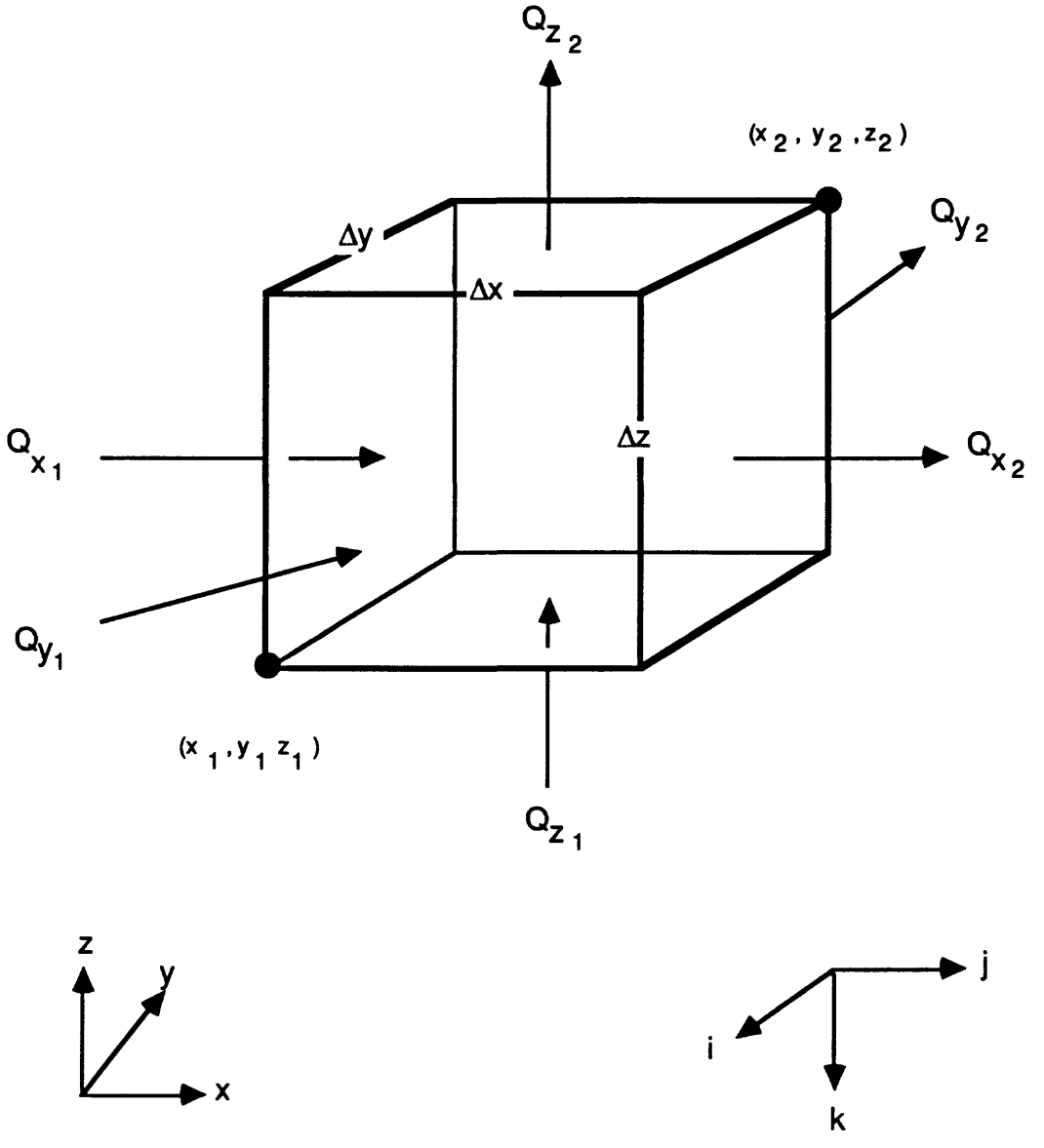


Figure 1. --- Finite-difference cell showing definitions of x-y-z and i-j-k coordinate systems and face flow terms.

The left side of equation 3 represents the net volume rate of outflow per unit volume of the cell, and the right side represents the net volume rate of production per unit volume due to internal sources and sinks. Substitution of Darcy's law for each of the flow terms in equation 3 results in a set of algebraic equations expressed in terms of heads at nodes located at the cell centers . The solution of that set of algebraic equations yields the values of head at the node points. Once the head solution has been obtained, the intercell flow rates can be computed from Darcy's law using the values of head at the node points. The U. S. Geological Survey modular three-dimensional finite-difference ground-water flow model (McDonald and Harbaugh, 1988) solves for head and calculates intercell flow rates.

In order to compute path lines, a method must be established to compute values of the principal components of the velocity vector at every point in the flow field based on the intercell flow rates from the finite difference model. The algorithm described in this report uses simple linear interpolation to compute the principal velocity components at points within a cell. Using simple linear interpolation, the principal velocity components can be expressed in the form,

$$v_x = A_x (x - x_1) + v_{x_1} \tag{4a}$$

$$v_y = A_y (y - y_1) + v_{y_1} \tag{4b}$$

$$v_z = A_z (z - z_1) + v_{z_1} \tag{4c}$$

where A_x, A_y, and A_z are constants that correspond to the components of the velocity gradient within the cell,

$$A_x = (v_{x_2} - v_{x_1})/\Delta x \tag{5a}$$

$$A_y = (v_{y_2} - v_{y_1})/\Delta y \tag{5b}$$

$$A_z = (v_{z_2} - v_{z_1})/\Delta z \tag{5c}$$

Linear interpolation produces a continuous velocity vector field within each individual cell that identically satisfies the differential conservation of mass equation (equation 1) everywhere within the cell. That point can be illustrated by noting that when the linear velocity component functions (equations 4a-4c) are substituted into equation 1, the three derivatives on the left side of equation 1 become constants that are identically equal to the three terms on the left side of equation 3 (provided that porosity is considered constant within a cell). Consequently, linear interpolation of the six cell face velocity components results in a velocity vector field within the cell that automatically satisfies equation 1 at every point inside the cell, if it is assumed that internal sources or sinks are considered to be uniformly distributed within the cell. The fact that the velocity vector field within each cell satisfies the differential mass balance equation assures that

path lines will distribute water throughout the flow field in a way that is consistent with the overall movement of water in the system as indicated by the solution of the finite-difference flow equations.

Consider the movement of a particle, p, through a three-dimensional finite-difference cell. The rate of change in the particle's x-component of velocity as it moves through the cell is given by,

$$(dv_x/dt)_p = (dv_x/dx)\,(dx/dt)_p \tag{6}$$

To simplify notation, the subscript, p, is used to indicate that a term is evaluated at the location of the particle [denoted by the x-y-z coordinates (x_p, y_p, z_p)]. For example, the term, $(dv_x/dt)_p$, is the time rate of change in the x-component of velocity evaluated at the location of the particle.

In equation (6), the term $(dx/dt)_p$ is the time rate of change of the x-location of the particle. By definition,

$$vx_p = (dx/dt)p \tag{7}$$

where v_{x_p} is the x-component of velocity for the particle. Differentiating equation (4a) with respect to x yields the additional relation,

$$(dv_x/dx) = A_x \tag{8}$$

Substituting equations (7) and (8) into equation (6) gives,

$$(dv_x/dt)_p = A_x\, v_{x_p} \tag{9a}$$

Analogous equations are obtained for the y and z directions,

$$(dv_y/dt)p = Ay\, v_{y_p} \tag{9b}$$

$$(dv_z/dt)p = A_z\, v_{z_p} \tag{9c}$$

Equations (9a) through (9c) can be rearranged to the form,

$$(1/v_{x_p})\, d(v_{x_p}) = A_x\, dt \tag{10}$$

Equation (10) can be integrated and evaluated between times t_1 and t_2, $(t_2 > t_1)$ to give,

$$\ln [v_{x_p}(t_2)/v_{x_p}(t_1)] = A_x\, \Delta t \tag{11}$$

where $\Delta t = t_2 - t_1$. By taking the exponential of each side of equation (11), substituting equation (4a) for $v_{x_p}(t_2)$, and rearranging, we obtain,

$$x_p(t_2) = x_1 + (1/A_x)[v_{x_p}(t_1) \exp(A_x \Delta t) - v_{x_1}] \quad (12a)$$

Analogous equations can be developed for the y and z directions,

$$y_p(t_2) = y_1 + (1/A_y)[v_{y_p}(t_1) \exp(A_y \Delta t) - v_{y_1}] \quad (12b)$$

$$z_p(t_2) = z_1 + (1/A_z)[v_{z_p}(t_1) \exp(A_z \Delta t) - v_{z_1}] \quad (12c)$$

The velocity components $v_{x_p}(t_1)$, $v_{y_p}(t_1)$, and $v_{z_p}(t_1)$ are known functions of the particle's coordinates at time t_1, consequently the coordinates of the particle at any future time (t_2) can be computed directly from equations (12a) through (12c).

For steady-state flow, the direct integration method described above can be imbedded in a simple algorithm that allows a particle's exit point from a cell to be determined directly given any known starting location within the cell. To illustrate the method, consider the two-dimensional example shown in figure 2. Cell (i,j) is in the x-y plane and contains a particle, P, located at (x_p,y_p) at time t_p. For this example, it is assumed that v_{x_1} and v_{x_2} are greater than zero. That is, water flows into the cell through face x_1 and out of the cell through face x_2. Similarly, it is assumed that v_{y_1} and v_{y_2} are also greater than zero, so that water flows into the cell through face y_1 and out of the cell through face y_2.

The first step is to determine the face across which particle P passes as it leaves cell (i,j). For the present example, this is accomplished by noting that the velocity components at the four faces require that particle P leave the cell through either face x_2 or face y_2. Consider the x-direction first. From equation (4a) v_{x_p} can be calculated at the point (x_p,y_p). Since we also know v_x equals v_{x_2} at face x_2, equation 11 can be used to determine the time that would be required for particle P to reach face x_2,

$$\Delta t_x = (1/A_x) \ln(v_{x_2}/v_{x_p}) \quad (13a)$$

An analogous calculation can be made to determine the time required for particle P to reach face y_2,

$$\Delta t_y = (1/A_y) \ln(v_{y_2}/v_{y_p}) \quad (13b)$$

where v_{x_p} and v_{y_p} are the x and y components of velocity of particle P at (x_p,y_p). If Δt_x is less than Δt_y, particle P will leave the cell across face x_2 and enter cell (i,j+1). Conversely, if Δt_y is less than Δt_x, particle P will leave the cell across face y_2 and enter cell (i-1,j). A third possibility is that Δt_x and Δt_y are equal, in which case particle P would leave through the corner of cell (i,j) and enter cell (i-1,j+1). The particle trajectory shown in figure 2 corresponds to a situation where Δt_y is less than Δt_x. The length of

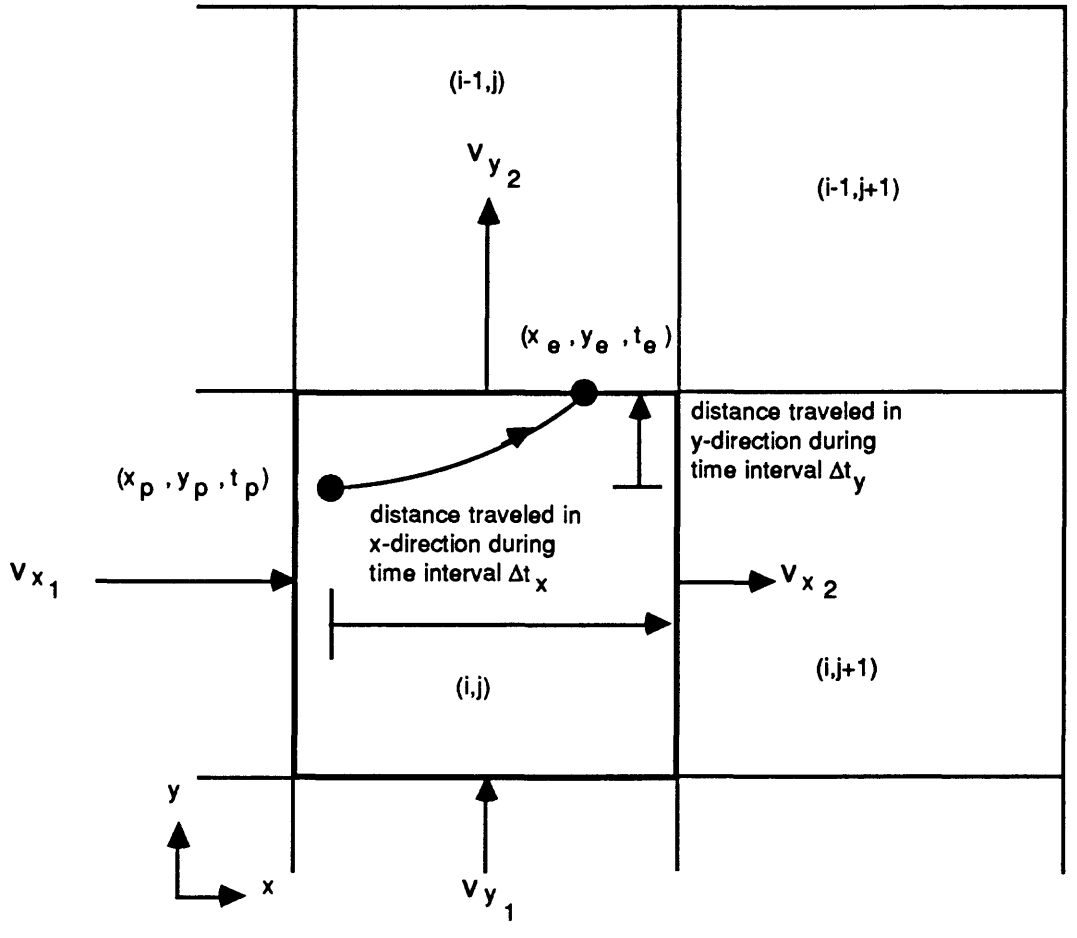


Figure 2. --- Schematic illustration of the computation of exit point and time of travel for a particle in a two-dimensional cell.

time required for particle P to travel from point (x_p,y_p) to a boundary face of cell (i,j) is taken to be the smaller of Δt_x and Δt_y, and is denoted as Δt_e. The value Δt_e is then used in equations (12a) through (12c) to determine the exit coordinates (x_e,y_e) for particle P as it leaves cell (i,j),

$$x_e = x_1 + (1/A_x)[v_{x_p}(t_p) \exp(A_x \Delta t_e) - v_{x_1}] \tag{14a}$$

and

$$y_e = y_1 + (1/A_y)[v_{y_p}(t_p) \exp(A_y \Delta t_e) - v_{y_1}] \tag{14b}$$

The time at which particle P leaves the cell is given by: $t_e = t_p + \Delta t_e$. This sequence of calculations is repeated, cell by cell, until the particle reaches a discharge point. The approach can be generalized to three dimensions in a straight forward way by performing all of the calculations for the z-direction in addition to the x- and y-directions A flow chart outlining the algorithm for a steady-state flow system is presented in figure 3.

It is often desirable to calculate the location of a particle at specific points in time that will not generally correspond to those points in time at which the particle passes from one cell to another. For these cases, the coordinates of a particle at any intermediate time can be computed directly from equations (12a) through (12c) using an appropriate value for Δt within the range 0 to Δt_e. It should be emphasized, however, that the use of specific time steps is solely for convenience. The method described in this report does not require discrete time steps. In addition, the accuracy of pathline computations for steady state flow fields is not affected by time step size because pathlines are integrated analytically with respect to time. The ultimate discharge point and total time of travel for a given particle will be identical regardless of how many (if any) discrete time steps are taken.

For the purposes of illustration, the preceding example considered a specific case where all of the velocity components at the cell faces were non-zero and in the positive x or positive y directions. Of course, those conditions will not always exist. Figure 4 illustrates the other possible situations that can occur in any of the three coordinate directions. Figure 4a shows the case where v_{x_1} and v_{x_2} are in opposite directions and flow is into the cell through both faces x_1 and x_2. For this case, it is obvious that once a particle enters the cell, it cannot leave the cell in the x-direction. When implementing this algorithm, a check is made to determine if this condition exists for a given coordinate direction. If so, a flag is set to indicate that the particle cannot leave the cell across either of the faces in that direction. When this situation prevails in all three coordinate directions, it indicates that a strong sink is present within the cell and no outflow can occur. Figure 4b shows a second alternative in which v_{x_1} and v_{x_2} are in opposite directions and flow is out of the cell through faces x_1 and x_2. This condition implies that a local flow divide exists for the x-direction somewhere within the cell. For this situation, the potential exit face in the x-direction is

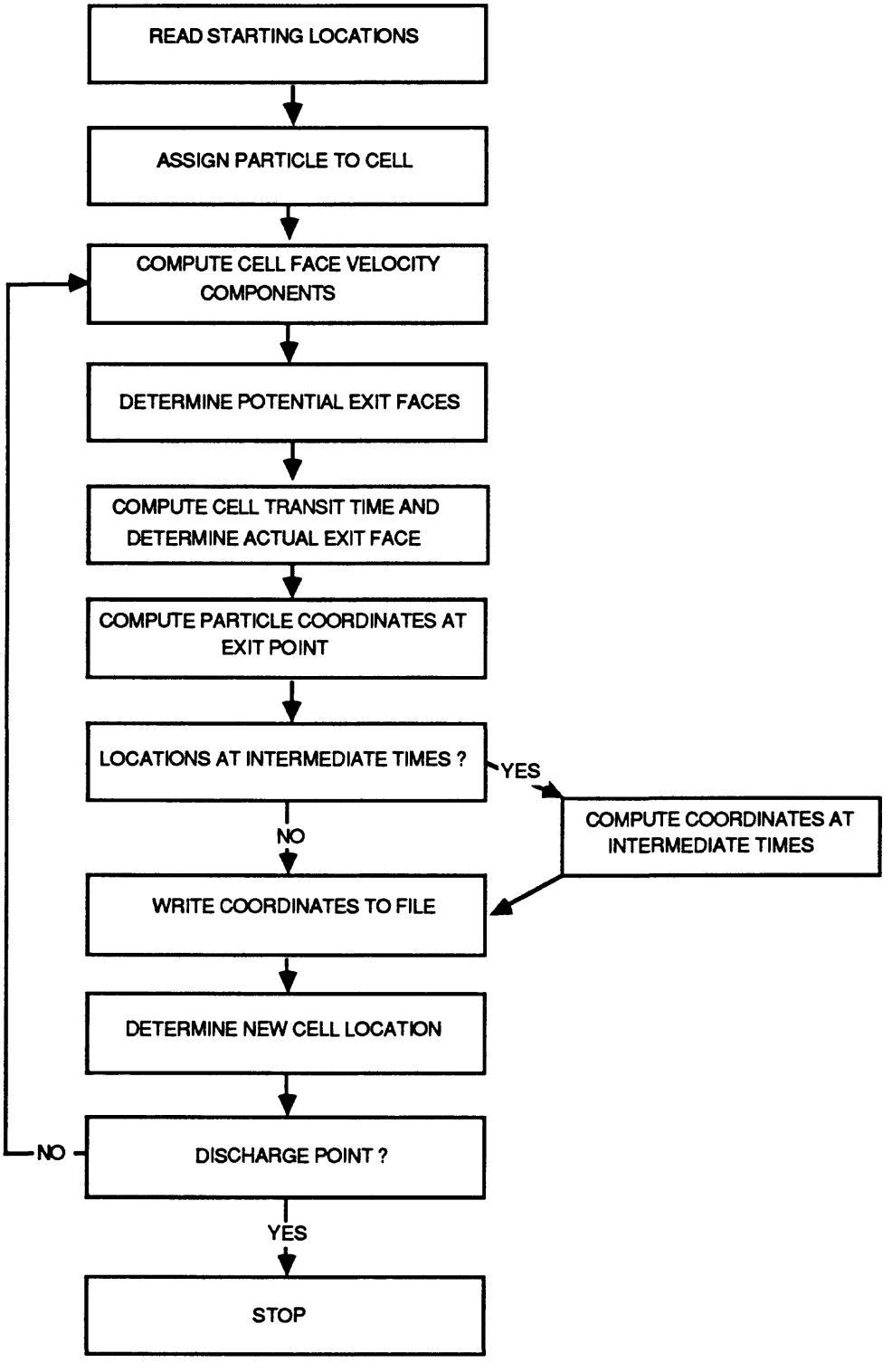


Figure 3. --- Flow chart for the particle tracking algorithm.

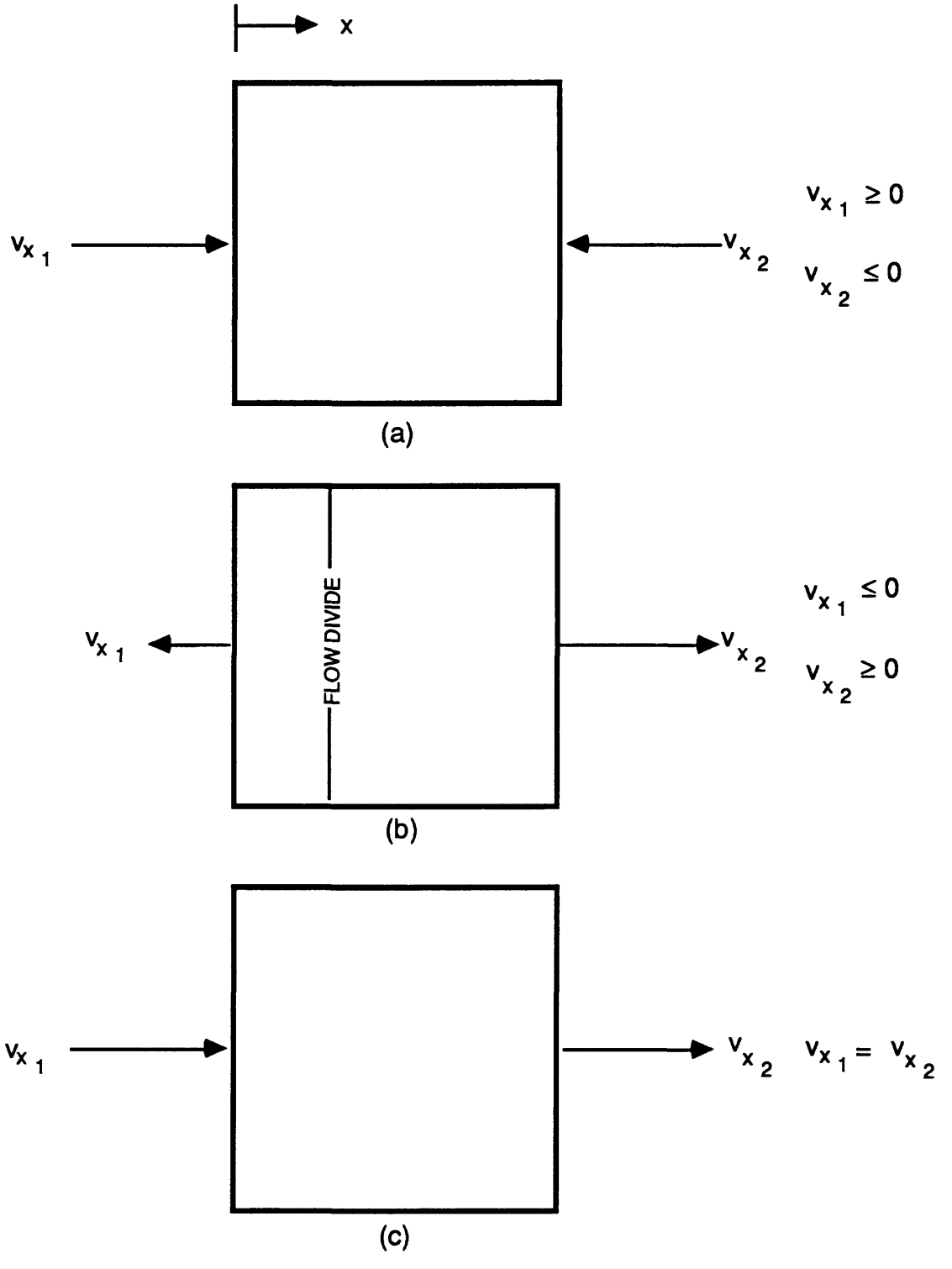


Figure 4. --- Additional combinations of velocities at cell-face pairs.

determined by checking the sign of v_{x_p}. If v_{x_p} is less than zero, the particle has the potential to leave the cell across face x_1. On the other hand, if v_{x_p} is greater than zero, the particle has the potential to leave the cell in the x-direction only through face x_2. Once the appropriate potential exit face has been determined for a coordinate direction, the transit time for that direction can be computed as outlined in the preceding discussion. Finally, the case where v_{x_1} is non-zero and equal to v_{x_2} (figure 4c) also must be considered a special case because the quotient ln(1)/0 that results from the formulation of equation (13a) is indeterminate and cannot be computed. When that is the case, equation (13a) is bypassed and the transit time in the x-direction is computed from the simple relations,

$$\Delta t_x = (x_2 - x_p)/v_{x_1}, \quad (v_{x_1} > 0) \tag{15a}$$

or

$$\Delta t_x = (x_1 - x_p)/v_{x_1}, \quad (v_{x_1} < 0) \tag{15b}$$

MODIFICATIONS FOR SPECIAL CASES

The basic algorithm described in the preceding section has been adapted in the computer program MODPATH to deal with three special cases:

1. grids with non-rectangular vertical discretization
2. water table layers
3. quasi 3-D representation of confining layers

Non-Rectangular Vertical Discretization

The development presented in the previous section was based on the assumption that the flow domain was discretized into a three-dimensional rectangular grid of horizontal rectangular cells. However, in practice, many three-dimensional finite difference simulations use a rectangular grid in the horizontal plane and a deformed grid in the vertical direction to allow grid cells to conform to stratigraphic units that vary in thickness and are not perfectly horizontal. The particle tracking algorithm described above can be used to compute approximate path lines for deformed, or "stratigraphic", three-dimensional grids. Figure 5a shows a hydrogeologic system that has been discretized using deformed cells that conform to the stratigraphy. Figures 5b and 5c show how deformed finite-difference cells are represented in the particle tracking algorithm.

Cells are assumed to be horizontal and rectangular with top and bottom elevations equal to the top and bottom elevations of the cell at the node. A local coordinate, z_L, can be defined for each cell as,

$$z_L = (z - z_1)/(z_2 - z_1) \tag{16}$$

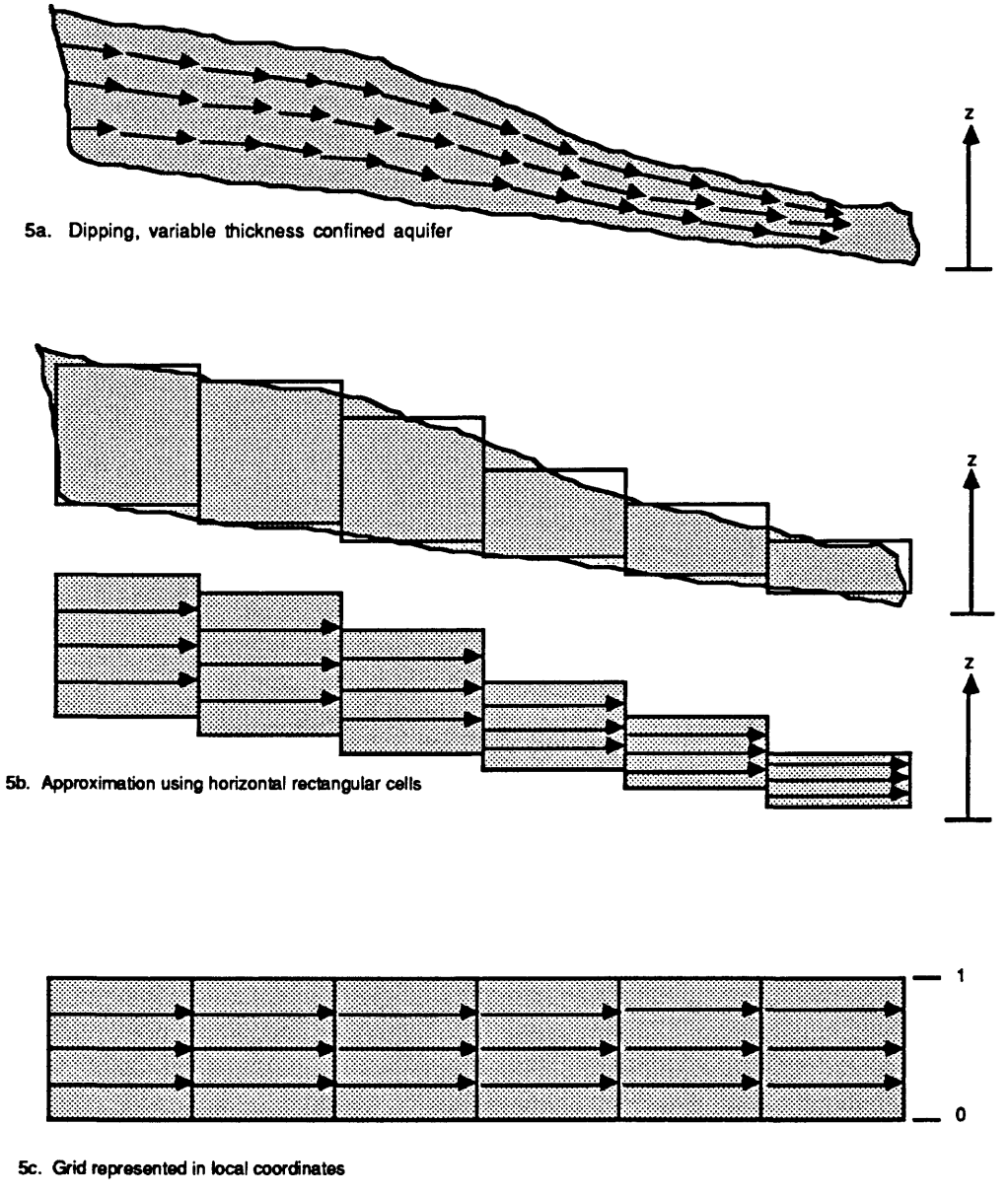


Figure 5. --- Schematic illustration of a finite-difference representation of an inclined aquifer with variable thickness.

where z_1 and z_2 are the elevations of the bottom and top of the cell, respectively. According to equation (16), the local z-coordinate equals 0 at the bottom of the cell and 1 at the top of the cell. When a particle is transferred laterally from one cell to another, its local z-coordinate remains the same. That is, if a particle leaves a cell at a position half way between the top and bottom of the cell, it is assumed to enter the neighboring cell half way between the top and bottom of that cell, regardless of how the thickness or absolute elevation of the layer changes from one cell to the next. This procedure is illustrated schematically in figure 5 for the case of lateral flow in a confined aquifer of variable thickness and dip. When all layers are constant in thickness and horizontal, this approach reduces exactly to the algorithm developed above for true rectangular grids.

The advantage of a stratigraphic three-dimensional grid is that complex hydrogeologic systems can be simulated with fewer layers than would be necessary to adequately represent them with a three-dimensional rectangular grid. The principal disadvantage is that spatial discretization errors are introduced that are difficult to quantify, especially with respect to path line computations. Nevertheless, the method described above produces results that are at least semi-quantitative for flow models based on stratigraphic vertical discretization.

Water Table Layers

For water table layers the saturated thickness of cells changes areally in relation to the slope of the water table and the bottom elevation of cells within the layer. The top elevation of a cell in a water table layer is set equal to the head in the cell. Consequently, water table layers vary in thickness even for true three-dimensional rectangular grids. The particle tracking algorithm treats water table layers in the same way as the variable thickness stratigraphic layers described in the preceding section.

Quasi 3-D Representation of Confining Layers

Ground water systems typically are characterized by the presence of highly transmissive, subhorizontal aquifers separated from one another in the vertical by confining layers of much lower transmissivity. Because of the large contrast in hydraulic conductivity between aquifers and confining layers, ground water flow in these systems is predominantly lateral in aquifers and vertical through confining layers. In these systems, confining layers function primarily as low-conductivity vertical connections between aquifer layers. Confining layers often are not simulated as active layers in finite-difference models. Instead, their effect on vertical flow between aquifers is accounted for implicitly by computing the effective vertical hydraulic conductance between aquifers based on the vertical conductivity and thickness of the confining layers. This approach is referred to as a quasi three-dimensional representation. In MODPATH, each unsimulated confining layer is assumed to be part of the active model layer directly above it. For cells with underlying confining layers, the local z-coordinate within the confining layer varies linearly from -1 at the

base of the confining layer to 0 at the top of the confining layer (Figure 6). It is assumed that one-dimensional, steady state, vertical flow exists throughout the confining layer. That assumption implies that the average vertical linear velocity is constant throughout the confining layer and that its magnitude equals the volumetric flow rate between adjacent model layers divided by the area of the cell and the porosity of the confining layer. When a particle reaches a top or bottom face of a cell that is recognized to be a boundary of a confining layer, the particle is moved vertically across the confining layer into the next active model layer. Time of travel across the confining layer is computed by dividing the thickness of the confining layer by the average vertical linear velocity within the confining layer. A value for the porosity of the confining layer must be specified to compute travel time across the layer.

BOUNDARY CONDITIONS AND DISCHARGE POINTS

The intercell flow rates for all active cells are computed and stored as output from the Block Centered Flow (BCF) package of the modular flow model. Those values are then input to the MODPATH where they are used to compute cell face velocity components. Special consideration often is necessary for cells that incorporate boundary conditions. The modular flow model accounts for three types of boundary conditions:

1) Specified head at a node
2) Specified flow rate to or from a cell
3) head-dependent flow rate to or from a cell

Specified head cells act as net sources or sinks of water to the flow system. For the purposes of path line computations, specified head cells are treated as active cells that contain a net source or sink. Path lines computed through an isolated specified head cell surrounded entirely by variable head cells are consistent with the existence of a uniformly distributed source or sink of water within the cell. When a number of specified head cells are adjacent to one another, path lines computed through those cells usually will be misleading and inappropriate due to the fact that the modular flow model does not compute rates of flow between adjacent specified head cells. Consequently, path lines in the vicinity of specified head cells should always be examined critically to make sure that the specified head cells are not exerting an unrealistic artificial constraint on direction or time of travel.

Finite-difference flow equations represent volumetric water balances for individual grid cells. At the scale of an individual grid cell, the finite-difference representation of the ground-water flow equation contains no information about the spatial distribution of specific components of flow into or out of the cell. For example, the finite-difference flow equations cannot distinguish between the case where water flows into a cell across a boundary face and the case where water is injected into the cell at the same rate by a well with no water flowing across the boundary face. For this reason, specified flux boundaries are accounted for in the modular flow model by placing wells in cells that contain flux boundaries. In the preceding example the

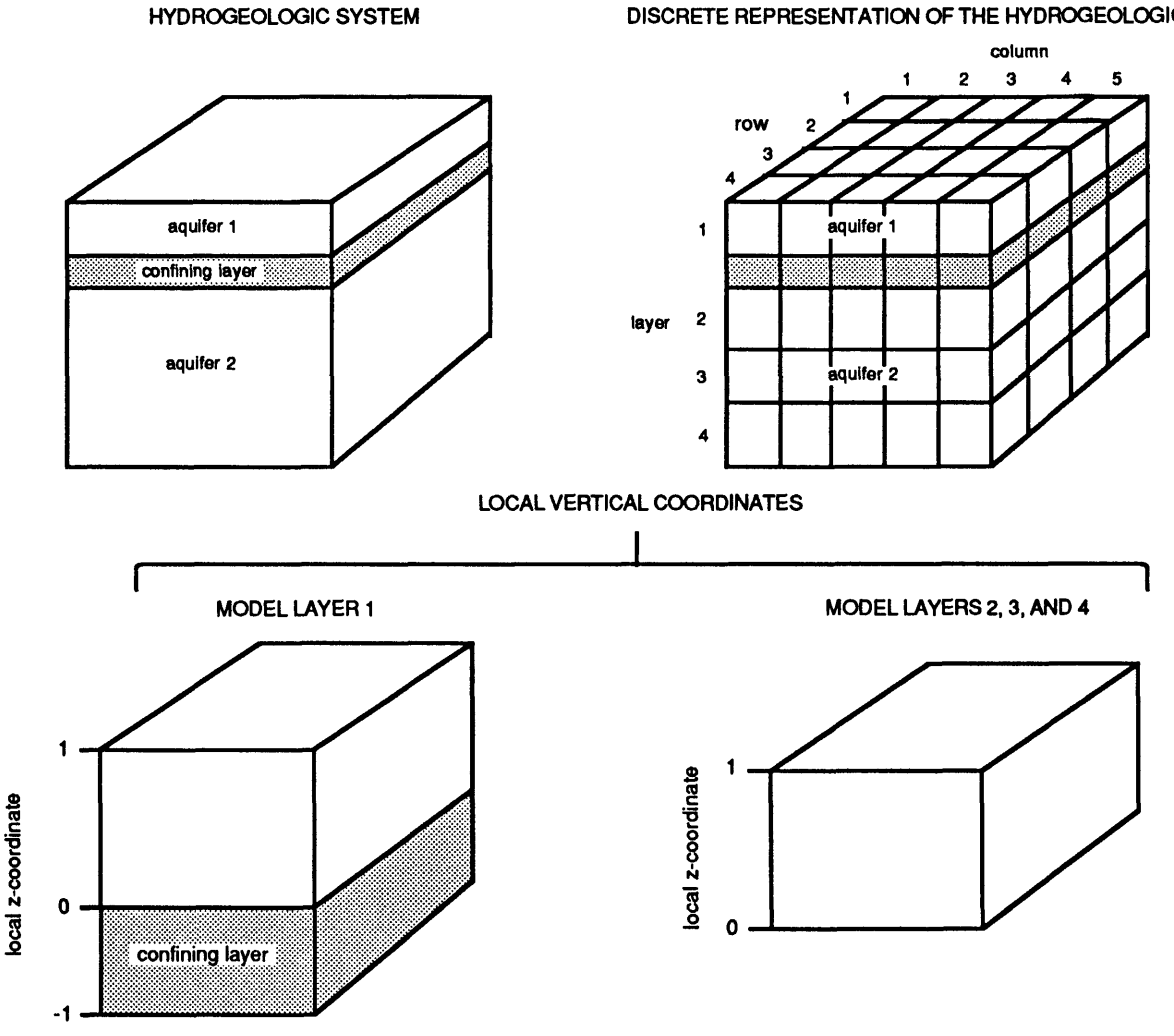


Figure 6. -- Definition of local vertical coordinates for the cases of true three dimensional and quasi-three dimensional systems.

head distributions resulting from the solution of the finite-difference equations would be identical. In contrast, path lines computed for these two cases depend strongly on the distribution of flow within the cell.

Previously, during the development of the velocity interpolation algorithm, it was shown that the simple linear velocity interpolation scheme used in MODPATH implicitly accounts for any internal source or sink (such as a well) as if it was uniformly distributed throughout the volume of the cell. However, when a well is used to represent flow across a specific boundary, a more accurate representation of path lines sometimes can be obtained by explicitly assigning that rate of flow to one or more boundary faces of the cell. These two approaches can be thought of as extrapolations of the results of the finite-difference flow model in the sense that both are equally consistent with the head distribution and the volumetric water balance produced by the flow model. The decision about whether one of these interpretations is more realistic than the other for a particular system is entirely a matter of professional judgement. Often, neither approach is able to accurately represent flow paths at the scale of an individual cell because the actual distribution of wells or boundary flows is much more complex than can be represented at the given level of finite-difference discretization.

In MODPATH, the cell face across which flow occurs may be specified for wells that represent boundary fluxes; the appropriate velocity component is then computed and assigned to that face. For those wells that represent true internal sources or sinks, rather than boundary flows, no cell face is specified and the well is treated as a distributed source or sink. In addition to wells, the recharge package in the modular flow model provides another way to specify flux at a boundary. In MODPATH, recharge may be assigned to the top face of a cell, or treated as a distributed source. The distributed source approximation is usually only appropriate for two-dimensional areal flow models. Because specified fluxes are part of the input to the modular flow model, they are entered directly into MODPATH using the same data sets developed for the well and recharge packages of the modular flow model.

The modular flow model includes options to simulate rivers, drains, general boundary fluxes, and evapotranspiration as head-dependent fluxes. Unlike specified fluxes, head-dependent fluxes are part of the solution to the flow problem, and, therefore, must be stored as budget output from the modular flow model. Those values then are entered as data to MODPATH and either assigned to a specific cell face or treated as distributed sources or sinks. In the case of evapotranspiration, fluxes can be assigned to the top face of a cell, or treated as a distributed sink.

Cells that contain internal sinks are flagged as potential discharge points. When a particle enters the flow system, it moves through the system until it reaches a boundary where flow is out of the system, or until it enters a cell containing an internal sink. If the sink is sufficiently strong, flow will be into the cell from all directions. In that case, every particle that enters the cell discharges to the sink. If the sink is weak,

it is possible that only part of the water flowing into the cell discharges to the sink. When a particle enters a cell containing a weak sink, there is no way of determining whether that particular particle should discharge to the sink or pass through the cell. In MODPATH, the user has the option of (1) stopping particles when they enter cells with any amount of discharge to internal sinks, (2) letting particles pass through cells with weak sinks, so that they will discharge only at discharge boundaries or strong sink cells, or (3) stopping particles when they enter cells in which discharge to sinks is larger than a specified fraction of the total inflow to the cells.

BACKWARD TRACKING

If all velocity terms are multiplied by -1, the tracking algorithm can be operated "in reverse" to track particles backwards along their path lines. This option can be useful for determining recharge source areas for localized zones of discharge, such as well fields. When operated in reverse, particles are terminated when they reach inflow boundaries or when they enter cells containing strong internal sources.

LIMITATIONS

MODPATH has a number of limitations that must be understood if it is to be used effectively. These limitations are related to (1) underlying assumptions in the particle tracking scheme, (2) discretization effects, and (3) uncertainty in parameters and boundary conditions.

LIMITATIONS DUE TO UNDERLYING ASSUMPTIONS OF THE METHOD

Perhaps the most important limitation of MODPATH is its inability to deal with transient flow systems. This restriction is not due to any inherent limitation of the method for transient systems, but results instead from the desire to design a path line analysis model that will execute efficiently as a post-processor for the modular flow model with a manageable amount of numerical output from the finite difference flow model.

The semi-analytical particle tracking method used in MODPATH is valid only for the simple linear velocity interpolation scheme described in the section THEORY. The method is a consistent approach for computing and interpolating velocities from intercell flow rates for the standard 7-point, three-dimensional block-centered finite difference approximation of the ground-water flow equation (such as the USGS modular three-dimensional ground-water flow model with the standard "block-centered flow" package). The method cannot be used to compute path lines for other types of numerical approximations of the flow equation, such as lattice-centered finite difference grids and finite element models.

LIMITATIONS DUE TO DISCRETIZATION EFFECTS

The accuracy of numerically-generated path lines, and a proper interpretation of what they represent, depends on the extent to which ground-water systems can be realistically represented by discrete networks of finite-difference cells. The degree of spatial discretization in a finite-difference model influences (1) the level of detail at which hydrogeologic and system boundaries can be represented, (2) the accuracy of velocity calculations, and (3) the ability to accurately and unambiguously represent internal sinks. Often, a level of spatial discretization that is adequate for a flow simulation analysis oriented toward water supply may not be adequate for a path line analysis.

The effect of spatial discretization on the representation of internal sinks is especially important for particle tracking analyses because of the ambiguity associated with the movement of particles through weak sink cells. These cells contain sinks that do not discharge at a large enough rate to consume all of the water entering the cell. The net result is a flow-through cell in which water enters the cell across some faces and leaves it across others. Path lines computed for these cells are consistent with the assumption of a uniformly distributed sink within the cell; however, it is difficult to interpret the results of particle tracking analyses in systems with weak sink cells because:

1. There is no way to know whether a specific particle should discharge to the sink or pass through the cell. That means individual particles will not correspond to a fixed volume of water, nor will flow tubes defined by adjacent pathlines represent a fixed quantity of flow.
2. Path lines through weak sink cells may not accurately represent the path of any water in the system if they contain point sinks that cannot be represented accurately as being uniformly distributed throughout the cells.

These problems are a direct result of spatial discretization that is too coarse. Using a finer grid may eliminate the problem by turning weak sink cells into strong sink cells that clearly correspond to discharge points for all particles entering those cells. From a practical point of view, however, it usually is impossible to entirely avoid weak sinks when simulating real systems.

A common example of the weak sink dilemma occurs in two-dimensional areal simulations that account for the effect of rivers. In areal simulations, rivers often are represented as distributed sinks or sources of water within cells. In many systems, shallow ground water discharges to the river while deeper ground water flows underneath the river and discharges elsewhere. The shallow and deep flow systems cannot be distinguished from one another because of the averaging effect of the areal model; the averaging results in a flow through cell in which the river acts as a weak sink. This type of model cannot be used to quantitatively delineate the zone of contribution for recharge water to a well near a river with an underflow component because it is impossible to determine which particles discharge to the river and which pass under the river and enter the well. In spite of that important limitation, particle tracking still may be able to provide useful, but more qualitative, information about the zone of contribution to the well. Although this example illustrated the problem posed by rivers in areal models, it is only one example of how discretization can affect path line computations. The important point is that discretization characteristics of the flow model place major, unavoidable constraints on particle tracking analyses and the conclusions that can be drawn from them.

Even when a cell contains a strong sink, the finite difference discretization in the immediate vicinity of the sink will not be adequate to accurately describe the pattern of flow near the sink. The only way to improve the accuracy of path line computations near internal sources and sinks is to refine the grid of the finite-difference flow model.

LIMITATIONS DUE TO UNCERTAINTY IN PARAMETERS AND BOUNDARY CONDITIONS

So far, all of the limitations that have been discussed relate to discretization effects and to underlying assumptions of the methodology. In fact, the most important limitation in any ground water analysis is the uncertainty in boundary conditions and hydrogeologic parameters used to define the system, and particle tracking is no exception. Models are always idealized approximations of reality. At best, a particle tracking

analysis only provides information about how water moves in the idealized system described by the model. The degree to which the model accurately represents the real system places additional constraints on interpreting the results of a particle tracking analysis beyond those relating to discretization effects and limitations of the method.

ORGANIZATION AND STRUCTURE

The primary purpose of the main program in MODPATH is to perform start-up operations such as opening data files and allocating space for arrays. As in the modular flow model, array data in MODPATH is stored in a single master array dimensioned as an unlabeled common block in the main program. Once files have been opened and array space allocated, the execution of the program is turned over to a subroutine named DRIVER that controls the overall sequence of computations. Flow charts for the main program and for DRIVER are shown in Figure 7. Descriptions of procedures and decision points are enclosed in rectangular boxes. Subroutines are designated by ovals. Only the major subroutines referenced by the main program and DRIVER are shown. Many of those subroutines call additional subroutines that do not appear on the flow chart.

Data is input to MODPATH through a combination of files and interactive input. Files contain the basic information about geometry, boundary conditions, and cell-by-cell budget terms required to compute the velocity vector field. These files are referred to as "flow system" files, and predominantly consist of input and output files of the modular flow model. The interactive input is devoted primarily to selecting options for the particle tracking analysis to indicate where particles will be located and what type of output is required.

FLOW SYSTEM FILES

The flow system files provide the information necessary to compute the velocity vector field for the flow system and locate internal sinks that represent potential discharge points for particles. The flow system files consist of:

1) Main data file.

2) Stress package data files from the modular flow model,including any auxiliary files referenced by the primary stress package data files.

3) Cell-by-cell budget files.

4) Heads stored as unformatted output from the modular flow model.

Main Data File

The main data file is the only flow system file that is not derived predominantly from an existing modular flow model data file. It contains information about grid size and geometry, FORTRAN unit numbers for data files, and porosity data. Input instructions and definitions of variables for the main data file are presented in Appendix I. Array data is input using utility subroutines patterned after those in the modular flow model. Data formats for the array input utility subroutines are summarized in Appendix II.

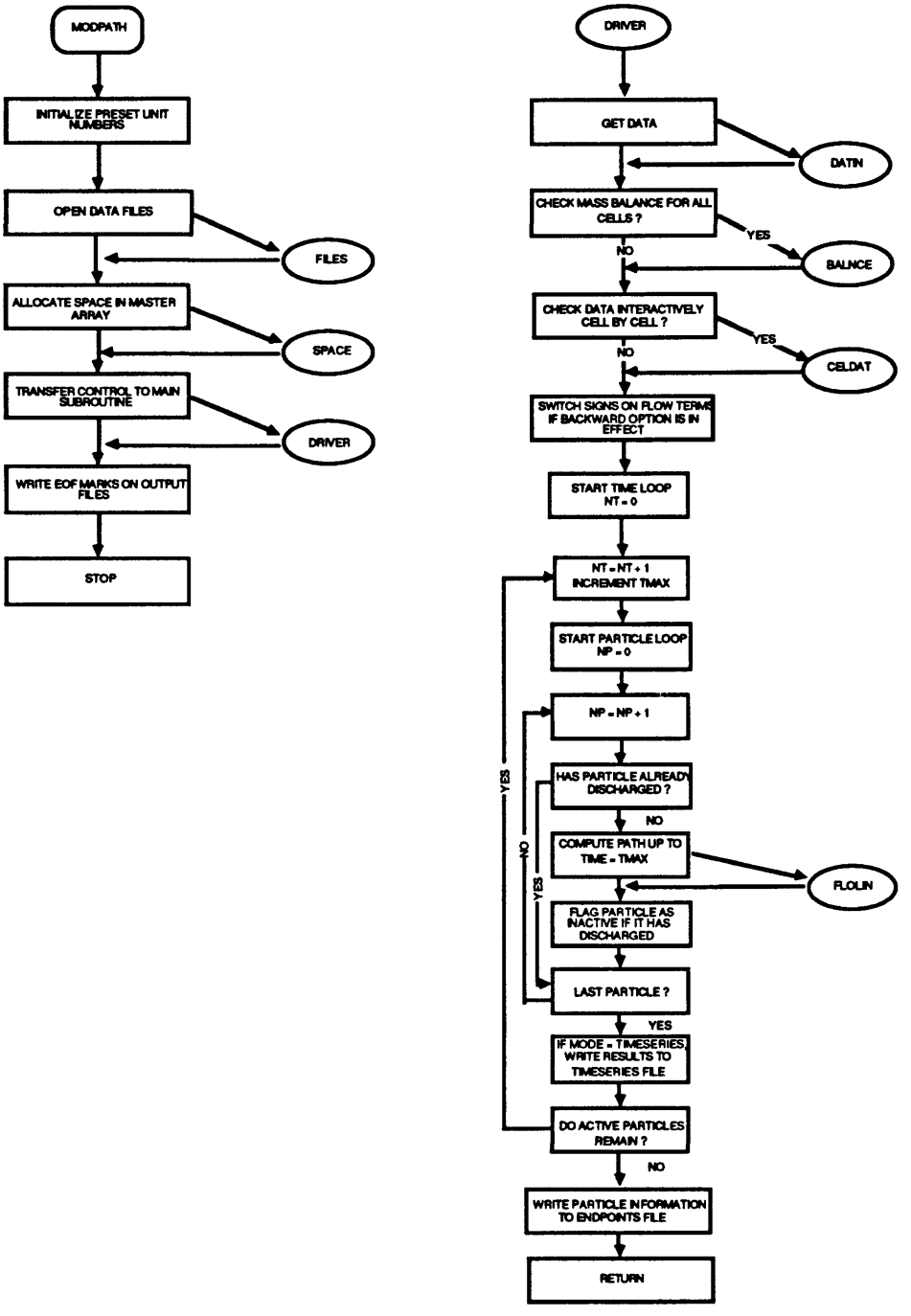


Figure 7. --- Flow chart for MODPATH main program and DRIVER subroutine

Stress Package Data Files

MODPATH requires data files for all of the stress package options used in a simulation analysis. MODPATH reads the modular flow model stress package data files. Input instructions and variable definitions for the stress package data sets are presented in Appendix III. Stress package data sets must be modified to indicate whether individual flow terms are to be treated as internal sources and sinks, or assigned to specific faces of the cell. In the case of list-oriented stress packages (well, river, drain, and general head boundary), flow terms can be designated as internal sources and sinks, or they can be assigned to any of the six cell faces by entering an additional integer variable, IFACE, at the end of each data record. IFACE is defined as:

IFACE = 0 or IFACE > 6	flow term is treated as internal source or sink.
IFACE = 1 to 6	flow term is assigned to cell face corresponding to a number 1 through 6.
IFACE < 0	flow term is apportioned uniformly among faces perpendicular to the horizontal plane that form boundaries with inactive cells (IBOUND=0). This is accomplished by assigning a single average velocity to the "boundary" faces. The average velocity is computed by dividing the total flow rate by porosity and the total cross sectional area of the boundary faces. If none of the faces are boundaries with inactive cells, the flow term is treated as an internal source or sink.

The cell faces that correspond to numbers 1 through 6 are shown in Figure 8.

To illustrate the use of IFACE, consider Figure 9 which shows a portion of a model layer along an irregular boundary with variable, specified flux across the boundary. In this case, wells would be used to account for the boundary fluxes. Well flow rates are computed by multiplying the flux by the area of the cell face. If the layer is 100 feet thick and DELR is 50 feet and DELC is 100 feet, the boundary fluxes could be accounted for with the following entries in the well data set:

layer	row	column	Q	IFACE
1	3	1	0.005	4
1	2	2	0.010	1
1	2	2	0.005	4
1	1	3	0.010	1
1	1	3	0.010	4

By convention, flow into a cell is positive, flow out of a cell is negative. MODPATH converts these flow rates to corresponding values of velocity at the specified cell faces.

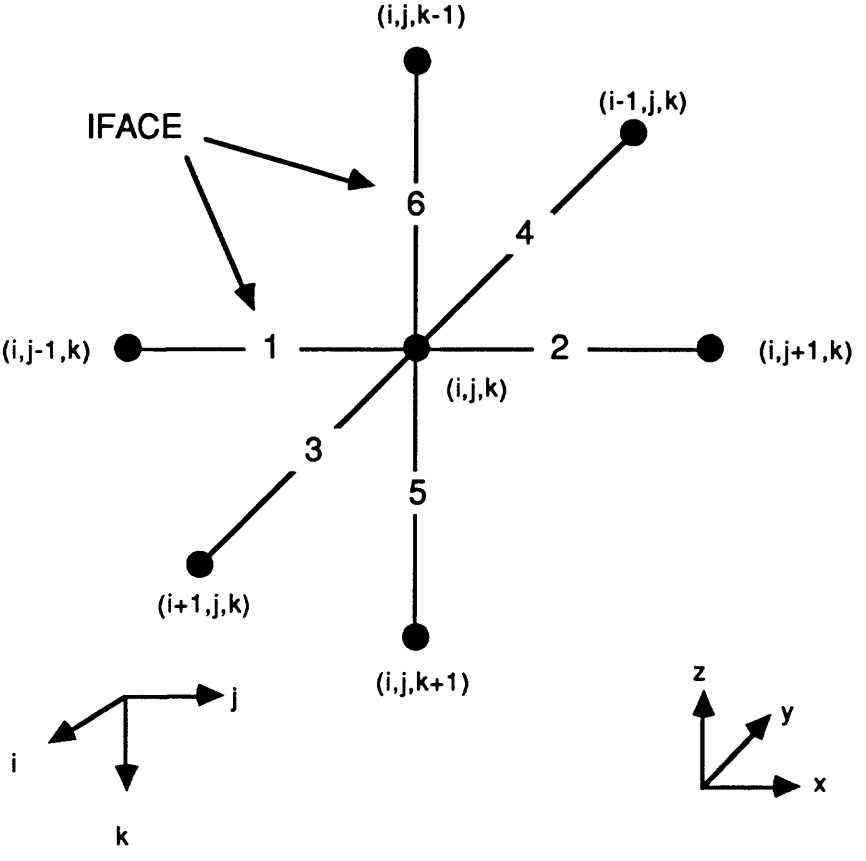


Figure 8. --- Definition of IFACE for a finite-difference cell.

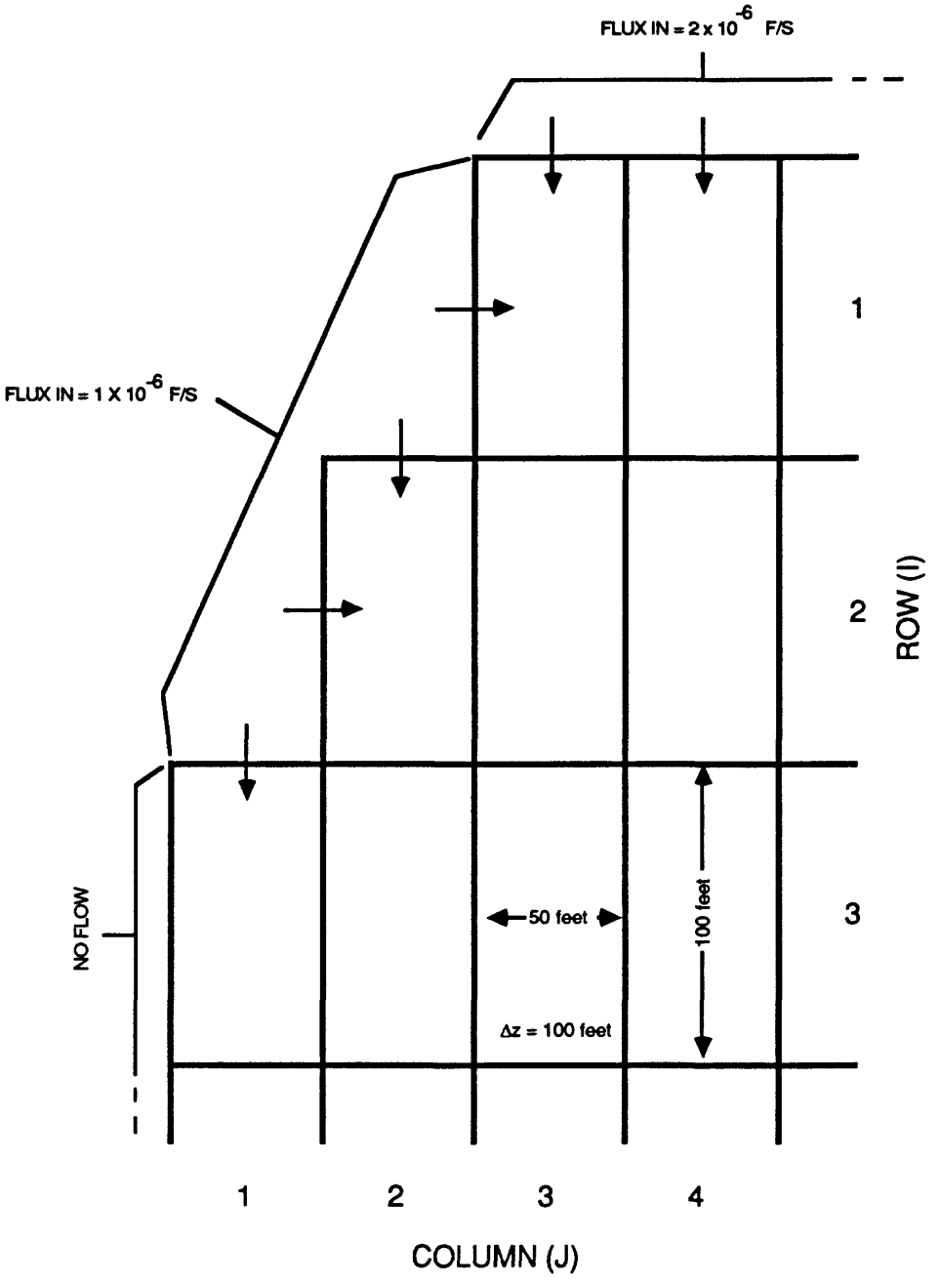


Figure 9. --- Example of how boundary fluxes are assigned to cell faces.

The same result could be obtained with the following data set:

layer	row	column	Q	IFACE
1	3	1	0.005	4
1	2	2	0.015	-1
1	1	3	0.010	1
1	1	3	0.010	4

In this case, MODPATH would determine how to distribute flow terms among the faces. For the cell in row 2, column 2, the program would determine that faces 1 and 4 form boundaries with inactive cells. The total length of those cell faces would be computed (100 + 50 = 150 feet), and the flow term would be uniformly apportioned between faces 1 and 4 as:

$$\text{Face 1: } Q = 0.015 \times (100/150) = 0.01 \text{ cfs}$$

and

$$\text{Face 4: } Q = 0.015 \times (50/150) = 0.005 \text{ cfs.}$$

Often it is convenient for grids with highly irregular boundaries to set IFACE equal to a negative number and let the computer program determine the appropriate cell face for each flow term. However, for extremely complex boundary configurations with variable flux distributions, it may be necessary to specify some of the cell faces explicitly to assure that velocities are distributed correctly. For example, the cell in row 1, column 3, requires two entries to achieve the desired velocity distribution.

The use of IFACE to designate how river, drain, and general head boundary package flow terms are assigned within cells is identical to that of the well package. However, in contrast to the well package, the flow terms for these three stress packages are not specified explicitly in the data set, but instead are computed as part of the solution to the flow problem and stored as output from the modular flow model. A single flow term is stored for each cell. When more than one river reach, drain, or general head boundary is specified for a single cell, the flow term computed and stored by the finite-difference flow model for that stress package represents a composite flow. MODPATH does not redistribute the composite flow among the multiple entries within the cell. Instead, MODPATH takes the value of IFACE for the first data entry for the cell and distributes the composite flow for that cell accordingly. For example, if two drains are placed in the same cell, and the first drain is given an IFACE value of 0 and the second is assigned a value of 1, the total composite flow to drains in that cell will be treated as discharge to an internal sink because the first entry encountered in the drain package data file had an IFACE value equal to 0.

Only two options for distributing flow terms are provided for the array-oriented stress packages, recharge and evapotranspiration. For these packages, a variable ITOP is defined as:

ITOP = 0; flow terms are treated as internal sources and sinks for all cells.

ITOP = 1; flow terms are assigned to the top face of all cells.

Input instructions are provided for these data sets in Appendix III.

It is important to remember that the recharge and evapotranspiration data sets may reference auxiliary data files containing array data. When that is the case, those data sets also must be supplied to MODPATH.

Cell-By-Cell Budget File

The cell-by-cell budget file for the Block Centered Flow (BCF) package is required for all cases. Budget files for the river, drain, general head boundary, and evapotranspiration packages also are required whenever those packages are used in a flow simulation. Budget files are not required for the well and recharge packages because those flow terms are specified explicitly in the stress package data sets. Cell-by-cell budget terms are stored as unformatted output by the modular flow model when the appropriate output flags are specified for the modular flow model. The cell-by-cell flow terms for the BCF package and the individual stress packages may be stored in a single file or in separate files.

Head File

In order to compute the saturated thickness and vertical coordinates within water table layers, MODPATH requires that the heads for all unconfined and unconfined/confined model layers (LAYCON = 1, 2, or 3) be stored as unformatted output from the modular flow model. Heads may be stored by setting the appropriate flags in the output control data file of the modular flow model. MODPATH sets up an array containing heads at every cell, but only uses those heads in unconfined or confined/unconfined layers. Heads are set equal to zero in layers for which no head data was stored. If desired, heads may be stored and input to MODPATH for confined layers even though those values are not used by the program.

Opening Flow System Files

All of the files used by MODPATH are opened internally within the program. Names and FORTRAN unit numbers are supplied to MODPATH by the user in the form of an additional data file that contains a one line data record for each flow system file. The data record is free-format and consists of a unit number, followed by a space or comma, followed by the file name in single quotes. Any valid unit number for a given operating system may be specified, with the exception of numbers 101 through 109, which are reserved for internal use by the computer program. The first entry in this file must correspond to the main data file; the remaining flow system files can be entered in any order. Consider a simple example of a system with recharge, wells, and rivers. The file containing the list of names and unit numbers might look like:

```
  5   'MAIN.DATA'
  6   'RECHARGE.DATA'
  7   'WELL.DATA'
  8   'RIVER.DATA'
 -9   'RIVER.BUDGET'
-10   'BCF.BUDGET'
-11   'HEADS'
```

MODPATH prompts the user for the name of this file and opens it internally within the program. The unit numbers specified for the stress package data files, the BCF budget file, and the head file must match those specified in the IUNIT array for data group 2 of the main data file. The fortran unit numbers for the stress package budget files must correspond to the unit numbers specified on the first line of each stress package data file. Unformatted binary budget and head files are flagged by specifying a negative unit number.

Output Summary of Flow Field Data

A summary of the flow field data is printed in a file named SUMMARY.PTH as the flow system files are read. Each time a data set is about to be read, a message to that effect is written in the summary file. If the values of an array are set equal to a single constant, its value is printed in SUMMARY.PTH. Otherwise, the option exists to print or suppress the array values according to the flag set in the control record for the array input subroutine (see Appendix II). If the output of array values is suppressed, a message is printed in SUMMARY.PTH after the array has been read indicating that the data was successfully obtained. This approach is helpful in the early stages of an analysis when it often is necessary to locate errors in the input data.

INTERACTIVE INPUT

In most cases, several different kinds of particle tracking analyses will be performed during the course of a study. Therefore, for convenience and speed, particle tracking program options are specified interactively. In this section, the interactive input for MODPATH is described in approximately the same order as it is entered in the program.

Name of File Containing Flow System Files

MODPATH prompts the user to enter the name of the file containing the list of flow system files and unit numbers. The data is read and space is allocated for arrays based on grid discretization data in the main data file. The remainder of the space in the master array is used for a number of arrays that are dimensioned equal to the maximum number of particles. The value of the maximum number of particles for a given run is then printed at the terminal. The maximum number of particles that can be accommodated depends on the size and type of finite-difference grid and the length to which the master array is dimensioned in MODPATH.

Output Mode

In order to optimize the output for various types of graphical options provided by MODPATH-PLOT, one of three types of output can be specified in response to the prompt:

```
SELECT THE MODE FOR STORING OUTPUT DATA:
  0 = ENDPOINTS AND STARTING POINTS ONLY
  1 = PATH LINE COORDINATES
  2 = TIME SERIES DATA
```

MODE 0: When mode 0 is selected, only the endpoint and starting point data for each particle is recorded in a file named ENDPOINT. The ENDPOINT file is generated for modes 0, 1, and 2. Mode 0 produces the minimum amount of output from MODPATH. It is most useful when the main objective of an analysis is to map recharge areas to specified discharge zones. (Discharge zones are defined in the plotting program MODPATH-PLOT using the IBOUND array, which can be modified interactively in MODPATH-PLOT to define zones). A detailed description of the structure and format of the ENDPOINT file is presented in Appendix IV.

MODE 1: Coordinates along the path of each particle are recorded in a file named PATHLINE. The PATHLINE file contains the starting coordinates of a particle and the coordinates at every point where a particle enters a new cell or a confining layer. In addition, coordinates of intermediate points are recorded whenever the cumulative travel time corresponds to a point in time for which a data point was requested. A detailed description of the structure and format of the PATHLINE file is presented in Appendix IV.

The user is asked whether particle locations should be computed and recorded at specific points in time along the path line. If the response is yes, MODPATH prompts the user to indicate how the time data will be specified. Two options are provided:

1) Times can be computed based on a constant time step size.

   If this option is selected, the time step and a units conversion factor are entered interactively. The specified time step size is multiplied by the units conversion factor to obtain the time step size in the same time unit used in the modular flow model.

2) Time data can be read from a file.

   When this option is selected, the user is prompted to enter a file name. The file has the following format:

   Line 1: Number of time periods for which distinct time step data will be entered.

variable:	NPER
format:	I10

NPER lines: Time step data for the specified time period

variables:	PERLEN	NSTP	TSMULT	TFAC
format:	F10.0	I10	F10.0	F10.0

PERLEN = length of the time period

NSTP = number of time steps within the time period

TSMULT = geometric multiplication factor for increasing time step length within the time period

TFAC = units conversion factor for converting the value input for PERLEN to the time units used by the finite difference flow model.

For the semi-analytical integration scheme used in MODPATH, time step size has no affect on the accuracy of the path line computation for steady state flow fields.

MODE 2: The locations of particles at specified points in time are computed and recorded in a file named TIMESERS. In this mode, the locations of all particles are computed for a specified point in time and recorded in sequence in file TIMESERS. The procedure is repeated at each point in time for which output is specified. Mode 2 produces a series of particle locations that are stacked in time in file TIMESERS. When points in TIMESERS are plotted, the effect is to follow the progress of a group of particles as a series of snapshots in time. A detailed description of the structure and format of the timeseries file is presented in Appendix IV. For mode 2, MODPATH prompts the user to indicate how the time data will be specified. The same options apply as described above for mode 1.

Starting Location Data

One of two options for entering starting locations for particles can be selected in response to the prompt:

```
HOW ARE STARTING LOCATIONS TO BE ENTERED?
  1 = FROM A FILE
  2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
```

OPTION 1: Particle coordinates are read from a file. MODPATH prompts the user to enter a file name. The file consists of one line of data for each particle. Data is entered in free-format with a space or comma separating entries. For each particle, enter:

J I K x(local) y(local) z(local)

Starting locations within the cell are specified using local coordinates. Local coordinates vary within a finite difference cell from zero to one in each of the coordinate directions. Local coordinates are defined so that point (0, 0, 0) corresponds to point (x_1, y_1, z_1) and point (1, 1, 1) corresponds to point (x_2, y_2, z_2), as shown in figure 1. If the model layer has an underlying quasi-three-dimensional confining layer, the local z coordinate within the confining layer varies from -1 at the bottom of the confining layer to 0 at its top (figure 7).

OPTION 2: This option provides a method for specifying large, regularly spaced arrays of particles over three-dimensional subregions of the grid. When this option is in effect, the user is prompted to define a subregion of the grid by entering the minimum and maximum J, I, and K cell indices for the region. Once the subregion has been defined, the user is asked to select a pattern for distributing particles within a cell. Identical distributions of particles are generated for all cells in the subregion. MODPATH prompts the user to select one of two methods for distributing the particles:

a) regularly distribute a three-dimensional array of points within a cell

b) regularly distribute a two-dimensional array of points on one or more of the six cell faces

If the choice is made to distribute particles in 3-D arrays within cells, the user receives the following prompt:

```
ENTER: NJ NI NK
  NJ = NUMBER OF PARTICLES PER CELL IN THE J DIRECTION
  NI = NUMBER OF PARTICLES PER CELL IN THE I DIRECTION
  NK = NUMBER OF PARTICLES PER CELL IN THE K DIRECTION
```

The cell is evenly subdivided in each of the coordinate directions according to the values of NJ, NI, and NK to produce (NJ x NI x NK) subvolumes. Particles are then placed at the center of each subvolume.

If particles are to be distributed in 2-D arrays on individual cell faces, MODPATH asks the user if particles are to be placed on a specific face. Using face 1 as an example, the program would issue the prompt:

```
DO YOU WANT TO PLACE PARTICLES ON FACE 1 ?
  [Y = YES;    N OR <CR> =NO]
```

If the response is no (N), face 1 is skipped and the same prompt is issued for face 2. If the answer is yes (Y), the following prompt is issued asking the user to define the array of particles to be placed on face 1:

```
ENTER: NI NK
  NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 1
  NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FACE 1
```

Face 1 then is divided in the I and K directions into NI and NK subdivisions, respectively, to form (NI x NK) subareas. Particles then are placed on face 1 at the center of each subarea. Once the values of NI and NK have been entered for face 1, the user is asked for the same type of information for face 2. This process is repeated for all 6 faces. Examples of particle placement are shown in Figure 10.

When particle locations are generated internally, the user is asked if the locations should be stored in a file. If so, the user is prompted to enter a file name. The data are written in the same order as described under option 1:

J I K x(local) y(local) z(local)

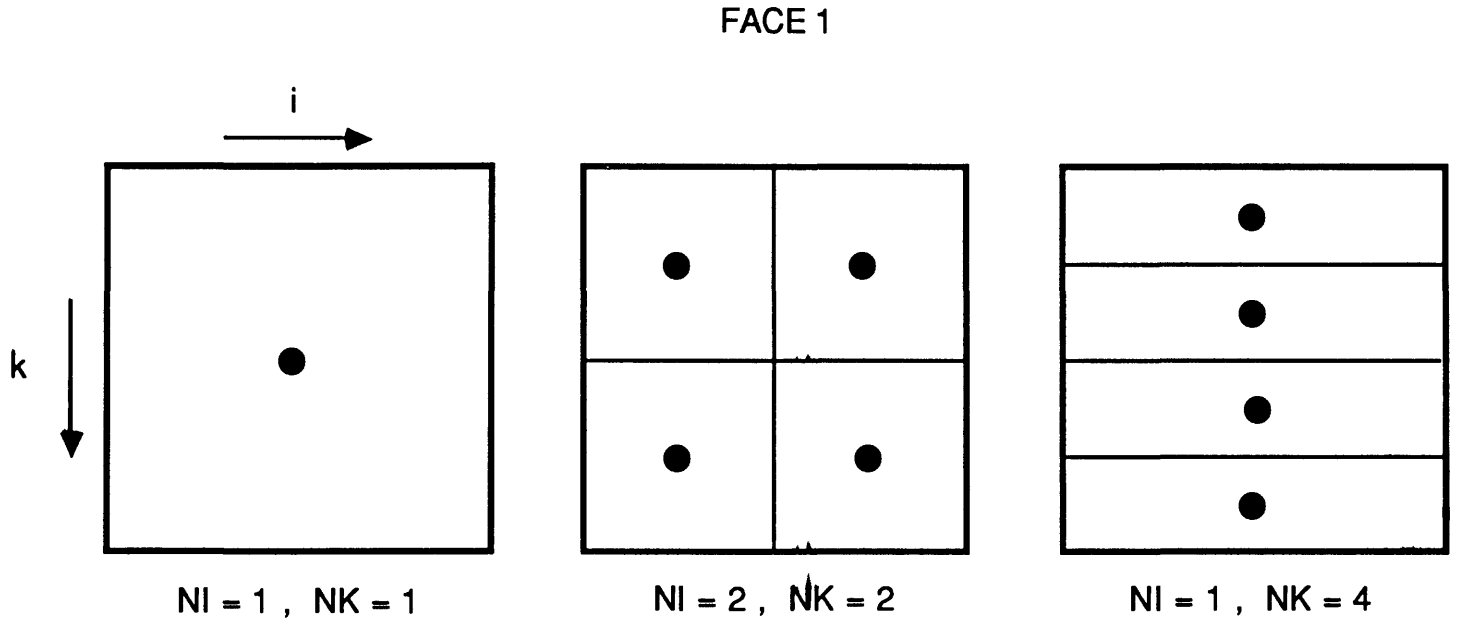


Figure 10. --- Possible arrangements of particles on a cell face using the automatic generation option.

Direction of Tracking Computation

MODPATH provides the option of tracking particles forward in the direction of ground water flow, or backwards in the up-flow direction toward points of recharge. MODPATH prompts the user to select either the forward or backward option. Backward tracking is accomplished by multiplying all velocity components by -1. Once the sign of the velocity components has been changed, computations are carried out in exactly the same way as for forward tracking. For backward tracking, particles terminate at points of recharge, rather than points of discharge. The backward tracking option often provides an efficient means of delineating the source of recharge to localized points of discharge, such as well fields or drains.

Criteria for Terminating Path Lines

Particle path lines are automatically terminated when (1) they reach a cell face that is a boundary of the active grid, or (2) they enter a cell with a strong sink from which there is no outflow to other cells (or, for backward tracking, a strong source cell with no inflow from other cells). For through-flow cells that have internal sinks, an arbitrary decision must be made about whether to stop particles. As described previously, MODPATH provides three options:

1) particles pass through cells with weak sinks.

2) particles are stopped when they enter cells with internal sinks.

3) particles are stopped when they enter cells where discharge to sinks is larger than a specified fraction of the total inflow to the cell.

MODPATH prompts the user to select one of these three options. If the last option is selected, the user is requested to enter a fraction between 0 and 1. These options apply for backward tracking as well as forward tracking.

MODPATH also provides the option of stopping particles whenever they enter a cell with an IBOUND value equal to a specially designated zone code (regardless of whether the cell is a potential discharge point). MODPATH issues the prompt:

```
DO YOU WANT TO SPECIFY A ZONE IN WHICH TO STOP PARTICLES WHENEVER
THEY ENTER ?
  [Y = YES;    N OR <CR> =NO]
```

If the response is yes, the user is asked to enter the number of the zone. Only one zone number can be specified. In addition, if the response is yes, the user is given the chance to define the zone by changing values in the IBOUND array interactively.

Mass Balance Check

An option is provided to check mass balances on a cell-by-cell basis for all active cells in the grid (except constant head cells). If the user chooses to compute mass balances, a prompt is issued requesting that an error tolerance be specified. The value should be entered as a percent. Mass balances for all cells are then computed, and those cells with errors exceeding the tolerance are recorded in SUMMARY.PTH. Only the first 100 exceedences are recorded. A message is printed to the screen indicating the number of exceedences. When exceedences are encountered, the user is asked whether execution should continue or stop. Cell by cell mass balance checking is very useful as a means of assuring that all of the appropriate stress packages have been included and that the stress package flow rates have been assigned to the appropriate faces. For example, if a well flow rate was mistakenly assigned to a face that is not actually a boundary face, the mass balance calculation will show an error for that cell. Although the cell by cell mass balance calculation can be skipped, it is strongly recommended that it be included at least during the initial stages of an analysis to provide an additional check for data errors.

Cell-By-Cell Data Summary

MODPATH uses and produces a huge volume of data that is difficult to digest when printed as large arrays. Consequently, a provision exists to write to the screen a summary of relevant data for individual grid cells. The user is prompted to enter the cell indices. Output includes head, flow rate and velocity components at the six cell faces, coordinates of the cell faces, and a summary of the mass balance components for the cell. Cells are specified one at a time and any number of cells may be checked. When the user is through checking cells, a prompt is issued asking if the data is correct. If the response is no, execution is stopped. Cell-by-cell data summaries are useful in the early stage of a study to help find errors in data.

OUTPUT

MODPATH produces the following output files as a function of the output mode specified by the user:

Mode	Files
0	SUMMARY.PTH ENDPOINT
1	SUMMARY.PTH ENDPOINT PATHLINE
2	SUMMARY.PTH ENDPOINT TIMESERS

The files ENDPOINT, PATHLINE, and TIMESERS contain the basic numerical data on particle coordinates and other attributes required by MODPATH-PLOT to produce graphical output. These files also can serve as input data for other user-defined graphics programs. Detailed descriptions of the structure, contents, and format of the ENDPOINT, PATHLINE, and TIMESERS files are presented in Appendix IV to facilitate the development of user-defined graphics programs.

EXECUTION WITHOUT INTERACTIVE INPUT

Data that is normally input to MODPATH interactively at the terminal can be read from a file instead. When data is read entirely from files the prompts to the terminal screen are suppressed so that no input or output occurs at the terminal. MODPATH requires that the file containing the "interactive" input data be named "TERMIN.PTH". MODPATH looks for a file by that name. If one does not exist, it is created by MODPATH and the word "SCREEN" is written in the first line of the file. A flag is then set for that run forcing data to be entered interactively at the terminal. Each response to a prompt is then written to the file TERMIN.PTH. After execution, the file TERMIN.PTH contains a record of the responses to prompts at the terminal.

The next time MODPATH is executed, it looks again for a file named TERMIN.PTH. If one exists from a previous run, it reads the first line of the file and searches for the words "SCREEN" and "BATCH". If the first line contains the word "SCREEN", data is forced to be input at the terminal and is once again recorded in the file TERMIN.PTH. However, if the first line contains the word "BATCH", MODPATH suppresses the prompts to the screen and reads the "interactive" input from the file TERMIN.PTH. This approach allows the user to make one interactive run to generate the file TERMIN.PTH, and then, if desired, change the first line of the file from "SCREEN" to "BATCH" so that future runs of the same type can be made without interactive input at the terminal.

MODPATH-PLOT

ORGANIZATION AND STRUCTURE

The organization and structure of MODPATH-PLOT is similar to that of MODPATH. The main program controls the opening of files and the allocation of space for arrays. Control then is transferred to a main subroutine named DRIVER. Flow charts for the main program and subroutine DRIVER are shown in Figure 11. Only major subroutines are shown in Figure 11. MODPATH-PLOT uses the DISSPLA graphics subroutine library (Computer Associates, 1981). DISSPLA subroutines are not shown explicitly in Figure 11.

MODPATH-PLOT produces graphical output based on data generated by MODPATH. Input to MODPATH-PLOT consists of (1) flow system files, (2) particle data stored in ENDPOINT, PATHLINE, and TIMESERS files, and (3) interactive input to select plot options.

FLOW SYSTEM FILES

MODPATH-PLOT uses only some of the flow system files needed to execute MODPATH. They are:

1) main data file
2) head file (unformatted from modular flow model)
3) well package data file
4) river package data file
5) drain package data file
6) general head boundary package data file

The main data file and the head file are required to draw the grid. The four stress package data files are used only to identify and plot (if desired) the location of cells that contain wells, rivers, drains, or general head boundary stresses. None of the budget files are needed because no flow calculations are made by MODPATH-PLOT. The stress package data files used to run MODPATH may be changed for the purposes of MODPATH-PLOT in order to customize a plot. For example, values of IFACE in the well package data set may be changed to control which wells are shown on the plot.

INTERACTIVE INPUT AND PROGRAM OPTIONS

Name of File Containing Flow System Files

MODPATH-PLOT prompts the user to enter the name of the file containing the list of flow system files and unit numbers. In most cases, it is convenient to specify the same list of flow system files as supplied to MODPATH, even though that list will contain some files that are not required by MODPATH-PLOT.

Title

The user is prompted to enter a short title of up to 80 characters.

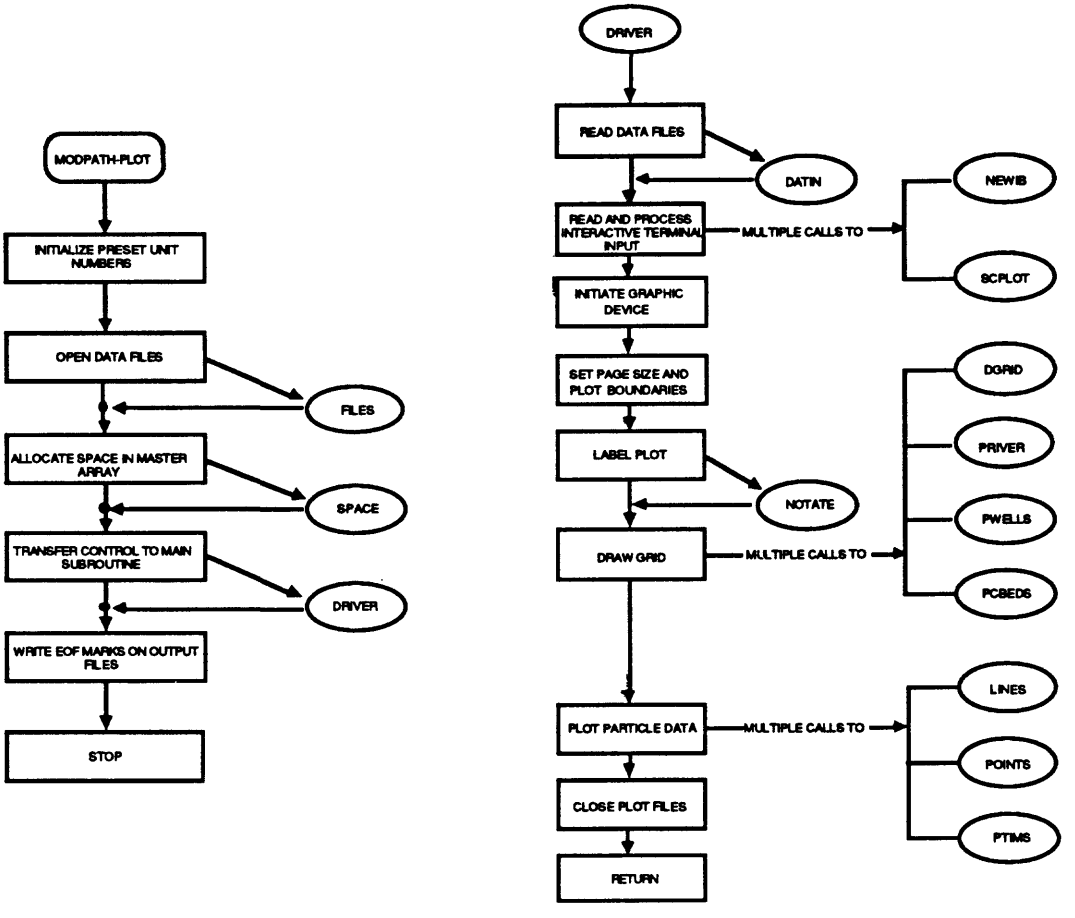


Figure 11. — Flow chart for MODPATH-PLOT main program and DRIVER subroutine.

Device Code

Options are provided for specifying the following graphics devices:

1) TEKTRONIX 4010 mode.

2) TEKTRONIX 4105/4205 mode.

3) TEKTRONIX 4115/4125 mode.

4) TEKTRONIX 4107/4109/4207 mode.

5) META file mode. This mode stores DISSPLA graphics output in a file in device-independent format. Graphic output stored in META files can be plotted to a variety of hardware devices using the DISSPLA post-processor.

Grid

MODPATH-PLOT always displays the boundary of the active grid. The user is given the option to specify whether interior grid lines will be drawn.

Plot Type

One of five types of plots may be specified in response to the prompt:

ENTER THE TYPE OF PLOT:
1 = PATH LINE PLOT
2 = MAP VIEW OF STARTING LOCATIONS (FORWARD TRACKING)
3 = MAP VIEW OF FINAL LOCATIONS (FORWARD TRACKING)
4 = MAP VIEW OF FINAL LOCATIONS (BACKWARD TRACKING)
5 = TIME SERIES PLOT

PLOT TYPES 1 and 5: Plot type 1 draws path lines using data from a PATHLINE file generated by MODPATH. Plot type 5 plots data points from a TIMESERS file to show the locations of a group of particles as a series of snapshots in time.

If a path line plot or a time series plot is specified, the user is asked to select one of three orientations for the plot:

1) map view
2) cross section along a column
3) cross section along a row

Cross sections are plotted as true rectangular grids using average thicknesses for variable thickness grid layers. In cases where layers undergo extreme changes thickness from one cell to the next, this approach can lead to distorted plots. In most cases, however, that is not a problem.

The user also is asked to choose one of two ways of plotting the data:

1) plot all data by projecting the data onto the 2-D slice (map or cross section)

2) plot only data within the layer, column, or row corresponding to the 2-D slice

For complex three-dimensional flow fields, the projection method can lead to strange and confusing pictures with path lines that cross or data points that plot on top of one another. In those situations, it may be possible to obtain a clearer picture by plotting only those points and portions of path lines that fall within the slice. However, very often neither of these approaches produces clear results for complex 3-D paths. By using the slice-only method, sometimes it is possible to capture more of the three-dimensional structure of the path lines by producing a series of plots for successive slices that can be displayed together as a series of views.

After the orientation and the data plotting option have been selected, MODPATH-PLOT prompts the user to specify the layer, row, or column along which the slice is to be taken. This information is used to draw the grid. If a map view is selected and all data are plotted by projection, the active grid for layer 1 is drawn. In all other cases, the active grid is drawn for the layer, row, or column specified by the user.

PLOT TYPES 2, 3, AND 4: These three options plot either starting or final locations of particle paths based on data in an ENDPOINT file from MODPATH. These plots are useful in delineating sources of water to major discharge points and (or) to hydrogeologic units within a flow system.

Plot type 2 displays starting coordinates from a forward tracking analysis. A map of source areas can be produced by placing particles at the top of cells receiving areal recharge. The color and symbol used for the points is determined by the zone code of the cell containing the final location of the particle. For plot type 2, particles that terminate in cells with a zone code of 1 (IBOUND = 1) are not plotted.

Plot type 3 displays final locations from a forward tracking run. This type of plot is useful for showing the distribution of recharge entering a model layer or a designated hydrogeologic zone. Points are plotted as a small "+". The color of a point is determined by the zone code of the cell containing the starting location for the particle. If plot type 3 is selected, the user is asked if a zone should be specified to indicate that only points terminating in cells within that zone should be plotted. If yes, the user is asked to enter a zone number.

Plot type 4 displays final locations from a backward tracking run. If particles are placed around a discharge point, this type of plot maps source areas for the discharge point. Points are plotted as a small "+". The color of a point is determined by the zone code of the cell containing the starting location for the particle. If plot type 4 is selected, the user is asked if a zone should be specified to indicate that only points terminating in cells within that zone should be plotted. If yes, the user is asked to enter a zone number. If no, the endpoints of all particles are plotted.

Plot Boundaries and Scale

MODPATH-PLOT allows all or part of the grid to be plotted. The user is asked to define the portion of the grid to be plotted by entering the minimum and maximum values for the appropriate cell indices. If the plot is a cross section, the user is asked to specify a vertical exaggeration. MODPATH-PLOT then

determines the maximum possible plot size and calculates a map scale for that sized plot. The map scale then is printed at the terminal and the user is asked if the scale is acceptable. If not, the user can specify a map scale to produce a smaller plot.

Zone Codes

Zone codes for grid cells are equal to the absolute value of entries in the IBOUND array. Zone codes are used extensively by MODPATH-PLOT to determine whether or not to plot lines or points and to determine colors, symbols, and line patterns. Zone codes may be set explicitly in the IBOUND array of the main data file or may be reset interactively during the execution of MODPATH-PLOT. The user is asked if any of the zone codes should be changed. If so, one of three options is selected:

1) change the entire grid to be a single value
2) change a single cell
3) change a 3-D subregion of the grid to be a single value

MODPATH-PLOT then asks that the new zone code be entered. Depending of the option selected, additional information may be requested to define the cells to be changed. After the changes have been made, a prompt is issued asking if there are more changes. If yes, the process is repeated.

Color Plots

Either color or black and white plots may be specified. If black and white is selected and a path line plot has been chosen, MODPATH-PLOT issues the prompt:

```
WHAT TYPE OF LINE PATTERNS SHOULD BE USED ?
  0 = CYCLE THROUGH LINE PATTERNS (SOLID, DASH, DOT)
  1 = USE A SINGLE LINE PATTERN FOR ALL PATH LINES
```

If option 0 is selected, path lines cycle through a solid, dash, dot sequence according to the value of the zone code values:

Zone Code	Line Pattern
1	solid
2	dash
3	dot
4	solid
5	dash
6	dot
(sequence repeats)	

If option 1 is selected, the user is asked to specify either solid, dash, or dot.

When a color plot is requested, MODPATH-PLOT issues the prompt:

```
HOW SHOULD COLORS BE CHOSEN FOR DIFFERENT DATA GROUPS ?
  0 = CYCLE THROUGH COLORS
  1 = ALL LINES AND (OR) DATA POINTS PLOTTED IN ONE COLOR
```

If option 0 is selected, colors are cycled according to zone code, where:

Zone Code	Color
1	black/white
2	red
3	green
4	blue
5	black/white
6	red
7	green
8	blue

(sequence repeats)

If option 1 is selected, the user selects one of the four colors.

Border

An option is provided to draw the page border. When previewing plots at the terminal it is useful to draw the page border to evaluate the layout of the plot. However, it is better not to draw the border for certain types of pen plotters.

SYMBOLS FOR STRESS PACKAGE FEATURES

An option is provided to show the location of cells containing wells, rivers, drains, and general head boundaries. No option is provided for plotting recharge and evapotranspiration zones. The options for plotting stress package features are summarized below:

River Cells

In map view, rivers are shown as blue rectangles outlining the cell when IFACE is any value from 0 to 6. River cells are shown in map view regardless of the layer in which they occur. In cross section, rivers are shown in blue as dashed rectangles outlining the cell only when IFACE is 0. Only those river cells in the plane of the cross section are plotted.

Wells, Drains, and General Head Boundaries

These three stress package features are all represented in the same way. These features are shown on a plot only when IFACE equals 0. In map view, they are displayed as red octagons. In cross section, they are displayed as red dashed rectangles.

SAMPLE PROBLEM

A sample problem is illustrated in Figure 12. It consists of two aquifers separated by a 20-foot-thick confining layer. The upper aquifer is unconfined. The lower aquifer is 200 feet thick and confined. A partially-penetrating well in the lower aquifer is located in the center of the flow system. The well discharges at a rate of 80,000 cubic feet per day. Boundary conditions are:

1. uniform areal recharge to the upper aquifer (0.0045 feet/day)
2. no flow on all sides and along the bottom of the lower aquifer.
3. a shallow, partially penetrating river located along one side at the top of the upper aquifer.

In the finite-difference approximation, a single unconfined model layer represents the upper aquifer. The lower aquifer is divided into four 50-foot-thick layers, and the confining layer separating the two aquifers is accounted for using a quasi-three-dimensional approach. The areal grid consists of 27 rows and 27 columns with spacing that varies from 40 feet by 40 feet at the location of the well to 400 feet by 400 feet away from the well. Modular flow model data sets are presented in Appendix V along with those for MODPATH and MODPATH-PLOT. Several examples are presented to illustrate how MODPATH and MODPATH-PLOT can be used to analyze problems using endpoint, path line, and time series modes.

Figure 13 shows a path line plot for a cross section taken perpendicular to the river along row 14. Because of the symmetry of the flow system, pathlines along the center of row 14 are perpendicular to the river. The path lines shown in figure 13 were generated using backward tracking from the well in cell (14, 14, 4) and the river in cell (14, 27, 1). Tables 1 and 2 show the interactive dialogue from running MODPATH and MODPATH-PLOT for this case. The circles plotted along the path lines represent 7300 days increments in time (20 years) with time zero corresponding to the starting location of the particles at the faces of cell (14, 14, 4). Time of travel always is measured from the starting point of the particle, therefore its meaning depends on whether the path lines were computed backward or forward.

A principal objective of particle tracking analyses often is to delineate recharge areas that supply water to specific points of discharge in the flow system. One way to accomplish that objective is to track particles backward from points of discharge to points of recharge. Figure 14 shows a map view of recharge locations at the water table for 204 particles that were tracked backward from their points of origin at the faces of cell (14, 14, 4). The distribution of those particles at the water table provides an estimate of the extent of the contributing recharge area for the well. However, it should be stressed that an advective particle tracking analysis does not account for mixing of waters due to the effects of dispersion. Tables 3 and 4 record the interactive dialogue from running MODPATH and MODPATH-PLOT for this case.

If the objective of the particle tracking analysis is to produce an endpoints plot such as figure 14, it is not necessary to store intermediate particle locations in a file. Consequently, when "endpoint" mode is

selected in MODPATH, path line coordinates are not recorded; only the initial and final particle locations are recorded in the file ENDPOINT.

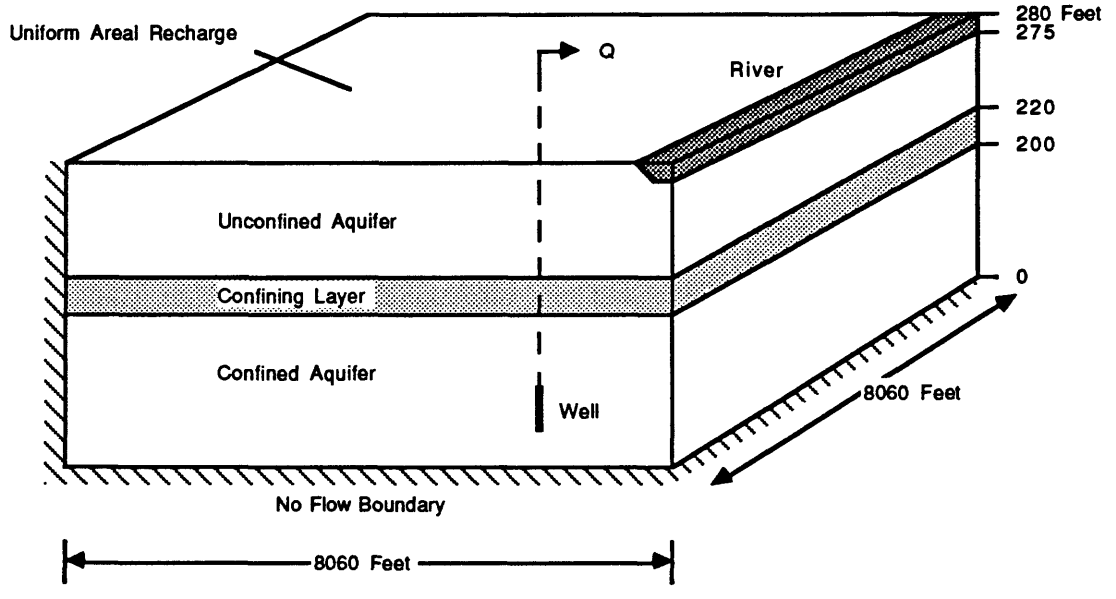


Figure 12. --- Block diagram showing hydrogeology, geometry, and boundary conditions for sample problem.

Table 1.
Interactive dialogue for MODPATH for cross sectional path line analysis

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS    83165
 SELECT THE MODE FOR STORING OUTPUT DATA:
     0 = ENDPOINTS AND STARTING POINTS ONLY
     1 = FLOW LINE COORDINATES
     2 = TIME SERIES DATA FOR SCATTER PLOTS
1

DO YOU WANT TO COMPUTE LOCATIONS AT INTERMEDIATE TIMES ?
    [Y = YES;    N OR <CR> = NO]
Y

 HOW SHOULD POINTS IN TIME BE SPECIFIED ?
     0 = WITH A CONSTANT TIME STEP
     1 = VALUES OF TIME READ FROM A FILE
0

 ENTER: TIME STEP LENGTH, FACTOR FOR CONVERTING UNITS
20 365

 HOW ARE STARTING LOCATIONS TO BE ENTERED?
     1 = FROM AN EXISTING DATA FILE
     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
2

DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
    [Y = YES;    N OR <CR> = NO]
N

 IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
     0 = FORWARD IN THE DIRECTION OF FLOW
     1 = BACKWARDS TOWARD RECHARGE LOCATIONS
1

 HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
     0 = PARTICLES PASS THROUGH CELLS WITH WEAK SINKS
     1 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WITH INTERNAL SINKS
     2 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WHERE DISCHARGE TO SINKS
         IS LARGER THAN A SPECIFIED FRACTION OF THE TOTAL INFLOW TO THE CELL
0

DO YOU WANT TO SPECIFY ONE ZONE IN WHICH TO STOP PARTICLES WHENEVER THEY ENTER ?
    [Y = YES;    N OR <CR> = NO]
N

 READING AND PROCESSING INPUT DATA...
 STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.

SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
    [Y = YES;    N OR <CR> = NO]
N
 ENTER DATA FOR SUBREGION  1 :

 DEFINE THE SUBREGION BY ENTERING THE MINIMUM AND MAXIMUM J,I,K COORDINATES.
   ENTER:  MINIMUM J, MAXIMUM J, MINIMUM I, MAXIMUM I, MINIMUM K, MAXIMUM K
14 14 14 14 4 4
 WHERE SHOULD THE PARTICLES BE LOCATED ?
   0 = WITHIN A CELL
   1 = ON ONE OR MORE OF THE CELL FACES
1
DO YOU WANT TO PLACE PARTICLES ON FACE 1 ?
    [Y = YES;    N OR <CR> = NO]
Y
```

```
 ENTER:  NI  NK
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 1
    NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FACE 1
1 5
DO YOU WANT TO PLACE PARTICLES ON FACE 2 ?
    [Y = YES;    N OR <CR> = NO]
Y
 ENTER:  NI  NK
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 2
    NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FACE 2
1 3
DO YOU WANT TO PLACE PARTICLES ON FACE 3 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 4 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 5 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 6 ?
    [Y = YES;    N OR <CR> = NO]
Y
 ENTER:  NJ  NI
    NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FACE 6
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 6
1 1

 TOTAL NUMBER OF PARTICLES IS      9
 MAXIMUM NUMBER OF PARTICLES IS  83165
DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
    [Y = YES;    N OR <CR> = NO]
Y

 ENTER DATA FOR SUBREGION  2 :

 DEFINE THE SUBREGION BY ENTERING THE MINIMUM AND MAXIMUM J,I,K COORDINATES.
   ENTER:  MINIMUM J, MAXIMUM J, MINIMUM I, MAXIMUM I, MINIMUM K, MAXIMUM K
27 27 14 14 1 1

 WHERE SHOULD THE PARTICLES BE LOCATED ?
   0 = WITHIN A CELL
   1 = ON ONE OR MORE OF THE CELL FACES
1

DO YOU WANT TO PLACE PARTICLES ON FACE 1 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 2 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 3 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 4 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 5 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 6 ?
    [Y = YES;    N OR <CR> = NO]
Y
 ENTER:  NJ  NI
    NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FACE 6
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 6
3 1
```

```
 TOTAL NUMBER OF PARTICLES IS     12
 MAXIMUM NUMBER OF PARTICLES IS  83165
DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
    [Y = YES;     N OR <CR> = NO]
N

DO YOU WANT TO CHECK THE MASS BALANCE FOR ALL CELLS ?
    [Y = YES;     N OR <CR> = NO]
Y

 SPECIFY AN ERROR TOLERANCE (IN PERCENT):
1.

MASS BALANCES FOR ALL CELLS ARE <  1.0000%

 DO YOU WANT TO CHECK DATA CELL BY CELL ?
    [Y = YES;     N OR <CR> = NO]
Y
 ENTER J I K COORDINATES OF CELL:  J   I   K
27 14 1

DATA FOR CELL ( 27, 14,  1)
IBOUND=       1 HEAD=     0.28327E+03 POROSITY= 0.300
X1=     0.76600E+04 X2=     0.80600E+04 (X2-X1)=     0.40000E+03
Y1=     0.40100E+04 Y2=     0.40500E+04 (Y2-Y1)=     0.40000E+02
Z1=     0.22000E+03 Z2=     0.28327E+03 (Z2-Z1)=     0.63267E+02
 VOLUMETRIC FLOW (Q1-Q6) AND VELOCITY (V1-V6) AT CELL FACES
X (FACES 1&2): Q1=  9.11772E+02 Q2=  0.00000E-01 V1=  1.201E+00 V2=  0.000E-01
Y (FACES 3&4): Q3=  5.28046E-01 Q4= -1.76885E-01 V3=  6.955E-05 V4= -2.330E-05
Z (FACES 5&6): Q5=  0.61025E+02 Q6=  0.97343E+03 V5=  0.127E-01 V6=  0.203E+00
TOTAL IN ACROSS CELL FACES =  0.97350E+03
TOTAL OUT ACROSS CELL FACES =  0.97343E+03
NET SOURCES =  0.00000E+00
 MASS BALANCE ERROR IN PERCENT IS DEFINED AS:
  100 X [(IN - OUT + NET SOURCES)/(IN + OUT)/2)]
MASS BALANCE ERROR =   0.00744 PERCENT
CONFINING BED DATA:
  THICKNESS =  2.00000E+01
  POROSITY =  3.00000E-01
  VERTICAL VELOCITY =  1.27136E-02

 DO YOU WANT TO CHECK DATA FOR ANOTHER CELL ?
    [Y = YES;     N OR <CR> = NO]
N

 DOES THE DATA APPEAR TO BE CORRECT?
    [Y = YES;     N OR <CR> = NO]
Y

DO YOU WANT TO WRITE ENDPOINT INFORMATION TO THE SCREEN ?
    [Y = YES;     N OR <CR> = NO]
Y

 NOW COMPUTING PARTICLE PATHS...
 PARTICLE   10 STOPS IN (I= 14, J= 24, K=  1)  TRAVEL TIME =   9.9282E+02
 PARTICLE   11 STOPS IN (I= 14, J= 20, K=  1)  TRAVEL TIME =   4.2060E+03
 PARTICLE    5 STOPS IN (I= 14, J=  4, K=  1)  TRAVEL TIME =   1.3548E+04
 PARTICLE    8 STOPS IN (I= 14, J=  8, K=  1)  TRAVEL TIME =   1.3186E+04
 PARTICLE    9 STOPS IN (I= 14, J=  6, K=  1)  TRAVEL TIME =   9.5430E+03
 PARTICLE   12 STOPS IN (I= 14, J=  9, K=  1)  TRAVEL TIME =   9.2201E+03
 PARTICLE    4 STOPS IN (I= 14, J=  3, K=  1)  TRAVEL TIME =   1.8909E+04
 PARTICLE    3 STOPS IN (I= 14, J=  2, K=  1)  TRAVEL TIME =   2.5459E+04
 PARTICLE    2 STOPS IN (I= 14, J=  1, K=  1)  TRAVEL TIME =   3.5317E+04
 PARTICLE    7 STOPS IN (I= 14, J=  8, K=  1)  TRAVEL TIME =   3.0129E+04
 PARTICLE    1 STOPS IN (I= 14, J=  1, K=  1)  TRAVEL TIME =   5.6151E+04
 PARTICLE    6 STOPS IN (I= 15, J=  8, K=  1)  TRAVEL TIME =   9.9183E+04
```

Table 2
Interactive dialogue from MODPATH-PLOT for the cross section plot

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS   117470
 READING MODPATH DATA FILES...

 ENTER TITLE (80 CHARACTERS OR LESS):
CROSS SECTION, BACK TRACK PATH LINES

 ENTER GRAPHICS DEVICE CODE:
     1 = TEKTRONIX 4010 (GRAPHICS TAB)
     2 = TEKTRONIX 4105 OR 4205
     3 = TEKTRONIX 4115 OR 4125
     4 = TEKTRONIX 4107, 4207, OR 4109
     5 = CREATE META FILE
5

DRAW INTERIOR GRID LINES ?
    [ Y = YES;      N OR <CR> = NO ]
N

 ENTER THE TYPE OF GRAPH:
     1 = FLOW LINE PLOT
     2 = MAP VIEW OF STARTING LOCATIONS (FORWARD TRACKING)
     3 = MAP VIEW OF FINAL LOCATIONS (FORWARD TRACKING)
     4 = MAP VIEW OF FINAL LOCATIONS (BACKWARD TRACKING)
     5 = TIME SERIES PLOT
1

 WHAT IS THE ORIENTATION OF THE PLOT ?
   1 = MAP VIEW
   2 = CROSS SECTION VIEW ALONG A COLUMN
   3 = CROSS SECTION VIEW ALONG A ROW
3

 WHAT DATA SHOULD BE PLOTTED ?
    0 = PLOT ALL DATA BY PROJECTION ONTO THE 2D SLICE
    1 = PLOT ONLY THE DATA WITHIN THE ROW CORRESPONDING TO THE CROSS SECTION
0
 ALONG WHAT ROW SHOULD THE CROSS SECTION BE TAKEN ?
14

 WERE THE FLOWLINES GENERATED BY FORWARD OR BACKWARD TRACKING ?
    0 = FORWARD
    1 = BACKWARD
1

DO YOU WANT TO PLOT POINTS AT SPECIFIED TIME INTERVALS ?
    [ Y = YES;      N OR <CR> = NO ]
Y

DO YOU WANT TO STOP DRAWING PATH LINES AT A SPECIFIED TIME?
    [ Y = YES;      N OR <CR> = NO ]
N

 ENTER NAME OF ENDPOINT FILE (<CR>="ENDPOINT"):
ENDPOINT

 ENTER NAME OF PATHLINE FILE (<CR>="PATHLINE"):
PATHLINE

 ENTER GRID COORDINATES:
      MINIMUM COLUMN VALUE,  MAXIMUM COLUMN VALUE
1 27
```

```
      MINIMUM LAYER VALUE,  MAXIMUM LAYER VALUE
1 5

 WHAT IS THE VERTICAL EXAGGERATION ?
5

 ENTER PAGE SIZE:
    1 = 8.5 X 11
    2 = 11 X 17
1

 WHAT ARE THE UNITS OF DISTANCE IN THE MODEL ?
    1 = FEET
    2 = METERS
1

MAXIMUM SIZE PLOT IS AT SCALE OF (1:   10181.)
IS THIS SCALE ACCEPTABLE ?
    [ Y = YES;      N OR <CR> = NO ]
Y

DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
    [ Y = YES;      N OR <CR> = NO ]
Y

 WHAT TYPE OF CHANGE DO YOU WANT TO MAKE ?
    1 = CHANGE AN ENTIRE LAYER
    2 = CHANGE AN INDIVIDUAL CELL
    3 = CHANGE ALL CELLS IN A BLOCK OF CELLS
2

 ENTER THE CELL INDICES:  J  I  K
14 14 4

 ENTER THE NEW ZONE CODE:
2

DO YOU WANT TO CHANGE SOME MORE ZONE CODES ?
    [ Y = YES;      N OR <CR> = NO ]
Y

 WHAT TYPE OF CHANGE DO YOU WANT TO MAKE ?
    1 = CHANGE AN ENTIRE LAYER
    2 = CHANGE AN INDIVIDUAL CELL
    3 = CHANGE ALL CELLS IN A BLOCK OF CELLS
2

 ENTER THE CELL INDICES:  J  I  K
27 14 1

 ENTER THE NEW ZONE CODE:
4

DO YOU WANT TO CHANGE SOME MORE ZONE CODES ?
    [ Y = YES;      N OR <CR> = NO ]
N

DO YOU WANT A COLOR PLOT ?
    [ Y = YES;      N OR <CR> = NO ]
Y

 HOW SHOULD COLORS BE CHOSEN FOR DIFFERENT DATA GROUPS ?
    0 = CYCLE THROUGH COLORS
    1 = ALL LINES AND (OR) DATA POINTS PLOTTED IN ONE  COLOR
0
DRAW THE PAGE BORDER ?
    [ Y = YES;      N OR <CR> = NO ]
N
```

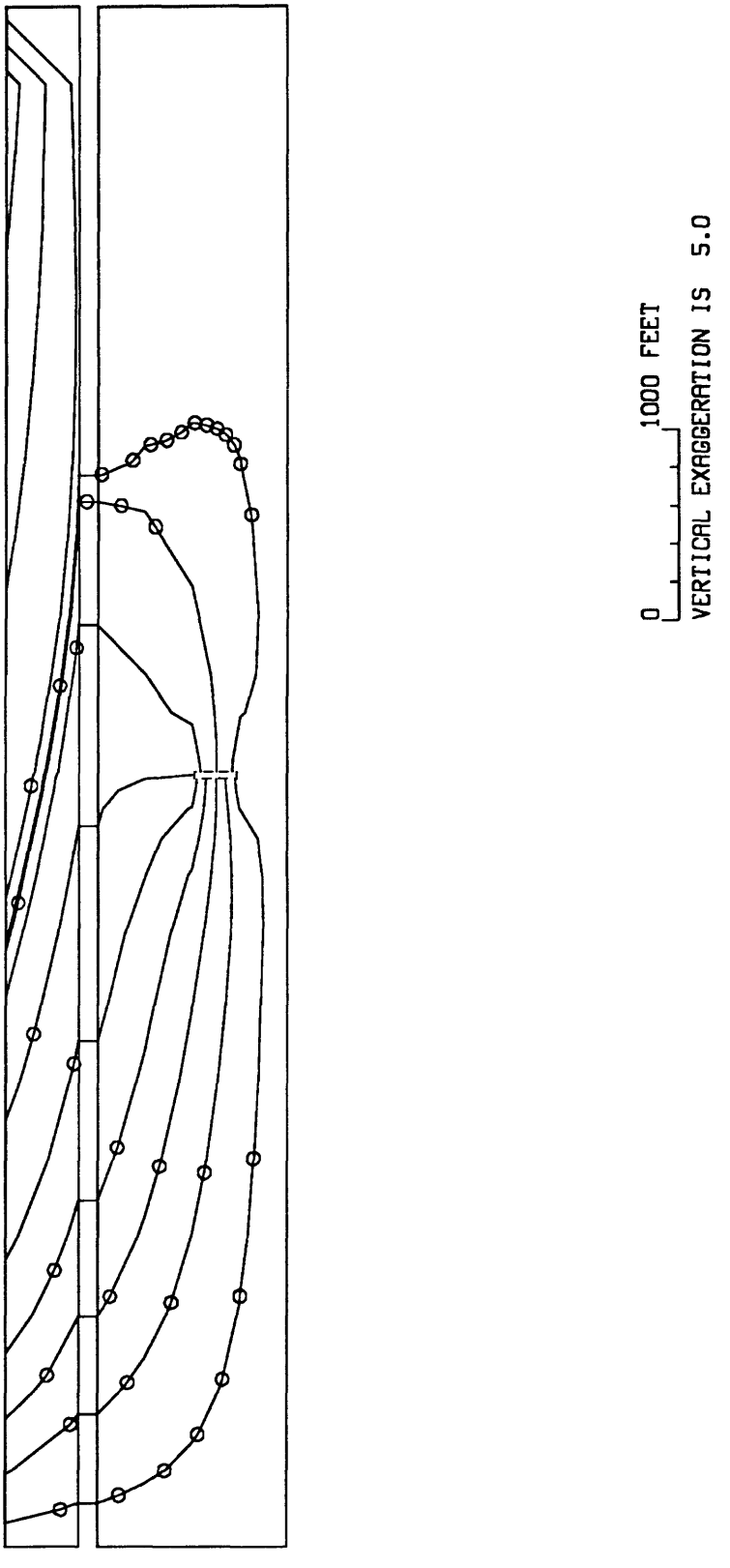


Figure 13. Cross-section along row 14 showing path lines and time of travel points for backward tracking analysis

Table 3
Interactive dialogue from MODPATH for the backward endpoint analysis

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS    83165

 SELECT THE MODE FOR STORING OUTPUT DATA:
     0 = ENDPOINTS AND STARTING POINTS ONLY
     1 = FLOW LINE COORDINATES
     2 = TIME SERIES DATA FOR SCATTER PLOTS
0

 HOW ARE STARTING LOCATIONS TO BE ENTERED?
     1 = FROM AN EXISTING DATA FILE
     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
2

DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
    [Y = YES;     N OR <CR> = NO]
N

 IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
     0 = FORWARD IN THE DIRECTION OF FLOW
     1 = BACKWARDS TOWARD RECHARGE LOCATIONS
1

 HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
     0 = PARTICLES PASS THROUGH CELLS WITH WEAK SINKS
     1 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WITH INTERNAL SINKS
     2 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WHERE DISCHARGE TO SINKS
         IS LARGER THAN A SPECIFIED FRACTION OF THE TOTAL INFLOW TO THE CELL
0

DO YOU WANT TO SPECIFY ONE ZONE IN WHICH TO STOP PARTICLES WHENEVER THEY ENTER ?
    [Y = YES;     N OR <CR> = NO]
N

 READING AND PROCESSING INPUT DATA...

 STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.

SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
    [Y = YES;     N OR <CR> = NO]
N

 ENTER DATA FOR SUBREGION  1 :

 DEFINE THE SUBREGION BY ENTERING THE MINIMUM AND MAXIMUM J,I,K COORDINATES.
   ENTER:  MINIMUM J, MAXIMUM J, MINIMUM I, MAXIMUM I, MINIMUM K, MAXIMUM K
14 14 14 14 4 4

 WHERE SHOULD THE PARTICLES BE LOCATED ?
   0 = WITHIN A CELL
   1 = ON ONE OR MORE OF THE CELL FACES
1

DO YOU WANT TO PLACE PARTICLES ON FACE 1 ?
    [Y = YES;     N OR <CR> = NO]
Y
 ENTER:  NI  NK
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 1
    NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FACE 1
10 5
DO YOU WANT TO PLACE PARTICLES ON FACE 2 ?
    [Y = YES;     N OR <CR> = NO]
Y
```

```
 ENTER:  NI  NK
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 2
    NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FACE 2
10 5
DO YOU WANT TO PLACE PARTICLES ON FACE 3 ?
    [Y = YES;    N OR <CR> = NO]
Y
 ENTER:  NJ  NK
    NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FACE 3
    NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FACE 3
10 5
DO YOU WANT TO PLACE PARTICLES ON FACE 4 ?
    [Y = YES;    N OR <CR> = NO]
Y
 ENTER:  NJ  NK
    NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FACE 4
    NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FACE 4
10 5
DO YOU WANT TO PLACE PARTICLES ON FACE 5 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 6 ?
    [Y = YES;    N OR <CR> = NO]
Y
 ENTER:  NJ  NI
    NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FACE 6
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 6
2 2

 TOTAL NUMBER OF PARTICLES IS    204
 MAXIMUM NUMBER OF PARTICLES IS  83165

DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
    [Y = YES;    N OR <CR> = NO]
N

DO YOU WANT TO CHECK THE MASS BALANCE FOR ALL CELLS ?
    [Y = YES;    N OR <CR> = NO]
N

 DO YOU WANT TO CHECK DATA CELL BY CELL ?
    [Y = YES;    N OR <CR> = NO]
N

DO YOU WANT TO WRITE ENDPOINT INFORMATION TO THE SCREEN ?
    [Y = YES;    N OR <CR> = NO]
N

 NOW COMPUTING PARTICLE PATHS...
```

Table 4
Interactive dialogue from MODPATH-PLOT for the backward endpoint analysis

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS   117470

 READING MODPATH DATA FILES...

 ENTER TITLE (80 CHARACTERS OR LESS):
RECHARGE LOCATIONS FOR PARTICLES BACKTRACKED FROM WELL AT (14,14,4)

 ENTER GRAPHICS DEVICE CODE:
     1 = TEKTRONIX 4010 (GRAPHICS TAB)
     2 = TEKTRONIX 4105 OR 4205
     3 = TEKTRONIX 4115 OR 4125
     4 = TEKTRONIX 4107, 4207, OR 4109
     5 = CREATE META FILE
5

DRAW INTERIOR GRID LINES ?
    [ Y = YES;      N OR <CR> = NO ]
N

 ENTER THE TYPE OF GRAPH:
     1 = FLOW LINE PLOT
     2 = MAP VIEW OF STARTING LOCATIONS (FORWARD TRACKING)
     3 = MAP VIEW OF FINAL LOCATIONS (FORWARD TRACKING)
     4 = MAP VIEW OF FINAL LOCATIONS (BACKWARD TRACKING)
     5 = TIME SERIES PLOT
4

DO YOU WANT TO PLOT ONLY THOSE POINTS THAT TERMINATE IN ONE SPECIFIC ZONE ?
    [ Y = YES;      N OR <CR> = NO ]
N

 ENTER NAME OF ENDPOINT FILE (<CR>="ENDPOINT"):
ENDPOINT

 ENTER GRID COORDINATES:
      MINIMUM COLUMN VALUE,  MAXIMUM COLUMN VALUE
1 27
      MINIMUM ROW VALUE,  MAXIMUM ROW VALUE
1 27
 ENTER PAGE SIZE:
    1 = 8.5 X 11
    2 = 11 X 17
1
 WHAT ARE THE UNITS OF DISTANCE IN THE MODEL ?
    1 = FEET
    2 = METERS
1
MAXIMUM SIZE PLOT IS AT SCALE OF (1:   16120.)
IS THIS SCALE ACCEPTABLE ?
    [ Y = YES;      N OR <CR> = NO ]
Y

DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
    [ Y = YES;      N OR <CR> = NO ]
N

DO YOU WANT A COLOR PLOT ?
    [ Y = YES;      N OR <CR> = NO ]
N
DRAW THE PAGE BORDER ?
    [ Y = YES;      N OR <CR> = NO ]
N
```

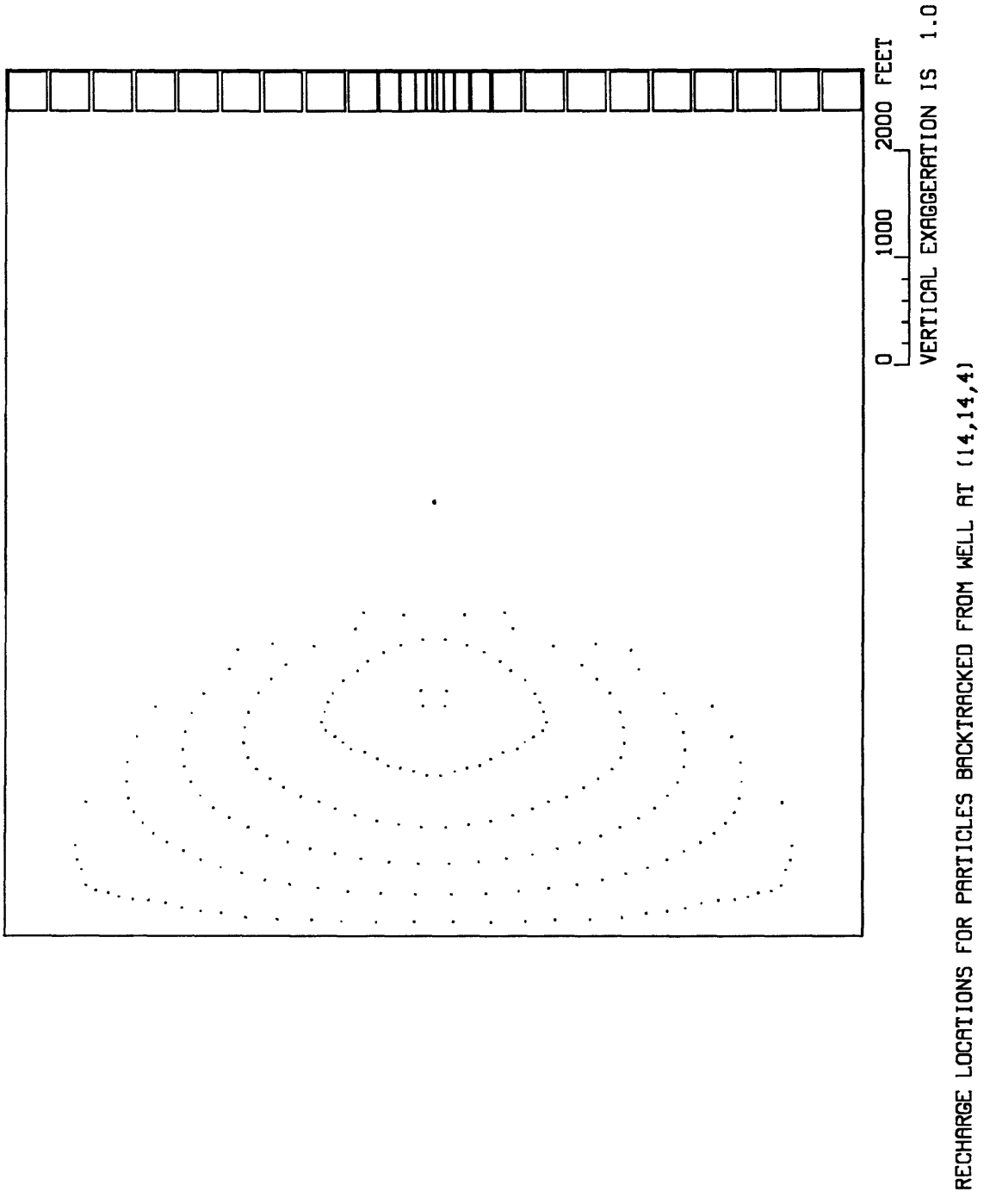


Figure 14. Map view of recharge locations at the water table for an array of particles tracked backwards from cell (14, 14, 4).

In the preceding example, final locations were plotted for a backward tracking analysis in order to define a contributing recharge area for a well. An alternative approach is to use a forward tracking analysis to define the contributing recharge area. For a forward tracking analysis, particles are placed over a wide area that includes all of the possible recharge points for water to the well. The particles are then tracked forward to their points of discharge and the results are recorded in the file ENDPOINT. Once ENDPOINT has been generated by MODPATH, a plot is made of those particles that discharged to the well in cell (14, 14, 4). The plot is produced by changing the IBOUND value for cell (14, 14, 4) to a number with an absolute value greater than 1. That IBOUND value is the "zone code" for the cell. MODPATH-PLOT then reads ENDPOINT, checks the cell coordinates of the discharge point for each particle, and plots those with zone codes greater than 1. Figure 15 shows a map of the contributing recharge area for the well that was generated from a forward tracking analysis. Tables 5 and 6 show the interactive dialogue generated during the creation of the plot shown in figure 15. For this analysis, a 4 by 4 array of 16 particles was placed on the top face (water table) of each cell in layer 1 (11,664 particles in total). A total of 2,624 particles discharge to cell (14, 14, 4). The total rate of recharge captured by the well equals the product of the constant rate of recharge and the area of recharge captured. To estimate the area of recharge captured, it was assumed that each particle represented one-sixteenth of the area of the top face of the cell in which it originated. The recharge area corresponding to each of the particles that discharged to cell (14, 14, 4) was summed to give the total area of recharge captured. The total area was then multiplied by the recharge rate to obtain a volumetric rate of 79,883 cubic feet/day of captured recharge. That value is within 0.15 percent of the pumping rate of 80,000 cubic feet/day. The good agreement between the computed rate of recharge captured and the pumping rate is indirect evidence that the velocity interpolation scheme is consistent with the finite-difference approximations of the ground-water flow equations.

Table 5
Interactive dialogue from MODPATH for the forward endpoint analysis

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS    83165

 SELECT THE MODE FOR STORING OUTPUT DATA:
     0 = ENDPOINTS AND STARTING POINTS ONLY
     1 = FLOW LINE COORDINATES
     2 = TIME SERIES DATA FOR SCATTER PLOTS
0

 HOW ARE STARTING LOCATIONS TO BE ENTERED?
     1 = FROM AN EXISTING DATA FILE
     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
2

DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
    [Y = YES;    N OR <CR> = NO]
N

 IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
     0 = FORWARD IN THE DIRECTION OF FLOW
     1 = BACKWARDS TOWARD RECHARGE LOCATIONS
0

 HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
     0 = PARTICLES PASS THROUGH CELLS WITH WEAK SINKS
     1 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WITH INTERNAL SINKS
     2 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WHERE DISCHARGE TO SINKS
         IS LARGER THAN A SPECIFIED FRACTION OF THE TOTAL INFLOW TO THE CELL
0

DO YOU WANT TO SPECIFY ONE ZONE IN WHICH TO STOP PARTICLES WHENEVER THEY ENTER ?
    [Y = YES;    N OR <CR> = NO]
N

 READING AND PROCESSING INPUT DATA...

 STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.

SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
    [Y = YES;    N OR <CR> = NO]
N

 ENTER DATA FOR SUBREGION  1 :

 DEFINE THE SUBREGION BY ENTERING THE MINIMUM AND MAXIMUM J,I,K COORDINATES.
   ENTER:  MINIMUM J, MAXIMUM J, MINIMUM I, MAXIMUM I, MINIMUM K, MAXIMUM K
1 27 1 27 1 1
 WHERE SHOULD THE PARTICLES BE LOCATED ?
   0 = WITHIN A CELL
   1 = ON ONE OR MORE OF THE CELL FACES
1
DO YOU WANT TO PLACE PARTICLES ON FACE 1 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 2 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 3 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 4 ?
    [Y = YES;    N OR <CR> = NO]
N
```

```
DO YOU WANT TO PLACE PARTICLES ON FACE 5 ?
    [Y = YES;     N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 6 ?
    [Y = YES;     N OR <CR> = NO]
Y
 ENTER:  NJ  NI
    NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FACE 6
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 6
4 4

 TOTAL NUMBER OF PARTICLES IS  11664
 MAXIMUM NUMBER OF PARTICLES IS  83165

DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
    [Y = YES;     N OR <CR> = NO]
N

DO YOU WANT TO CHECK THE MASS BALANCE FOR ALL CELLS ?
    [Y = YES;     N OR <CR> = NO]
N

 DO YOU WANT TO CHECK DATA CELL BY CELL ?
    [Y = YES;     N OR <CR> = NO]
N

DO YOU WANT TO WRITE ENDPOINT INFORMATION TO THE SCREEN ?
    [Y = YES;     N OR <CR> = NO]
N

 NOW COMPUTING PARTICLE PATHS...
```

Table 6
Interactive dialogue from MODPATH-PLOT for the forward endpoint analysis

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS   117470

 READING MODPATH DATA FILES...

 ENTER TITLE (80 CHARACTERS OR LESS):
PARTICLES AT WATER TABLE THAT DISCHARGE TO WELL IN (14,14,4)

 ENTER GRAPHICS DEVICE CODE:
     1 = TEKTRONIX 4010 (GRAPHICS TAB)
     2 = TEKTRONIX 4105 OR 4205
     3 = TEKTRONIX 4115 OR 4125
     4 = TEKTRONIX 4107, 4207, OR 4109
     5 = CREATE META FILE
5

DRAW INTERIOR GRID LINES ?
    [ Y = YES;      N OR <CR> = NO ]
N

 ENTER THE TYPE OF GRAPH:
     1 = FLOW LINE PLOT
     2 = MAP VIEW OF STARTING LOCATIONS (FORWARD TRACKING)
     3 = MAP VIEW OF FINAL LOCATIONS (FORWARD TRACKING)
     4 = MAP VIEW OF FINAL LOCATIONS (BACKWARD TRACKING)
     5 = TIME SERIES PLOT
2

 ENTER NAME OF ENDPOINT FILE (<CR>="ENDPOINT"):
ENDPOINT

 ENTER GRID COORDINATES:
      MINIMUM COLUMN VALUE,  MAXIMUM COLUMN VALUE
1 27
      MINIMUM ROW VALUE,  MAXIMUM ROW VALUE
1 27

 ENTER PAGE SIZE:
    1 = 8.5 X 11
    2 = 11 X 17
1

 WHAT ARE THE UNITS OF DISTANCE IN THE MODEL ?
    1 = FEET
    2 = METERS
1
MAXIMUM SIZE PLOT IS AT SCALE OF (1:   16120.)
IS THIS SCALE ACCEPTABLE ?
    [ Y = YES;      N OR <CR> = NO ]
Y

DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
    [ Y = YES;      N OR <CR> = NO ]
Y

 WHAT TYPE OF CHANGE DO YOU WANT TO MAKE ?
    1 = CHANGE AN ENTIRE LAYER
    2 = CHANGE AN INDIVIDUAL CELL
    3 = CHANGE ALL CELLS IN A BLOCK OF CELLS
2

 ENTER THE CELL INDICES:  J  I  K
14 14 4
```

```
 ENTER THE NEW ZONE CODE:
2

DO YOU WANT TO CHANGE SOME MORE ZONE CODES ?
    [ Y = YES;      N OR <CR> = NO ]
N

DO YOU WANT A COLOR PLOT ?
    [ Y = YES;      N OR <CR> = NO ]
N

DRAW THE PAGE BORDER ?
    [ Y = YES;      N OR <CR> = NO ]
N
```

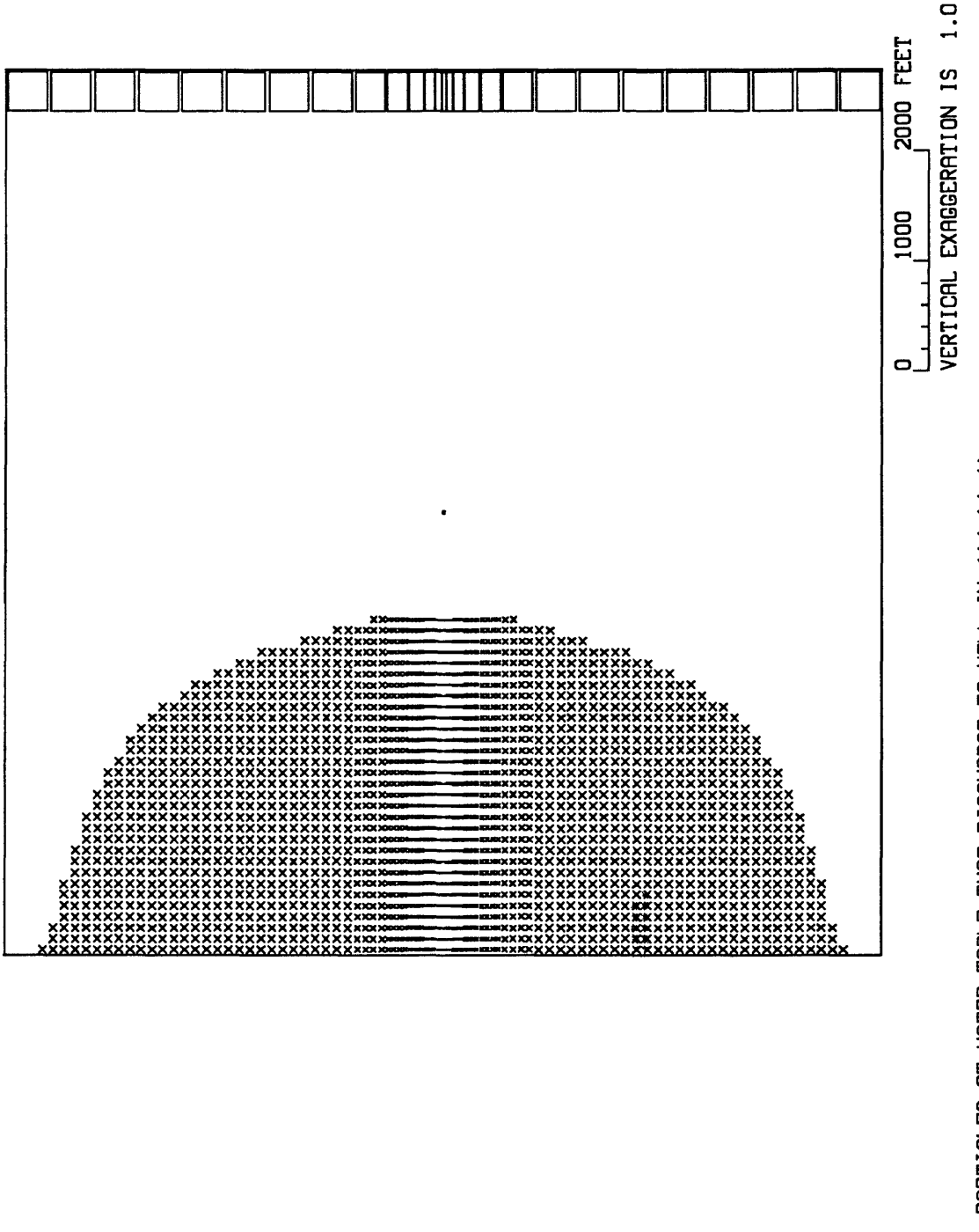


Figure 15. Map view of recharge locations for particles starting at the water table that discharge to cell (14, 14, 4).

The final example illustrates the use of "time series" mode in which the locations of a number of particles are computed, stored, and displayed at selected points in time. Figure 16 shows a map view of a cluster of 100 particles that originated at the water table in cell (5, 5, 1) after 0, 30, and 60 years. Time series mode is distinguished from the other modes by the fact that locations of particles at requested points of time are "stacked" in a file named TIMESERS. The first group of locations in the "stack" is always the initial locations of the particles. In contrast to path line mode, intermediate particle locations at entry points to cells are not recorded in a file. As with the other modes, time series mode produces a file named ENDPOINT that contains the initial and final locations of each particle. Tables 7 and 8 record the dialogue from MODPATH and MODPATH-PLOT for the time series analysis shown in figure 16. Note that in order to plot particle locations at time 0, as well as at 30 and 60 years, three time steps are specified in the interactive input to MODPATH-PLOT. The first time step corresponds to the initial conditions and always is designated as time step number 0. Particle locations after 30 and 60 years correspond to time step numbers 1 and 2, respectively. To plot just the initial conditions, the user would request only one time step and specify that time step to be number 0.

Table 7
Interactive dialogue from MODPATH for time series analysis

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS    83165

 SELECT THE MODE FOR STORING OUTPUT DATA:
     0 = ENDPOINTS AND STARTING POINTS ONLY
     1 = FLOW LINE COORDINATES
     2 = TIME SERIES DATA FOR SCATTER PLOTS
2

 HOW SHOULD POINTS IN TIME BE SPECIFIED ?
     0 = WITH A CONSTANT TIME STEP
     1 = VALUES OF TIME READ FROM A FILE
0

 ENTER: TIME STEP LENGTH, FACTOR FOR CONVERTING UNITS
30 365

 HOW ARE STARTING LOCATIONS TO BE ENTERED?
     1 = FROM AN EXISTING DATA FILE
     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
2

DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
    [Y = YES;     N OR <CR> = NO]
N

 IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
     0 = FORWARD IN THE DIRECTION OF FLOW
     1 = BACKWARDS TOWARD RECHARGE LOCATIONS
0

 HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
     0 = PARTICLES PASS THROUGH CELLS WITH WEAK SINKS
     1 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WITH INTERNAL SINKS
     2 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS WHERE DISCHARGE TO SINKS
         IS LARGER THAN A SPECIFIED FRACTION OF THE TOTAL INFLOW TO THE CELL
0

DO YOU WANT TO SPECIFY ONE ZONE IN WHICH TO STOP PARTICLES WHENEVER THEY ENTER ?
    [Y = YES;     N OR <CR> = NO]
N

 READING AND PROCESSING INPUT DATA...

 STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.

SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
    [Y = YES;     N OR <CR> = NO]
N

 ENTER DATA FOR SUBREGION  1 :

 DEFINE THE SUBREGION BY ENTERING THE MINIMUM AND MAXIMUM J,I,K COORDINATES.
   ENTER:  MINIMUM J, MAXIMUM J, MINIMUM I, MAXIMUM I, MINIMUM K, MAXIMUM K
5 5 5 5 1 1

 WHERE SHOULD THE PARTICLES BE LOCATED ?
   0 = WITHIN A CELL
   1 = ON ONE OR MORE OF THE CELL FACES
1
DO YOU WANT TO PLACE PARTICLES ON FACE 1 ?
    [Y = YES;     N OR <CR> = NO]
N
```

```
DO YOU WANT TO PLACE PARTICLES ON FACE 2 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 3 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 4 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 5 ?
    [Y = YES;    N OR <CR> = NO]
N
DO YOU WANT TO PLACE PARTICLES ON FACE 6 ?
    [Y = YES;    N OR <CR> = NO]
Y
 ENTER:  NJ  NI
    NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FACE 6
    NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FACE 6
10 10

 TOTAL NUMBER OF PARTICLES IS    100
 MAXIMUM NUMBER OF PARTICLES IS  83165

DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
    [Y = YES;    N OR <CR> = NO]
N

DO YOU WANT TO CHECK THE MASS BALANCE FOR ALL CELLS ?
    [Y = YES;    N OR <CR> = NO]
N

 DO YOU WANT TO CHECK DATA CELL BY CELL ?
    [Y = YES;    N OR <CR> = NO]
N

 NOW COMPUTING PARTICLE PATHS...
```

Table 8
Interactive dialogue from MODPATH-PLOT for time series analysis

```
 ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:
FILES.DAT

THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS   117470

 READING MODPATH DATA FILES...

 ENTER TITLE (80 CHARACTERS OR LESS):
TIME SERIES PLOT OF PARTICLE LOCATIONS AFTER 0, 30, AND 60 YEARS

 ENTER GRAPHICS DEVICE CODE:
     1 = TEKTRONIX 4010 (GRAPHICS TAB)
     2 = TEKTRONIX 4105 OR 4205
     3 = TEKTRONIX 4115 OR 4125
     4 = TEKTRONIX 4107, 4207, OR 4109
     5 = CREATE META FILE
5

DRAW INTERIOR GRID LINES ?
    [ Y = YES;     N OR <CR> = NO ]
N

 ENTER THE TYPE OF GRAPH:
     1 = FLOW LINE PLOT
     2 = MAP VIEW OF STARTING LOCATIONS (FORWARD TRACKING)
     3 = MAP VIEW OF FINAL LOCATIONS (FORWARD TRACKING)
     4 = MAP VIEW OF FINAL LOCATIONS (BACKWARD TRACKING)
     5 = TIME SERIES PLOT
5

 WHAT IS THE ORIENTATION OF THE PLOT ?
   1 = MAP VIEW
   2 = CROSS SECTION VIEW ALONG A COLUMN
   3 = CROSS SECTION VIEW ALONG A ROW
1

 WHAT DATA SHOULD BE PLOTTED ?
    0 = PLOT ALL DATA BY PROJECTION ONTO THE 2D SLICE
    1 = PLOT ONLY THE DATA WITHIN THE LAYER CORRESPONDING TO THE 2D SLICE
0

 ENTER NAME OF ENDPOINT FILE (<CR>="ENDPOINT"):
ENDPOINT

 ENTER NAME OF TIMESERIES FILE (<CR>="TIMESERS"):
TIMESERS

 HOW MANY TIME STEPS DO YOU WANT TO PLOT ?
   (YOU MAY PLOT UP TO 50 TIME STEPS)
   (TO PLOT ALL OF THE TIME STEPS, ENTER A NEGATIVE NUMBER)
3

 ENTER THE TIME STEP NUMBERS THAT YOU WANT TO PLOT:
0 1 2

 ENTER GRID COORDINATES:
      MINIMUM COLUMN VALUE,  MAXIMUM COLUMN VALUE
1 27

      MINIMUM ROW VALUE,  MAXIMUM ROW VALUE
1 27
 ENTER PAGE SIZE:
    1 = 8.5 X 11
    2 = 11 X 17
1
```

```
 WHAT ARE THE UNITS OF DISTANCE IN THE MODEL ?
    1 = FEET
    2 = METERS
1

MAXIMUM SIZE PLOT IS AT SCALE OF (1:   16120.)
IS THIS SCALE ACCEPTABLE ?
    [ Y = YES;      N OR <CR> = NO ]
Y

DO YOU WANT A COLOR PLOT ?
    [ Y = YES;      N OR <CR> = NO ]
N

DRAW THE PAGE BORDER ?
    [ Y = YES;      N OR <CR> = NO ]
N
```

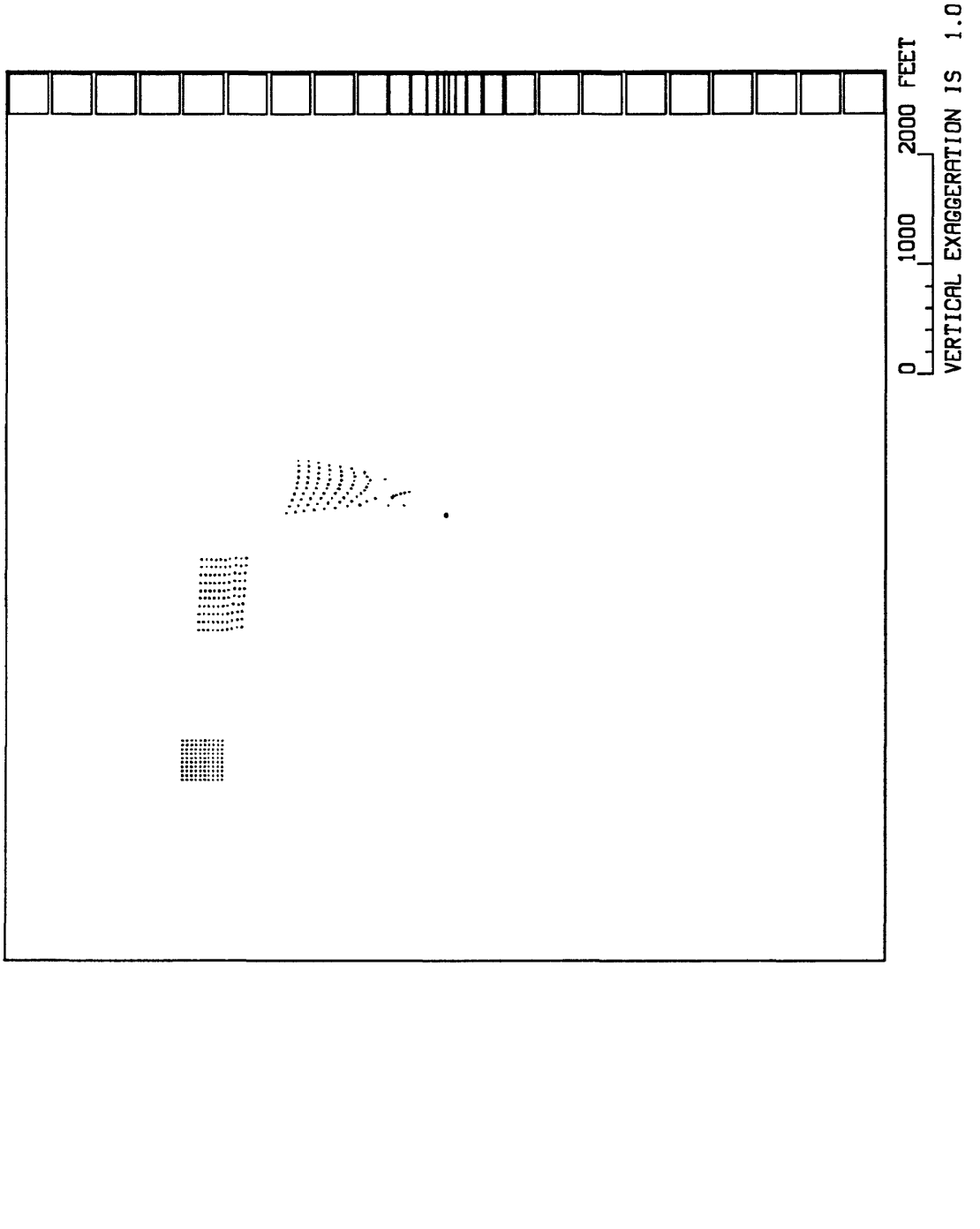


Figure 16. Map view showing locations of particles after 0, 30, and 60 years, based on a forward tracking time series analysis.

REFERENCES

Computer Associates, 1981, DISSPLA (Display Integrated Software System and Plotting LAnguage) User's Manual.

Garabedian, S. P. and L. F. Konikow, 1983, Front-tracking model for convective transport in flowing ground water: U. S. Geological Survey Water-Resources Investigations Report 83-4034, 55 p.

Konikow, L. F. and J. D. Bredehoeft, 1978, Computer model of two-dimensional solute transport and dispersion in ground water: U. S. Geological Survey Techniques of Water Resources Investigations, Book 7, 90 p.

McDonald, M.G., and Harbaugh, A.W., 1988, A modular three-dimensional finite-difference ground-water flow model: U.S. Geological Survey Techniques of Water Resources Investigations, Book 6, Chapter A1.

Mandle, R. J. and A. L. Kontis, 1986, Directions and rates of ground-water movement in the vicinity of Kesterson Reservoir, San Joaquin Valley, California: U. S. Geological Survey Water-Resources Investigations Report 86-4196, 57 p.

Pollock, David W., 1988, Semianalytical computation of path lines for finite difference models: Ground Water, v. 26, no. 6, p. 743-750.

Prickett, T. A., T. G. Naymik, and C. G. Lonnquist, 1981, A "random walk" solute transport model for selected ground-water quality evaluations,: Illinois State Water Survey, Champaign, IL, Bulletin 65, 103 p.

Shafer, J. M., 1987, Reverse pathline calculation of time-related capture zones in nonuniform flow: Ground Water, v. 25, no. 3, p. 283-289.

APPENDIX I -- INPUT FOR MAIN DATA FILE

DATA GROUP 1 -- GRID DIMENSIONS AND SPECIFICATIONS

1. DATA:	NCOL	NROW	NLAY	NCBL	IGRID
FORMAT:	8I10				

NCOL = number of columns
NROW = number of rows
NLAY = number of layers
NCBL = number of quasi-3D confining layers
IGRID = grid geometry code.
0 = grid with variable thickness and (or) nonhorizontal layers.
1 = true rectangular 3-D grid . (Layer 1 can be unconfined for IGRID = 1 provided that the bottom elevation of layer 1 is equal to a constant elevation.)

DATA GROUP 2 -- INPUT UNIT NUMBER ARRAY

2.DATA:	IUNIT (8)
FORMAT:	16I5

IUNIT = array containing FORTRAN unit numbers on which the following Modpath data sets will be read:

IUNIT (1) = recharge package data set
(2) = well package data set
(3) = evapotranspiration package data set
(4) = river package data set
(5) = drain package data set
(6) = general head package data set
(7) = unformatted (binary) BCF budget file from flow model output
(8) = unformatted (binary) head file from flow model output

DATA GROUP 3 -- LAYER TYPE CODES

3. DATA	LAYCON (NLAY)
FORMAT:	40I2

LAYCON = layer type
0 = confined
1 = unconfined (layer 1 only)
2 = convertible with constant transmissivity
3 = convertible with head-dependent transmissivity

DATA GROUP 4 -- CONFINING BED CODES
********** INCLUDE GROUP 4 ONLY IF NCBL > 0 **********

4. DATA:	NCON (NLAY)
FORMAT:	40I2

NCON = array containing the confining layer code for each model layer.

0 = no quasi-3D confining layer associated with the model layer
1 = a quasi-3D confining layer is associated with the model layer

---

DATA GROUP 5 -- HORIZONTAL GRID SPACING

5A. DATA: DELR (NCOL)
FORMAT: 1-D real array reader

5B. DATA: DELC (NROW)
FORMAT: 1-D real array reader

DELR = grid spacing along rows (x-direction in MODPATH)
DELC = grid spacing along columns (negative y-direction in MODPATH)

DATA GROUP 6 -- MODEL LAYER AND CONFINING LAYER THICKNESS
********** INCLUDE GROUP 6 ONLY IF IGRID = 1 **********

6A. DATA: DELZ (NLAY)
FORMAT: 1-D real array reader

6B. DATA: DELZCB (NLAY)
FORMAT: 1-D real array reader
**** INCLUDE 6B ONLY IF NCBL > 0 ****

6C. DATA: ZBL1
FORMAT: F10.0

DELZ = grid spacing (cell thickness) in the vertical direction (z-direction in MODPATH). The value input for DELZ of the top layer is ignored if the top layer is a water table layer (LAYCON = 1).

DELZCB = thickness of quasi-3D confining layers. NLAY values must be read even if only some of the model layers have underlying quasi-3D confining layers. A value of 0.0 should be entered for model layers that are not underlain by quasi-3D confining layers. The bottom layer cannot have an underlying quasi-3D confining layer, so DELZCB(NLAY) is always set equal to zero.

ZBL1 = bottom elevation of model layer 1. If layer 1 has an underlying quasi-3D confining layer, the value of ZBL1 corresponds to the top of that confining layer.

DATA GROUP 7 -- TOP AND BOTTOM ELEVATIONS FOR MODEL LAYERS
********** INCLUDE GROUP 7 ONLY IF IGRID = 0 **********

REPEAT THE FOLLOWING DATA IN SEQUENCE FOR EACH LAYER:

7A. DATA: ZTOP (NCOL,NROW)
FORMAT: 2-D real array reader
****DO NOT INCLUDE 7A FOR LAYER 1 IF LAYCON = 1 ****

7B. DATA: ZBOT (NCOL,NROW)
FORMAT: 2-D real array reader

ZTOP = array containing the elevation of the top of the model layer. A ZTOP array is not specified for layer 1 when layer 1 is unconfined (LAYCON = 1).

ZBOT = array containing the elevation of the bottom of the model layer. If the layer contains an underlying quasi-3D confining layer, ZBOT corresponds to the top of the confining layer. The thickness of that confining layer is calculated by taking the difference in elevation between ZBOT and the elevation of the top of the underlying model layer.

DATA GROUP 8 -- IBOUND

REPEAT THE FOLLOWING DATA IN SEQUENCE FOR EACH LAYER:

8. DATA:	IBOUND (NCOL,NROW)
FORMAT:	2-D integer array reader

IBOUND = array containing integer numbers designating cell type

IBOUND < 0	specified head cell
IBOUND = 0	inactive cell
IBOUND > 0	active cell

*********MODPATH requires that the absolute value of IBOUND be less than 1000 **********

DATA GROUP 9 -- POROSITY

REPEAT THE FOLLOWING DATA IN SEQUENCE FOR EACH LAYER:

9A. DATA:	POR (NCOL,NROW)
FORMAT:	2-D real array reader

If the model layer has a quasi-3D confining layer, then

9B. DATA:	Porcb (NCOL,NROW)
FORMAT:	2-D real array reader

POR = array containing porosity values for a model layer.

Porcb = array containing porosity values for a quasi-3D confining layer associated with the model layer. In MODPATH, all porosity values are stored in a 3-dimensional array, POR, that is dimensioned: POR (NCOL, NROW, NLAY + NCBL).

APPENDIX II -- INPUT FOR ARRAY UTILITY SUBROUTINES

Array data from the main data file (Appendix I) is read using utility subroutines patterned after those in the modular flow model. Input has the following format:

FOR REAL ARRAY READER (1-D AND 2-D)

Data:	LOCAT	CNSTNT	FMT	IPRN
Format	I10	F10.0	A20	I10

FOR INTEGER ARRAY READER (2-D)

Data:	LOCAT	ICON	FMT	IPRN
Format:	I10	I10	A20	I10

Explanation of Fields Used in Input Instructions

LOCAT -- Indicates the location of the data which will be put in the array.

If LOCAT ≤ 0, every element in the array will be set equal to the value CNSTNT/ICON

If LOCAT > 0,it is the unit number from which data values will be read in the format specified in the third field of the array-control record (FMT).

CNSTNT/ICON -- is a constant. Its use depends on the value of LOCAT.

If LOCAT ≤ 0, every element in the array is set equal to CNSTNT/ICON.

IF LOCAT > 0, when CNSTNT/ICON is not 0, every element in the array is multiplied by CNSTNT/ICON.

FMT -- is the format of records containing the array values. It is used only if LOCAT is positive. The format must be enclosed in parentheses. If the field containing FMT in the array-control record is left blank, the array data is handled as list-directed input which is read in format-free style. Format-free style requires that the entries in the data set be separated by a blank or comma.

IPRN -- is a flag indicating whether an array should be printed.

If IPRN = 0, the array is not printed. A message is written in the SUMMARY.PTH or SUMMARY.PLT file after the data is successfully read.

If IPRN = 1, the array is printed in (10E13.5) or (25I5) format.

APPENDIX III -- INPUT FOR STRESS PACKAGE DATA FILES

The input instructions listed below are for a steady-state simulation using one stress period. Only those variables that have a special interpretation in MODPATH are defined. More detailed information is provided in McDonald and Harbaugh (1988).

Recharge Package:

The input to the recharge package is read on the unit specified in IUNIT(1) in MODPATH.

1. Record: NRCHOP IRCHCB ITOP
   Format: I10 I10 I10

2. Record: INRECH INIRCH
   Format: I10 I10

3. Array: RECH (NCOL,NROW)
   Format: 2-D real array reader

If NRCHOP is equal to 2 then

4. Array: IRCH (NCOL,NROW)
   Format: 2-D integer array reader

IRCHCB -- is a flag or unit number for storing cell-by-cell budget terms. Cell-by-cell budget terms do not need to be stored for the recharge package.

ITOP -- is a flag indicating how recharge is distributed in the cell.

If ITOP = 0, recharge is treated as a distributed source.

If ITOP = 1, recharge is assigned to the top face of the cell.

Well Package:

Input to the well package is read on the unit specified in IUNIT(2) in MODPATH.

1. Record: MXWELL IWELCB
   Format: I10 I10

2. Record: ITMP
   Format: I10

3. Record: Layer Row Column Q IFACE
   Format: I10 I10 I10 F10.0 I10

MXWELL -- is the maximum number of wells used during simulation.

IWELCB -- is a flag or unit number for cell-by-cell flow terms. Cell-by-cell budget terms do not need to be stored for the well package.

ITMP -- is a counter. ITMP is the number of wells active during the steady state simulation (ITMP $\geq$ 0).

IFACE -- is a flag indicating how well flow rates are to be handled.

If IFACE = 0 or > 6, flow term is treated as an internal source or sink.

If IFACE ≥ 1 and ≤ 6, flow term is assigned to cell face corresponding to a number 1 through 6.

If IFACE < 0, flow term is apportioned uniformly among all faces perpendicular to the horizontal plane that bound inactive cells. This is accomplished by assigning a single average velocity to the "boundary" faces. The average velocity is computed by dividing the total flow rate by porosity and the total cross sectional area of the boundary faces. If there are no inactive boundary cells, the flow term is treated as an internal source or sink.

Evapotranspiration Package:

Input to the evapotranspiration package is read on the unit specified in IUNIT(3) in MODPATH.

1. Record: NEVTOP IEVTCB ITOP
   Format: I10 I10 I10

2. Record: INSURF INEVTR INEXDP INIEVT
   Format: I10 I10 I10 I10

3. Array: SURF (NCOL,NROW)
   Format: 2-D real array reader

4. Array: EVTR (NCOL,NROW)
   Format: 2-D real array reader

5. Array: EXDP (NCOL,NROW)
   Format: 2-D real array reader

If NEVTOP is equal to 2 then

6. Array: IEVT (NCOL,NROW)
   Format: 2-D integer array reader

IEVTCB -- is a unit number for storing cell-by-cell budget terms. In MODPATH, it is the unit from which the cell-by-cell budget terms are read. IEVTCB > 0. To run MODPATH, cell-by-cell budget terms must be stored.

ITOP -- is a flag indicating how evapotranspiration flows are to be handled in MODPATH.

If ITOP = 0, flow terms are treated as internal sinks.

If ITOP = 1, flow terms are assigned to the top of cells.

River Package:

The input to the river package is read on the unit specified in IUNIT(4) in MODPATH.

1. Record: MXRIVR IRIVCB
   Format: I10 I10

2. Record: ITMP
   Format: I10

3. Record: Layer Row Column Stage Cond Rbot IFACE
   Format: I10 I10 I10 F10.0 F10.0 F10.0 I10
   (Input item 3 normally consists of one record for each river reach. If ITMP is negative or zero, item 3 is not read.)

MXRIVR -- is the maximum number of reaches active at one time.

IRIVCB -- is a unit number for storing cell-by-cell budget terms. In MODPATH, it is the unit from which the cell-by-cell budget terms are read. IRIVCB > 0. To run MODPATH, cell-by-cell budget terms must be stored in a file.

ITMP -- is a counter. ITMP is the number of reaches active during the steady state simulation. (ITMP ≥ 0)

IFACE -- is a flag indicating how budget terms are to be handled.

If IFACE = 0 or > 6, the flow term is treated as an internal source or sink.

If IFACE ≥1 and ≤ 6, the flow term is assigned to the cell face corresponding to a number 1 through 6.

If IFACE < 0, the flow term is uniformly apportioned between all faces perpendicular to the horizontal plane that bound inactive cells, as previously described for the well package. If there are no inactive boundary cells, the flow term is treated as an internal source or sink.

Drain Package:

The input to the drain package is read on the unit specified in IUNIT(5) in MODPATH.

1. Record: MXDRN IDRNCB
   Format: I10 I10

2. Record: ITMP
   Format: I10

3. Record: Layer Row Column Elevation Cond IFACE
   Format: I10 I10 I10 F10.0 F10.0 I10
   (Input item 3 normally consists of one record for each drain reach. If ITMP is negative or zero, item 3 is not read.)

MXDRN -- is the maximum number of drains active at one time.

IDRNCB -- is a unit number for storing cell-by-cell budget terms. In MODPATH, it is the unit from which the cell-by-cell budget terms are read. IDRNCB > 0. To run MODPATH, cell-by-cell budget terms must be stored in a file.

ITMP -- is a counter. ITMP is the number of reaches active during the steady state simulation. (ITMP ≥ 0)

IFACE -- is a flag indicating how budget terms are to be handled.

If IFACE = 0 or > 6, the flow term is treated as an internal source or sink.

If IFACE ≥1 and ≤ 6, the flow term is assigned to the cell face corresponding to a number 1 to 6.

If IFACE < 0, the flow term is uniformly apportioned between all faces perpendicular to the horizontal plane that bound inactive cells, as previously described for the well package. If there are no inactive boundary cells, the flow term is treated as an internal source or sink.

General Head Boundary Package:

The input to the general head boundary package is read on the unit specified in IUNIT(6) in MODPATH.

1. Record: MXBND IGHBCB
   Format: I10 I10

2. Record: ITMP
   Format: I10

3.

Record:	Layer	Row	Column	Boundary Head	Cond	IFACE
Format:	I10	I10	I10	F10.0	F10.0	I10

(Input item 3 normally consists of one record for each general head boundary. If ITMP is negative or zero, item 3 is not read.)

MXBND -- is the maximum number of general head boundaries active at one time.

IGHBCB -- is a unit number for storing cell-by-cell budget terms. In MODPATH, it is the unit from which the cell-by-cell budget terms are read. IGHBCB > 0. To run MODPATH, cell-by-cell budget terms must be stored in a file.

ITMP -- is a counter. ITMP is the number of reaches active during the steady state simulation. (ITMP ≥ 0)

IFACE -- is a flag indicating how budget terms are to be handled.

If IFACE = 0 or > 6, the flow term is treated as an internal source or sink.

If IFACE ≥1 and ≤ 6, the flow term is assigned to the cell face corresponding to a number 1 to 6.

If IFACE < 0, the flow term is uniformly apportioned between all faces perpendicular to the horizontal plane that bound inactive cells, as previously described for the well package. If there are no inactive boundary cells, the flow term is treated as an internal source or sink.

APPENDIX IV -- OUTPUT FILES

ENDPOINT FILE

The ENDPOINT file contains a one-line record for each particle. Each record contains 16 variables, and is written with the format,

FORMAT (4I4,8E12.4,4I4).

The variables, in the order they appear on the record, are defined as,

(1) Value of IBOUND for the cell containing the final location of the particle

(2) J (column) index for the cell containing the final location

(3) I (row) index for the cell containing the final location

(4) K (layer) index for the cell containing the final location

(5) Global coordinate in the x-direction (J index direction) for the final location

(6) Global coordinate in the y-direction (I index direction) for the final location

(7) Global coordinate in the z-direction (K index direction) for the final location

(8) Local coordinate for the z-direction within the grid cell (0 to 1 within the model layer, -1 to 0 within an underlying confining layer)

(9) Total time of travel

(10) Global coordinate in the x-direction for starting location

(11) Global coordinate in the y-direction for starting location

(12) Local coordinate in the z-direction within the cell for starting location

(13) J index for cell containing starting location

(14) I index for cell containing starting location

(15) K index for cell containing starting location

(16) Value of IBOUND for cell containing starting location

PATHLINE FILE

When mode 1 is selected, coordinates along the path of each particle are recorded in a file named PATHLINE. The file contains the starting coordinates of a particle, and the coordinates at every point where a particle enters a new cell or a confining layer. In addition, coordinates of intermediate points are recorded whenever the cumulative travel time corresponds to a point in time for which a data point was requested. These intermediate points are flagged in the PATHLINE file by multiplying the travel time by -1 and storing the negative value. The PATHLINE file contains a sequence of one-line records, each line containing coordinate and location information for one point on a path line. Each record contains nine variables, and is written with the format,

FORMAT (I5,1X,5(E20.12,1X),2(I3,1X),I3).

The variables, in the order they appear on the line, are defined as:

(1) Particle index number

(2) Global coordinate in the x-direction

(3) Global coordinate in the y-direction

(4) Local coordinate in the z-direction within the cell

(5) Global coordinate in the z-direction

(6) Cumulative travel time

(7) J index of cell containing the point

(8) I index of cell containing the point

(9) K index of cell containing the point

TIME SERIES FILE

When mode 2 is selected, the locations of particles at specified points in time are computed and recorded in a file named TIMESERS. In this mode, the locations of all particles are computed for a specified point in time, and recorded in sequence in TIMESERS. The procedure is repeated at each point in time for which output is specified. The TIMESERS file contains a series of one-line records, each containing coordinate and location information for a particle. Initial locations are always recorded first in the TIMESERS file and are given a time step index number of 0. Each record contains 10 variables, and is written in the format,

FORMAT (2(I4,1X),3(I3,1X),4(E20.12,1X),E20.12)

The variables, in order, are defined as:

(1) Time step index number

(2) Particle index number

(3) J index of cell containing particle

(4) I index of cell containing particle

(5) K index of cell containing particle

(6) Global coordinate in x-direction

(7) Global coordinate in y-direction

(8) Global coordinate in z-direction

(9) Local coordinate in z-direction with the cell

(10) Cumulative time of travel

APPENDIX V -- SAMPLE PROBLEM DATA FILES

MODULAR FINITE-DIFFERENCE FLOW MODEL DATA FILES

BAS Package File (BAS.DAT)

```
MODPATH SAMPLE PROBLEM

         5         27         27          1          4
 14 15 00 16 00 00 00 17 18 00 00 19
         0          0
         0          1
         0          1
         0          1
         0          1
         0          1
     999.9
         0       280.
         0       280.
         0       280.
         0       280.
         0       280.
       1.0          1        1.0
```

BCF Package File (BCF.DAT)

```
         1         50
 1 0 0 0 0
         0        1.0
        14        1.0     (16F5.0)
 400. 400. 400. 400. 400. 400. 400. 300. 200. 150. 100.  60.  40.  60. 100.
 150. 200. 300. 400. 400. 400. 400. 400. 400. 400. 400.
        14        1.0     (16F5.0)
 400. 400. 400. 400. 400. 400. 400. 400. 300. 200. 150. 100.  60.  40.  60. 100.
 150. 200. 300. 400. 400. 400. 400. 400. 400. 400. 400.
         0        50.
         0       220.
         0    0.0005
         0      1250.
         0       0.01
         0      1250.
         0       0.01
         0      1250.
         0       0.01
         0      1250.
```

Recharge Package File (RCH.DAT)

```
         1          0          1
         1          0
         0    0.0045
```

Well Package File (WEL.DAT)

```
         1          0
         1
         4         14         14   -80000.          0
```

River Package File (RIV.DAT)

```
27      50
27
 1       1      27     280.    3200.    275.      6
 1       2      27     280.    3200.    275.      6
 1       3      27     280.    3200.    275.      6
 1       4      27     280.    3200.    275.      6
 1       5      27     280.    3200.    275.      6
 1       6      27     280.    3200.    275.      6
 1       7      27     280.    3200.    275.      6
 1       8      27     280.    3200.    275.      6
 1       9      27     280.    2400.    275.      6
 1      10      27     280.    1600.    275.      6
 1      11      27     280.    1200.    275.      6
 1      12      27     280.     800.    275.      6
 1      13      27     280.     480.    275.      6
 1      14      27     280.     320.    275.      6
 1      15      27     280.     480.    275.      6
 1      16      27     280.     800.    275.      6
 1      17      27     280.    1200.    275.      6
 1      18      27     280.    1600.    275.      6
 1      19      27     280.    2400.    275.      6
 1      20      27     280.    3200.    275.      6
 1      21      27     280.    3200.    275.      6
 1      22      27     280.    3200.    275.      6
 1      23      27     280.    3200.    275.      6
 1      24      27     280.    3200.    275.      6
 1      25      27     280.    3200.    275.      6
 1      26      27     280.    3200.    275.      6
 1      27      27     280.    3200.    275.      6
```

Output Control File (OC.DAT)

```
 4       4      52       0
 0       1       1       1
 1       0       1       0
```

Strongly Implicit Procedure Package File (SIP.DAT)

```
400      5
 1.    .001      0    0.0005      1
```

Main Data File (MAIN.DAT)

```
        27         27          5          1          1
   17   15    0    16     0     0    50    52
 1 0 0 0 0
 1 0 0 0 0
       100        1.0     (16F5.0)
 400. 400. 400. 400. 400. 400. 400. 400. 300. 200. 150. 100.  60.  40.  60. 100.
 150. 200. 300. 400. 400. 400. 400. 400. 400. 400. 400.
       100        1.0     (16F5.0)
 400. 400. 400. 400. 400. 400. 400. 400. 300. 200. 150. 100.  60.  40.  60. 100.
 150. 200. 300. 400. 400. 400. 400. 400. 400. 400. 400.
       100        1.0     (16F5.0)
 100.  50.  50.  50.  50.
       100        1.0
  20.   0.   0.   0.   0.
 220.
         0          1
         0          1
         0          1
         0          1
         0          1
         0        0.3
         0        0.3
         0        0.3
         0        0.3
         0        0.3
         0        0.3
```

Name and Unit Number List File (FILES.DAT)

```
100  'MAIN.DAT'
 15  'WEL.DAT'
 16  'RIV.DAT'
 17  'RCH.DAT'
-50  'BUDGET.OUT'
-52  'HEADS.OUT'
```

```
C---Version 1.0   July 21, 1989
C*************************************************************************# A00020
C                              MODPATH                                   *# A00030
C                                                                        *# A00040
C  A three-dimensional particle tracking post-processing program for     *# A00050
C      the USGS modular three-dimensional ground-water flow model        *# A00060
C                                                                        *# A00070
C                                  by                                    *# A00080
C                                                                        *# A00090
C                           David W. Pollock                             *# A00100
C                                                                        *# A00110
C*************************************************************************# A00120
C                                                                         # A00130
C----- REDIMENSION MASTER ARRAY BY CHANGING THE NEXT TWO STATEMENTS ----# A00140
C----- MAKE SURE THAT LENA IS SET EQUAL TO THE LENGTH OF THE A ARRAY ---# A00150
      COMMON A(700000)                                                   # A00160
      LENA=700000                                                        # A00170
C-------------------------------------------------------------------------# A00180
C                                                                         # A00190
C  SET UNIT NUMBERS FOR SCREEN I/O AND FILES THAT ARE SPECIFIED BY        # A00200
C  INTERACTIVE I/O                                                        # A00210
C                                                                         # A00220
C  I0 IS THE UNIT NUMBER OF FILE CONTAINING FILE NAMES AND UNIT NUMBERS # A00230
      I0=101                                                             # A00240
C  I2 IS THE UNIT NUMBER FOR THE "PATHLINE"  FILE                         # A00250
      I2=102                                                             # A00260
C  I3 IS THE UNIT NUMBER FOR THE "ENDPOINT"  FILE                         # A00270
      I3=103                                                             # A00280
C  I4 IS THE UNIT NUMBER FOR THE "TIMESERS"   FILE                        # A00290
      I4=104                                                             # A00300
C  I5 IS THE UNIT NUMBER FOR A SCRATCH FILE                               # A00310
      I5=105                                                             # A00320
C  I6 IS THE UNIT NUMBER FOR THE STARTING LOCATIONS FILE                  # A00330
      I6=106                                                             # A00340
C  I7 IS THE UNIT NUMBER FOR THE "SUMMARY.PTH"  FILE                      # A00350
      I7=107                                                             # A00360
C  I8 IS THE UNIT NUMBER FOR THE TIME STEP FILE                           # A00370
      I8=108                                                             # A00380
C  I9 IS THE UNIT NUMBER FOR THE FILE TO WHICH SCREEN INPUT IS            # A00390
C     ECHOED IF THE RUN IS INTERACTIVE (IBATCH=0). IF THE RUN IS          # A00400
C     BATCH (IBATCH NOT EQUAL TO 0), "SCREEN" INPUT IS READ FROM          # A00410
C     A FILE OPENED TO UNIT I9                                            # A00420
C                                                                         # A00430
      I9=109                                                             # A00440
C                                                                         # A00450
C                                                                         # A00460
C  "IBATCH" IS A FLAG INDICATING WHETHER THE RUN HAS INTERACTIVE          # A00470
C  SCREEN INPUT OR IS "BATCH"                                             # A00480
C                                                                         # A00490
C  IBATCH = 0 --> INPUT AT SCREEN                                         # A00500
C  IBATCH NOT EQUAL TO 0 --> ALL INPUT READ FROM FILES                    # A00510
C                                                                         # A00520
C  THE VALUE OF IBATCH IS SET BY A CALL TO SUBROUTINE IOTYPE WHICH        # A00530
C  OPENS A FILE NAMED "TERMIN.PTH". IF TERMIN.PTH DOES NOT EXIST, IT IS # A00540
C  CREATED BY IOTYPE. IOTYPE READS THE FIRST LINE OF TERMIN.PTH. IBATCH # A00550
C  IS SET TO 0 IF THE FIRST LINE CONTAINS "SCREEN". IT IS SET TO 1 IF    # A00560
C  THE FIRST LINE CONTAINS "BATCH". IF THE FILE HAS JUST BEEN CREATED,   # A00570
C  IBATCH IS SET TO 0 AND THE WORD "SCREEN" IS WRITTEN IN THE FIRST      # A00580
C  LINE OF THE FILE.                                                      # A00590
C                                                                         # A00600
      CALL IOTYPE(IBATCH,I9,I7)                                          # A00610
C                                                                         # A00620
      IA=1                                                               # A00630
C                                                                         # A00640
C  "IA" IS AN EXTRA UNIT NUMBER THAT CAN BE SET TO THE DEFAULT SCREEN    # A00650
C  UNIT NUMBER FOR THE COMPUTER SYSTEM OR TO ANOTHER OUTPUT DEVICE.      # A00660
C  THE PARAMETER IS PASSED TO SEVERAL SUBROUTINES, BUT IS NOT USED.      # A00670
C                                                                         # A00680
C  ALL SCREEN INPUT AND OUTPUT IS DONE USING THE "*" CHARACTER FOR       # A00690
```

```
C  THE UNIT PARAMETER IN READ AND WRITE STATEMENTS                      # A00700
C                                                                       # A00710
C-----------------------------------------------------------------------# A00720
C                                                                       # A00730
C  OPEN THE DATA FILES CONTAINING THE BASIC INFORMATION ABOUT THE FLOW  # A00740
C  SYSTEM. ALSO OPEN STANDARD OUTPUT AND SCRATCH FILES.                 # A00750
C                                                                       # A00760
      CALL FILES (IA,I0,I1,I2,I3,I4,I5,I6,I7,I8,I9,IBATCH)             # A00770
C                                                                       # A00780
C  ALLOCATE SPACE FOR ARRAYS                                            # A00790
C                                                                       # A00800
      CALL SPACE(LENA,NCOL,NROW,NLAY,NCBL,IGRID,NLPOR,NZDIM,NUNIT,LCQX, # A00810
     1LCQY,LCQZ,LCPOR,LCIBOU,LCXMAX,LCDX,LCYMAX,LCDY,LCZBOT,LCZTOP,     # A00820
     2LCDZ,LCHEAD,LCBUFF,LCLAYC,LCNCON,LCDZCB,LCIBUF,LCIUN,LCXLC,LCQSS, # A00830
     3LCYLC,LCZLC,LCZLL,LCTOT,LCJLC,LCILC,LCKLC,NPART,NCP1,NRP1,NLP1,IA,# A00840
     4I1,I7,I9,IBATCH)                                                  # A00850
C                                                                       # A00860
C  EXECUTE TRACKING PROGRAM                                             # A00870
C                                                                       # A00880
      CALL DRIVER (NCOL,NROW,NLAY,NCBL,IGRID,NLPOR,NZDIM,A(LCQX),       # A00890
     1A(LCQY),A(LCQZ),A(LCPOR),A(LCIBOU),A(LCXMAX),A(LCDX),A(LCYMAX),   # A00900
     2A(LCDY),A(LCZBOT),A(LCZTOP),A(LCDZ),A(LCHEAD),A(LCBUFF),          # A00910
     3A(LCLAYC),A(LCNCON),A(LCDZCB),A(LCIBUF),A(LCIUN),A(LCXLC),        # A00920
     4A(LCYLC),A(LCZLC),A(LCZLL),A(LCTOT),A(LCJLC),A(LCILC),A(LCKLC),   # A00930
     5A(LCQSS),NPART,NCP1,NRP1,NLP1,NUNIT,MODE,IA,I1,I2,I3,I4,          # A00940
     6I5,I6,I7,I8,I9,IBATCH)                                            # A00950
C                                                                       # A00960
      STOP                                                              # A00970
      END                                                               # A00980
C-----END OF ROUTINE--------------------------------------------------# A00990
```

```
C                                                                       # B00010
C---Version 1.0   July 21, 1989                                         # B00020
C***********************************************************************# B00030
C                                                                       # B00040
C                              DRIVER                                   # B00050
C                                                                       # B00060
C  THIS IS THE MAIN SUBROUTINE THAT CONTROLS THE OVERALL SEQUENCE OF    # B00070
C  COMPUTATIONS FOR THE PARTICLE TRACKING ANALYSIS.                     # B00080
C                                                                       # B00090
C***********************************************************************# B00100
C                                                                       # B00110
      SUBROUTINE DRIVER (NCOL,NROW,NLAY,NCBL,IGRID,NLPOR,NZDIM,QX,      # B00120
     1QY,QZ,POR,IBOUND,XMAX,DELR,YMAX,DELC,ZBOT,ZTOP,DELZ,HEAD,BUFF,    # B00130
     2LAYCON,NCON,DELZCB,IBUFF,IUNIT,XLC,YLC,ZLC,ZLLC,TOT,JLC,ILC,KLC,  # B00140
     3QSS,NPART,NCP1,NRP1,NLP1,NUNIT,MODE,IA,I1,I2,I3,I4,I5,            # B00150
     4I6,I7,I8,I9,IBATCH)                                               # B00160
C                                                                       # B00170
      DIMENSION TOT(NPART),XLC(NPART),YLC(NPART),ZLC(NPART),            # B00180
     1ZLLC(NPART),JLC(NPART),ILC(NPART),KLC(NPART),QX(NCP1,NROW,NLAY),  # B00190
     2QY(NCOL,NRP1,NLAY),QZ(NCOL,NROW,NLP1),POR(NCOL,NROW,NLPOR),       # B00200
     3IBOUND(NCOL,NROW,NLAY),XMAX(NCOL),DELR(NCOL),YMAX(NROW),          # B00210
     4DELC(NROW),ZBOT(NZDIM),ZTOP(NZDIM),DELZ(NLAY),DELZCB(NLAY),       # B00220
     5IBUFF(NCOL,NROW,NLAY),IUNIT(NUNIT),HEAD(NCOL,NROW,NLAY),          # B00230
     6LAYCON(NLAY),NCON(NLAY),BUFF(NCOL,NROW,NLAY),                     # B00240
     7QSS(NCOL,NROW,NLAY)                                               # B00250
      CHARACTER*80 MES                                                  # B00260
C                                                                       # B00270
      CALL DATIN (LAYCON,NCON,HEAD,DELT,XMAX,YMAX,DELR,DELC,DELZ,       # B00280
     1DELZCB,ZTOP,ZBOT,IBOUND,POR,QX,QY,QZ,IUNIT,NCOL,NROW,NLAY,NLPOR,  # B00290
     2NCP1,NRP1,NLP1,NUNIT,NZDIM,IRP,NPART,NPRT,MODE,FRAC,BUFF,IREV,    # B00300
     3IBUFF,IGRID,JLC,ILC,KLC,XLC,YLC,ZLC,ZLLC,NCBL,ITMS,QSS,ISNK,      # B00310
     4ITREAD,IA,I1,I2,I3,I4,I5,I6,I7,I8,I9,IZSTOP,IPEP,                 # B00320
     5IBATCH)                                                           # B00330
C                                                                       # B00340
C  CHECK MASS BALANCES                                                  # B00350
C                                                                       # B00360
      MES= 'DO YOU WANT TO CHECK THE MASS BALANCE FOR ALL CELLS ?'      # B00370
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # B00380
      IF(IANS.EQ.1) THEN                                                # B00390
      IF (IBATCH.EQ.0)                                                  # B00400
     1WRITE (*,*) 'SPECIFY AN ERROR TOLERANCE (IN PERCENT):'            # B00410
      IF(IBATCH.EQ.0) THEN                                              # B00420
      READ (*,*) ETOL                                                   # B00430
      WRITE(I9,*) ETOL                                                  # B00440
      ELSE                                                              # B00450
      READ(I9,*) ETOL                                                   # B00460
      END IF                                                            # B00470
      CALL BALNCE (QX,QY,QZ,QSS,IBOUND,HEAD,NCOL,NROW,NLAY,             # B00480
     1NCP1,NRP1,NLP1,KOUNT,IA,I7,I9,IBATCH,ETOL)                        # B00490
      IF(KOUNT.GT.0) THEN                                               # B00500
      MES= 'DO YOU WANT TO CONTINUE ?'                                  # B00510
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # B00520
      IF(IANS.EQ.0) STOP                                                # B00530
      END IF                                                            # B00540
      END IF                                                            # B00550
C                                                                       # B00560
C  CHECK DATA CELL BY CELL                                              # B00570
C                                                                       # B00580
      MES= ' DO YOU WANT TO CHECK DATA CELL BY CELL ?'                  # B00590
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # B00600
      IF(IANS.EQ.1) THEN                                                # B00610
      CALL CELDAT (XMAX,YMAX,DELR,DELC,ZBOT,ZTOP,DELZ,                  # B00620
     1DELZCB,POR,IBOUND,HEAD,QX,QY,QZ,LAYCON,NCOL,NROW,NLAY,NZDIM,      # B00630
     2IGRID,NLPOR,NCP1,NRP1,NLP1,NCON,QSS,IA,I7,I9,IBATCH)              # B00640
C                                                                       # B00650
      MES= ' DOES THE DATA APPEAR TO BE CORRECT?'                       # B00660
      CALL YESNO(IBATCH,I9,MES,IA,IANS)                                 # B00670
      IF(IANS.EQ.0) RETURN                                              # B00680
      END IF                                                            # B00690
C                                                                       # B00700
C                                                                       # B00710
C  SWITCH SIGNS OF FLOWS IF REVERSE TRACKING IS USED                    # B00720
```

```
C                                                                       # B00730
      IF(IREV.EQ.1) THEN                                                # B00740
      DO 10 K=1,NLAY                                                    # B00750
      DO 10 I=1,NROW                                                    # B00760
      DO 10 J=1,NCP1                                                    # B00770
10    QX(J,I,K)= -QX(J,I,K)                                             # B00780
      DO 20 K=1,NLAY                                                    # B00790
      DO 20 I=1,NRP1                                                    # B00800
      DO 20 J=1,NCOL                                                    # B00810
20    QY(J,I,K)= - QY(J,I,K)                                            # B00820
      DO 30 K=1,NLP1                                                    # B00830
      DO 30 I=1,NROW                                                    # B00840
      DO 30 J=1,NCOL                                                    # B00850
30    QZ(J,I,K)= -QZ(J,I,K)                                             # B00860
      END IF                                                            # B00870
C                                                                       # B00880
      IOP=0                                                             # B00890
      IF(MODE.NE.2) THEN                                                # B00900
      MES= 'DO YOU WANT TO WRITE ENDPOINT INFORMATION TO THE SCREEN ?'   # B00910
      CALL YESNO(IBATCH,I9,MES,IA,IOP)                                  # B00920
      END IF                                                            # B00930
C                                                                       # B00940
      IF (IBATCH.EQ.0) THEN                                             # B00950
      WRITE (*,*) ' '                                                   # B00960
      WRITE (*,*) 'NOW COMPUTING PARTICLE PATHS...'                     # B00970
      WRITE (*,*) ' '                                                   # B00980
      END IF                                                            # B00990
C                                                                       # B01000
      KOUNT=0                                                           # B01010
      KSTP=0                                                            # B01020
      NSTP=0                                                            # B01030
      KPER=0                                                            # B01040
      TIME=0.0E+0                                                       # B01050
      TMAX=0.0E+0                                                       # B01060
      DO 40 N=1,NPART                                                   # B01070
40     TOT(N)=TIME                                                      # B01080
      IF (ITMS.EQ.1.AND.ITREAD.EQ.1) THEN                               # B01090
      READ(I8,'(I10)') NPER                                             # B01100
      IF (NPER.EQ.0) THEN                                               # B01110
      WRITE (I7,*) 'TIME DATA FILE SAYS NPER = 0. RUN STOPPED.'         # B01120
      IF (IBATCH.EQ.0)                                                  # B01130
     1 WRITE (*,*) 'TIME DATA FILE SAY NPER = 0. RUN STOPPED.'          # B01140
      STOP                                                              # B01150
      END IF                                                            # B01160
      END IF                                                            # B01170
C                                                                       # B01180
C  IF TIME SERIES MODE, WRITE INITIAL CONDITIONS IN TIMESERIES FILE     # B01190
C                                                                       # B01200
      IF (MODE.EQ.2) THEN                                               # B01210
      CALL ZCALC (IGRID,NCOL,NROW,NLAY,ZLC,ZLLC,ZTOP,ZBOT,              # B01220
     1 HEAD,LAYCON,ILC,JLC,KLC,NZDIM,NPART,NPRT)                        # B01230
      DO 50 N=1,NPRT                                                    # B01240
      WRITE(I4,5020) KOUNT,N,JLC(N),ILC(N),KLC(N),XLC(N),YLC(N),ZLC(N), # B01250
     1ZLLC(N),TMAX                                                      # B01260
50    CONTINUE                                                          # B01270
      END IF                                                            # B01280
C                                                                       # B01290
C---------------------  START TIME LOOP  ------------------------       # B01300
C                                                                       # B01310
60    CONTINUE                                                          # B01320
      TOLD=TMAX                                                         # B01330
      KOUNT=KOUNT+1                                                     # B01340
      KSTP=KSTP+1                                                       # B01350
C                                                                       # B01360
C  INCREMENT TMAX                                                       # B01370
C                                                                       # B01380
C  ITMS = 0 -- LOCATIONS ARE NOT COMPUTED AT SPECIFIC POINTS IN TIME    # B01390
C  ITMS = 1 -- LOCATIONS ARE COMPUTED AT SPECIFIC POINTS IN TIME        # B01400
C                                                                       # B01410
      IF (ITMS.EQ.1) THEN                                               # B01420
C                                                                       # B01430
C  ITREAD = 0 -- CONSTANT TIME STEP LENGTH                              # B01440
```

```
C  ITREAD = 1 -- TIME STEP DATA READ FROM A FILE                          # B01450
C                                                                        # B01460
      IF (ITREAD.EQ.0) THEN                                              # B01470
      TMAX=TMAX+DELT                                                     # B01480
      ELSE                                                               # B01490
        IF (KSTP.GT.NSTP.OR.KOUNT.EQ.1) THEN                             # B01500
          IF (KPER.LT.NPER) THEN                                         # B01510
          READ (I8,5000) PERLEN,NSTP,TSMULT,TFAC                         # B01520
          KPER=KPER+1                                                    # B01530
          KSTP=1                                                         # B01540
          DELT= TFAC*PERLEN/FLOAT(NSTP)                                  # B01550
          IF(TSMULT.NE.1.)                                               # B01560
     1      DELT=TFAC*PERLEN*(1.-TSMULT)/(1.-TSMULT**NSTP)               # B01570
          TMAX=TMAX+DELT                                                 # B01580
          ELSE                                                           # B01590
          TMAX=1.0E+30                                                   # B01600
          ITMS=0                                                         # B01610
          END IF                                                         # B01620
        ELSE                                                             # B01630
          DELT=TSMULT*DELT                                               # B01640
          TMAX=TMAX+DELT                                                 # B01650
        END IF                                                           # B01660
      END IF                                                             # B01670
5000   FORMAT(F10.0,I10,2F10.0)                                          # B01680
      ELSE                                                               # B01690
      TMAX= 1.0E+30                                                      # B01700
      END IF                                                             # B01710
C                                                                        # B01720
C--------------------  START PARTICLE LOOP  -----------------------     # B01730
C                                                                        # B01740
      IEND=0                                                             # B01750
      DO 70 N=1,NPRT                                                     # B01760
      IF(TOT(N).GT.0.0E+0) GO TO 70                                      # B01770
C                                                                        # B01780
C  SET START TIME FOR PARTICLE                                           # B01790
C                                                                        # B01800
      TIME=TOLD                                                          # B01810
C                                                                        # B01820
      CALL FLOLIN (N,IGRID,MODE,TIME,TMAX,DELT,IDSCH,                    # B01830
     1JLC(N),ILC(N),KLC(N),XLC(N),YLC(N),ZLC(N),ZLLC(N),IBOUND,LAYCON,   # B01840
     2ZBOT,ZTOP,XMAX,YMAX,QX,QY,QZ,DELC,DELR,POR,HEAD,NCON,NCOL,NROW,    # B01850
     3NLAY,NCBL,NLPOR,NZDIM,NCP1,NRP1,NLP1,ISNK,IREV,FRAC,IZSTOP,I2,     # B01860
     4I7,QSS,1)                                                          # B01870
C                                                                        # B01880
C  ASSIGN TIME OF TRAVEL TO TOT ARRAY IF PARTICLE HAS DISCHARGED         # B01890
C                                                                        # B01900
C  IDSCH = 0 -- PARTICLE HAS DISCHARGED                                  # B01910
C  IDSCH = 1 -- PARTICLE HAS NOT DISCHARGED                              # B01920
C                                                                        # B01930
      IF(IDSCH.EQ.0) TOT(N)=TIME                                         # B01940
      IEND=IEND+IDSCH                                                    # B01950
C                                                                        # B01960
C  WRITE DISCHARGE INFO TO SCREEN IF IOP=1                               # B01970
C                                                                        # B01980
C                                                                        # B01990
      IF (IDSCH.EQ.0) THEN                                               # B02000
      IF(IOP.EQ.1.AND.IBATCH.EQ.0)                                       # B02010
     1WRITE (*,5010) N,ILC(N),JLC(N),KLC(N),TOT(N)                       # B02020
5010  FORMAT(' PARTICLE',I5,' STOPS IN (I=',I3,', J=',I3,', K=',         # B02030
     1I3,')  TRAVEL TIME = ',1PE12.4)                                    # B02040
      END IF                                                             # B02050
C                                                                        # B02060
C  DON'T WRITE RESULTS IN TIMESERIES FILE FOR TIME=TMAX IF PARTICLE      # B02070
C  DISCHARGED AT TIME < TMAX.                                            # B02080
C                                                                        # B02090
      IF(IDSCH.EQ.0.AND.TIME.LT.TMAX) GO TO 70                           # B02100
C                                                                        # B02110
C  WRITE RESULTS TO TIMESERIES FILE                                      # B02120
C                                                                        # B02130
      IF (MODE.EQ.2) THEN                                                # B02140
      WRITE(I4,5020) KOUNT,N,JLC(N),ILC(N),KLC(N),XLC(N),YLC(N),ZLC(N),  # B02150
     1ZLLC(N),TMAX                                                       # B02160
```

```
5020   FORMAT(2(I4,1X),3(I3,1X),4(E20.12,1X),E20.12)                    # B02170
      END IF                                                           # B02180
C                                                                      # B02190
70    CONTINUE                                                         # B02200
C                                                                      # B02210
C------------------  END PARTICLE LOOP  -------------------------      # B02220
C                                                                      # B02230
C  CHECK TO SEE IF ALL PARTICLES HAVE DISCHARGED. IF NOT GO THROUGH    # B02240
C  TIME LOOP AGAIN.                                                    # B02250
C                                                                      # B02260
      IF(IEND.GT.0) GO TO 60                                           # B02270
C                                                                      # B02280
C---------------------  END TIME LOOP  --------------------------      # B02290
C                                                                      # B02300
C  WRITE DISCHARGE DATA TO ENDPOINT  FILE                              # B02310
C                                                                      # B02320
      REWIND(I5)                                                       # B02330
      DO 80 N=1,NPRT                                                   # B02340
      READ(I5,*) NN,XSTRT,YSTRT,ZLSTRT,JFRST,IFRST,KFRST               # B02350
      IZONE=IBOUND(JLC(N),ILC(N),KLC(N))                               # B02360
      IF(IZONE.LT.0) IZONE=-IZONE                                      # B02370
      IF(IZONE.GE.1000) IZONE=IZONE/1000                               # B02380
      IZONE2=IBOUND(JFRST,IFRST,KFRST)                                 # B02390
      IF(IZONE2.LT.0) IZONE2=-IZONE2                                   # B02400
      IF(IZONE2.GE.1000) IZONE2=IZONE2/1000                            # B02410
      IF (IPEP.EQ.0) THEN                                              # B02420
      WRITE(I3,5030) IZONE,JLC(N),ILC(N),KLC(N),XLC(N),YLC(N),ZLC(N),  # B02430
     1ZLLC(N),TOT(N),XSTRT,YSTRT,ZLSTRT,JFRST,IFRST,KFRST,IZONE2       # B02440
      ELSE                                                             # B02450
      IF (IZONE.EQ.IZSTOP) WRITE (I3,5030) IZONE,JLC(N),ILC(N),KLC(N), # B02460
     1 XLC(N),YLC(N),ZLC(N),ZLLC(N),TOT(N),XSTRT,YSTRT,ZLSTRT,JFRST,   # B02470
     2 IFRST,KFRST,IZONE2                                              # B02480
      END IF                                                           # B02490
80    CONTINUE                                                         # B02500
5030   FORMAT(4I4,8E12.4,4I4)                                          # B02510
C                                                                      # B02520
      RETURN                                                           # B02530
      END                                                              # B02540
C                                                                      # B02550
C-----END OF ROUTINE-------------------------------------------------# B02560
```

```
C                                                                       # C00010
C---Version 1.0   July 21, 1989                                         # C00020
C***********************************************************************# C00030
C                                                                      *# C00040
C                              FLOLIN                                  *# C00050
C                                                                      *# C00060
C  THIS SUBROUTINE COMPUTES THE PATH OF A PARTICLE FROM AN INITIAL     *# C00070
C  STARTING LOCATION AND STARTING TIME TO A DISCHARGE POINT OR A       *# C00080
C  SPECIFIED POINT IN TIME.  IF THE LINE MODE IS USED (MODE=1), THE    *# C00090
C  COORDINATES OF EACH COMPUTED POINT ARE WRITTEN TO A FILE NAMED      *# C00100
C  "PATHLINE"                                                          *# C00110
C                                                                      *# C00120
C***********************************************************************# C00130
C                                                                       # C00140
      SUBROUTINE FLOLIN (IPART,IGRID,MODE,TIME,TMAX,DELT,               # C00150
     1IDSCH,JP,IP,KP,XP,YP,ZP,ZLOC,IBOUND,LAYCON,ZBOT,ZTOP,XMAX,YMAX,   # C00160
     2QX,QY,QZ,DELC,DELR,POR,HEAD,NCON,NCOL,NROW,NLAY,NCBL,NLPOR,NZDIM, # C00170
     3NCP1,NRP1,NLP1,ISNK,IREV,FRAC,IZSTOP,I2,I7,QSS,ISS)               # C00180
C                                                                       # C00190
      DIMENSION IBOUND(NCOL,NROW,NLAY),ZTOP(NZDIM),ZBOT(NZDIM),         # C00200
     1LAYCON(NLAY),XMAX(NCOL),YMAX(NROW),NCON(NLAY),QSS(NCOL,NROW,NLAY),# C00210
     2POR(NCOL,NROW,NLPOR),QX(NCP1,NROW,NLAY),QY(NCOL,NRP1,NLAY),       # C00220
     3QZ(NCOL,NROW,NLP1),DELC(NROW),DELR(NCOL),HEAD(NCOL,NROW,NLAY)     # C00230
C                                                                       # C00240
C  THE VARIABLE "ISS" IS A FLAG INDICATING STEADY STATE OR TRANSIENT    # C00250
C  CONDITIONS. IN THIS PROGRAM, ISS IS SET EQUAL TO 1 FOR STEADY STATE. # C00260
C                                                                       # C00270
      KOLD=KP                                                           # C00280
      NSEGS= NCOL*NROW*NLAY                                             # C00290
      IDSCH=1                                                           # C00300
      IBND=IBOUND(JP,IP,KP)                                             # C00310
      HED=HEAD(JP,IP,KP)                                                # C00320
C                                                                       # C00330
C  RETURN IF PARTICLE IS IN INACTIVE CELL                               # C00340
C                                                                       # C00350
      IF(IBND.EQ.0.OR.HED.GT.1.0E+29) THEN                              # C00360
      IDSCH=0                                                           # C00370
      RETURN                                                            # C00380
      END IF                                                            # C00390
C                                                                       # C00400
C  RETURN IF PARTICLE IS IN THE AUTOMATIC DISCHARGE ZONE                # C00410
C                                                                       # C00420
      IF(IBND.EQ.IZSTOP) THEN                                           # C00430
      IDSCH=0                                                           # C00440
      RETURN                                                            # C00450
      END IF                                                            # C00460
C                                                                       # C00470
C  INITIALIZE VARIABLES AND COMPUTE INITIAL VALUE OF                    # C00480
C  NORMALIZED Z COORDINATE                                              # C00490
C                                                                       # C00500
      ILAST=0                                                           # C00510
      IF(IGRID.EQ.1) THEN                                               # C00520
      NK=KP                                                             # C00530
      NKP=NK+1                                                          # C00540
      ELSE                                                              # C00550
      NK = (KP-1)*NCOL*NROW + (IP-1)*NCOL + JP                          # C00560
      NKP= NK + (NCOL*NROW)                                             # C00570
      END IF                                                            # C00580
      IF(KP.EQ.NLAY.AND.ZLOC.LT.0.0) THEN                               # C00590
      WRITE(I7,*) ' RUN STOPPED. INCONSISTENT CONFINING BED LOCATION'   # C00600
      STOP                                                              # C00610
      END IF                                                            # C00620
      ZMX=ZTOP(NK)                                                      # C00630
      HED=HEAD(JP,IP,KP)                                                # C00640
      IF(LAYCON(KP).EQ.1) ZMX=HED                                       # C00650
      IF(LAYCON(KP).GT.1.AND.HED.LT.ZMX) ZMX=HED                        # C00660
      IF(ZLOC.GE.0.0) THEN                                              # C00670
      ZP=ZLOC*ZMX + (1.0-ZLOC)*ZBOT(NK)                                 # C00680
      ELSE                                                              # C00690
      ZL= 1.0+ZLOC                                                      # C00700
      ZP=ZL*ZBOT(NK) + (1.0-ZL)*ZTOP(NKP)                               # C00710
      END IF                                                            # C00720
```

```
C                                                                       # C00730
C  WRITE STARTING COORDINATES                                           # C00740
C                                                                       # C00750
      IF(MODE.EQ.1)WRITE(I2,160) IPART,XP,YP,ZLOC,ZP,TIME,              # C00760
     1JP,IP,KP                                                          # C00770
C                                                                       # C00780
C  CHECK TO SEE IF PARTICLE IS LOCATED IN CONFINING BED.                # C00790
C  IF IT IS, MOVE PARTICLE TO NEXT ACTIVE LAYER.                        # C00800
C                                                                       # C00810
      IF(ZLOC.GT.0.0) GO TO 50                                          # C00820
      IF(ZLOC.GE.0.0.AND.QZ(JP,IP,KP+1).GE.0.0) GO TO 50                # C00830
      LCON=NCON(KP)                                                     # C00840
      KPOR=NLAY+LCON                                                    # C00850
      IF(LCON.EQ.0) GO TO 30                                            # C00860
      IF(KP.EQ.NLAY) GO TO 30                                           # C00870
      V= QZ(JP,IP,KP+1)/POR(JP,IP,KPOR)/DELC(IP)/DELR(JP)               # C00880
      IF(V.EQ.0.0) THEN                                                 # C00890
      IDSCH=0                                                           # C00900
      RETURN                                                            # C00910
      END IF                                                            # C00920
      ZCB= ZTOP(NKP)                                                    # C00930
      ZMN= ZBOT(NK)                                                     # C00940
      IF(V.GT.0.0) DT= -ZLOC*(ZMN-ZCB)/V                                # C00950
      IF(V.LT.0.0) DT= -(1.0+ZLOC)*(ZMN-ZCB)/V                          # C00960
      TIM=TIME+DT                                                       # C00970
C                                                                       # C00980
C  CHECK TO SEE IF MAXIMUM TIME HAS BEEN REACHED OR EXCEEDED.           # C00990
C  IF SO, SET TIME EQUAL TO TMAX, CALUCULATE PARTICLE LOCATION, &       # C01000
C  EXIT SUBROUTINE.                                                     # C01010
C                                                                       # C01020
      IF(TMAX.GT.TIM) GO TO 10                                          # C01030
      DTT=TMAX-TIME                                                     # C01040
      ZLC= ZLOC + V*DTT/(ZMN-ZCB)                                       # C01050
      ZP= -ZLC*ZCB + (1.0+ZLC)*ZMN                                      # C01060
      TTM=-TMAX                                                         # C01070
      IF(MODE.EQ.1) WRITE(I2,160) IPART,XP,YP,ZLC,ZP,TTM,               # C01080
     1JP,IP,KP                                                          # C01090
      TIME=TMAX                                                         # C01100
      ZLOC=ZLC                                                          # C01110
      RETURN                                                            # C01120
10    CONTINUE                                                          # C01130
C                                                                       # C01140
      TIME=TIME+DT                                                      # C01150
      ZP=ZMN                                                            # C01160
      ZLOC=0.0                                                          # C01170
      IF(V.GE.0.0) GO TO 20                                             # C01180
      KP=KP+1                                                           # C01190
      ZP=ZCB                                                            # C01200
      ZLOC=1.0                                                          # C01210
20    IF(MODE.EQ.1) WRITE(I2,160) IPART,XP,YP,ZLOC,ZP,TIME,JP,IP,KP     # C01220
      GO TO 40                                                          # C01230
30    WRITE(I7,*) ' RUN STOPPED. INCONSISTENT CONFINING BED LOCATION.'  # C01240
      STOP                                                              # C01250
40    CONTINUE                                                          # C01260
C                                                                       # C01270
C  THE PARTICLE IS NOW IN AN ACTIVE MODEL LAYER.  CHECK TO SEE IF IT IS # C01280
C  IN AN INACTIVE CELL WITHIN THE MODEL LAYER.  IF SO, RESET            # C01290
C  KP AND ZLOC TO THE VALUES AT THE EXIT POINT OF PREVIOUS ACTIVE CELL. # C01300
C                                                                       # C01310
      IBND=IBOUND(JP,IP,KP)                                             # C01320
      HED=HEAD(JP,IP,KP)                                                # C01330
      IF(IBND.EQ.0.OR.HED.GT.1.0E+29) THEN                              # C01340
      IDSCH=0                                                           # C01350
      KP=KOLD                                                           # C01360
      ZLOC= -1.0                                                        # C01370
      RETURN                                                            # C01380
      END IF                                                            # C01390
50    CONTINUE                                                          # C01400
C                                                                       # C01410
C       START TRACKING LOOP                                             # C01420
C                                                                       # C01430
      DO 150 N=1,NSEGS                                                  # C01440
```

```
C                                                                       # C01450
C  STORE CURRENT CELL LOCATION AND LOCAL Z COORDINATE                   # C01460
C                                                                       # C01470
      JOLD=JP                                                           # C01480
      IOLD=IP                                                           # C01490
      KOLD=KP                                                           # C01500
      ZLOLD=ZLOC                                                        # C01510
C                                                                       # C01520
C  ASSIGN SCALAR VARIABLES FOR CONVENIENCE                              # C01530
C                                                                       # C01540
      XMN=0.0                                                           # C01550
      IF(JP.GT.1) XMN=XMAX(JP-1)                                        # C01560
      XMX=XMAX(JP)                                                      # C01570
      YMN=0.0                                                           # C01580
      IF(IP.LT.NROW) YMN=YMAX(IP+1)                                     # C01590
      YMX=YMAX(IP)                                                      # C01600
      IF(IGRID.EQ.1) THEN                                               # C01610
      NK=KP                                                             # C01620
      ELSE                                                              # C01630
      NK= (KP-1)*NCOL*NROW + (IP-1)*NCOL + JP                           # C01640
      END IF                                                            # C01650
      ZMX=ZTOP(NK)                                                      # C01660
      HED=HEAD(JP,IP,KP)                                                # C01670
      IF(LAYCON(KP).EQ.1) ZMX=HED                                       # C01680
      IF(LAYCON(KP).GT.1.AND.HED.LT.ZMX) ZMX=HED                        # C01690
      ZMN= ZBOT(NK)                                                     # C01700
      IF(ZMX.EQ.ZMN) ZMX=ZMX+0.01                                       # C01710
      ZP= ZLOC*ZMX + (1.0-ZLOC)*ZMN                                     # C01720
      DZ= ZMX-ZMN                                                       # C01730
C                                                                       # C01740
C  ASSIGN FACE VELOCITIES                                               # C01750
C                                                                       # C01760
      PHI=POR(JP,IP,KP)                                                 # C01770
      VX1= QX(JP,IP,KP)/PHI/DELC(IP)/DZ                                 # C01780
      VX2= QX(JP+1,IP,KP)/PHI/DELC(IP)/DZ                               # C01790
      VY1= QY(JP,IP+1,KP)/PHI/DELR(JP)/DZ                               # C01800
      VY2= QY(JP,IP,KP)/PHI/DELR(JP)/DZ                                 # C01810
      VZ1= QZ(JP,IP,KP+1)/PHI/DELC(IP)/DELR(JP)                         # C01820
      VZ2= QZ(JP,IP,KP)/PHI/DELC(IP)/DELR(JP)                           # C01830
C                                                                       # C01840
C  COMPUTE CELL TRANSIT TIMES IN X, Y, AND Z DIRECTIONS                 # C01850
C                                                                       # C01860
      CALL DTCALC (VX1,VX2,VX,DVXDX,XMN,XMX,XP,DTX,IVXFLG)              # C01870
      CALL DTCALC (VY1,VY2,VY,DVYDY,YMN,YMX,YP,DTY,IVYFLG)              # C01880
      CALL DTCALC (VZ1,VZ2,VZ,DVZDZ,ZMN,ZMX,ZP,DTZ,IVZFLG)              # C01890
C                                                                       # C01900
C  DETERMINE EXIT FACE AND SET FLAGS                                    # C01910
C                                                                       # C01920
      IX=0                                                              # C01930
      IY=0                                                              # C01940
      IZ=0                                                              # C01950
      DT=DTX                                                            # C01960
      IX=1                                                              # C01970
      IF(DTY.GE.DT) GO TO 60                                            # C01980
      DT=DTY                                                            # C01990
      IX=0                                                              # C02000
      IY=1                                                              # C02010
      IZ=0                                                              # C02020
60    IF(DTZ.GE.DT) GO TO 70                                            # C02030
      DT=DTZ                                                            # C02040
      IY=0                                                              # C02050
      IX=0                                                              # C02060
      IZ=1                                                              # C02070
70    CONTINUE                                                          # C02080
      IF(VX.LT.0.0.AND.IX.GT.0) IX=-IX                                  # C02090
      IF(VY.LT.0.0.AND.IY.GT.0) IY=-IY                                  # C02100
      IF(VZ.LT.0.0.AND.IZ.GT.0) IZ=-IZ                                  # C02110
C                                                                       # C02120
C  CHECK TO SEE IF THIS CELL IS A DISCHARGE POINT                       # C02130
C                                                                       # C02140
C    CHECK TO SEE IF THERE IS NO OUTFLOW FROM THIS CELL.                # C02150
C    IF SIMULATION IS STEADY STATE, DISCHARGE PARTICLE AND RETURN.      # C02160
```

```
C                                                                       # C02170
      IXYZ=0                                                            # C02180
      IF(IVXFLG.GT.1) IXYZ=IXYZ+1                                       # C02190
      IF(IVYFLG.GT.1) IXYZ=IXYZ+1                                       # C02200
      IF(IVZFLG.GT.1) IXYZ=IXYZ+1                                       # C02210
      IF(IXYZ.EQ.3.AND.ISS.EQ.1) THEN                                   # C02220
      IDSCH=0                                                           # C02230
      RETURN                                                            # C02240
      END IF                                                            # C02250
C                                                                       # C02260
C  RETURN IF PARTICLE IS IN THE AUTOMATIC DISCHARGE ZONE                # C02270
C                                                                       # C02280
      IF(IZSTOP.GT.0) THEN                                              # C02290
      IBABS=IBOUND(JP,IP,KP)                                            # C02300
      IF(IBOUND(JP,IP,KP).LT.0) IBABS= -IBOUND(JP,IP,KP)                # C02310
      IF(IBABS.EQ.IZSTOP) THEN                                          # C02320
      IDSCH=0                                                           # C02330
      RETURN                                                            # C02340
      END IF                                                            # C02350
      END IF                                                            # C02360
C                                                                       # C02370
      IF(IBOUND(JP,IP,KP).GT.-1000.AND.IBOUND(JP,IP,KP).LT.1000)        # C02380
     1  GO TO 80                                                        # C02390
      IBD=0                                                             # C02400
      IF(IBOUND(JP,IP,KP).LE.-1000.OR.IBOUND(JP,IP,KP).GE.1000) IBD=1   # C02410
      IF(IBD.EQ.1.AND.ISNK.GE.1) THEN                                   # C02420
      IF(TIME.LE.0.0.AND.IREV.EQ.1) GO TO 80                            # C02430
      IF (ISNK.EQ.1) THEN                                               # C02440
      IDSCH=0                                                           # C02450
      RETURN                                                            # C02460
      END IF                                                            # C02470
      END IF                                                            # C02480
C                                                                       # C02490
C  DO A SECOND CHECK TO SEE IF PARTICLE CANNOT GET OUT OF CELL.         # C02500
C  IF IT CANNOT, DISCHARGE IT.                                          # C02510
C                                                                       # C02520
      IF(TIME.LE.0.0.AND.IREV.EQ.1) GO TO 80                            # C02530
      DXDY= DELR(JP)*DELC(IP)                                           # C02540
      DXDZ= DELR(JP)*(ZMX-ZMN)                                          # C02550
      DYDZ= DELC(IP)*(ZMX-ZMN)                                          # C02560
      VSIGN=1.0                                                         # C02570
      IF(IREV.EQ.1) VSIGN= -1.0                                         # C02580
      VLX1=VSIGN*VX1                                                    # C02590
      VLX2=VSIGN*VX2                                                    # C02600
      VLY1=VSIGN*VY1                                                    # C02610
      VLY2=VSIGN*VY2                                                    # C02620
      VLZ1=VSIGN*VZ1                                                    # C02630
      VLZ2=VSIGN*VZ2                                                    # C02640
      QINOUT= -QSS(JP,IP,KP)                                            # C02650
      IF(QINOUT.LE.0.0) GO TO 80                                        # C02660
      QI=0.0                                                            # C02670
      IF(VLX1.GT.0.0) QI=QI+VLX1*DYDZ                                   # C02680
      IF(VLX2.LT.0.0) QI=QI-VLX2*DYDZ                                   # C02690
      IF(VLY1.GT.0.0) QI=QI+VLY1*DXDZ                                   # C02700
      IF(VLY2.LT.0.0) QI=QI-VLY2*DXDZ                                   # C02710
      IF(VLZ1.GT.0.0) QI=QI+VLZ1*DXDY                                   # C02720
      IF(VLZ2.LT.0.0) QI=QI-VLZ2*DXDY                                   # C02730
      IF(QI.LE.0.0) GO TO 80                                            # C02740
      F=QINOUT/QI                                                       # C02750
      IF(F.GT.FRAC) THEN                                                # C02760
      IDSCH=0                                                           # C02770
      RETURN                                                            # C02780
      END IF                                                            # C02790
C                                                                       # C02800
80    CONTINUE                                                          # C02810
C                                                                       # C02820
C  CHECK TO SEE IF TIME IS GREATER THAN OR EQUAL TO TMAX.               # C02830
C  IF SO, SET TIME EQUAL TO TMAX, CALCULATE LOCATION & RETURN           # C02840
C                                                                       # C02850
      TIM=TIME+DT                                                       # C02860
      IF(TMAX.GT.TIM) GO TO 90                                          # C02870
      DTT= TMAX-TIME                                                    # C02880
```

```
      CALL NEWXYZ (VX,DVXDX,VX1,VX2,DTT,XP,XMN,XMX,XPP,JP,IVXFLG,0)     # C02890
      CALL NEWXYZ (VY,DVYDY,VY1,VY2,DTT,YP,YMN,YMX,YPP,IP,IVYFLG,0)     # C02900
      CALL NEWXYZ (VZ,DVZDZ,VZ1,VZ2,DTT,ZP,ZMN,ZMX,ZPP,KP,IVZFLG,0)     # C02910
C                                                                       # C02920
      ZLOC= (ZPP-ZMN)/(ZMX-ZMN)                                         # C02930
C                                                                       # C02940
C  WRITE COORINATES OF INTERMEDIATE POINTS                              # C02950
C                                                                       # C02960
      TTM= -TMAX                                                        # C02970
      IF(MODE.EQ.1) WRITE(I2,160) IPART,XPP,YPP,ZLOC,ZPP,TTM,           # C02980
     1JP,IP,KP                                                          # C02990
      XP=XPP                                                            # C03000
      YP=YPP                                                            # C03010
      ZP=ZPP                                                            # C03020
      TIME=TMAX                                                         # C03030
      RETURN                                                            # C03040
90    CONTINUE                                                          # C03050
C                                                                       # C03060
C  COMPUTE PARTICLE COORDINATES AT THE EXIT POINT OF CELL (JP,IP,KP)    # C03070
C                                                                       # C03080
      CALL NEWXYZ (VX,DVXDX,VX1,VX2,DT,XP,XMN,XMX,XNEW,JP,IVXFLG,IX)    # C03090
      CALL NEWXYZ (VY,DVYDY,VY1,VY2,DT,YP,YMN,YMX,YNEW,IP,IVYFLG,IY)    # C03100
      CALL NEWXYZ (VZ,DVZDZ,VZ1,VZ2,DT,ZP,ZMN,ZMX,ZNEW,KP,IVZFLG,IZ)    # C03110
      XP=XNEW                                                           # C03120
      YP=YNEW                                                           # C03130
      ZP=ZNEW                                                           # C03140
      TIME=TIME+DT                                                      # C03150
C                                                                       # C03160
      ZLOC= (ZP-ZMN)/(ZMX-ZMN)                                          # C03170
      IF(ZLOC.GT.1.0) ZLOC=1.0                                          # C03180
      IF(ZLOC.LT.0.0) ZLOC=0.0                                          # C03190
C                                                                       # C03200
      IF(IX.LT.0) JP=JP-1                                               # C03210
      IF(IX.GT.0) JP=JP+1                                               # C03220
      IF(IY.LT.0) IP=IP+1                                               # C03230
      IF(IY.GT.0) IP=IP-1                                               # C03240
      ICB=0                                                             # C03250
      IF(IZ.LT.0.AND.NCON(KP).NE.0) ICB=1                               # C03260
      IF(IZ.LT.0.AND.NCON(KP).EQ.0) KP=KP+1                             # C03270
      IF(IZ.GT.0) KP=KP-1                                               # C03280
C                                                                       # C03290
C  DISCHARGE PARTICLE IF IT REACHES BOUNDARY OF THE ACTIVE GRID         # C03300
C                                                                       # C03310
      IEXIT=0                                                           # C03320
      IF(JP.LT.1.OR.JP.GT.NCOL) IEXIT=1                                 # C03330
      IF(IP.LT.1.OR.IP.GT.NROW) IEXIT=1                                 # C03340
      IF(KP.LT.1.OR.KP.GT.NLAY) IEXIT=1                                 # C03350
      IF(IEXIT.EQ.1) GO TO 100                                          # C03360
      IBND=IBOUND(JP,IP,KP)                                             # C03370
      HED=HEAD(JP,IP,KP)                                                # C03380
      IF(IBND.EQ.0.OR.HED.GT.1.0E+29) IEXIT=1                           # C03390
100   CONTINUE                                                          # C03400
      IF(IEXIT.EQ.1) THEN                                               # C03410
      IDSCH=0                                                           # C03420
      JP=JOLD                                                           # C03430
      IP=IOLD                                                           # C03440
      KP=KOLD                                                           # C03450
      IF(MODE.EQ.1) WRITE(I2,160) IPART,XP,YP,ZLOC,ZP,TIME,JP,IP,KP     # C03460
      RETURN                                                            # C03470
      END IF                                                            # C03480
C                                                                       # C03490
C  SWITCH ZLOC FROM 0 TO 1 OR 1 TO 0 IF PARTICLE CHANGES LAYERS         # C03500
C                                                                       # C03510
      IF(IZ.LT.0.AND.NCON(KP).EQ.0) ZLOC=1.0                            # C03520
      IF(IZ.GT.0.AND.NCON(KP).EQ.0) ZLOC=0.0                            # C03530
      IF(IZ.GT.0.AND.NCON(KP).GT.0) THEN                                # C03540
      ZLOC= -1.0                                                        # C03550
      ICB=1                                                             # C03560
      END IF                                                            # C03570
C                                                                       # C03580
C  WRITE COORDINATES OF EXIT POINT                                      # C03590
C                                                                       # C03600
```

```
      IF(MODE.EQ.1) WRITE(I2,160) IPART,XP,YP,ZLOC,ZP,TIME,JP,IP,KP     # C03610
C                                                                        # C03620
C  CHECK TO SEE IF PARTICLE HAS ENTERED CONFINING BED.                   # C03630
C  IF SO, MOVE PARTICLE THROUGH CONFINING BED TO AN ACTIVE MODEL LAYER.  # C03640
C  OTHERWISE, GO TO END OF CELL LOOP.                                    # C03650
C                                                                        # C03660
      IF(ICB.EQ.0) GO TO 150                                             # C03670
      KOLD=KP                                                            # C03680
      LCON=NCON(KP)                                                      # C03690
      KPOR=NLAY+LCON                                                     # C03700
      IF(LCON.EQ.0) GO TO 130                                            # C03710
      IF(KP.EQ.NLAY) GO TO 130                                           # C03720
      V= QZ(JP,IP,KP+1)/POR(JP,IP,KPOR)/DELC(IP)/DELR(JP)                # C03730
      IF(V.EQ.0.0) THEN                                                  # C03740
      IDSCH=0                                                            # C03750
      RETURN                                                             # C03760
      END IF                                                             # C03770
      IF(IGRID.EQ.1) THEN                                                # C03780
      NK= KP                                                             # C03790
      NKP=NK+1                                                           # C03800
      ELSE                                                               # C03810
      NK= (KP-1)*NCOL*NROW + (IP-1)*NCOL + JP                            # C03820
      NKP=NK + (NCOL*NROW)                                               # C03830
      END IF                                                             # C03840
      ZCB= ZTOP(NKP)                                                     # C03850
      ZMN= ZBOT(NK)                                                      # C03860
      IF(V.GT.0.0) DT= -ZLOC*(ZMN-ZCB)/V                                 # C03870
      IF(V.LT.0.0) DT= -(1.0+ZLOC)*(ZMN-ZCB)/V                           # C03880
C                                                                        # C03890
C  CHECK TO SEE IF TIME IS GREATER THAN OR EQUAL TO TMAX.                # C03900
C  IF SO, SET TIME EQUAL TO TMAX, CALCULATE LOCATION, &                  # C03910
C  EXIT SUBROUTINE.                                                      # C03920
C                                                                        # C03930
      TIM=TIME+DT                                                        # C03940
      IF(TMAX.GT.TIM) GO TO 110                                          # C03950
      DTT=TMAX-TIME                                                      # C03960
      ZLC= ZLOC + V*DTT/(ZMN-ZCB)                                        # C03970
      ZP= -ZLC*ZCB + (1.0+ZLC)*ZMN                                       # C03980
      TTM=-TMAX                                                          # C03990
      IF(MODE.EQ.1) WRITE(I2,160) IPART,XP,YP,ZLC,ZP,TTM,JP,IP,KP        # C04000
      TIME=TMAX                                                          # C04010
      ZLOC=ZLC                                                           # C04020
      RETURN                                                             # C04030
110   CONTINUE                                                           # C04040
C                                                                        # C04050
      TIME=TIME+DT                                                       # C04060
      ZP=ZMN                                                             # C04070
      ZLOC=0.0                                                           # C04080
      IF(V.GE.0.0) GO TO 120                                             # C04090
      KP=KP+1                                                            # C04100
      ZP=ZCB                                                             # C04110
      ZLOC=1.0                                                           # C04120
120   IF(MODE.EQ.1) WRITE(I2,160) IPART,XP,YP,ZLOC,ZP,TIME,JP,IP,KP      # C04130
      GO TO 140                                                          # C04140
130   WRITE(I7,*) ' RUN STOPPED. INCONSISTENT CONFINING BED LOCATION.'   # C04150
      STOP                                                               # C04160
140   CONTINUE                                                           # C04170
C                                                                        # C04180
C  THE PARTICLE IS NOW IN AN ACTIVE MODEL LAYER.  CHECK TO SEE IF IT IS  # C04190
C  IN AN INACTIVE CELL WITHIN THE MODEL LAYER.  IF SO, RESET             # C04200
C  KP AND ZLOC TO THE VALUES AT THE EXIT POINT OF PREVIOUS ACTIVE CELL.  # C04210
C                                                                        # C04220
      IBND=IBOUND(JP,IP,KP)                                              # C04230
      HED=HEAD(JP,IP,KP)                                                 # C04240
      IF(IBND.EQ.0.OR.HED.GT.1.0E+29) THEN                               # C04250
      IDSCH=0                                                            # C04260
      KP=KOLD                                                            # C04270
      ZLOC= -1.0                                                         # C04280
      RETURN                                                             # C04290
      END IF                                                             # C04300
150   CONTINUE                                                           # C04310
C                                                                        # C04320
```

```
      RETURN                                                            # C04330
160   FORMAT(I5,1X,5(E20.12,1X),2(I3,1X),I3)                            # C04340
      END                                                               # C04350
C-----END OF ROUTINE-----------------------------------------------------# C04360
```

```
C                                                                       # D00010
C---Version 1.0   July 21, 1989                                         # D00020
C***********************************************************************# D00030
C                                                                       # D00040
C                            NEWXYZ                                     # D00050
C                                                                       # D00060
C  THIS SUBROUTINE COMPUTES A NEW X, Y, OR Z PARTICLE COORDINATE GIVEN  # D00070
C  FACE VELOCITIES AND A TIME INTERVAL.                                 # D00080
C                                                                       # D00090
C***********************************************************************# D00100
C                                                                       # D00110
      SUBROUTINE NEWXYZ (V,DVDX,V1,V2,DT,XP,XMIN,XMAX,XNEW,JP,          # D00120
     1  IVFLG,ISIDE)                                                    # D00130
C                                                                       # D00140
      IF(IVFLG.EQ.1) GO TO 10                                           # D00150
      IF(IVFLG.EQ.2) GO TO 20                                           # D00160
      XNEW= XP + V*(EXP(DVDX*DT) - 1.0E+0)/DVDX                         # D00170
      GO TO 30                                                          # D00180
10    XNEW= XP + V1*DT                                                  # D00190
      GO TO 30                                                          # D00200
20    XNEW= XP                                                          # D00210
      GO TO 40                                                          # D00220
30    IF(ISIDE.GT.0) XNEW=XMAX                                          # D00230
      IF(ISIDE.LT.0) XNEW=XMIN                                          # D00240
40    CONTINUE                                                          # D00250
      IF(XNEW.LE.XMIN.AND.V1.EQ.0.0E+0) XNEW=XMIN + 1.0E-3*(XMAX-XMIN)  # D00260
      IF(XNEW.GE.XMAX.AND.V2.EQ.0.0E+0) XNEW=XMAX - 1.0E-3*(XMAX-XMIN)  # D00270
      RETURN                                                            # D00280
      END                                                               # D00290
C-----END OF ROUTINE-------------------------------------------------- # D00300
```

```
C                                                                       # E00010
C---Version 1.0   July 21, 1989                                         # E00020
C***********************************************************************# E00030
C                                                                       # E00040
C                              DTCALC                                   # E00050
C                                                                       # E00060
C  THIS SUBROUTINE CALCULATES THE TRANSIT TIME (DT) FROM THE CURRENT    # E00070
C  LOCATION TO A POTENTIAL DISCHARGE FACE IN A GIVEN DIRECTION.         # E00080
C  IT ALSO DETERMINES IF THERE IS NO POTENTIAL FOR DISCHARGE ACROSS     # E00090
C  EITHER OF THE FACES PERPEDICULAR TO THE GIVEN COORDINATE DIRECTION.  # E00100
C  IT SETS AND RETURNS A FLAG "IVFLG" TO INDICATE WHAT CONDITION EXISTS # E00110
C  FOR THE GIVEN COORDINATE DIRECTION.                                  # E00120
C                                                                       # E00130
C     IVFLG = 0 -- "NORMAL" CONDITIONS. OUTFLOW CAN OCCUR ACROSS ONE    # E00140
C                  OR BOTH OF THE FACES PERPENDICULAR TO THIS           # E00150
C                  DIRECTION.                                           # E00160
C     IVFLG = 1 -- A CONSTANT, NON-ZERO VELOCITY COMPONENT EXISTS IN    # E00170
C                  THIS DIRCETION THROUGHOUT THE CELL. DT IS COMPUTED   # E00180
C                  USING THE CONSTANT VELOCITY INSTEAD OF THE LOG       # E00190
C                  EXPRESSION.                                          # E00200
C     IVFLG = 2 -- "MACHINE" ZERO (V<10**-15) VELOCITY COMPONENT EXISTS # E00210
C                  THROUGHOUT THE CELL IN THIS DIRECTION.               # E00220
C                  DT IS SET TO "MACHINE" INFINITY (10**20).            # E00230
C     IVFLG = 3 -- THERE IS NO POTENTIAL FOR OUTFLOW FROM EITHER FACE   # E00240
C                  PERPENDICULAR TO THIS DIRECTION.                     # E00250
C                  DT IS SET TO "MACHINE" INFINITY (10**20).            # E00260
C                                                                       # E00270
C***********************************************************************# E00280
C                                                                       # E00290
      SUBROUTINE DTCALC (V1,V2,V,DVDX,X1,X2,XP,DT,IVFLG)                # E00300
C                                                                       # E00310
      IVFLG=0                                                           # E00320
      DT= 1.0E+20                                                       # E00330
      V2A= ABS(V2)                                                      # E00340
      V1A= ABS(V1)                                                      # E00350
      DV= V2-V1                                                         # E00360
      DVA= ABS(DV)                                                      # E00370
      IF(V2A.LE.1.0E-15.AND.V1A.LE.1.0E-15) GO TO 30                    # E00380
      VV=V1A                                                            # E00390
      IF(V2A.GT.VV) VV=V2A                                              # E00400
      VVV= DVA/VV                                                       # E00410
      IF(VVV.LE.1.0E-5) GO TO 20                                        # E00420
      DX= X2-X1                                                         # E00430
      DVDX= DV/DX                                                       # E00440
      XF= (XP-X1)/DX                                                    # E00450
      V= (1.0E+0-XF)*V1 + XF*V2                                         # E00460
      IF(V1.GE.0.0E+0.AND.V2.LE.0.0E+0) GO TO 40                        # E00470
      IF (V1.LE.0.0.AND.V2.GE.0.0) THEN                                 # E00480
      IF (ABS(V).LE.0.0) THEN                                           # E00490
      V= 1.0E-20                                                        # E00500
      IF (V2.LE.0.0) V= -V                                              # E00510
      END IF                                                            # E00520
      END IF                                                            # E00530
      VR1= V1/V                                                         # E00540
      VR2= V2/V                                                         # E00550
      VR=VR1                                                            # E00560
      IF(VR.LE.0.0E+0) VR=VR2                                           # E00570
      V1V2= V1*V2                                                       # E00580
      IF(V1V2.LE.0.0) GO TO 10                                          # E00590
      IF(V.GT.0.0) VR=VR2                                               # E00600
      IF(V.LT.0.0) VR=VR1                                               # E00610
10    CONTINUE                                                          # E00620
      DT= ALOG(VR)/DVDX                                                 # E00630
      GO TO 50                                                          # E00640
20    CONTINUE                                                          # E00650
      IVFLG=1                                                           # E00660
      ZRO= 1.0E-15                                                      # E00670
      ZROM= -ZRO                                                        # E00680
      IF(V1.GT.ZRO) DT= (X2-XP)/V1                                      # E00690
      IF(V1.LT.ZROM) DT= (X1-XP)/V1                                     # E00700
      GO TO 50                                                          # E00710
30    IVFLG=2                                                           # E00720
```

```
      GO TO 50                                                          # E00730
40    IVFLG=3                                                           # E00740
50    CONTINUE                                                          # E00750
      RETURN                                                            # E00760
      END                                                               # E00770
C-----END OF ROUTINE-----------------------------------------------------# E00780
```

```
C                                                                       # F00010
C---Version 1.0   July 21, 1989                                         # F00020
C***********************************************************************# F00030
C     .                                                                 # F00040
C                                DATIN                                  # F00050
C                                                                       # F00060
C  THIS SUBROUTINE CONTROLS DATA INPUT TO MODPATH.                      # F00070
C                                                                       # F00080
C***********************************************************************# F00090
C                                                                       # F00100
      SUBROUTINE DATIN (LAYCON,NCON,HEAD,DELT,XMAX,YMAX,DELR,DELC,DELZ, # F00110
     1DELZCB,ZTOP,ZBOT,IBOUND,POR,QX,QY,QZ,IUNIT,NCOL,NROW,NLAY,NLPOR,  # F00120
     2NCP1,NRP1,NLP1,NUNIT,NZDIM,IRP,NPART,NPRT,MODE,FRAC,BUFF,IREV,    # F00130
     3IBUFF,IGRID,JLC,ILC,KLC,XLC,YLC,ZLC,ZLLC,NCBL,ITMS,QSS,ISNK,      # F00140
     4ITREAD,IA,I1,I2,I3,I4,I5,I6,I7,I8,I9,IZSTOP,IPEP,                 # F00150
     5IBATCH)                                                           # F00160
C                                                                       # F00170
      CHARACTER*80 FLHEAD,FLFLO1,FLFLO2,FLPART,MES,SUFFIX,FLOUT1,       # F00180
     1FLOUT2,FLOUT3,FLOUT4,FIL,OUT1,OUT2,OUT3                           # F00190
      CHARACTER*16 TEXT                                                 # F00200
C                                                                       # F00210
      DIMENSION LAYCON(NLAY),NCON(NLAY),DELR(NCOL),DELC(NROW),DELZ(NLAY)# F00220
     1,DELZCB(NLAY),XMAX(NCOL),YMAX(NROW),ZTOP(NZDIM),ZBOT(NZDIM),      # F00230
     2HEAD(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),POR(NCOL,NROW,NLPOR), # F00240
     3QX(NCP1,NROW,NLAY),QY(NCOL,NRP1,NLAY),QZ(NCOL,NROW,NLP1),         # F00250
     4IUNIT(NUNIT),BUFF(NCOL,NROW,NLAY),IBUFF(NCOL,NROW,NLAY),          # F00260
     5JLC(NPART),ILC(NPART),KLC(NPART),XLC(NPART),YLC(NPART),           # F00270
     6ZLC(NPART),ZLLC(NPART),QSS(NCOL,NROW,NLAY)                        # F00280
C                                                                       # F00290
      IU=I1                                                             # F00300
      NPRT=NPART                                                        # F00310
      OUT1= 'PATHLINE'                                                  # F00320
      OUT2= 'ENDPOINT'                                                  # F00330
      OUT3= 'TIMESERS'                                                  # F00340
C                                                                       # F00350
C  WRITE NCOL, NROW, NLAY, NCBL, AND IGRID TO SUMMARY.PTH               # F00360
C                                                                       # F00370
      WRITE(I7,5000)                                                    # F00380
5000  FORMAT('-------------------------------------------------------------# F00390
     1----------------------')                                          # F00400
      WRITE(I7,5010) NCOL,NROW,NLAY,NCBL                                # F00410
5010  FORMAT(I5,' COLUMNS',I5,' ROWS',I5,' LAYERS',I5,' CONFINING LAYERS# F00420
     1')                                                                # F00430
      WRITE(I7,5020) IGRID                                              # F00440
5020  FORMAT('IGRID (GRID TYPE CODE) IS',I2)                            # F00450
C                                                                       # F00460
C  READ UNIT NUMBERS FOR INPUT FILES                                    # F00470
C                                                                       # F00480
      READ(IU,5030) (IUNIT(N),N=1,NUNIT)                                # F00490
5030  FORMAT(16I5)                                                      # F00500
      WRITE(I7,5000)                                                    # F00510
      WRITE(I7,5040) (IUNIT(N),N=1,8)                                   # F00520
5040  FORMAT('IUNIT ARRAY: ',8I4)                                       # F00530
C                                                                       # F00540
C  ENTER MODE DATA                                                      # F00550
C                                                                       # F00560
      IF (IBATCH.EQ.0) THEN                                             # F00570
      WRITE (*,*) 'SELECT THE MODE FOR STORING OUTPUT DATA:'            # F00580
      WRITE (*,*) '    0 = ENDPOINTS AND STARTING POINTS ONLY'          # F00590
      WRITE (*,*) '    1 = FLOW LINE COORDINATES'                       # F00600
      WRITE (*,*) '    2 = TIME SERIES DATA FOR SCATTER PLOTS'          # F00610
      END IF                                                            # F00620
      IF(IBATCH.EQ.0) THEN                                              # F00630
      READ (*,*) MODE                                                   # F00640
      WRITE(I9,*) MODE                                                  # F00650
      ELSE                                                              # F00660
      READ(I9,*) MODE                                                   # F00670
      END IF                                                            # F00680
C                                                                       # F00690
C  ENTER TIME STEP INFORMATION FOR MODES 1 AND 2, AND DUMMY VALUES      # F00700
C  FOR MODE 0                                                           # F00710
C                                                                       # F00720
```

```
      IF (MODE.EQ.0) THEN                                                  # F00730
      ITMS=0                                                               # F00740
      END IF                                                               # F00750
      IF(MODE.EQ.1.OR.MODE.EQ.2) THEN                                      # F00760
      IF (MODE.EQ.1) THEN                                                  # F00770
      MES= 'DO YOU WANT TO COMPUTE LOCATIONS AT INTERMEDIATE TIMES ?'      # F00780
      CALL YESNO (IBATCH,I9,MES,IA,ITMS)                                   # F00790
      ELSE IF (MODE.EQ.2) THEN                                             # F00800
      ITMS=1                                                               # F00810
      END IF                                                               # F00820
      IF(ITMS.EQ.1) THEN                                                   # F00830
      IF (IBATCH.EQ.0) THEN                                                # F00840
      WRITE (*,*) 'HOW SHOULD POINTS IN TIME BE SPECIFIED ?'               # F00850
      WRITE (*,*) '    0 = WITH A CONSTANT TIME STEP'                      # F00860
      WRITE (*,*) '    1 = VALUES OF TIME READ FROM A FILE'                # F00870
      END IF                                                               # F00880
      IF(IBATCH.EQ.0) THEN                                                 # F00890
      READ (*,*) ITREAD                                                    # F00900
      WRITE(I9,*) ITREAD                                                   # F00910
      ELSE                                                                 # F00920
      READ(I9,*) ITREAD                                                    # F00930
      END IF                                                               # F00940
      IF(ITREAD.LT.0.OR.ITREAD.GT.1) THEN                                  # F00950
10    CONTINUE                                                             # F00960
      IF(IBATCH.EQ.0) WRITE (*,*) 'ENTER 0 OR 1:'                          # F00970
      IF(IBATCH.EQ.0) THEN                                                 # F00980
      READ (*,*) ITREAD                                                    # F00990
      WRITE(I9,*) ITREAD                                                   # F01000
      ELSE                                                                 # F01010
      READ(I9,*) ITREAD                                                    # F01020
      END IF                                                               # F01030
      IF(ITREAD.LT.0.OR.ITREAD.GT.1) GO TO 10                              # F01040
      END IF                                                               # F01050
      IF (ITREAD.EQ.1) THEN                                                # F01060
      IF (IBATCH.EQ.0)                                                     # F01070
     1WRITE (*,*) 'ENTER THE NAME OF THE DATA FILE:'                       # F01080
      IF(IBATCH.EQ.0) THEN                                                 # F01090
      READ (*,280) FIL                                                     # F01100
      WRITE(I9,280) FIL                                                    # F01110
      ELSE                                                                 # F01120
      READ(I9,280) FIL                                                     # F01130
      END IF                                                               # F01140
      CALL OPNFIL (I8,FIL,1,I7,IBATCH,1)                                   # F01150
5050  FORMAT (8F10.0)                                                      # F01160
      ELSE                                                                 # F01170
      IF (IBATCH.EQ.0) WRITE (*,*)                                         # F01180
     1'ENTER: TIME STEP LENGTH, FACTOR FOR CONVERTING UNITS'               # F01190
      IF(IBATCH.EQ.0) THEN                                                 # F01200
      READ (*,*) DELT,TMULT                                                # F01210
      WRITE(I9,*) DELT,TMULT                                               # F01220
      ELSE                                                                 # F01230
      READ(I9,*) DELT,TMULT                                                # F01240
      END IF                                                               # F01250
      DELT=DELT*TMULT                                                      # F01260
      END IF                                                               # F01270
      END IF                                                               # F01280
      END IF                                                               # F01290
      IF (IBATCH.EQ.0) THEN                                                # F01300
      WRITE (*,*) 'HOW ARE STARTING LOCATIONS TO BE ENTERED?'              # F01310
      WRITE (*,*) '    1 = FROM AN EXISTING DATA FILE'                     # F01320
      WRITE (*,*) '    2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNA# F01330
     1LLY'                                                                 # F01340
      END IF                                                               # F01350
      IF(IBATCH.EQ.0) THEN                                                 # F01360
      READ (*,*) IRP                                                       # F01370
      WRITE(I9,*) IRP                                                      # F01380
      ELSE                                                                 # F01390
      READ(I9,*) IRP                                                       # F01400
      END IF                                                               # F01410
      IF(IRP.LT.1.OR.IRP.GT.2) IRP=2                                       # F01420
C                                                                          # F01430
C  OPEN FILE FOR PARTICLE STARTING LOCATIONS                               # F01440
```

```
C                                                                       # F01450
      IF(IRP.EQ.1) THEN                                                 # F01460
      IF (IBATCH.EQ.0) WRITE (*,*)                                      # F01470
     1'ENTER NAME OF DATA FILE CONTAINING STARTING LOCATIONS:'           # F01480
      IF(IBATCH.EQ.0) THEN                                              # F01490
      READ (*,280) FLPART                                               # F01500
      WRITE(I9,280) FLPART                                              # F01510
      ELSE                                                              # F01520
      READ(I9,280) FLPART                                               # F01530
      END IF                                                            # F01540
      CALL OPNFIL (I6,FLPART,1,I7,IBATCH,1)                             # F01550
      ELSE IF (IRP.EQ.2) THEN                                           # F01560
      MES= 'DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS# F01570
     1 ON DISK ?'                                                       # F01580
      CALL YESNO(IBATCH,I9,MES,IA,ILSTOR)                               # F01590
      IF(ILSTOR.EQ.1) THEN                                              # F01600
      IF (IBATCH.EQ.0) WRITE (*,*) 'ENTER A FILE NAME:'                 # F01610
      IF(IBATCH.EQ.0) THEN                                              # F01620
      READ (*,280) FLPART                                               # F01630
      WRITE(I9,280) FLPART                                              # F01640
      ELSE                                                              # F01650
      READ(I9,280) FLPART                                               # F01660
      END IF                                                            # F01670
      CALL OPNFIL (I6,FLPART,2,I7,IBATCH,3)                             # F01680
      END IF                                                            # F01690
      END IF                                                            # F01700
C                                                                       # F01710
C  CREATE AND OPEN 'PATHLINE' AND (OR) 'TIMESERS' FILES                 # F01720
C                                                                       # F01730
      IF(MODE.EQ.1) CALL OPNFIL (I2,OUT1,4,I7,IBATCH,3)                 # F01740
      IF(MODE.EQ.2) CALL OPNFIL (I4,OUT3,4,I7,IBATCH,3)                 # F01750
C                                                                       # F01760
C  CHOOSE BETWEEN FORWARD AND REVERSE TRACKING                          # F01770
C                                                                       # F01780
      IF (IBATCH.EQ.0) THEN                                             # F01790
      WRITE (*,*) 'IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?'     # F01800
      WRITE (*,*) '    0 = FORWARD IN THE DIRECTION OF FLOW'            # F01810
      WRITE (*,*) '    1 = BACKWARDS TOWARD RECHARGE LOCATIONS'         # F01820
      END IF                                                            # F01830
      IF(IBATCH.EQ.0) THEN                                              # F01840
      READ (*,*) IREV                                                   # F01850
      WRITE(I9,*) IREV                                                  # F01860
      ELSE                                                              # F01870
      READ(I9,*) IREV                                                   # F01880
      END IF                                                            # F01890
C                                                                       # F01900
C  SELECT DISCHARGE CRITERION                                           # F01910
C                                                                       # F01920
      IF (IBATCH.EQ.0) THEN                                             # F01930
      WRITE (*,*) 'HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS# F01940
     1 WITH INTERNAL SINKS ?'                                           # F01950
      WRITE (*,*) '    0 = PARTICLES PASS THROUGH CELLS WITH WEAK SINKS'# F01960
      WRITE (*,*) '    1 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS W# F01970
     1ITH INTERNAL SINKS'                                               # F01980
      WRITE (*,*) '    2 = PARTICLES ARE STOPPED WHEN THEY ENTER CELLS W# F01990
     1HERE DISCHARGE TO SINKS'                                          # F02000
      WRITE (*,*) '        IS LARGER THAN A SPECIFIED FRACTION OF THE TO# F02010
     1TAL INFLOW TO THE CELL'                                           # F02020
      END IF                                                            # F02030
      IF(IBATCH.EQ.0) THEN                                              # F02040
      READ (*,*) ISNK                                                   # F02050
      WRITE(I9,*) ISNK                                                  # F02060
      ELSE                                                              # F02070
      READ(I9,*) ISNK                                                   # F02080
      END IF                                                            # F02090
      FRAC=0.99                                                         # F02100
      IF(ISNK.EQ.2) THEN                                                # F02110
      IF (IBATCH.EQ.0) THEN                                             # F02120
      WRITE (*,*) 'ENTER A NUMBER BETWEEN 0 AND 1:'                     # F02130
      WRITE (*,*) '    (0.0 => NONE OF THE INFLOW TO THE CELL IS DISCHAR# F02140
     1GED TO INTERNAL SINKS)'                                           # F02150
      WRITE (*,*) '    (1.0 => ALL INFLOW TO THE CELL IS DISCHARGED TO I# F02160
```

```
     1NTERNAL SINKS)'                                                    # F02170
      END IF                                                             # F02180
      IF(IBATCH.EQ.0) THEN                                               # F02190
      READ (*,*) FRAC                                                    # F02200
      WRITE(I9,*) FRAC                                                   # F02210
      ELSE                                                               # F02220
      READ(I9,*) FRAC                                                    # F02230
      END IF                                                             # F02240
      IF(FRAC.GE.0.99) FRAC=0.99                                         # F02250
      END IF                                                             # F02260
C                                                                        # F02270
C  SPECIFY AUTOMATIC DISCHARGE ZONE                                      # F02280
C                                                                        # F02290
      IZSTOP=0                                                           # F02300
      IPEP=0                                                             # F02310
      MES= 'DO YOU WANT TO SPECIFY ONE ZONE IN WHICH TO STOP PARTICLES W# F02320
     1HENEVER THEY ENTER ?'                                              # F02330
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                 # F02340
      IF(IANS.EQ.1) THEN                                                 # F02350
      IF (IBATCH.EQ.0)                                                   # F02360
     1WRITE (*,*) 'ENTER THE ZONE NUMBER (MUST BE > 1):'                 # F02370
      IF(IBATCH.EQ.0) THEN                                               # F02380
      READ (*,*) IZSTOP                                                  # F02390
      WRITE(I9,*) IZSTOP                                                 # F02400
      ELSE                                                               # F02410
      READ(I9,*) IZSTOP                                                  # F02420
      END IF                                                             # F02430
      IF(IZSTOP.LT.2) IZSTOP=0                                           # F02440
      IF (IBATCH.EQ.0) THEN                                              # F02450
      WRITE (*,*) 'SPECIFY WHICH ENDPOINTS TO RECORD:'                   # F02460
      WRITE (*,*) '    0 = ENDPOINT DATA RECORDED FOR ALL PARTICLES'     # F02470
      WRITE (*,*) '    1 = ENDPOINT DATA RECORDED ONLY FOR PARTICLES'    # F02480
      WRITE (*,5060) IZSTOP                                              # F02490
      END IF                                                             # F02500
5060  FORMAT ('         TERMINATING IN ZONE ',I3)                        # F02510
      IF(IBATCH.EQ.0) THEN                                               # F02520
      READ (*,*) IPEP                                                    # F02530
      WRITE(I9,*) IPEP                                                   # F02540
      ELSE                                                               # F02550
      READ(I9,*) IPEP                                                    # F02560
      END IF                                                             # F02570
      IF (IPEP.LT.0.OR.IPEP.GT.1) IPEP=0                                 # F02580
      END IF                                                             # F02590
C                                                                        # F02600
      IF (IBATCH.EQ.0)                                                   # F02610
     1WRITE (*,*) 'READING AND PROCESSING INPUT DATA...'                 # F02620
C                                                                        # F02630
C  LAYER TYPE CODES                                                      # F02640
C                                                                        # F02650
      READ(IU,5070) (LAYCON(N),N=1,NLAY)                                 # F02660
5070  FORMAT(40I2)                                                       # F02670
      WRITE(I7,5000)                                                     # F02680
      WRITE(I7,*) 'LAYCON (LAYER TYPE CODES):'                           # F02690
      WRITE(I7,5080) (LAYCON(N),N=1,NLAY)                                # F02700
5080   FORMAT(25I3)                                                      # F02710
C                                                                        # F02720
C  CONFINING BED CODES                                                   # F02730
C                                                                        # F02740
      IF(NCBL.GT.0) THEN                                                 # F02750
      READ(IU,5070) (NCON(N),N=1,NLAY)                                   # F02760
      ELSE                                                               # F02770
      DO 20 N=1,NLAY                                                     # F02780
20    NCON(N)=0                                                          # F02790
      END IF                                                             # F02800
      IF(NCBL.EQ.0) THEN                                                 # F02810
      WRITE(I7,5000)                                                     # F02820
      WRITE(I7,*) 'NO CONFINING LAYERS.  NCON = 0 FOR ALL LAYERS.'       # F02830
      ELSE                                                               # F02840
      WRITE(I7,5000)                                                     # F02850
      WRITE(I7,*) 'NCON (CONFINING LAYER CODES):'                        # F02860
      WRITE(I7,5080) (NCON(N),N=1,NLAY)                                  # F02870
      END IF                                                             # F02880
```

```
      IF(NCBL.GT.0) THEN                                                # F02890
      NN=0                                                              # F02900
      DO 30 N=1,NLAY                                                    # F02910
      IF(NCON(N).EQ.0) GO TO 30                                         # F02920
      NN=NN+1                                                           # F02930
      NCON(N)=NN                                                        # F02940
30    CONTINUE                                                          # F02950
      END IF                                                            # F02960
C                                                                       # F02970
C  DELR, GRID SPACING ALONG A ROW (X-DIRECTION)                         # F02980
C                                                                       # F02990
      WRITE(I7,5000)                                                    # F03000
      WRITE(I7,*) 'DELR ARRAY NOW BEING READ...'                        # F03010
      CALL IN1DR (IU,DELR,NCOL,I7)                                      # F03020
C                                                                       # F03030
C     COMPUTE XMAX ARRAY                                                # F03040
C                                                                       # F03050
      XMAX(1)=DELR(1)                                                   # F03060
      DO 40 J=2,NCOL                                                    # F03070
40     XMAX(J)=XMAX(J-1)+DELR(J)                                        # F03080
C                                                                       # F03090
C  DELC, GRID SPACING ALONG A COLUMN (Y-DIRECTION)                      # F03100
C                                                                       # F03110
      WRITE(I7,5000)                                                    # F03120
      WRITE(I7,*) 'DELC ARRAY NOW BEING READ...'                        # F03130
      CALL IN1DR (IU,DELC,NROW,I7)                                      # F03140
C                                                                       # F03150
C     COMPUTE YMAX ARRAY                                                # F03160
C                                                                       # F03170
      YMAX(NROW)=DELC(NROW)                                             # F03180
      DO 50 I=2,NROW                                                    # F03190
      II=NROW+1-I                                                       # F03200
50    YMAX(II)=YMAX(II+1)+DELC(II)                                      # F03210
C                                                                       # F03220
C  READ VERTICAL COORDINATE DATA                                        # F03230
C                                                                       # F03240
      IF (IGRID.EQ.1) THEN                                              # F03250
      WRITE(I7,5000)                                                    # F03260
      WRITE(I7,*) 'DELZ ARRAY NOW BEING READ...'                        # F03270
      CALL IN1DR (IU,DELZ,NLAY,I7)                                      # F03280
      IF(NCBL.GT.0) THEN                                                # F03290
      WRITE(I7,5000)                                                    # F03300
      WRITE(I7,*) 'DELZCB ARRAY NOW BEING READ...'                      # F03310
      CALL IN1DR (IU,DELZCB,NLAY,I7)                                    # F03320
      ELSE                                                              # F03330
      DO 60 N=1,NLAY                                                    # F03340
60    DELZCB(N)=0.0                                                     # F03350
      END IF                                                            # F03360
      READ(IU,5090) ZBL1                                                # F03370
5090  FORMAT(F10.0)                                                     # F03380
      ZBOT(1)=ZBL1                                                      # F03390
      WRITE(I7,5000)                                                    # F03400
      WRITE(I7,5100) ZBOT(1)                                            # F03410
5100  FORMAT('BOTTOM ELEVATION OF LAYER 1 IS',E13.5)                    # F03420
      ZTOP(1)= ZBOT(1) + DELZ(1)                                        # F03430
      DO 70 K=2,NLAY                                                    # F03440
      ZTOP(K)= ZBOT(K-1) - DELZCB(K-1)                                  # F03450
      ZBOT(K)= ZTOP(K) - DELZ(K)                                        # F03460
70    CONTINUE                                                          # F03470
      ELSE                                                              # F03480
      DO 80 K=1,NLAY                                                    # F03490
      KP= 1 + (K-1)*NCOL*NROW                                           # F03500
      IF(LAYCON(K).NE.1) THEN                                           # F03510
      WRITE(I7,5000)                                                    # F03520
      WRITE(I7,5110) K                                                  # F03530
5110  FORMAT('TOP ELEVATION OF LAYER',I4,' NOW BEING READ...')          # F03540
      CALL IN2DR (IU,ZTOP(KP),NCOL,NROW,I7)                             # F03550
      END IF                                                            # F03560
      WRITE(I7,5000)                                                    # F03570
      WRITE(I7,5120) K                                                  # F03580
5120  FORMAT('BOTTOM ELEVATION OF LAYER',I4,' NOW BEING READ...')       # F03590
      CALL IN2DR (IU,ZBOT(KP),NCOL,NROW,I7)                             # F03600
```

```
80    CONTINUE                                                          # F03610
      END IF                                                            # F03620
C                                                                       # F03630
C  IBOUND DATA                                                          # F03640
C                                                                       # F03650
      DO 90 K=1,NLAY                                                    # F03660
      WRITE(I7,5000)                                                    # F03670
      WRITE(I7,5130) K                                                  # F03680
5130  FORMAT('IBOUND ARRAY FOR LAYER',I4,' NOW BEING READ...')          # F03690
      CALL IN2DI (IU,IBOUND(1,1,K),NCOL,NROW,I7)                        # F03700
90    CONTINUE                                                          # F03710
C                                                                       # F03720
C  POROSITY DATA                                                        # F03730
C                                                                       # F03740
      DO 100 K=1,NLAY                                                   # F03750
      WRITE(I7,5000)                                                    # F03760
      WRITE(I7,5140) K                                                  # F03770
5140  FORMAT('POROSITY ARRAY FOR LAYER',I4,' NOW BEING READ...')        # F03780
      CALL IN2DR (IU,POR(1,1,K),NCOL,NROW,I7)                           # F03790
      LCON=NCON(K)                                                      # F03800
      IF(LCON.GT.0) THEN                                                # F03810
      KCB=NLAY+LCON                                                     # F03820
      WRITE(I7,5000)                                                    # F03830
      WRITE(I7,5150) K                                                  # F03840
5150  FORMAT('POROSITY ARRAY FOR CONFINING LAYER UNDERLYING MODEL LAYER'# F03850
     1,I4,' NOW BEING READ...')                                         # F03860
      CALL IN2DR (IU,POR(1,1,KCB),NCOL,NROW,I7)                         # F03870
      END IF                                                            # F03880
100    CONTINUE                                                         # F03890
C                                                                       # F03900
C  READ HEADS                                                           # F03910
C                                                                       # F03920
      DO 110 K=1,NLAY                                                   # F03930
      DO 110 I=1,NROW                                                   # F03940
      DO 110 J=1,NCOL                                                   # F03950
110    HEAD(J,I,K)=0.                                                   # F03960
      IUHED=IUNIT(8)                                                    # F03970
      IF (IUHED.NE.0) THEN                                              # F03980
120    CONTINUE                                                         # F03990
      READ (IUHED,END=130) KSTP,KPER,PERTIM,TOTIM,TEXT,NC,NR,K          # F04000
      IF (TEXT.NE.'            HEAD') THEN                              # F04010
      IF (IBATCH.EQ.0)                                                  # F04020
     1WRITE (*,*) 'HEAD FILE DOES NOT CONTAIN HEAD DATA.'               # F04030
      WRITE (I7,*) 'HEAD FILE DOES NOT CONTAIN HEAD DATA.'              # F04040
      STOP                                                              # F04050
      END IF                                                            # F04060
      WRITE(I7,5000)                                                    # F04070
      WRITE(I7,5160) K                                                  # F04080
5160  FORMAT('HEADS NOW BEING READ FOR LAYER',I4)                       # F04090
      READ(IUHED) ((HEAD(J,I,K),J=1,NCOL),I=1,NROW)                     # F04100
      GO TO 120                                                         # F04110
      END IF                                                            # F04120
130    CONTINUE                                                         # F04130
      WRITE(I7,*) 'HEADS HAVE BEEN READ'                                # F04140
      WRITE(I7,*) '**** HEADS FOR INDIVIDUAL CELLS CAN BE CHECKED USING # F04150
     1THE CELL-BY-CELL OPTION ****'                                     # F04160
C                                                                       # F04170
C  SET IBOUND = 0 FOR DRY CELLS                                         # F04180
C                                                                       # F04190
      DO 140 K=1,NLAY                                                   # F04200
      DO 140 I=1,NROW                                                   # F04210
      DO 140 J=1,NCOL                                                   # F04220
      IF (HEAD(J,I,K).GT.1.0E+29) IBOUND(J,I,K)=0                       # F04230
140    CONTINUE                                                         # F04240
C                                                                       # F04250
      CALL CELFLO (QX,QY,QZ,BUFF,IUNIT,IBOUND,HEAD,IBUFF,DELR,          # F04260
     1DELC,NCOL,NROW,NLAY,NCP1,NRP1,NLP1,NUNIT,QSS,IA,I7,IBATCH,I9)     # F04270
C                                                                       # F04280
C  READ PARTICLE LOCATIONS FROM DATA FILE                               # F04290
C                                                                       # F04300
      IF(IRP.EQ.1) THEN                                                 # F04310
      WRITE(I7,5000)                                                    # F04320
```

```
      WRITE(I7,*) 'STARTING LOCATIONS NOW BEING READ FROM FILE...'      # F04330
      N=0                                                               # F04340
150    N=N+1                                                            # F04350
      IF(N.GT.NPART) GO TO 190                                          # F04360
160   READ(I6,*,END=190) JLC(N),ILC(N),KLC(N),XLC(N),YLC(N),ZLLC(N)     # F04370
      NPRT=N                                                            # F04380
      JP=JLC(N)                                                         # F04390
      IP=ILC(N)                                                         # F04400
      IF(KLC(N).EQ.0) THEN                                              # F04410
      ZLLC(N)=1.0                                                       # F04420
      DO 170 K=1,NLAY                                                   # F04430
      KLC(N)=K                                                          # F04440
      IF(IBOUND(JP,IP,K).EQ.0) GO TO 170                                # F04450
      IF(HEAD(JP,IP,K).GE.1.0E+29) GO TO 170                            # F04460
      GO TO 180                                                         # F04470
170    CONTINUE                                                         # F04480
      GO TO 160                                                         # F04490
      END IF                                                            # F04500
180    CONTINUE                                                         # F04510
      IF(IBOUND(JP,IP,KLC(N)).EQ.0) GO TO 160                           # F04520
C                                                                       # F04530
C  CONVERT LOCAL X AND Y TO GLOBAL COORDINATES                          # F04540
C                                                                       # F04550
      IF(XLC(N).LE.0.0) XLC(N)=0.001                                    # F04560
      IF(XLC(N).GE.1.0) XLC(N)=0.999                                    # F04570
      IF(YLC(N).LE.0.0) YLC(N)=0.001                                    # F04580
      IF(YLC(N).GE.1.0) YLC(N)=0.999                                    # F04590
      IF(ZLLC(N).LE.0.0) ZLLC(N)=0.001                                  # F04600
      IF(ZLLC(N).GE.1.0) ZLLC(N)=0.999                                  # F04610
      XMN=0.0                                                           # F04620
      XMX=XMAX(JP)                                                      # F04630
      IF(JP.GT.1) XMN= XMAX(JP-1)                                       # F04640
      XLC(N)= (1.0-XLC(N))*XMN + XLC(N)*XMX                             # F04650
      YMN=0.0                                                           # F04660
      YMX=YMAX(IP)                                                      # F04670
      IF(IP.LT.NROW) YMN=YMAX(IP+1)                                     # F04680
      YLC(N)= (1.0-YLC(N))*YMN + YLC(N)*YMX                             # F04690
      WRITE(I5,5170) N,XLC(N),YLC(N),ZLLC(N),JLC(N),ILC(N),KLC(N)       # F04700
5170  FORMAT(I10,3E15.6,3I5)                                            # F04710
      GO TO 150                                                         # F04720
190   CONTINUE                                                          # F04730
      WRITE(I7,*) '  STARTING LOCATIONS HAVE BEEN READ FROM FILE'       # F04740
      END IF                                                            # F04750
      IF(IRP.EQ.2) THEN                                                 # F04760
      MES= ' STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.'  # F04770
      IF (IBATCH.EQ.0) WRITE (*,5180) MES                               # F04780
5180  FORMAT(A)                                                         # F04790
      IF (IBATCH.EQ.0) WRITE (*,*) ' '                                  # F04800
      MES= 'SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?'        # F04810
      CALL YESNO (IBATCH,I9,MES,IA,ICHEAD)                              # F04820
      N=0                                                               # F04830
      KO=0                                                              # F04840
200   CONTINUE                                                          # F04850
      KO=KO+1                                                           # F04860
      IF (IBATCH.EQ.0) THEN                                             # F04870
      WRITE (*,5190) KO                                                 # F04880
5190  FORMAT(' ENTER DATA FOR SUBREGION ',I2,' :')                      # F04890
      WRITE (*,*) ' '                                                   # F04900
      WRITE (*,*) 'DEFINE THE SUBREGION BY ENTERING THE MINIMUM AND MAXI# F04910
     1MUM J,I,K COORDINATES.'                                           # F04920
      WRITE (*,*) '  ENTER:  MINIMUM J, MAXIMUM J, MINIMUM I, MAXIMUM I,# F04930
     1 MINIMUM K, MAXIMUM K'                                            # F04940
      END IF                                                            # F04950
      IF(IBATCH.EQ.0) THEN                                              # F04960
      READ (*,*) JMIN,JMAX,IMIN,IMAX,KMIN,KMAX                          # F04970
      WRITE(I9,*) JMIN,JMAX,IMIN,IMAX,KMIN,KMAX                         # F04980
      ELSE                                                              # F04990
      READ(I9,*) JMIN,JMAX,IMIN,IMAX,KMIN,KMAX                          # F05000
      END IF                                                            # F05010
      IF (JMAX.LT.1.OR.JMAX.GT.NCOL) JMAX=NCOL                          # F05020
      IF (IMAX.LT.1.OR.IMAX.GT.NROW) IMAX=NCOL                          # F05030
      IF (KMAX.LT.1.OR.KMAX.GT.NLAY) KMAX=NLAY                          # F05040
```

```
 KWT=0                                                                  # F05050
 IF(KMIN.EQ.0) THEN                                                     # F05060
 KMIN=1                                                                 # F05070
 KMAX=1                                                                 # F05080
 KWT=1                                                                  # F05090
 END IF                                                                 # F05100
 IF (IBATCH.EQ.0) THEN                                                  # F05110
 WRITE (*,*) 'WHERE SHOULD THE PARTICLES BE LOCATED ?'                  # F05120
 WRITE (*,*) '  0 = WITHIN A CELL'                                      # F05130
 WRITE (*,*) '  1 = ON ONE OR MORE OF THE CELL FACES'                   # F05140
 END IF                                                                 # F05150
 IF(IBATCH.EQ.0) THEN                                                   # F05160
 READ (*,*) ILOC                                                        # F05170
 WRITE(I9,*) ILOC                                                       # F05180
 ELSE                                                                   # F05190
 READ(I9,*) ILOC                                                        # F05200
 END IF                                                                 # F05210
 IF(ILOC.EQ.0) THEN                                                     # F05220
 IF (IBATCH.EQ.0) THEN                                                  # F05230
 WRITE (*,*) 'ENTER: NJ  NI  NK'                                        # F05240
 WRITE (*,*) '   NJ = NUMBER OF PARTICLES PER CELL IN THE J DIRECTI# F05250
1ON'                                                                    # F05260
 WRITE (*,*) '   NI = NUMBER OF PARTICLES PER CELL IN THE I DIRECTI# F05270
1ON'                                                                    # F05280
 WRITE (*,*) '   NK = NUMBER OF PARTICLES PER CELL IN THE K DIRECTI# F05290
1ON'                                                                    # F05300
 END IF                                                                 # F05310
 IF(IBATCH.EQ.0) THEN                                                   # F05320
 READ (*,*) NJ,NI,NK                                                    # F05330
 WRITE(I9,*) NJ,NI,NK                                                   # F05340
 ELSE                                                                   # F05350
 READ(I9,*) NJ,NI,NK                                                    # F05360
 END IF                                                                 # F05370
 IF(NJ.EQ.0) NJ=1                                                       # F05380
 IF(NI.EQ.0) NI=1                                                       # F05390
 IF(NK.EQ.0) NK=1                                                       # F05400
 ELSE                                                                   # F05410
 MES= 'DO YOU WANT TO PLACE PARTICLES ON FACE 1 ?'                      # F05420
 CALL YESNO (IBATCH,I9,MES,IA,IF1)                                      # F05430
 IF(IF1.EQ.1) THEN                                                      # F05440
 IF (IBATCH.EQ.0) THEN                                                  # F05450
 WRITE (*,*) 'ENTER:  NI  NK'                                           # F05460
 WRITE (*,*) '   NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FA# F05470
1CE 1'                                                                  # F05480
 WRITE (*,*) '   NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FA# F05490
1CE 1'                                                                  # F05500
 END IF                                                                 # F05510
 IF(IBATCH.EQ.0) THEN                                                   # F05520
 READ (*,*) NI1,NK1                                                     # F05530
 WRITE(I9,*) NI1,NK1                                                    # F05540
 ELSE                                                                   # F05550
 READ(I9,*) NI1,NK1                                                     # F05560
 END IF                                                                 # F05570
 END IF                                                                 # F05580
 MES= 'DO YOU WANT TO PLACE PARTICLES ON FACE 2 ?'                      # F05590
 CALL YESNO (IBATCH,I9,MES,IA,IF2)                                      # F05600
 IF(IF2.EQ.1) THEN                                                      # F05610
 IF (IBATCH.EQ.0) THEN                                                  # F05620
 WRITE (*,*) 'ENTER:  NI  NK'                                           # F05630
 WRITE (*,*) '   NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FA# F05640
1CE 2'                                                                  # F05650
 WRITE (*,*) '   NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FA# F05660
1CE 2'                                                                  # F05670
 END IF                                                                 # F05680
 IF(IBATCH.EQ.0) THEN                                                   # F05690
 READ (*,*) NI2,NK2                                                     # F05700
 WRITE(I9,*) NI2,NK2                                                    # F05710
 ELSE                                                                   # F05720
 READ(I9,*) NI2,NK2                                                     # F05730
 END IF                                                                 # F05740
 END IF                                                                 # F05750
 MES= 'DO YOU WANT TO PLACE PARTICLES ON FACE 3 ?'                      # F05760
```

```
 CALL YESNO (IBATCH,I9,MES,IA,IF3)                                      # F05770
 IF(IF3.EQ.1) THEN                                                      # F05780
 IF (IBATCH.EQ.0) THEN                                                  # F05790
 WRITE (*,*) 'ENTER:  NJ  NK'                                           # F05800
 WRITE (*,*) '   NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FA# F05810
1CE 3'                                                                  # F05820
 WRITE (*,*) '   NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FA# F05830
1CE 3'                                                                  # F05840
 END IF                                                                 # F05850
 IF(IBATCH.EQ.0) THEN                                                   # F05860
 READ (*,*) NJ3,NK3                                                     # F05870
 WRITE(I9,*) NJ3,NK3                                                    # F05880
 ELSE                                                                   # F05890
 READ(I9,*) NJ3,NK3                                                     # F05900
 END IF                                                                 # F05910
 END IF                                                                 # F05920
 MES= 'DO YOU WANT TO PLACE PARTICLES ON FACE 4 ?'                      # F05930
 CALL YESNO (IBATCH,I9,MES,IA,IF4)                                      # F05940
 IF(IF4.EQ.1) THEN                                                      # F05950
 IF (IBATCH.EQ.0) THEN                                                  # F05960
 WRITE (*,*) 'ENTER:  NJ  NK'                                           # F05970
 WRITE (*,*) '   NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FA# F05980
1CE 4'                                                                  # F05990
 WRITE (*,*) '   NK = NUMBER OF PARTICLES IN THE K DIRECTION FOR FA# F06000
1CE 4'                                                                  # F06010
 END IF                                                                 # F06020
 IF(IBATCH.EQ.0) THEN                                                   # F06030
 READ (*,*) NJ4,NK4                                                     # F06040
 WRITE(I9,*) NJ4,NK4                                                    # F06050
 ELSE                                                                   # F06060
 READ(I9,*) NJ4,NK4                                                     # F06070
 END IF                                                                 # F06080
 END IF                                                                 # F06090
 MES= 'DO YOU WANT TO PLACE PARTICLES ON FACE 5 ?'                      # F06100
 CALL YESNO (IBATCH,I9,MES,IA,IF5)                                      # F06110
 IF(IF5.EQ.1) THEN                                                      # F06120
 IF (IBATCH.EQ.0) THEN                                                  # F06130
 WRITE (*,*) 'ENTER:  NJ  NI'                                           # F06140
 WRITE (*,*) '   NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FA# F06150
1CE 5'                                                                  # F06160
 WRITE (*,*) '   NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FA# F06170
1CE 5'                                                                  # F06180
 END IF                                                                 # F06190
 IF(IBATCH.EQ.0) THEN                                                   # F06200
 READ (*,*) NJ5,NI5                                                     # F06210
 WRITE(I9,*) NJ5,NI5                                                    # F06220
 ELSE                                                                   # F06230
 READ(I9,*) NJ5,NI5                                                     # F06240
 END IF                                                                 # F06250
 END IF                                                                 # F06260
 MES= 'DO YOU WANT TO PLACE PARTICLES ON FACE 6 ?'                      # F06270
 CALL YESNO (IBATCH,I9,MES,IA,IF6)                                      # F06280
 IF(IF6.EQ.1) THEN                                                      # F06290
 IF (IBATCH.EQ.0) THEN                                                  # F06300
 WRITE (*,*) 'ENTER:  NJ  NI'                                           # F06310
 WRITE (*,*) '   NJ = NUMBER OF PARTICLES IN THE J DIRECTION FOR FA# F06320
1CE 6'                                                                  # F06330
 WRITE (*,*) '   NI = NUMBER OF PARTICLES IN THE I DIRECTION FOR FA# F06340
1CE 6'                                                                  # F06350
 END IF                                                                 # F06360
 IF(IBATCH.EQ.0) THEN                                                   # F06370
 READ (*,*) NJ6,NI6                                                     # F06380
 WRITE(I9,*) NJ6,NI6                                                    # F06390
 ELSE                                                                   # F06400
 READ(I9,*) NJ6,NI6                                                     # F06410
 END IF                                                                 # F06420
 END IF                                                                 # F06430
 END IF                                                                 # F06440
 DO 230 K=KMIN,KMAX                                                     # F06450
 DO 230 I=IMIN,IMAX                                                     # F06460
 DO 230 J=JMIN,JMAX                                                     # F06470
 KKK=K                                                                  # F06480
```

```
      IF(KWT.EQ.1) THEN                                                 # F06490
      DO 210 KK=1,NLAY                                                  # F06500
      KKK=KK                                                            # F06510
      IF(IBOUND(J,I,KKK).EQ.0) GO TO 210                                # F06520
      IF(HEAD(J,I,KKK).GE.1.00E+29) GO TO 210                           # F06530
      GO TO 220                                                         # F06540
210   CONTINUE                                                          # F06550
      GO TO 230                                                         # F06560
      END IF                                                            # F06570
      IF(IBOUND(J,I,KKK).EQ.0) GO TO 230                                # F06580
220   CONTINUE                                                          # F06590
      IF (ICHEAD.EQ.0.AND.IBOUND(J,I,KKK).LT.0) GO TO 230               # F06600
      IF (ILOC.EQ.0) THEN                                               # F06610
      CALL VOLDIV(J,I,KKK,JLC,ILC,KLC,XLC,YLC,ZLLC,NI,NJ,NK,NPART,N,    # F06620
     1IA,I9,IBATCH)                                                     # F06630
      ELSE                                                              # F06640
      IF(IF1.EQ.1) CALL FACDIV (J,I,KKK,JLC,ILC,KLC,XLC,YLC,ZLLC,1,     # F06650
     1NI1,NK1,NPART,N,IA,I9,IBATCH)                                     # F06660
      IF(IF2.EQ.1) CALL FACDIV (J,I,KKK,JLC,ILC,KLC,XLC,YLC,ZLLC,2,     # F06670
     1NI2,NK2,NPART,N,IA,I9,IBATCH)                                     # F06680
      IF(IF3.EQ.1) CALL FACDIV (J,I,KKK,JLC,ILC,KLC,XLC,YLC,ZLLC,3,     # F06690
     1NJ3,NK3,NPART,N,IA,I9,IBATCH)                                     # F06700
      IF(IF4.EQ.1) CALL FACDIV (J,I,KKK,JLC,ILC,KLC,XLC,YLC,ZLLC,4,     # F06710
     1NJ4,NK4,NPART,N,IA,I9,IBATCH)                                     # F06720
      IF(IF5.EQ.1) CALL FACDIV (J,I,KKK,JLC,ILC,KLC,XLC,YLC,ZLLC,5,     # F06730
     1NJ5,NI5,NPART,N,IA,I9,IBATCH)                                     # F06740
      IF(IF6.EQ.1) CALL FACDIV (J,I,KKK,JLC,ILC,KLC,XLC,YLC,ZLLC,6,     # F06750
     1NJ6,NI6,NPART,N,IA,I9,IBATCH)                                     # F06760
      END IF                                                            # F06770
230   CONTINUE                                                          # F06780
      IF (IBATCH.EQ.0) WRITE (*,5200) N,NPART                           # F06790
5200  FORMAT(' TOTAL NUMBER OF PARTICLES IS ',I6/                       # F06800
     1' MAXIMUM NUMBER OF PARTICLES IS ',I6)                            # F06810
      MES= 'DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?'                 # F06820
      CALL YESNO(IBATCH,I9,MES,IA,IANS)                                 # F06830
      IF(IANS.EQ.1) GO TO 200                                           # F06840
      NPRT=N                                                            # F06850
      DO 240 N=1,NPRT                                                   # F06860
      IF(ILSTOR.EQ.1) WRITE(I6,5210) JLC(N),ILC(N),KLC(N),XLC(N),YLC(N),# F06870
     1ZLLC(N)                                                           # F06880
5210  FORMAT(3I10,3F14.7)                                               # F06890
      JP=JLC(N)                                                         # F06900
      IP=ILC(N)                                                         # F06910
      XMN=0.0                                                           # F06920
      XMX=XMAX(JP)                                                      # F06930
      IF(JP.GT.1) XMN= XMAX(JP-1)                                       # F06940
      XLC(N)= (1.0-XLC(N))*XMN + XLC(N)*XMX                             # F06950
      YMN=0.0                                                           # F06960
      YMX=YMAX(IP)                                                      # F06970
      IF(IP.LT.NROW) YMN=YMAX(IP+1)                                     # F06980
      YLC(N)= (1.0-YLC(N))*YMN + YLC(N)*YMX                             # F06990
      WRITE(I5,5170) N,XLC(N),YLC(N),ZLLC(N),JLC(N),ILC(N),KLC(N)       # F07000
240   CONTINUE                                                          # F07010
      IF(ILSTOR.EQ.1) CLOSE (I6)                                        # F07020
      END IF                                                            # F07030
C                                                                       # F07040
C  CHANGE ZONE CODES IN THE IBOUND ARRAY                                # F07050
C                                                                       # F07060
      IF (IZSTOP.NE.0) THEN                                             # F07070
      MES= 'DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND AR# F07080
     1RAY ?'                                                            # F07090
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # F07100
      IF (IANS.EQ.1) THEN                                               # F07110
250   CONTINUE                                                          # F07120
      IF (IBATCH.EQ.0) THEN                                             # F07130
      WRITE (*,*) 'WHAT TYPE OF CHANGE DO YOU WANT TO MAKE ?'           # F07140
      WRITE (*,*) '    1 = CHANGE AN ENTIRE LAYER'                      # F07150
      WRITE (*,*) '    2 = CHANGE AN INDIVIDUAL CELL'                   # F07160
      WRITE (*,*) '    3 = CHANGE ALL CELLS IN A SUBREGION OF THE GRID' # F07170
      END IF                                                            # F07180
      IF(IBATCH.EQ.0) THEN                                              # F07190
      READ (*,*) IANS                                                   # F07200
```

```
      WRITE(I9,*) IANS                                                  # F07210
      ELSE                                                              # F07220
      READ(I9,*) IANS                                                   # F07230
      END IF                                                            # F07240
      IF(IANS.EQ.1) THEN                                                # F07250
      IF (IBATCH.EQ.0) WRITE (*,*) 'ENTER THE LAYER NUMBER:'            # F07260
      IF(IBATCH.EQ.0) THEN                                              # F07270
      READ (*,*) NLAYER                                                 # F07280
      WRITE(I9,*) NLAYER                                                # F07290
      ELSE                                                              # F07300
      READ(I9,*) NLAYER                                                 # F07310
      END IF                                                            # F07320
      IF (IBATCH.EQ.0) WRITE (*,*) 'ENTER THE NEW ZONE CODE:'           # F07330
      IF(IBATCH.EQ.0) THEN                                              # F07340
      READ (*,*) NUM                                                    # F07350
      WRITE(I9,*) NUM                                                   # F07360
      ELSE                                                              # F07370
      READ(I9,*) NUM                                                    # F07380
      END IF                                                            # F07390
      DO 260 I=1,NROW                                                   # F07400
      DO 260 J=1,NCOL                                                   # F07410
      ITEMP=IBOUND(J,I,NLAYER)                                          # F07420
      IF(ITEMP.NE.0) THEN                                               # F07430
      IBOUND(J,I,NLAYER)=NUM                                            # F07440
      IF(ITEMP.LT.0) IBOUND(J,I,NLAYER)= -IBOUND(J,I,NLAYER)            # F07450
      END IF                                                            # F07460
260   CONTINUE                                                          # F07470
      ELSE IF (IANS.EQ.2) THEN                                          # F07480
      IF (IBATCH.EQ.0)                                                  # F07490
     1WRITE (*,*) 'ENTER THE CELL INDICES:  J  I  K'                    # F07500
      IF(IBATCH.EQ.0) THEN                                              # F07510
      READ (*,*) J,I,K                                                  # F07520
      WRITE(I9,*) J,I,K                                                 # F07530
      ELSE                                                              # F07540
      READ(I9,*) J,I,K                                                  # F07550
      END IF                                                            # F07560
      IF (IBATCH.EQ.0) WRITE (*,*) 'ENTER THE NEW ZONE CODE:'           # F07570
      IF(IBATCH.EQ.0) THEN                                              # F07580
      READ (*,*) NUM                                                    # F07590
      WRITE(I9,*) NUM                                                   # F07600
      ELSE                                                              # F07610
      READ(I9,*) NUM                                                    # F07620
      END IF                                                            # F07630
      ITEMP=IBOUND(J,I,K)                                               # F07640
      IF(ITEMP.NE.0) THEN                                               # F07650
      IBOUND(J,I,K)=NUM                                                 # F07660
      IF(ITEMP.LT.0) IBOUND(J,I,K)= -IBOUND(J,I,K)                      # F07670
      END IF                                                            # F07680
      ELSE IF (IANS.EQ.3) THEN                                          # F07690
      IF (IBATCH.EQ.0) THEN                                             # F07700
      WRITE (*,*) 'DEFINE THE SUBREGION BY ENTERING THE MINIMUM AND MAXI# F07710
     1MUM J,I,K COORDINATES.'                                           # F07720
      WRITE (*,*) '  ENTER:  MINIMUM J, MAXIMUM J, MINIMUM I, MAXIMUM I,# F07730
     1 MINIMUM K, MAXIMUM K'                                            # F07740
      END IF                                                            # F07750
      IF(IBATCH.EQ.0) THEN                                              # F07760
      READ (*,*) JJ1,JJ2,II1,II2,KK1,KK2                               # F07770
      WRITE(I9,*) JJ1,JJ2,II1,II2,KK1,KK2                              # F07780
      ELSE                                                              # F07790
      READ(I9,*) JJ1,JJ2,II1,II2,KK1,KK2                               # F07800
      END IF                                                            # F07810
      IF(JJ2.LT.1.OR.JJ2.GT.NCOL) JJ2=NCOL                              # F07820
      IF(II2.LT.1.OR.II2.GT.NROW) II2=NROW                              # F07830
      IF(KK2.LT.1.OR.KK2.GT.NLAY) KK2=NLAY                              # F07840
      IF (IBATCH.EQ.0) WRITE (*,*) 'ENTER THE NEW ZONE CODE:'           # F07850
      IF(IBATCH.EQ.0) THEN                                              # F07860
      READ (*,*) NUM                                                    # F07870
      WRITE(I9,*) NUM                                                   # F07880
      ELSE                                                              # F07890
      READ(I9,*) NUM                                                    # F07900
      END IF                                                            # F07910
      DO 270 K=KK1,KK2                                                  # F07920
```

```
      DO 270 I=II1,II2                                                  # F07930
      DO 270 J=JJ1,JJ2                                                  # F07940
      ITEMP=IBOUND(J,I,K)                                               # F07950
      IF(ITEMP.NE.0) THEN                                               # F07960
      IBOUND(J,I,K)=NUM                                                 # F07970
      IF(ITEMP.LT.0) IBOUND(J,I,K)= -IBOUND(J,I,K)                      # F07980
      END IF                                                            # F07990
270   CONTINUE                                                          # F08000
      END IF                                                            # F08010
      MES= 'DO YOU WANT TO CHANGE SOME MORE ZONE CODES ?'               # F08020
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # F08030
      IF (IANS.EQ.1) GO TO 250                                          # F08040
      END IF                                                            # F08050
      END IF                                                            # F08060
C                                                                       # F08070
      RETURN                                                            # F08080
280   FORMAT(A)                                                         # F08090
      END                                                               # F08100
C-----END OF ROUTINE-----------------------------------------------------# F08110
```

```
C                                                                       # G00010
C---Version 1.0   July 21, 1989                                         # G00020
C***********************************************************************# G00030
C                                                                       # G00040
C                          VOLDIV                                       # G00050
C                                                                       # G00060
C  THIS SUBROUTINE PLACES A THREE DIMENSIONAL ARRAY OF PARTICLES WITHIN # G00070
C  A FINITE DIFFERENCE CELL.                                            # G00080
C                                                                       # G00090
C***********************************************************************# G00100
C                                                                       # G00110
      SUBROUTINE VOLDIV (J,I,K,JLC,ILC,KLC,XLC,YLC,ZLLC,NDIV,MDIV,LDIV, # G00120
     1NPART,NP,IA,I9,IBATCH)                                            # G00130
      DIMENSION JLC(NPART),ILC(NPART),KLC(NPART),XLC(NPART),YLC(NPART), # G00140
     1ZLLC(NPART)                                                       # G00150
C                                                                       # G00160
      DELN= 1.0E+0/FLOAT(NDIV)                                          # G00170
      S1F= DELN/2.0E+0                                                  # G00180
      DELM= 1.0E+0/FLOAT(MDIV)                                          # G00190
      S2F= DELM/2.0E+0                                                  # G00200
      DELL= 1.0E+0/FLOAT(LDIV)                                          # G00210
      S3F= DELL/2.0E+0                                                  # G00220
      S1=S1F                                                            # G00230
      S2=S2F                                                            # G00240
      S3=S3F                                                            # G00250
      DO 30 L=1,LDIV                                                    # G00260
      S1=S1F                                                            # G00270
      DO 20 N=1,NDIV                                                    # G00280
      S2=S2F                                                            # G00290
      DO 10 M=1,MDIV                                                    # G00300
      NP=NP+1                                                           # G00310
      IF(NP.GT.NPART) THEN                                              # G00320
      IF (IBATCH.EQ.0)                                                  # G00330
     1WRITE (*,*) 'MAXIMUM NUMBER OF PARTICLES EXCEEDED. RUN STOPPED.'  # G00340
      STOP                                                              # G00350
      END IF                                                            # G00360
      JLC(NP)=J                                                         # G00370
      ILC(NP)=I                                                         # G00380
      KLC(NP)=K                                                         # G00390
      XLC(NP)=S2                                                        # G00400
      YLC(NP)=S1                                                        # G00410
      ZLLC(NP)=S3                                                       # G00420
      S2=S2+DELM                                                        # G00430
10    CONTINUE                                                          # G00440
      S1=S1+DELN                                                        # G00450
20    CONTINUE                                                          # G00460
      S3=S3+DELL                                                        # G00470
30    CONTINUE                                                          # G00480
      RETURN                                                            # G00490
      END                                                               # G00500
C-----END OF ROUTINE-------------------------------------------------- # G00510
```

```
C                                                                       # H00010
C---Version 1.0   July 21, 1989                                         # H00020
C***********************************************************************# H00030
C                                                                       # H00040
C                              FACDIV                                   # H00050
C                                                                       # H00060
C  THIS SUBROUTINE PLACES A TWO DIMENSIONAL ARRAY OF PARTICLES ON ONE   # H00070
C  FACE OF A FINITE DIFFERENCE CELL.                                    # H00080
C                                                                       # H00090
C***********************************************************************# H00100
C                                                                       # H00110
      SUBROUTINE FACDIV (J,I,K,JLC,ILC,KLC,XLC,YLC,ZLLC,IS,NDIV,        # H00120
     1MDIV,NPART,NP,IA,I9,IBATCH)                                       # H00130
      DIMENSION JLC(NPART),ILC(NPART),KLC(NPART),XLC(NPART),YLC(NPART), # H00140
     1ZLLC(NPART)                                                       # H00150
C                                                                       # H00160
      DELN= 1.0E+0/FLOAT(NDIV)                                          # H00170
      S1F= DELN/2.0E+0                                                  # H00180
      DELM= 1.0E+0/FLOAT(MDIV)                                          # H00190
      S2F= DELM/2.0E+0                                                  # H00200
      S1=S1F                                                            # H00210
      S2=S2F                                                            # H00220
      DO 20 N=1,NDIV                                                    # H00230
      S2=S2F                                                            # H00240
      DO 10 M=1,MDIV                                                    # H00250
      NP=NP+1                                                           # H00260
      IF(NP.GT.NPART) THEN                                              # H00270
      IF (IBATCH.EQ.0)                                                  # H00280
     1WRITE (*,*) 'MAXIMUM NUMBER OF PARTICLES EXCEEDED. RUN STOPPED.'  # H00290
      STOP                                                              # H00300
      END IF                                                            # H00310
      JLC(NP)=J                                                         # H00320
      ILC(NP)=I                                                         # H00330
      KLC(NP)=K                                                         # H00340
      IF(IS.EQ.1) THEN                                                  # H00350
      XLC(NP)=0.0001                                                    # H00360
      YLC(NP)=S1                                                        # H00370
      ZLLC(NP)=S2                                                       # H00380
      ELSE IF(IS.EQ.2) THEN                                             # H00390
      XLC(NP)=0.9999                                                    # H00400
      YLC(NP)=S1                                                        # H00410
      ZLLC(NP)=S2                                                       # H00420
      ELSE IF(IS.EQ.3) THEN                                             # H00430
      XLC(NP)=S1                                                        # H00440
      YLC(NP)=0.0001                                                    # H00450
      ZLLC(NP)=S2                                                       # H00460
      ELSE IF(IS.EQ.4) THEN                                             # H00470
      XLC(NP)=S1                                                        # H00480
      YLC(NP)=0.9999                                                    # H00490
      ZLLC(NP)=S2                                                       # H00500
      ELSE IF(IS.EQ.5) THEN                                             # H00510
      XLC(NP)=S1                                                        # H00520
      YLC(NP)=S2                                                        # H00530
      ZLLC(NP)=0.0001                                                   # H00540
      ELSE IF(IS.EQ.6) THEN                                             # H00550
      XLC(NP)=S1                                                        # H00560
      YLC(NP)=S2                                                        # H00570
      ZLLC(NP)=0.9999                                                   # H00580
      END IF                                                            # H00590
      S2=S2+DELM                                                        # H00600
10    CONTINUE                                                          # H00610
      S1=S1+DELN                                                        # H00620
20    CONTINUE                                                          # H00630
      RETURN                                                            # H00640
      END                                                               # H00650
C-----END OF ROUTINE-------------------------------------------------- # H00660
```

```
C                                                                       # I00010
C---Version 1.0   July 21, 1989                                         # I00020
C***********************************************************************# I00030
C                                                                       # I00040
C                                YESNO                                  # I00050
C                                                                       # I00060
C  THIS SUBROUTINE WRITES A QUESTION TO THE SCREEN THAT REQUIRES A      # I00070
C  YES-NO RESPONSE. THE YES-NO RESPONSE IS CONVERTED TO AN INTEGER      # I00080
C  RESPONSE (0 = NO, 1 =YES)                                            # I00090
C                                                                       # I00100
C***********************************************************************# I00110
C                                                                       # I00120
      SUBROUTINE YESNO(IBATCH,I9,MES,IA,IANS)                           # I00130
      CHARACTER*80 MES                                                  # I00140
      CHARACTER*1 IN                                                    # I00150
      CHARACTER*10 ANSWR                                                # I00160
10    CONTINUE                                                          # I00170
      IF (IBATCH.EQ.0) THEN                                             # I00180
      WRITE (*,5000) MES                                                # I00190
      WRITE (*,*) '   [Y = YES;    N OR <CR> = NO]'                     # I00200
      END IF                                                            # I00210
      IF(IBATCH.EQ.0) THEN                                              # I00220
      READ (*,'(A10)') ANSWR                                            # I00230
      IF (ANSWR.EQ.' ') WRITE(I9,*) 'N'                                 # I00240
      IF (ANSWR.NE.' ') WRITE(I9,'(A10)') ANSWR                         # I00250
      ELSE                                                              # I00260
      READ(I9,'(A10)') ANSWR                                            # I00270
      END IF                                                            # I00280
5000  FORMAT(A)                                                         # I00290
      IN=' '                                                            # I00300
      DO 20 N=1,10                                                      # I00310
      IF (ANSWR(N:N).NE.' ') THEN                                       # I00320
      IN= ANSWR(N:N)                                                    # I00330
      GO TO 30                                                          # I00340
      END IF                                                            # I00350
20    CONTINUE                                                          # I00360
30    CONTINUE                                                          # I00370
      IANS= -1                                                          # I00380
      IF(IN.EQ.'Y'.OR.IN.EQ.'y') IANS=1                                 # I00390
      IF(IN.EQ.'N'.OR.IN.EQ.'n'.OR.IN.EQ.' ') IANS=0                    # I00400
      IF(IANS.EQ.-1) GO TO 10                                           # I00410
      RETURN                                                            # I00420
      END                                                               # I00430
C-----END OF ROUTINE-------------------------------------------------- # I00440
```

```
C                                                                       # J00010
C---Version 1.0   July 21, 1989                                         # J00020
C***********************************************************************# J00030
C                                                                       # J00040
C                            CELFLO                                     # J00050
C                                                                       # J00060
C  THIS SUBROUTINE OVERSEES BUDGET DATA INPUT.                          # J00070
C                                                                       # J00080
C***********************************************************************# J00090
C                                                                       # J00100
      SUBROUTINE CELFLO (QX,QY,QZ,BUFF,IUNIT,IBOUND,HEAD,IBUFF,         # J00110
     1DELR,DELC,NCOL,NROW,NLAY,NCP1,NRP1,NLP1,NUNIT,QSS,IA,I7,IBATCH,   # J00120
     2I9)                                                               # J00130
C                                                                       # J00140
      DIMENSION DELC(NROW),DELR(NCOL),QX(NCP1,NROW,NLAY),               # J00150
     1QY(NCOL,NRP1,NLAY),QZ(NCOL,NROW,NLP1),BUFF(NCOL,NROW,NLAY),       # J00160
     2IUNIT(NUNIT),IBUFF(NCOL,NROW,NLAY),HEAD(NCOL,NROW,NLAY),          # J00170
     3IBOUND(NCOL,NROW,NLAY),QSS(NCOL,NROW,NLAY)                        # J00180
      CHARACTER*16 LABEL,TEXT                                           # J00190
C                                                                       # J00200
      IUBCF=IUNIT(7)                                                    # J00210
C                                                                       # J00220
C  ZERO ARRAY CONTAINING NET FLOW RATE TO SOURCES & SINKS               # J00230
C                                                                       # J00240
      DO 1 K=1,NLAY                                                     # J00250
      DO 1 I=1,NROW                                                     # J00260
      DO 1 J=1,NCOL                                                     # J00270
1     QSS(J,I,K)=0.0                                                    # J00280
C                                                                       # J00290
C  GENERATE PROCESSED FACE FLOW TERMS                                   # J00300
C                                                                       # J00310
      WRITE(I7,5500)                                                    # J00320
5500  FORMAT('-----------------------------------------------------------# J00330
     1----------------------')                                          # J00340
      WRITE(I7,*) 'FLOW RATES FROM THE BCF PACKAGE NOW BEING READ...'   # J00350
C                                                                       # J00360
C  READ CONSTANT HEAD FLOWS                                             # J00370
C                                                                       # J00380
      TEXT= '   CONSTANT HEAD'                                          # J00390
      CALL RDBUDG (BUFF,TEXT,NCOL,NROW,NLAY,IUBCF,I7)                   # J00400
C                                                                       # J00410
C  POTENTIAL DISCHARGE CELLS ARE FLAGGED IN THE IBOUND ARRAY            # J00420
C  BY MULTIPLYING IBOUND VALUE BY 1000. THIS LOOP FLAGS CH              # J00430
C  CELLS THAT HAVE NET DISCHARGE.                                       # J00440
C                                                                       # J00450
      DO 10 K=1,NLAY                                                    # J00460
      DO 10 I=1,NROW                                                    # J00470
      DO 10 J=1,NCOL                                                    # J00480
      IF(IBOUND(J,I,K).GE.0) GO TO 10                                   # J00490
      IF(BUFF(J,I,K).LE.0.0) IBOUND(J,I,K)=1000*IBOUND(J,I,K)           # J00500
10     CONTINUE                                                         # J00510
C                                                                       # J00520
C  READ X FACE FLOWS                                                    # J00530
C                                                                       # J00540
      IF(NCOL.GT.1) THEN                                                # J00550
      TEXT= 'FLOW RIGHT FACE '                                          # J00560
      CALL RDBUDG (BUFF,TEXT,NCOL,NROW,NLAY,IUBCF,I7)                   # J00570
      ELSE                                                              # J00580
      CALL ZERO (BUFF,NCOL,NROW,NLAY)                                   # J00590
      END IF                                                            # J00600
      DO 20 K=1,NLAY                                                    # J00610
      DO 20 I=1,NROW                                                    # J00620
      QX(1,I,K)= 0.0                                                    # J00630
      DO 20 J=1,NCOL                                                    # J00640
20    QX(J+1,I,K)= BUFF(J,I,K)                                          # J00650
C                                                                       # J00660
C  READ Y FACE FLOWS                                                    # J00670
C                                                                       # J00680
      IF(NROW.GT.1) THEN                                                # J00690
      TEXT= 'FLOW FRONT FACE '                                          # J00700
      CALL RDBUDG (BUFF,TEXT,NCOL,NROW,NLAY,IUBCF,I7)                   # J00710
      ELSE                                                              # J00720
```

```
      CALL ZERO (BUFF,NCOL,NROW,NLAY)                                    # J00730
      END IF                                                             # J00740
      DO 30 K=1,NLAY                                                     # J00750
      DO 30 J=1,NCOL                                                     # J00760
      QY(J,1,K)= 0.0                                                     # J00770
      DO 30 I=1,NROW                                                     # J00780
30    QY(J,I+1,K)= -BUFF(J,I,K)                                          # J00790
C                                                                        # J00800
C  READ Z FACE FLOWS                                                     # J00810
C                                                                        # J00820
      IF(NLAY.GT.1) THEN                                                 # J00830
      TEXT= 'FLOW LOWER FACE '                                           # J00840
      CALL RDBUDG (BUFF,TEXT,NCOL,NROW,NLAY,IUBCF,I7)                    # J00850
      ELSE                                                               # J00860
      CALL ZERO (BUFF,NCOL,NROW,NLAY)                                    # J00870
      END IF                                                             # J00880
      DO 40 I=1,NROW                                                     # J00890
      DO 40 J=1,NCOL                                                     # J00900
      QZ(J,I,1)= 0.0                                                     # J00910
      DO 40 K=1,NLAY                                                     # J00920
40    QZ(J,I,K+1)= -BUFF(J,I,K)                                          # J00930
      WRITE(I7,*) '  BCF FLOW RATES HAVE BEEN READ'                     # J00940
C                                                                        # J00950
C  MODIFY FACE FLOW FOR BOUNDARY FLUXES AND SET IBOUND FLAGS             # J00960
C  FOR THOSE CELLS WITH DISTRIBUTED SINKS AND SOURCES                    # J00970
C                                                                        # J00980
C---WELLS                                                                # J00990
      IF(IUNIT(2).NE.0) CALL WELLS (IUNIT(2),QX,QY,QZ,IBOUND,HEAD,       # J01000
     1NCOL,NROW,NLAY,NCP1,NRP1,NLP1,QSS,DELR,DELC,IA,I7,I9,IBATCH)       # J01010
C---DRAINS                                                               # J01020
      IF(IUNIT(5).NE.0) CALL RDGHB (2,IUNIT(5),QX,QY,QZ,BUFF,IBOUND,     # J01030
     1NCOL,NROW,NLAY,NCP1,NRP1,NLP1,QSS,IBUFF,DELR,DELC,IA,I7,I9,        # J01040
     2IBATCH)                                                            # J01050
C---RECHARGE                                                             # J01060
      IF(IUNIT(1).NE.0) CALL RCHRGE (IUNIT(1),QZ,BUFF,IBUFF,HEAD,        # J01070
     1IBOUND,DELR,DELC,NCOL,NROW,NLP1,NLAY,QSS,IA,I7,I9,IBATCH)          # J01080
C---ET                                                                   # J01090
      IF(IUNIT(3).NE.0) CALL EVAPO (IUNIT(3),QZ,BUFF,IBUFF,NCOL,NROW,    # J01100
     1NLAY,NLP1,QSS,IBOUND,IA,I7,I9,IBATCH)                              # J01110
C---RIVERS                                                               # J01120
      IF(IUNIT(4).NE.0) CALL RDGHB (1,IUNIT(4),QX,QY,QZ,BUFF,IBOUND,     # J01130
     1NCOL,NROW,NLAY,NCP1,NRP1,NLP1,QSS,IBUFF,DELR,DELC,IA,I7,I9,        # J01140
     2IBATCH)                                                            # J01150
C---GENERAL HEAD BOUNDARIES                                              # J01160
      IF(IUNIT(6).NE.0) CALL RDGHB (3,IUNIT(6),QX,QY,QZ,BUFF,IBOUND,     # J01170
     1NCOL,NROW,NLAY,NCP1,NRP1,NLP1,QSS,IBUFF,DELR,DELC,IA,I7,I9,        # J01180
     2IBATCH)                                                            # J01190
      WRITE(I7,*) ' '                                                    # J01200
      WRITE(I7,*) '**** FLOW RATES FROM BCF AND ALL STRESS PACKAGES HAVE# J01210
     1 BEEN READ ****'                                                   # J01220
      WRITE(I7,*) '**** INTERCELL FLOW RATES CAN BE CHECKED USING THE CE# J01230
     1LL-BY-CELL OPTION ****'                                            # J01240
      WRITE(I7,5500)                                                     # J01250
C                                                                        # J01260
      RETURN                                                             # J01270
      END                                                                # J01280
C-----END OF ROUTINE-------------------------------------------------------# J01290
```

```
C                                                                       # K00010
C---Version 1.0   July 21, 1989                                         # K00020
C***********************************************************************# K00030
C                                                                       # K00040
C                               ZERO                                    # K00050
C                                                                       # K00060
C  THIS SUBROUTINE FILLS A THREE-DIMENSIONAL ARRAY WITH ZEROS.          # K00070
C                                                                       # K00080
C***********************************************************************# K00090
C                                                                       # K00100
      SUBROUTINE ZERO(X,NJ,NI,NK)                                       # K00110
      DIMENSION X(NJ,NI,NK)                                             # K00120
      DO 10 K=1,NK                                                      # K00130
      DO 10 I=1,NI                                                      # K00140
      DO 10 J=1,NJ                                                      # K00150
10    X(J,I,K)=0.0                                                      # K00160
      RETURN                                                            # K00170
      END                                                               # K00180
C-----END OF ROUTINE--------------------------------------------------# K00190
```

```
C                                                                       # L00010
C---Version 1.0   July 21, 1989                                         # L00020
C***********************************************************************# L00030
C                                                                       # L00040
C                              RCHRGE                                   # L00050
C                                                                       # L00060
C  THIS SUBROUTINE READS THE RECHARGE DATA FILE FROM THE USGS MODULAR    # L00070
C  FINITE-DIFFERENCE FLOW MODEL.                                        # L00080
C                                                                       # L00090
C***********************************************************************# L00100
C                                                                       # L00110
      SUBROUTINE RCHRGE (IU,QZ,RECH,IRCH,HEAD,IBOUND,DELR,DELC,         # L00120
     1NCOL,NROW,NLP1,NLAY,QSS,IA,I7,I9,IBATCH)                          # L00130
C                                                                       # L00140
      DIMENSION QZ(NCOL,NROW,NLP1),RECH(NCOL,NROW,NLAY),IRCH(NCOL,NROW),# L00150
     1HEAD(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),DELR(NCOL),           # L00160
     2DELC(NROW),QSS(NCOL,NROW,NLAY)                                    # L00170
C                                                                       # L00180
      CHARACTER*80 MES                                                  # L00190
C                                                                       # L00200
      WRITE(I7,5000)                                                    # L00210
5000  FORMAT('-------------------------------------------------------------# L00220
     1----------------------')                                          # L00230
      WRITE(I7,*) 'RECHARGE DATA NOW BEING READ...'                     # L00240
      READ(IU,5010) NRCHOP,ID1,IFACE                                    # L00250
      IF (IFACE.EQ.0) THEN                                              # L00260
      MES= 'DO YOU WANT RECHARGE ASSIGNED TO THE TOP FACE OF CELLS ?'   # L00270
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # L00280
      IF (IANS.EQ.1) IFACE=1                                            # L00290
      END IF                                                            # L00300
      READ(IU,5010) ID1,ID2                                             # L00310
5010  FORMAT(3I10)                                                      # L00320
      WRITE(I7,*) 'RECHARGE ARRAY NOW BEING READ...'                    # L00330
      CALL IN2DR (IU,RECH(1,1,1),NCOL,NROW,I7)                          # L00340
      IF(NRCHOP.EQ.2) THEN                                              # L00350
      WRITE(I7,*) 'RECHARGE LAYER ARRAY NOW BEING READ...'              # L00360
      CALL IN2DI (IU,IRCH,NCOL,NROW,I7)                                 # L00370
      END IF                                                            # L00380
      IF(NRCHOP.EQ.1) THEN                                              # L00390
      DO 10 I=1,NROW                                                    # L00400
      DO 10 J=1,NCOL                                                    # L00410
      IF(IBOUND(J,I,1).LE.0.OR.HEAD(J,I,1).GT.1.0E+29) GO TO 10         # L00420
      IF(IFACE.GT.0) THEN                                               # L00430
      QZ(J,I,1)= QZ(J,I,1) - RECH(J,I,1)*DELR(J)*DELC(I)                # L00440
      ELSE                                                              # L00450
      QSS(J,I,1)=QSS(J,I,1) + RECH(J,I,1)*DELR(J)*DELC(I)               # L00460
      END IF                                                            # L00470
10    CONTINUE                                                          # L00480
      END IF                                                            # L00490
      IF(NRCHOP.EQ.2) THEN                                              # L00500
      DO 20 I=1,NROW                                                    # L00510
      DO 20 J=1,NCOL                                                    # L00520
      K= IRCH(J,I)                                                      # L00530
      IF(IBOUND(J,I,K).LE.0.OR.HEAD(J,I,K).GT.1.0E+29) GO TO 20         # L00540
      IF(IFACE.GT.0) THEN                                               # L00550
      CALL FACTYP (K,1,J,I,K-1,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,6,5,I7)  # L00560
      IF (NBD.EQ.1) THEN                                                # L00570
      QZ(J,I,K)= QZ(J,I,K) - RECH(J,I,1)*DELR(J)*DELC(I)                # L00580
      ELSE                                                              # L00590
      QSS(J,I,K)=QSS(J,I,K) + RECH(J,I,1)*DELR(J)*DELC(I)               # L00600
      END IF                                                            # L00610
      ELSE                                                              # L00620
      QSS(J,I,K)=QSS(J,I,K) + RECH(J,I,1)*DELR(J)*DELC(I)               # L00630
      END IF                                                            # L00640
20    CONTINUE                                                          # L00650
      END IF                                                            # L00660
      IF(NRCHOP.EQ.3) THEN                                              # L00670
      DO 50 I=1,NROW                                                    # L00680
      DO 50 J=1,NCOL                                                    # L00690
      DO 30 K=1,NLAY                                                    # L00700
      KK=K                                                              # L00710
      IF(IBOUND(J,I,K).LT.0) GO TO 50                                   # L00720
```

```
      IF(IBOUND(J,I,K).EQ.0.OR.HEAD(J,I,K).GT.1.0E+29) GO TO 30          # L00730
      GO TO 40                                                           # L00740
30    CONTINUE                                                           # L00750
      GO TO 50                                                           # L00760
40    CONTINUE                                                           # L00770
      IF(IFACE.GT.0) THEN                                                # L00780
      QZ(J,I,KK)= QZ(J,I,KK) - RECH(J,I,1)*DELC(I)*DELR(J)               # L00790
      ELSE                                                               # L00800
      QSS(J,I,KK)=QSS(J,I,KK) + RECH(J,I,1)*DELC(I)*DELR(J)              # L00810
      END IF                                                             # L00820
50    CONTINUE                                                           # L00830
      END IF                                                             # L00840
C                                                                        # L00850
      RETURN                                                             # L00860
      END                                                                # L00870
C-----END OF ROUTINE-------------------------------------------------------# L00880
```

```
C                                                                       # M00010
C---Version 1.0   July 21, 1989                                         # M00020
C***********************************************************************# M00030
C                                                                       # M00040
C                                  WELLS                                # M00050
C                                                                       # M00060
C  THIS SUBROUTINE READS THE WELL PACKAGE DATA FILE FROM THE USGS       # M00070
C  MODULAR FINITE-DIFFERENCE FLOW MODEL.                                # M00080
C                                                                       # M00090
C***********************************************************************# M00100
C                                                                       # M00110
      SUBROUTINE WELLS (IU,QX,QY,QZ,IBOUND,HEAD,NCOL,NROW,NLAY,         # M00120
     1NCP1,NRP1,NLP1,QSS,DELR,DELC,IA,I7,I9,IBATCH)                     # M00130
C                                                                       # M00140
      DIMENSION QX(NCP1,NROW,NLAY),QY(NCOL,NRP1,NLAY),                  # M00150
     1QZ(NCOL,NROW,NLP1),IBOUND(NCOL,NROW,NLAY),HEAD(NCOL,NROW,NLAY),   # M00160
     2QSS(NCOL,NROW,NLAY),DELR(NCOL),DELC(NROW)                         # M00170
      CHARACTER*80 MES                                                  # M00180
C                                                                       # M00190
      WRITE(I7,5000)                                                    # M00200
5000  FORMAT('-------------------------------------------------------------# M00210
     1----------------------')                                          # M00220
      WRITE(I7,*) 'WELL DATA NOW BEING READ...'                         # M00230
      READ(IU,5010) ID1,ID2                                             # M00240
      READ(IU,5010) ITMP                                                # M00250
5010   FORMAT (2I10)                                                    # M00260
      IF(ITMP.LE.0) RETURN                                              # M00270
      IERR=0                                                            # M00280
      DO 10 N=1,ITMP                                                    # M00290
      READ(IU,5050) K,I,J,Q,IFACE                                       # M00300
      IB=IBOUND(J,I,K)                                                  # M00310
      IF (IB.LE.0) THEN                                                 # M00320
      IF (IB.EQ.0) WRITE (I7,5020) J,I,K                                # M00330
      IF (IB.LT.0) WRITE (I7,5030) J,I,K                                # M00340
5020  FORMAT ('A WELL WAS SPECIFIED FOR INACTIVE CELL (J,I,K) = (',I3,  # M00350
     1',',I3,',',I3,')')                                                # M00360
5030  FORMAT ('A WELL WAS SPECIFIED FOR CONSTANT HEAD CELL (J,I,K) = (',# M00370
     1I3,',',I3,',',I3,')')                                             # M00380
      IERR=IERR+1                                                       # M00390
      END IF                                                            # M00400
      IF (IFACE.LT.0) THEN                                              # M00410
      CALL FLUXBC (Q,QSS,QX,QY,IBOUND,DELR,DELC,J,I,K,NCOL,NROW,NLAY,   # M00420
     1 NCP1,NRP1)                                                       # M00430
      ELSE IF(IFACE.EQ.0.OR.IFACE.GT.6) THEN                            # M00440
      IF(Q.LE.0.0.AND.ABS(IB).LT.1000) IBOUND(J,I,K)= 1000*IB           # M00450
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # M00460
      ELSE IF(IFACE.EQ.1) THEN                                          # M00470
      CALL FACTYP (J,1,J-1,I,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,4, # M00480
     1 I7)                                                              # M00490
      IF (NBD.EQ.1) THEN                                                # M00500
      QX(J,I,K)= QX(J,I,K) + Q                                          # M00510
      ELSE                                                              # M00520
      IF(Q.LE.0.0.AND.ABS(IB).LT.1000) IBOUND(J,I,K)= 1000*IB           # M00530
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # M00540
      END IF                                                            # M00550
      ELSE IF(IFACE.EQ.2) THEN                                          # M00560
      CALL FACTYP (J,NCOL,J+1,I,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,# M00570
     1 4,I7)                                                            # M00580
      IF (NBD.EQ.1) THEN                                                # M00590
      QX(J+1,I,K)= QX(J+1,I,K) - Q                                      # M00600
      ELSE                                                              # M00610
      IF(Q.LE.0.0.AND.ABS(IB).LT.1000) IBOUND(J,I,K)= 1000*IB           # M00620
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # M00630
      END IF                                                            # M00640
      ELSE IF(IFACE.EQ.3) THEN                                          # M00650
      CALL FACTYP (I,NROW,J,I+1,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,# M00660
     1 4,I7)                                                            # M00670
      IF (NBD.EQ.1) THEN                                                # M00680
      QY(J,I+1,K)= QY(J,I+1,K) + Q                                      # M00690
      ELSE                                                              # M00700
      IF(Q.LE.0.0.AND.ABS(IB).LT.1000) IBOUND(J,I,K)= 1000*IB           # M00710
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # M00720
```

```
      END IF                                                            # M00730
      ELSE IF(IFACE.EQ.4) THEN                                          # M00740
      CALL FACTYP (I,1,J,I-1,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,4, # M00750
     1 I7)                                                              # M00760
      IF (NBD.EQ.1) THEN                                                # M00770
      QY(J,I,K)= QY(J,I,K) - Q                                          # M00780
      ELSE                                                              # M00790
      IF(Q.LE.0.0.AND.ABS(IB).LT.1000) IBOUND(J,I,K)= 1000*IB           # M00800
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # M00810
      END IF                                                            # M00820
      ELSE IF(IFACE.EQ.5) THEN                                          # M00830
      CALL FACTYP (K,NLAY,J,I,K+1,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,# M00840
     1 4,I7)                                                            # M00850
      IF (NBD.EQ.1) THEN                                                # M00860
      QZ(J,I,K+1)= QZ(J,I,K+1) + Q                                      # M00870
      ELSE                                                              # M00880
      IF(Q.LE.0.0.AND.ABS(IB).LT.1000) IBOUND(J,I,K)= 1000*IB           # M00890
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # M00900
      END IF                                                            # M00910
      ELSE IF(IFACE.EQ.6) THEN                                          # M00920
      CALL FACTYP (K,1,J,I,K-1,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,4, # M00930
     1 I7)                                                              # M00940
      IF (NBD.EQ.1) THEN                                                # M00950
      QZ(J,I,K)= QZ(J,I,K) - Q                                          # M00960
      ELSE                                                              # M00970
      IF(Q.LE.0.0.AND.ABS(IB).LT.1000) IBOUND(J,I,K)= 1000*IB           # M00980
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # M00990
      END IF                                                            # M01000
      END IF                                                            # M01010
10    CONTINUE                                                          # M01020
      WRITE(I7,*) '  WELL DATA HAS BEEN READ'                          # M01030
      IF (IERR.GT.0) THEN                                               # M01040
      IF (IBATCH.EQ.0) THEN                                             # M01050
      WRITE (*,5040) IERR                                               # M01060
5040  FORMAT ('WELLS WERE SPECIFIED FOR ',I4,' INACTIVE OR CONSTANT HEAD# M01070
     1 CELLS')                                                          # M01080
      WRITE (*,*) 'A LIST IS PROVIDED IN THE "SUMMARY.PTH"  FILE'       # M01090
      END IF                                                            # M01100
      MES= 'DO YOU WANT TO CONTINUE ?'                                  # M01110
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # M01120
      IF(IANS.EQ.0) STOP                                                # M01130
      END IF                                                            # M01140
      RETURN                                                            # M01150
5050   FORMAT(3I10,F10.0,I10)                                           # M01160
      END                                                               # M01170
C-----END OF ROUTINE-------------------------------------------------------# M01180
```

```
C                                                                       # N00010
C---Version 1.0   July 21, 1989                                         # N00020
C***********************************************************************# N00030
C                                                                       # N00040
C                                EVAPO                                  # N00050
C                                                                       # N00060
C  THIS SUBROUTINE READS THE EVAPOTRANSPIRATION DATA FILE FOR THE USGS  # N00070
C  MODULAR FINITE-DIFFERENCE FLOW MODEL.  IT ALSO CALLS A SUBROUTINE    # N00080
C  TO READ THE BUDGET FILE FOR ET.                                      # N00090
C                                                                       # N00100
C***********************************************************************# N00110
C                                                                       # N00120
      SUBROUTINE EVAPO (IU,QZ,BUFF,IEVT,NCOL,NROW,NLAY,NLP1,QSS,IBOUND, # N00130
     1IA,I7,I9,IBATCH)                                                  # N00140
C                                                                       # N00150
      DIMENSION QZ(NCOL,NROW,NLP1),BUFF(NCOL,NROW,NLAY),                # N00160
     1IEVT(NCOL,NROW),QSS(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY)        # N00170
      CHARACTER*16 LABEL                                                # N00180
C                                                                       # N00190
      CHARACTER*80 MES                                                  # N00200
C                                                                       # N00210
      LABEL= '              ET'                                         # N00220
      WRITE(I7,5000)                                                    # N00230
5000  FORMAT('-------------------------------------------------------#  N00240
     1----------------------')                                          # N00250
      WRITE(I7,*) 'EVAPOTRANSPIRATION DATA NOW BEING READ...'           # N00260
      READ(IU,5010) NEVTOP,IEVTCB,IFACE                                 # N00270
      IF (IFACE.EQ.0) THEN                                              # N00280
      MES= 'DO YOU WANT TO ASSIGN EVAPOTRANSPIRATION TO THE TOP FACE OF # N00290
     1CELLS ?'                                                          # N00300
      CALL YESNO (IBATCH,I9,MES,IA,IANS)                                # N00310
      IF (IANS.EQ.1) IFACE=1                                            # N00320
      END IF                                                            # N00330
      READ(IU,5010) INSURF,INEVTR,INEXDP,INIEVT                         # N00340
      IF (INSURF.GE.0) CALL IN2DR (IU,BUFF,NCOL,NROW,I7)                # N00350
      IF (INEVTR.GE.0) CALL IN2DR (IU,BUFF,NCOL,NROW,I7)                # N00360
      IF (INEVTR.GE.0) CALL IN2DR (IU,BUFF,NCOL,NROW,I7)                # N00370
      IF (INIEVT.GE.0.AND.NEVTOP.EQ.2) CALL IN2DR(IU,BUFF,NCOL,NROW,I7) # N00380
5010   FORMAT (4I10)                                                    # N00390
      WRITE(I7,*) 'ET RATES NOW BEING READ...'                          # N00400
C---READ FLOW ET FLOW RATES INTO BUFFER                                 # N00410
      CALL RDBUDG (BUFF,LABEL,NCOL,NROW,NLAY,IEVTCB,I7)                 # N00420
C                                                                       # N00430
      WRITE(I7,*) '  ET RATES HAVE BEEN READ'                           # N00440
      IF(NEVTOP.EQ.2) THEN                                              # N00450
      WRITE(I7,*) 'ET LAYER NUMBER ARRAY NOW BEING READ...'             # N00460
      CALL IN2DI (IU,IEVT,NCOL,NROW,I7)                                 # N00470
      DO 10 I=1,NROW                                                    # N00480
      DO 10 J=1,NCOL                                                    # N00490
      K=IEVT(J,I)                                                       # N00500
      IF(IFACE.GT.0) THEN                                               # N00510
      CALL FACTYP (K,1,J,I,K-1,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,6,6,I7)  # N00520
      IF (NBD.EQ.1) THEN                                                # N00530
      QZ(J,I,K)= QZ(J,I,K) - BUFF(J,I,K)                                # N00540
      ELSE                                                              # N00550
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                               # N00560
      IF(BUFF(J,I,K).LE.0.0.AND.IBOUND(J,I,K).LT.1000)                  # N00570
     1 IBOUND(J,I,K)= 1000*IBOUND(J,I,K)                                # N00580
      END IF                                                            # N00590
      ELSE                                                              # N00600
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                               # N00610
      IF(BUFF(J,I,K).LE.0.0.AND.IBOUND(J,I,K).LT.1000)                  # N00620
     1 IBOUND(J,I,K)= 1000*IBOUND(J,I,K)                                # N00630
      END IF                                                            # N00640
10    CONTINUE                                                          # N00650
      ELSE                                                              # N00660
      DO 20 I=1,NROW                                                    # N00670
      DO 20 J=1,NCOL                                                    # N00680
      IF(IFACE.GT.0) THEN                                               # N00690
      QZ(J,I,1)= QZ(J,I,1) - BUFF(J,I,1)                                # N00700
      ELSE                                                              # N00710
      QSS(J,I,1)=QSS(J,I,1) + BUFF(J,I,1)                               # N00720
```

```
      IF(BUFF(J,I,1).LE.0.0.AND.IBOUND(J,I,1).LT.1000)                  # N00730
     1 IBOUND(J,I,1)= 1000*IBOUND(J,I,1)                                # N00740
      END IF                                                            # N00750
20    CONTINUE                                                          # N00760
      END IF                                                            # N00770
      RETURN                                                            # N00780
      END                                                               # N00790
C-----END OF ROUTINE-----------------------------------------------------# N00800
```

```
C                                                                       # O00010
C---Version 1.0   July 21, 1989                                         # O00020
C***********************************************************************# O00030
C                                                                       # O00040
C                                RDGHB                                  # O00050
C                                                                       # O00060
C  THIS SUBROUTINE READS THE DATA FILES FOR THE RIVER, DRAIN, AND       # O00070
C  GENERAL HEAD BOUNDARY PACKAGES FOR THE USGS MODULAR FINITE-DIFFERENCE# O00080
C  FLOW MODEL. IT ALSO READS THE BUDGET FILES FOR THOSE STRESS PACKAGES.# O00090
C                                                                       # O00100
C***********************************************************************# O00110
C                                                                       # O00120
      SUBROUTINE RDGHB (IPK,IU,QX,QY,QZ,BUFF,IBOUND,NCOL,NROW,NLAY,     # O00130
     1NCP1,NRP1,NLP1,QSS,IBUFF,DELR,DELC,IA,I7,I9,IBATCH)               # O00140
C                                                                       # O00150
      DIMENSION QX(NCP1,NROW,NLAY),QY(NCOL,NRP1,NLAY),                  # O00160
     1QZ(NCOL,NROW,NLP1),IBOUND(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY),   # O00170
     2QSS(NCOL,NROW,NLAY),IBUFF(NCOL,NROW,NLAY),DELR(NCOL),DELC(NROW)   # O00180
      CHARACTER*16 LABEL                                                # O00190
C                                                                       # O00200
      WRITE(I7,5000)                                                    # O00210
5000  FORMAT('-------------------------------------------------------#  # O00220
     1----------------------')                                          # O00230
      IF(IPK.EQ.1) THEN                                                 # O00240
      WRITE(I7,*) 'RIVER DATA NOW BEING READ...'                        # O00250
      LABEL= '   RIVER LEAKAGE'                                         # O00260
      ELSE IF(IPK.EQ.2) THEN                                            # O00270
      WRITE(I7,*) 'DRAIN DATA NOW BEING READ...'                        # O00280
      LABEL= '           DRAINS'                                        # O00290
      ELSE IF(IPK.EQ.3) THEN                                            # O00300
      WRITE(I7,*) 'GENERAL HEAD BOUNDARY DATA NOW BEING READ...'        # O00310
      LABEL= ' HEAD DEP BOUNDS'                                         # O00320
      END IF                                                            # O00330
      READ(IU,5010) MX,ICB                                              # O00340
      READ(IU,5010) ITMP                                                # O00350
5010   FORMAT (2I10)                                                    # O00360
      IF(ITMP.LE.0) RETURN                                              # O00370
      DO 10 K=1,NLAY                                                    # O00380
      DO 10 I=1,NROW                                                    # O00390
      DO 10 J=1,NCOL                                                    # O00400
10     IBUFF(J,I,K)=0                                                   # O00410
      WRITE(I7,*) '  FLOW RATES NOW BEING READ...'                      # O00420
      CALL RDBUDG (BUFF,LABEL,NCOL,NROW,NLAY,ICB,I7)                    # O00430
      WRITE(I7,*) '    FLOW RATES HAVE BEEN READ'                       # O00440
      DO 20 N=1,ITMP                                                    # O00450
      IF(IPK.EQ.1) THEN                                                 # O00460
      READ(IU,5020) K,I,J,H,C,E,IFACE                                   # O00470
5020   FORMAT(3I10,3F10.0,I10)                                          # O00480
      IF (IBOUND(J,I,K).EQ.0) WRITE (I7,5030) J,I,K                     # O00490
5030   FORMAT ('A RIVER WAS SPECIFIED FOR INACTIVE CELL (J,I,K) =',      # O00500
     12(I4,','),I4)                                                     # O00510
      ELSE                                                              # O00520
      READ(IU,5050) K,I,J,H,C,IFACE                                     # O00530
      IF (IBOUND(J,I,K).EQ.0) WRITE (I7,5040) J,I,K                     # O00540
5040   FORMAT('A DRAIN OR GHB WAS SPECIFIED FOR INACTIVE CELL (J,I,K)=',# O00550
     12(I4,','),I4)                                                     # O00560
5050   FORMAT(3I10,2F10.0,I10)                                          # O00570
      END IF                                                            # O00580
      IF(IBUFF(J,I,K).EQ.1) GO TO 20                                    # O00590
      IBUFF(J,I,K)=1                                                    # O00600
      IB=IBOUND(J,I,K)                                                  # O00610
      IF (IFACE.LT.0) THEN                                              # O00620
      CALL FLUXBC (BUFF(J,I,K),QSS,QX,QY,IBOUND,DELR,DELC,J,I,K,        # O00630
     1 NCOL,NROW,NLAY,NCP1,NRP1)                                        # O00640
      ELSE IF(IFACE.EQ.0.OR.IFACE.GT.6) THEN                            # O00650
      IF(BUFF(J,I,K).LE.0.0.AND.ABS(IB).LT.1000)IBOUND(J,I,K)=1000*IB   # O00660
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                               # O00670
      ELSE IF(IFACE.EQ.1) THEN                                          # O00680
      JM1=J-1                                                           # O00690
      CALL FACTYP (J,1,J-1,I,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,   # O00700
     1 IPK,I7)                                                          # O00710
      IF (NBD.EQ.1) THEN                                                # O00720
```

```
      QX(J,I,K)= QX(J,I,K) + BUFF(J,I,K)                                    # O00730
      ELSE                                                                  # O00740
      IF(BUFF(J,I,K).LE.0.0.AND.ABS(IB).LT.1000)IBOUND(J,I,K)= 1000*IB      # O00750
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                                   # O00760
      END IF                                                                # O00770
      ELSE IF(IFACE.EQ.2) THEN                                              # O00780
      CALL FACTYP (J,NCOL,J+1,I,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,    # O00790
     1 IPK,I7)                                                              # O00800
      IF (NBD.EQ.1) THEN                                                    # O00810
      QX(J+1,I,K)= QX(J+1,I,K) - BUFF(J,I,K)                                # O00820
      ELSE                                                                  # O00830
      IF(BUFF(J,I,K).LE.0.0.AND.ABS(IB).LT.1000)IBOUND(J,I,K)= 1000*IB      # O00840
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                                   # O00850
      END IF                                                                # O00860
      ELSE IF(IFACE.EQ.3) THEN                                              # O00870
      CALL FACTYP (I,NROW,J,I+1,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,    # O00880
     1 IPK,I7)                                                              # O00890
      IF (NBD.EQ.1) THEN                                                    # O00900
      QY(J,I+1,K)= QY(J,I+1,K) + BUFF(J,I,K)                                # O00910
      ELSE                                                                  # O00920
      IF(BUFF(J,I,K).LE.0.0.AND.ABS(IB).LT.1000)IBOUND(J,I,K)= 1000*IB      # O00930
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                                   # O00940
      END IF                                                                # O00950
      ELSE IF(IFACE.EQ.4) THEN                                              # O00960
      CALL FACTYP (I,1,J,I-1,K,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,       # O00970
     1 IPK,I7)                                                              # O00980
      IF (NBD.EQ.1) THEN                                                    # O00990
      QY(J,I,K)= QY(J,I,K) - BUFF(J,I,K)                                    # O01000
      ELSE                                                                  # O01010
      IF(BUFF(J,I,K).LE.0.0.AND.ABS(IB).LT.1000)IBOUND(J,I,K)= 1000*IB      # O01020
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                                   # O01030
      END IF                                                                # O01040
      ELSE IF(IFACE.EQ.5) THEN                                              # O01050
      CALL FACTYP (K,NLAY,J,I,K+1,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,    # O01060
     1 IPK,I7)                                                              # O01070
      IF (NBD.EQ.1) THEN                                                    # O01080
      QZ(J,I,K+1)= QZ(J,I,K+1) + BUFF(J,I,K)                                # O01090
      ELSE                                                                  # O01100
      IF(BUFF(J,I,K).LE.0.0.AND.ABS(IB).LT.1000)IBOUND(J,I,K)= 1000*IB      # O01110
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                                   # O01120
      END IF                                                                # O01130
      ELSE IF(IFACE.EQ.6) THEN                                              # O01140
      CALL FACTYP (K,1,J,I,K-1,NBD,IBOUND,NCOL,NROW,NLAY,J,I,K,IFACE,       # O01150
     1 IPK,I7)                                                              # O01160
      IF (NBD.EQ.1) THEN                                                    # O01170
      QZ(J,I,K)= QZ(J,I,K) - BUFF(J,I,K)                                    # O01180
      ELSE                                                                  # O01190
      IF(BUFF(J,I,K).LE.0.0.AND.ABS(IB).LT.1000)IBOUND(J,I,K)= 1000*IB      # O01200
      QSS(J,I,K)=QSS(J,I,K) + BUFF(J,I,K)                                   # O01210
      END IF                                                                # O01220
      END IF                                                                # O01230
20    CONTINUE                                                              # O01240
      IF(IPK.EQ.1) THEN                                                     # O01250
      WRITE(I7,*) '  RIVER DATA HAS BEEN READ'                              # O01260
      ELSE IF(IPK.EQ.2) THEN                                                # O01270
      WRITE(I7,*) '  DRAIN DATA HAS BEEN READ'                              # O01280
      ELSE IF(IPK.EQ.3) THEN                                                # O01290
      WRITE(I7,*) '  GENERAL HEAD BOUNDARY DATA HAS BEEN READ'              # O01300
      END IF                                                                # O01310
      RETURN                                                                # O01320
      END                                                                   # O01330
C-----END OF ROUTINE-------------------------------------------------------# O01340
```

```
C                                                                       # P00010
C---Version 1.0   July 21, 1989                                         # P00020
C***********************************************************************# P00030
C                                                                       # P00040
C                              IN2DR                                    # P00050
C                                                                       # P00060
C  THIS SUBROUTINE READS A TWO DIMENSIONAL ARRAY OF REAL NUMBERS.       # P00070
C                                                                       # P00080
C***********************************************************************# P00090
C                                                                       # P00100
      SUBROUTINE IN2DR (IU,X,JJ,II,I7)                                  # P00110
      DIMENSION X(JJ,II)                                                # P00120
      CHARACTER*20 FMT                                                  # P00130
C                                                                       # P00140
      READ(IU,5000) LOCAT,CNSTNT,FMT,IPRN                               # P00150
5000  FORMAT(I10,F10.0,A20,I10)                                         # P00160
      IF(LOCAT.LE.0) THEN                                               # P00170
      DO 10 I=1,II                                                      # P00180
      DO 10 J=1,JJ                                                      # P00190
10    X(J,I)= CNSTNT                                                    # P00200
      WRITE(I7,5010) CNSTNT                                             # P00210
5010  FORMAT('  CONSTANT VALUE OF',E13.5)                              # P00220
      RETURN                                                            # P00230
      END IF                                                            # P00240
C                                                                       # P00250
      IF(FMT.EQ.'                    ') THEN                            # P00260
      DO 20 I=1,II                                                      # P00270
      READ(LOCAT,*) (X(J,I),J=1,JJ)                                     # P00280
20    CONTINUE                                                          # P00290
      ELSE                                                              # P00300
      DO 30 I=1,II                                                      # P00310
      READ(LOCAT,FMT) (X(J,I),J=1,JJ)                                   # P00320
30    CONTINUE                                                          # P00330
      END IF                                                            # P00340
      DO 40 I=1,II                                                      # P00350
      DO 40 J=1,JJ                                                      # P00360
40    X(J,I)= CNSTNT*X(J,I)                                             # P00370
      IF(IPRN.GT.0) THEN                                                # P00380
      DO 50 I=1,II                                                      # P00390
      WRITE(I7,5020) (X(J,I),J=1,JJ)                                    # P00400
50    CONTINUE                                                          # P00410
      ELSE                                                              # P00420
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # P00430
      END IF                                                            # P00440
5020  FORMAT(10E13.5)                                                   # P00450
C                                                                       # P00460
      RETURN                                                            # P00470
      END                                                               # P00480
C-----END OF ROUTINE-------------------------------------------------------# P00490
```

```
C                                                                       # Q00010
C---Version 1.0   July 21, 1989                                         # Q00020
C***********************************************************************# Q00030
C                                                                       # Q00040
C                              IN1DR                                    # Q00050
C                                                                       # Q00060
C  THIS SUBROUTINE READS A ONE DIMENSIONAL ARRAY OF REAL NUMBERS.       # Q00070
C                                                                       # Q00080
C***********************************************************************# Q00090
C                                                                       # Q00100
      SUBROUTINE IN1DR (IU,X,JJ,I7)                                     # Q00110
      DIMENSION X(JJ)                                                   # Q00120
      CHARACTER*20 FMT                                                  # Q00130
C                                                                       # Q00140
      READ(IU,5000) LOCAT,CNSTNT,FMT,IPRN                               # Q00150
5000  FORMAT(I10,F10.0,A20,I10)                                         # Q00160
      IF(LOCAT.LE.0) THEN                                               # Q00170
      DO 10 J=1,JJ                                                      # Q00180
10    X(J)= CNSTNT                                                      # Q00190
      WRITE(I7,5010) CNSTNT                                             # Q00200
5010  FORMAT('  CONSTANT VALUE OF',E13.5)                              # Q00210
      RETURN                                                            # Q00220
      END IF                                                            # Q00230
C                                                                       # Q00240
      IF(FMT.EQ.'                    ') THEN                            # Q00250
      READ(LOCAT,*) (X(J),J=1,JJ)                                       # Q00260
      ELSE                                                              # Q00270
      READ(LOCAT,FMT) (X(J),J=1,JJ)                                     # Q00280
      END IF                                                            # Q00290
      DO 20 J=1,JJ                                                      # Q00300
20    X(J)= CNSTNT*X(J)                                                 # Q00310
C                                                                       # Q00320
      IF(IPRN.GT.0) THEN                                                # Q00330
      WRITE(I7,5020) (X(J),J=1,JJ)                                      # Q00340
5020  FORMAT(10E13.5)                                                   # Q00350
      ELSE                                                              # Q00360
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # Q00370
      END IF                                                            # Q00380
      RETURN                                                            # Q00390
      END                                                               # Q00400
C-----END OF ROUTINE-------------------------------------------------------# Q00410
```

```
C                                                                       # R00010
C---Version 1.0   July 21, 1989                                         # R00020
C***********************************************************************# R00030
C                                                                       # R00040
C                              IN2DI                                    # R00050
C                                                                       # R00060
C  THIS SUBROUTINE READS A TWO DIMENSIONAL ARRAY OF INTEGERS.           # R00070
C                                                                       # R00080
C***********************************************************************# R00090
C                                                                       # R00100
      SUBROUTINE IN2DI (IU,N,JJ,II,I7)                                  # R00110
      DIMENSION N(JJ,II)                                                # R00120
      CHARACTER*20 FMT                                                  # R00130
C                                                                       # R00140
      READ(IU,5000) LOCAT,ICON,FMT,IPRN                                 # R00150
5000  FORMAT(2I10,A20,I10)                                              # R00160
      IF(LOCAT.LE.0) THEN                                               # R00170
      DO 10 I=1,II                                                      # R00180
      DO 10 J=1,JJ                                                      # R00190
10    N(J,I)= ICON                                                      # R00200
      WRITE(I7,5010) ICON                                               # R00210
5010  FORMAT('  CONSTANT VALUE OF',I10)                                 # R00220
      RETURN                                                            # R00230
      END IF                                                            # R00240
C                                                                       # R00250
      IF(FMT.EQ.'                    ') THEN                            # R00260
      DO 20 I=1,II                                                      # R00270
      READ(LOCAT,*) (N(J,I),J=1,JJ)                                     # R00280
20    CONTINUE                                                          # R00290
      ELSE                                                              # R00300
      DO 30 I=1,II                                                      # R00310
      READ(LOCAT,FMT) (N(J,I),J=1,JJ)                                   # R00320
30    CONTINUE                                                          # R00330
      END IF                                                            # R00340
      DO 40 I=1,II                                                      # R00350
      DO 40 J=1,JJ                                                      # R00360
40    N(J,I)= ICON*N(J,I)                                               # R00370
      IF(IPRN.GT.0) THEN                                                # R00380
      DO 50 I=1,II                                                      # R00390
      WRITE(I7,5020) (N(J,I),J=1,JJ)                                    # R00400
50    CONTINUE                                                          # R00410
      ELSE                                                              # R00420
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # R00430
      END IF                                                            # R00440
5020  FORMAT(25I5)                                                      # R00450
C                                                                       # R00460
      RETURN                                                            # R00470
      END                                                               # R00480
C-----END OF ROUTINE-------------------------------------------------------# R00490
```

```
C                                                                       # S00010
C---Version 1.0   July 21, 1989                                         # S00020
C***********************************************************************# S00030
C                                                                       # S00040
C                              IN1DI                                    # S00050
C                                                                       # S00060
C  THIS SUBROUTINE READS A ONE DIMENSIONAL ARRAY OF INTEGERS.           # S00070
C                                                                       # S00080
C***********************************************************************# S00090
C                                                                       # S00100
      SUBROUTINE IN1DI (IU,N,JJ,I7)                                     # S00110
      DIMENSION N(JJ)                                                   # S00120
      CHARACTER*20 FMT                                                  # S00130
C                                                                       # S00140
      READ(IU,5000) LOCAT,ICON,FMT,IPRN                                 # S00150
5000  FORMAT(2I10,A20,I10)                                              # S00160
      IF(LOCAT.LE.0) THEN                                               # S00170
      DO 10 J=1,JJ                                                      # S00180
10    N(J)= ICON                                                        # S00190
      WRITE(I7,5010) ICON                                               # S00200
5010  FORMAT('  CONSTANT VALUE OF',I10)                                 # S00210
      RETURN                                                            # S00220
      END IF                                                            # S00230
C                                                                       # S00240
      IF(FMT.EQ.'                    ') THEN                            # S00250
      READ(LOCAT,*) (N(J),J=1,JJ)                                       # S00260
      ELSE                                                              # S00270
      READ(LOCAT,FMT) (N(J),J=1,JJ)                                     # S00280
      END IF                                                            # S00290
      DO 20 J=1,JJ                                                      # S00300
20    N(J)= ICON*N(J)                                                   # S00310
      IF(IPRN.GT.0) THEN                                                # S00320
      WRITE(I7,5020) (N(J),J=1,JJ)                                      # S00330
      ELSE                                                              # S00340
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # S00350
      END IF                                                            # S00360
5020  FORMAT(25I5)                                                      # S00370
C                                                                       # S00380
      RETURN                                                            # S00390
      END                                                               # S00400
C-----END OF ROUTINE--------------------------------------------------- # S00410
```

```
C                                                                       # T00010
C---Version 1.0   July 21, 1989                                         # T00020
C***********************************************************************# T00030
C                                                                       # T00040
C                              SPACE                                    # T00050
C                                                                       # T00060
C  THIS SUBROUTINE ALLOCATES SPACE FOR ARRAYS AND CALCULATES POINTERS   # T00070
C  FOR THE MASTER ARRAY.                                                # T00080
C                                                                       # T00090
C***********************************************************************# T00100
C                                                                       # T00110
      SUBROUTINE SPACE(LENA,NCOL,NROW,NLAY,NCBL,IGRID,NLPOR,NZDIM,NUNIT,# T00120
     1LCQX,LCQY,LCQZ,LCPOR,LCIBOU,LCXMAX,LCDX,LCYMAX,LCDY,LCZBOT,LCZTOP,# T00130
     2LCDZ,LCHEAD,LCBUFF,LCLAYC,LCNCON,LCDZCB,LCIBUF,LCIUN,LCXLC,LCQSS, # T00140
     3LCYLC,LCZLC,LCZLL,LCTOT,LCJLC,LCILC,LCKLC,NPART,NCP1,NRP1,NLP1,IA,# T00150
     4I1,I7,I9,IBATCH)                                                  # T00160
C                                                                       # T00170
      CHARACTER*80 FNAME                                                # T00180
      CHARACTER*80 MES                                                  # T00190
C                                                                       # T00200
      IU=I1                                                             # T00210
      NUNIT=8                                                           # T00220
C                                                                       # T00230
C  ENTER DIMENSION DATA                                                 # T00240
C                                                                       # T00250
      READ(IU,5000) NCOL,NROW,NLAY,NCBL,IGRID                           # T00260
5000   FORMAT(8I10)                                                     # T00270
C                                                                       # T00280
      NCRL=NCOL*NROW*NLAY                                               # T00290
      NCP1=NCOL+1                                                       # T00300
      NRP1=NROW+1                                                       # T00310
      NLP1=NLAY+1                                                       # T00320
      NLPOR= NLAY+NCBL                                                  # T00330
C                                                                       # T00340
      IF(IGRID.EQ.1) THEN                                               # T00350
      NZDIM=NLAY                                                        # T00360
      ELSE                                                              # T00370
      NZDIM=NCRL                                                        # T00380
      END IF                                                            # T00390
C                                                                       # T00400
      ISUM=1                                                            # T00410
      LCQX=ISUM                                                         # T00420
      ISUM=ISUM+NCRL+(NROW*NLAY)                                        # T00430
      LCQY=ISUM                                                         # T00440
      ISUM=ISUM+NCRL+(NCOL*NLAY)                                        # T00450
      LCQZ=ISUM                                                         # T00460
      ISUM=ISUM+NCRL+(NCOL*NROW)                                        # T00470
      LCPOR=ISUM                                                        # T00480
      ISUM=ISUM+(NCOL*NROW*NLPOR)                                       # T00490
      LCIBOU=ISUM                                                       # T00500
      ISUM=ISUM+NCRL                                                    # T00510
      LCXMAX=ISUM                                                       # T00520
      ISUM=ISUM+NCOL                                                    # T00530
      LCDX=ISUM                                                         # T00540
      ISUM=ISUM+NCOL                                                    # T00550
      LCYMAX=ISUM                                                       # T00560
      ISUM=ISUM+NROW                                                    # T00570
      LCDY=ISUM                                                         # T00580
      ISUM=ISUM+NROW                                                    # T00590
      LCZBOT=ISUM                                                       # T00600
      ISUM=ISUM+NZDIM                                                   # T00610
      LCZTOP=ISUM                                                       # T00620
      ISUM=ISUM+NZDIM                                                   # T00630
      LCDZ=ISUM                                                         # T00640
      ISUM=ISUM+NLAY                                                    # T00650
      LCHEAD=ISUM                                                       # T00660
      ISUM=ISUM+NCRL                                                    # T00670
      LCBUFF=ISUM                                                       # T00680
      ISUM=ISUM+NCRL                                                    # T00690
      LCLAYC=ISUM                                                       # T00700
      ISUM=ISUM+NLAY                                                    # T00710
      LCNCON=ISUM                                                       # T00720
```

```
      ISUM=ISUM+NLAY                                                    # T00730
      LCDZCB=ISUM                                                       # T00740
      ISUM=ISUM+NLAY                                                    # T00750
      LCIBUF=ISUM                                                       # T00760
      ISUM=ISUM + NCRL                                                  # T00770
      LCIUN=ISUM                                                        # T00780
      ISUM=ISUM + NUNIT                                                 # T00790
      LCQSS=ISUM                                                        # T00800
      ISUM=ISUM+NCRL                                                    # T00810
      ISM1=ISUM-1                                                       # T00820
      NDIF=LENA-ISM1                                                    # T00830
      NPART=NDIF/8                                                      # T00840
      IF (NPART.LT.1) THEN                                              # T00850
      IF (IBATCH.EQ.0) THEN                                             # T00860
      WRITE (*,*)                                                       # T00870
     1'THE SIZE OF THE MASTER ARRAY MUST BE INCREASED.'                 # T00880
      WRITE (*,*)                                                       # T00890
     1'REDIMENSION THE MASTER ARRAY:  A(LENA)'                          # T00900
      WRITE (*,5010) ISM1                                               # T00910
5010   FORMAT('WITH   LENA =',I8,' + 8 X (NUMBER OF PARTICLES)')        # T00920
      END IF                                                            # T00930
      WRITE (I7,*)                                                      # T00940
     1'THE SIZE OF THE MASTER ARRAY MUST BE INCREASED.'                 # T00950
      WRITE (I7,*)                                                      # T00960
     1'REDIMENSION THE MASTER ARRAY:  A(LENA)'                          # T00970
      WRITE (I7,5010) ISM1                                              # T00980
      STOP                                                              # T00990
      END IF                                                            # T01000
      LCXLC=ISUM                                                        # T01010
      ISUM=ISUM+NPART                                                   # T01020
      LCYLC=ISUM                                                        # T01030
      ISUM=ISUM+NPART                                                   # T01040
      LCZLC=ISUM                                                        # T01050
      ISUM=ISUM+NPART                                                   # T01060
      LCZLL=ISUM                                                        # T01070
      ISUM=ISUM+NPART                                                   # T01080
      LCJLC=ISUM                                                        # T01090
      ISUM=ISUM+NPART                                                   # T01100
      LCILC=ISUM                                                        # T01110
      ISUM=ISUM+NPART                                                   # T01120
      LCKLC=ISUM                                                        # T01130
      ISUM=ISUM+NPART                                                   # T01140
      LCTOT=ISUM                                                        # T01150
      ISUM=ISUM+NPART                                                   # T01160
      ISM1=ISUM-1                                                       # T01170
      IF (IBATCH.EQ.0) THEN                                             # T01180
      WRITE (*,5020) NPART                                              # T01190
5020  FORMAT ('THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS ',       # T01200
     1 I8)                                                              # T01210
      END IF                                                            # T01220
      WRITE (I7,5030) NPART                                             # T01230
5030  FORMAT ('MAXIMUM NUMBER OF PARTICLES IS ',I8)                     # T01240
      WRITE (I7,5040) ISM1,LENA                                         # T01250
5040  FORMAT (I8,' ELEMENTS OUT OF ',I8,' USED IN THE "A" ARRAY')       # T01260
      RETURN                                                            # T01270
      END                                                               # T01280
C-----END OF ROUTINE----------------------------------------------------# T01290
```

```
C                                                                       # U00010
C---Version 1.0   July 21, 1989                                         # U00020
C***********************************************************************# U00030
C                                                                       # U00040
C                              CELDAT                                   # U00050
C                                                                       # U00060
C  THIS SUBROUTINE PROVIDES A SUMMARY OF IMPORTANT DATA FOR A SPECIFIED # U00070
C  GRID CELL.                                                           # U00080
C                                                                       # U00090
C***********************************************************************# U00100
C                                                                       # U00110
      SUBROUTINE CELDAT (XMAX,YMAX,DELR,DELC,ZBOT,ZTOP,DELZ,DELZCB,POR, # U00120
     1IBOUND,HEAD,QX,QY,QZ,LAYCON,NCOL,NROW,NLAY,NZDIM,IGRID,NLPOR,     # U00130
     2NCP1,NRP1,NLP1,NCON,QSS,IA,I7,I9,IBATCH)                          # U00140
C                                                                       # U00150
      CHARACTER*80 MES                                                  # U00160
C                                                                       # U00170
      DIMENSION XMAX(NCOL),YMAX(NROW),DELR(NCOL),DELC(NROW),ZBOT(NZDIM),# U00180
     1ZTOP(NZDIM),DELZ(NLAY),DELZCB(NLAY),POR(NCOL,NROW,NLPOR),         # U00190
     2IBOUND(NCOL,NROW,NLAY),HEAD(NCOL,NROW,NLAY),QX(NCP1,NROW,NLAY),  # U00200
     3QY(NCOL,NRP1,NLAY),QZ(NCOL,NROW,NLP1),NCON(NLAY),LAYCON(NLAY),    # U00210
     4QSS(NCOL,NROW,NLAY)                                               # U00220
C                                                                       # U00230
      MES= ' DO YOU WANT TO CHECK DATA FOR ANOTHER CELL ?'              # U00240
10    IF (IBATCH.EQ.0) WRITE (*,*)                                      # U00250
     1'ENTER J I K COORDINATES OF CELL:  J   I   K'                     # U00260
      IF(IBATCH.EQ.0) THEN                                              # U00270
      READ (*,*) J,I,K                                                  # U00280
      WRITE(I9,*) J,I,K                                                 # U00290
      ELSE                                                              # U00300
      READ(I9,*) J,I,K                                                  # U00310
      END IF                                                            # U00320
      IF (IBATCH.EQ.0) WRITE (*,5000) J,I,K                             # U00330
5000   FORMAT('DATA FOR CELL (',I3,',',I3,',',I3,')')                   # U00340
      IF(IBOUND(J,I,K).GT.-1000.AND.IBOUND(J,I,K).LT.1000) THEN         # U00350
      IB=IBOUND(J,I,K)                                                  # U00360
      ISINK=0                                                           # U00370
      ELSE                                                              # U00380
      IB=IBOUND(J,I,K)/1000                                             # U00390
      ISINK=1                                                           # U00400
      END IF                                                            # U00410
      IF (IBATCH.EQ.0) WRITE (*,5010) IB,HEAD(J,I,K),POR(J,I,K)         # U00420
5010   FORMAT('IBOUND=',I7,' HEAD=',E15.5,' POROSITY=',F6.3)            # U00430
      X1=0.0                                                            # U00440
      X2=XMAX(J)                                                        # U00450
      IF(J.GT.1) X1=XMAX(J-1)                                           # U00460
      IF (IBATCH.EQ.0) WRITE (*,5020) X1,X2,DELR(J)                     # U00470
5020   FORMAT('X1=',E15.5,' X2=',E15.5,' (X2-X1)=',E15.5)               # U00480
      Y1=0.0                                                            # U00490
      Y2=YMAX(I)                                                        # U00500
      IF(I.LT.NROW) Y1=YMAX(I+1)                                        # U00510
      IF (IBATCH.EQ.0) WRITE (*,5030) Y1,Y2,DELC(I)                     # U00520
5030   FORMAT('Y1=',E15.5,' Y2=',E15.5,' (Y2-Y1)=',E15.5)               # U00530
      IF(IGRID.EQ.1) THEN                                               # U00540
      N=K                                                               # U00550
      ELSE                                                              # U00560
      N= (K-1)*NCOL*NROW + (I-1)*NCOL + J                               # U00570
      END IF                                                            # U00580
      ZMX=ZTOP(N)                                                       # U00590
      IF(LAYCON(K).EQ.1) ZMX=HEAD(J,I,K)                                # U00600
      IF(LAYCON(K).GT.1.AND.HEAD(J,I,K).LT.ZTOP(N)) ZMX=HEAD(J,I,K)     # U00610
      DZ=ZMX-ZBOT(N)                                                    # U00620
      IF (IBATCH.EQ.0) WRITE (*,5040) ZBOT(N),ZMX,DZ                    # U00630
5040   FORMAT('Z1=',E15.5,' Z2=',E15.5,' (Z2-Z1)=',E15.5)               # U00640
      QX1=QX(J,I,K)                                                     # U00650
      QX2=QX(J+1,I,K)                                                   # U00660
      IF (ABS(DZ).GT.1.0E-20) THEN                                      # U00670
      VX1=QX1/(Y2-Y1)/(ZMX-ZBOT(N))/POR(J,I,K)                          # U00680
      VX2=QX2/(Y2-Y1)/(ZMX-ZBOT(N))/POR(J,I,K)                          # U00690
      ELSE                                                              # U00700
      VX1=0.0                                                           # U00710
      VX2=0.0                                                           # U00720
```

```
      END IF                                                              # U00730
      IF (IBATCH.EQ.0) THEN                                               # U00740
      WRITE (*,*)                                                         # U00750
     1'VOLUMETRIC FLOW (Q1-Q6) AND VELOCITY (V1-V6) AT CELL FACES'        # U00760
      WRITE (*,5050) QX1,QX2,VX1,VX2                                      # U00770
5050   FORMAT('X (FACES 1&2): Q1=',1PE13.5,' Q2=',E13.5,' V1=',E11.3,     # U00780
     1 ' V2=',1PE11.3)                                                    # U00790
      END IF                                                              # U00800
      QY1=QY(J,I+1,K)                                                     # U00810
      QY2=QY(J,I,K)                                                       # U00820
      IF (ABS(DZ).GT.1.0E-20) THEN                                        # U00830
      VY1=QY1/(X2-X1)/(ZMX-ZBOT(N))/POR(J,I,K)                            # U00840
      VY2=QY2/(X2-X1)/(ZMX-ZBOT(N))/POR(J,I,K)                            # U00850
      ELSE                                                                # U00860
      VY1=0.0                                                             # U00870
      VY2=0.0                                                             # U00880
      END IF                                                              # U00890
      IF (IBATCH.EQ.0) WRITE (*,5060) QY1,QY2,VY1,VY2                     # U00900
5060   FORMAT('Y (FACES 3&4): Q3=',1PE13.5,' Q4=',E13.5,' V3=',E11.3,     # U00910
     1 ' V4=',E11.3)                                                      # U00920
      QZ1=QZ(J,I,K+1)                                                     # U00930
      QZ2=QZ(J,I,K)                                                       # U00940
      VZ1=QZ1/(X2-X1)/(Y2-Y1)/POR(J,I,K)                                  # U00950
      VZ2=QZ2/(X2-X1)/(Y2-Y1)/POR(J,I,K)                                  # U00960
      IF (IBATCH.EQ.0) WRITE (*,5070) QZ1,QZ2,VZ1,VZ2                     # U00970
5070   FORMAT('Z (FACES 5&6): Q5=',E13.5,' Q6=',E13.5,' V5=',E11.3,       # U00980
     1 ' V6=',E11.3)                                                      # U00990
      QI=0.0                                                              # U01000
      QO=0.0                                                              # U01010
      IF(QX1.GT.0.0) THEN                                                 # U01020
      QI=QI+QX1                                                           # U01030
      ELSE                                                                # U01040
      QO=QO-QX1                                                           # U01050
      END IF                                                              # U01060
      IF(QX2.LT.0.0) THEN                                                 # U01070
      QI=QI-QX2                                                           # U01080
      ELSE                                                                # U01090
      QO=QO+QX2                                                           # U01100
      END IF                                                              # U01110
      IF(QY1.GT.0.0) THEN                                                 # U01120
      QI=QI+QY1                                                           # U01130
      ELSE                                                                # U01140
      QO=QO-QY1                                                           # U01150
      END IF                                                              # U01160
      IF(QY2.LT.0.0) THEN                                                 # U01170
      QI=QI-QY2                                                           # U01180
      ELSE                                                                # U01190
      QO=QO+QY2                                                           # U01200
      END IF                                                              # U01210
      IF(QZ1.GT.0.0) THEN                                                 # U01220
      QI=QI+QZ1                                                           # U01230
      ELSE                                                                # U01240
      QO=QO-QZ1                                                           # U01250
      END IF                                                              # U01260
      IF(QZ2.LT.0.0)  THEN                                                # U01270
      QI=QI-QZ2                                                           # U01280
      ELSE                                                                # U01290
      QO=QO+QZ2                                                           # U01300
      END IF                                                              # U01310
      QINOUT= QI - QO + QSS(J,I,K)                                        # U01320
      IF (IBATCH.EQ.0) THEN                                               # U01330
      WRITE (*,5080) QI,QO,QSS(J,I,K)                                     # U01340
5080   FORMAT('TOTAL IN ACROSS CELL FACES =',E13.5/                        # U01350
     1'TOTAL OUT ACROSS CELL FACES =',E13.5/                              # U01360
     1'NET SOURCES =',E13.5)                                              # U01370
      WRITE (*,*) 'MASS BALANCE ERROR IN PERCENT IS DEFINED AS:'          # U01380
      WRITE (*,*) ' 100 X [(IN - OUT + NET SOURCES)/(IN + OUT)/2)]'       # U01390
      END IF                                                              # U01400
      QAVE=(QI+QO)/2.0                                                    # U01410
      IF(QAVE.GT.1.0E-20) THEN                                            # U01420
      ERROR= 100.0*(QINOUT/QAVE)                                          # U01430
      IF (IBATCH.EQ.0) WRITE (*,5090) ERROR                               # U01440
```

```
      ELSE                                                              # U01450
      IF (IBATCH.EQ.0) THEN                                             # U01460
      WRITE (*,*) 'NO FLOW INTO OR OUT OF THIS CELL'                    # U01470
      WRITE (*,*) '  NO MASS BALANCE COMPUTED'                         # U01480
      END IF                                                            # U01490
      END IF                                                            # U01500
5090   FORMAT('MASS BALANCE ERROR =',F10.5,' PERCENT')                  # U01510
      IF(NCON(K).GT.0) THEN                                             # U01520
      LCON=NCON(K)                                                      # U01530
      NLP=NLAY+LCON                                                     # U01540
      VZCB=QZ1/(X2-X1)/(Y2-Y1)/POR(J,I,NLP)                             # U01550
      IF(IBATCH.EQ.0) WRITE (*,5100) DELZCB(K),POR(J,I,NLP),VZCB        # U01560
5100   FORMAT('CONFINING BED DATA:'/'  THICKNESS =',1PE13.5/            # U01570
     1 '  POROSITY =',E13.5/'  VERTICAL VELOCITY =',E13.5)              # U01580
      END IF                                                            # U01590
C                                                                       # U01600
      CALL YESNO(IBATCH,I9,MES,IA,IANS)                                 # U01610
      IF(IANS.EQ.1) GO TO 10                                            # U01620
      RETURN                                                            # U01630
      END                                                               # U01640
C-----END OF ROUTINE-----------------------------------------------------# U01650
```

```
C                                                                       # V00010
C---Version 1.0   July 21, 1989                                         # V00020
C***********************************************************************# V00030
C                                                                       # V00040
C                            BALNCE                                     # V00050
C                                                                       # V00060
C  THIS SUBROUTINE CALCULATES VOLUMETRIC BALANCES FOR ALL VARIABLE HEAD # V00070
C  CELLS IN THE FINITE DIFFERENCE GRID.                                 # V00080
C                                                                       # V00090
C***********************************************************************# V00100
      SUBROUTINE BALNCE (QX,QY,QZ,QSS,IBOUND,HEAD,NCOL,NROW,NLAY,       # V00110
     1NCP1,NRP1,NLP1,KOUNT,IA,I7,I9,IBATCH,ETOL)                        # V00120
C                                                                       # V00130
      DIMENSION QX(NCP1,NROW,NLAY),QY(NCOL,NRP1,NLAY),                  # V00140
     1QZ(NCOL,NROW,NLP1),QSS(NCOL,NROW,NLAY),HEAD(NCOL,NROW,NLAY),      # V00150
     2IBOUND(NCOL,NROW,NLAY)                                            # V00160
      CHARACTER*80 MES                                                  # V00170
C                                                                       # V00180
      KOUNT=0                                                           # V00190
      EMAX=0.0                                                          # V00200
      DO 10  K=1,NLAY                                                   # V00210
      DO 10  I=1,NROW                                                   # V00220
      DO 10  J=1,NCOL                                                   # V00230
      IF(IBOUND(J,I,K).LE.0) GO TO 10                                   # V00240
      IF(HEAD(J,I,K).GT.1.0E+29) GO TO 10                               # V00250
      QX1=QX(J,I,K)                                                     # V00260
      QX2=QX(J+1,I,K)                                                   # V00270
      QY1=QY(J,I+1,K)                                                   # V00280
      QY2=QY(J,I,K)                                                     # V00290
      QZ1=QZ(J,I,K+1)                                                   # V00300
      QZ2=QZ(J,I,K)                                                     # V00310
C                                                                       # V00320
      QI=0.0                                                            # V00330
      QO=0.0                                                            # V00340
      IF(QX1.GT.0.0) THEN                                               # V00350
      QI=QI+QX1                                                         # V00360
      ELSE                                                              # V00370
      QO=QO-QX1                                                         # V00380
      END IF                                                            # V00390
      IF(QX2.LT.0.0) THEN                                               # V00400
      QI=QI-QX2                                                         # V00410
      ELSE                                                              # V00420
      QO=QO+QX2                                                         # V00430
      END IF                                                            # V00440
      IF(QY1.GT.0.0) THEN                                               # V00450
      QI=QI+QY1                                                         # V00460
      ELSE                                                              # V00470
      QO=QO-QY1                                                         # V00480
      END IF                                                            # V00490
      IF(QY2.LT.0.0) THEN                                               # V00500
      QI=QI-QY2                                                         # V00510
      ELSE                                                              # V00520
      QO=QO+QY2                                                         # V00530
      END IF                                                            # V00540
      IF(QZ1.GT.0.0) THEN                                               # V00550
      QI=QI+QZ1                                                         # V00560
      ELSE                                                              # V00570
      QO=QO-QZ1                                                         # V00580
      END IF                                                            # V00590
      IF(QZ2.LT.0.0)  THEN                                              # V00600
      QI=QI-QZ2                                                         # V00610
      ELSE                                                              # V00620
      QO=QO+QZ2                                                         # V00630
      END IF                                                            # V00640
      QIMO= QI-QO                                                       # V00650
      QINOUT= QIMO + QSS(J,I,K)                                         # V00660
      QAVE=(QI+QO)/2.0                                                  # V00670
      IF(QAVE.GT.1.0E-20) THEN                                          # V00680
      ERROR= 100.*(QINOUT/QAVE)                                         # V00690
      ELSE                                                              # V00700
      GO TO 10                                                          # V00710
      END IF                                                            # V00720
```

```
      ABERR= ABS(ERROR)                                                 # V00730
      ABEMAX= ABS(EMAX)                                                 # V00740
      IF(ABERR.GT.ABEMAX) EMAX=ERROR                                    # V00750
      IF(ABERR.GT.ETOL) THEN                                            # V00760
      KOUNT=KOUNT+1                                                     # V00770
      IF(KOUNT.LE.100) WRITE(I7,5010) ERROR,J,I,K,QIMO                  # V00780
      END IF                                                            # V00790
10    CONTINUE                                                          # V00800
      IF(KOUNT.EQ.0) THEN                                               # V00810
      IF (IBATCH.EQ.0) WRITE (*,5000) ETOL                              # V00820
5000   FORMAT('MASS BALANCES FOR ALL CELLS ARE <',F8.4,'%')             # V00830
      ELSE                                                              # V00840
      IF (IBATCH.EQ.0) THEN                                             # V00850
      WRITE (*,5020) KOUNT,ETOL                                         # V00860
      WRITE (*,5030) EMAX                                               # V00870
      WRITE (*,*) 'SEE "SUMMARY.PTH"  FILE FOR MORE INFORMATION'        # V00880
      IF(KOUNT.GT.100) WRITE (*,*) '(ONLY THE FIRST 100 CELLS WITH ERROR# V00890
     1S > THE TOLERANCE ARE RECORDED)'                                  # V00900
      END IF                                                            # V00910
      END IF                                                            # V00920
      RETURN                                                            # V00930
5010   FORMAT('MASS BALANCE ERROR OF ',F7.1,'% IN CELL (',I4,',',I4,',',# V00940
     1I4,')  (IN-OUT) =',E12.3)                                         # V00950
5020   FORMAT(I6,' CELLS HAD ERRORS >',F8.4)                            # V00960
5030   FORMAT('MAXIMUM ERROR IS ',F9.4,'%')                             # V00970
      END                                                               # V00980
C-----END OF ROUTINE-------------------------------------------------# V00990
```

```
C                                                                       # W00010
C---Version 1.0   July 21, 1989                                         # W00020
C***********************************************************************# W00030
C                                                                       # W00040
C                            FLUXBC                                     # W00050
C                                                                       # W00060
C  THIS SUBROUTINE APPORTIONS FLOW ALONG SPECIFIED FLOW CELL BOUNDARIES.# W00070
C                                                                       # W00080
C***********************************************************************# W00090
C                                                                       # W00100
      SUBROUTINE FLUXBC (Q,QSS,QX,QY,IBOUND,DELR,DELC,J,I,K,NCOL,       # W00110
     1NROW,NLAY,NCP1,NRP1)                                              # W00120
C                                                                       # W00130
      DIMENSION DELR(NCOL),DELC(NROW),IBOUND(NCOL,NROW,NLAY),           # W00140
     1 QSS(NCOL,NROW,NLAY),QX(NCP1,NROW,NLAY),QY(NCOL,NRP1,NLAY)        # W00150
      DIMENSION QSIDE(4)                                                # W00160
C                                                                       # W00170
      SUM=0.0E+0                                                        # W00180
      DO 10 N=1,4                                                       # W00190
10    QSIDE(N)=0.0                                                      # W00200
C                                                                       # W00210
      IF(J.EQ.1) THEN                                                   # W00220
      SUM=SUM+DELC(I)                                                   # W00230
      QSIDE(1)=DELC(I)                                                  # W00240
      ELSE IF(IBOUND(J-1,I,K).EQ.0) THEN                                # W00250
      SUM=SUM+DELC(I)                                                   # W00260
      QSIDE(1)=DELC(I)                                                  # W00270
      END IF                                                            # W00280
C                                                                       # W00290
      IF(J.EQ.NCOL) THEN                                                # W00300
      SUM=SUM+DELC(I)                                                   # W00310
      QSIDE(2)=DELC(I)                                                  # W00320
      ELSE IF (IBOUND(J+1,I,K).EQ.0) THEN                               # W00330
      SUM=SUM+DELC(I)                                                   # W00340
      QSIDE(2)=DELC(I)                                                  # W00350
      END IF                                                            # W00360
C                                                                       # W00370
      IF(I.EQ.1) THEN                                                   # W00380
      SUM=SUM+DELR(J)                                                   # W00390
      QSIDE(4)=DELR(J)                                                  # W00400
      ELSE IF (IBOUND(J,I-1,K).EQ.0) THEN                               # W00410
      SUM=SUM+DELR(J)                                                   # W00420
      QSIDE(4)=DELR(J)                                                  # W00430
      END IF                                                            # W00440
C                                                                       # W00450
      IF(I.EQ.NROW) THEN                                                # W00460
      SUM=SUM+DELR(J)                                                   # W00470
      QSIDE(3)=DELR(J)                                                  # W00480
      ELSE IF (IBOUND(J,I+1,K).EQ.0) THEN                               # W00490
      SUM=SUM+DELR(J)                                                   # W00500
      QSIDE(3)=DELR(J)                                                  # W00510
      END IF                                                            # W00520
C                                                                       # W00530
      ASUM= ABS(SUM)                                                    # W00540
      IF (ASUM.LT.1.0E-20) THEN                                         # W00550
      QSS(J,I,K)=QSS(J,I,K) + Q                                         # W00560
      IF(Q.LE.0.0.AND.IBOUND(J,I,K).LT.1000)                            # W00570
     1  IBOUND(J,I,K)=1000*IBOUND(J,I,K)                                # W00580
      RETURN                                                            # W00590
      END IF                                                            # W00600
C                                                                       # W00610
      DO 20 N=1,4                                                       # W00620
20    QSIDE(N)= Q*(QSIDE(N)/SUM)                                        # W00630
C                                                                       # W00640
      QX(J,I,K)= QX(J,I,K) + QSIDE(1)                                   # W00650
      QX(J+1,I,K)= QX(J+1,I,K) - QSIDE(2)                               # W00660
      QY(J,I+1,K)= QY(J,I+1,K) + QSIDE(3)                               # W00670
      QY(J,I,K)= QY(J,I,K) - QSIDE(4)                                   # W00680
C                                                                       # W00690
      RETURN                                                            # W00700
      END                                                               # W00710
C-----END OF ROUTINE-------------------------------------------------- # W00720
```

```
C                                                                       # X00010
C---Version 1.0   July 21, 1989                                         # X00020
C***********************************************************************# X00030
C                                                                       # X00040
C                              FILES                                    # X00050
C                                                                       # X00060
C  THIS SUBROUTINE OPENS FILES.                                         # X00070
C                                                                       # X00080
C***********************************************************************# X00090
C                                                                       # X00100
      SUBROUTINE FILES (IA,I0,I1,I2,I3,I4,I5,I6,I7,I8,I9,IBATCH)       # X00110
      CHARACTER*80 FNAME                                                # X00120
C                                                                       # X00130
C  CREATE AND OPEN STANDARD OUTPUT AND SCRATCH FILES                    # X00140
C                                                                       # X00150
      FNAME= 'SUMMARY.PTH'                                              # X00160
      CALL OPNFIL (I7,FNAME,4,I7,IBATCH,3)                              # X00170
      FNAME= 'ENDPOINT '                                                # X00180
      CALL OPNFIL (I3,FNAME,4,I7,IBATCH,3)                              # X00190
      CALL OPNFIL (I5,FNAME,3,I7,IBATCH,3)                              # X00200
C                                                                       # X00210
C  OPEN FILE CONTAINING FILE NAMES AND FORTRAN UNITS                    # X00220
C                                                                       # X00230
      IF (IBATCH.EQ.0) WRITE (*,*)                                      # X00240
     1'ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DATA FILES:'     # X00250
      IF(IBATCH.EQ.0) THEN                                              # X00260
      READ (*,5000) FNAME                                               # X00270
      WRITE(I9,5000) FNAME                                              # X00280
      ELSE                                                              # X00290
      READ(I9,5000) FNAME                                               # X00300
      END IF                                                            # X00310
5000   FORMAT (A)                                                       # X00320
C                                                                       # X00330
      CALL OPNFIL (I0,FNAME,1,I7,IBATCH,1)                              # X00340
C                                                                       # X00350
      KFIRST=0                                                          # X00360
10    READ (I0,*,END=20 ) IU,FNAME                                      # X00370
      IF(KFIRST.EQ.0) I1=IU                                             # X00380
      KFIRST=1                                                          # X00390
C                                                                       # X00400
C  OPEN DATA FILES                                                      # X00410
C                                                                       # X00420
      CALL OPNFIL (IU,FNAME,1,I7,IBATCH,1)                              # X00430
C                                                                       # X00440
      WRITE(I7,5010) IU,FNAME                                           # X00450
5010   FORMAT('FORTRAN UNIT ',I3,' ASSIGNED TO ',A60)                   # X00460
      GO TO 10                                                          # X00470
20    CONTINUE                                                          # X00480
      RETURN                                                            # X00490
      END                                                               # X00500
C-----END OF ROUTINE-------------------------------------------------- # X00510
```

```
C                                                                       # Y00010
C---Version 1.0   July 21, 1989                                         # Y00020
C***********************************************************************# Y00030
C                                                                       # Y00040
C                                RDBUDG                                 # Y00050
C                                                                       # Y00060
C  THIS SUBROUTINE READS AN ENTRY IN A BINARY BUDGET FILE PRODUCED BY   # Y00070
C  THE USGS MODULAR FINITE-DIFFERENCE FLOW MODEL.                       # Y00080
C                                                                       # Y00090
C***********************************************************************# Y00100
C                                                                       # Y00110
      SUBROUTINE RDBUDG (BUFF,LABEL,NCOL,NROW,NLAY,IU,IO)               # Y00120
      DIMENSION BUFF(NCOL,NROW,NLAY)                                    # Y00130
      CHARACTER*16 LABEL,TEXT                                           # Y00140
C                                                                       # Y00150
      IPASS=0                                                           # Y00160
10    IPASS=IPASS+1                                                     # Y00170
C                                                                       # Y00180
      IF(IPASS.GT.2) THEN                                               # Y00190
      IF (IBATCH.EQ.0) WRITE (*,5000) LABEL,IU                          # Y00200
      WRITE (IO,5000) LABEL,IU                                          # Y00210
5000  FORMAT (A16,' BUDGET TERMS WERE NOT FOUND ON UNIT ',I3)           # Y00220
      STOP                                                              # Y00230
      END IF                                                            # Y00240
C                                                                       # Y00250
20    READ (IU,END=30 ) KSTP,KPER,TEXT,NC,NR,NL                         # Y00260
      IF (TEXT.EQ.LABEL) THEN                                           # Y00270
      READ (IU) BUFF                                                    # Y00280
      RETURN                                                            # Y00290
      ELSE                                                              # Y00300
      READ (IU)                                                         # Y00310
      END IF                                                            # Y00320
      GO TO 20                                                          # Y00330
30    REWIND (IU)                                                       # Y00340
      GO TO 10                                                          # Y00350
      END                                                               # Y00360
C-----END OF ROUTINE-------------------------------------------------- # Y00370
```

```
C                                                                       # Z00010
C---Version 1.0   July 21, 1989                                         # Z00020
C***********************************************************************# Z00030
C                                                                       # Z00040
C                                IOTYPE                                 # Z00050
C                                                                       # Z00060
C  THIS ROUTINE READS THE FIRST LINE OF THE FILE NAMED "TERMIN". IT     # Z00070
C  LOOKS FOR THE FIRST NON-BLANK CHARACTER. IF THAT CHARACTER IS "S",   # Z00080
C  IBATCH IS SET TO 0 FOR SCREEN INPUT. IF THE CHARACTER IS "B", IBATCH # Z00090
C  IS SET TO 1 TO READ "SCREEN" INPUT FROM FILE "TERMIN". FILE "TERMIN" # Z00100
C  IS OPENED BY THIS ROUTINE.                                           # Z00110
C                                                                       # Z00120
C***********************************************************************# Z00130
C                                                                       # Z00140
      SUBROUTINE IOTYPE (IBATCH,I9,I7)                                  # Z00150
C                                                                       # Z00160
      CHARACTER*132 LINE1                                               # Z00170
      CHARACTER*6 S                                                     # Z00180
      CHARACTER*5 B                                                     # Z00190
      CHARACTER*80 FNAME                                                # Z00200
C                                                                       # Z00210
      S= 'SCREEN'                                                       # Z00220
      B= 'BATCH'                                                        # Z00230
C                                                                       # Z00240
      FNAME= 'TERMIN.PTH'                                               # Z00250
      CALL OPNFIL (I9,FNAME,4,I7,IBATCH,3)                              # Z00260
C                                                                       # Z00270
      READ (I9,'(A)',END=10 ) LINE1                                     # Z00280
      IS= INDEX (LINE1,S)                                               # Z00290
      IB= INDEX (LINE1,B)                                               # Z00300
      IF (IS.NE.0) THEN                                                 # Z00310
      IBATCH=0                                                          # Z00320
      ELSE IF (IB.NE.0) THEN                                            # Z00330
      IBATCH=1                                                          # Z00340
      ELSE IF (IS.EQ.0.AND.IB.EQ.0) THEN                                # Z00350
      GO TO 10                                                          # Z00360
      END IF                                                            # Z00370
      RETURN                                                            # Z00380
10    CONTINUE                                                          # Z00390
      REWIND (I9)                                                       # Z00400
      WRITE (I9,'(A)') 'SCREEN'                                         # Z00410
      IBATCH=0                                                          # Z00420
      RETURN                                                            # Z00430
      END                                                               # Z00440
C                                                                       # Z00450
C-----END OF ROUTINE-------------------------------------------------- # Z00460
```

```
C                                                                       #AA00010
C---Version 1.0   July 21, 1989                                         #AA00020
C***********************************************************************#AA00030
C                                                                       #AA00040
C                                ZCALC                                  #AA00050
C                                                                       #AA00060
C  THIS ROUTINE COMPUTES THE GLOBAL Z COORDINATE GIVEN THE LOCAL        #AA00070
C  Z COORDINATE WITHIN THE CELL                                         #AA00080
C                                                                       #AA00090
C***********************************************************************#AA00100
C                                                                       #AA00110
      SUBROUTINE ZCALC (IGRID,NCOL,NROW,NLAY,ZLC,ZLLC,ZTOP,ZBOT,        #AA00120
     1 HEAD,LAYCON,ILC,JLC,KLC,NZDIM,NPART,NPRT)                        #AA00130
C                                                                       #AA00140
      DIMENSION ZLLC(NPART),ZLC(NPART),ZTOP(NZDIM),ZBOT(NZDIM),         #AA00150
     1 HEAD(NCOL,NROW,NLAY),LAYCON(NLAY),ILC(NPART),JLC(NPART),         #AA00160
     2 KLC(NPART)                                                       #AA00170
C                                                                       #AA00180
      DO 10  NPT=1,NPRT                                                 #AA00190
      IP=ILC(NPT)                                                       #AA00200
      JP=JLC(NPT)                                                       #AA00210
      KP=KLC(NPT)                                                       #AA00220
      ZLOC= ZLLC(NPT)                                                   #AA00230
      IF(IGRID.EQ.1) THEN                                               #AA00240
      NK=KP                                                             #AA00250
      NKP=NK+1                                                          #AA00260
      ELSE                                                              #AA00270
      NK = (KP-1)*NCOL*NROW + (IP-1)*NCOL + JP                          #AA00280
      NKP= NK + (NCOL*NROW)                                             #AA00290
      END IF                                                            #AA00300
      IF(KP.EQ.NLAY.AND.ZLOC.LT.0.0) THEN                               #AA00310
      WRITE(I7,*) ' RUN STOPPED. INCONSISTENT CONFINING BED LOCATION'   #AA00320
      STOP                                                              #AA00330
      END IF                                                            #AA00340
      ZMX=ZTOP(NK)                                                      #AA00350
      HED=HEAD(JP,IP,KP)                                                #AA00360
      IF(LAYCON(KP).EQ.1) ZMX=HED                                       #AA00370
      IF(LAYCON(KP).GT.1.AND.HED.LT.ZMX) ZMX=HED                        #AA00380
      IF(ZLOC.GE.0.0) THEN                                              #AA00390
      ZP=ZLOC*ZMX + (1.0-ZLOC)*ZBOT(NK)                                 #AA00400
      ELSE                                                              #AA00410
      ZL= 1.0+ZLOC                                                      #AA00420
      ZP=ZL*ZBOT(NK) + (1.0-ZL)*ZTOP(NKP)                               #AA00430
      END IF                                                            #AA00440
      ZLC(NPT)=ZP                                                       #AA00450
10    CONTINUE                                                          #AA00460
      RETURN                                                            #AA00470
      END                                                               #AA00480
C                                                                       #AA00490
C-----END OF ROUTINE--------------------------------------------------#AA00500
```

```
C                                                                       #AB00010
C---Version 1.0   July 21, 1989                                         #AB00020
C***********************************************************************#AB00030
C                                                                       #AB00040
C                           OPNFIL                                      #AB00050
C                                                                       #AB00060
C  THIS ROUTINE OPENS A SINGLE FILE.                                    #AB00070
C                                                                       #AB00080
C***********************************************************************#AB00090
C                                                                       #AB00100
      SUBROUTINE OPNFIL (IU,FNAME,NSTAT,IOUT,IBATCH,NACT)               #AB00110
      CHARACTER*80 FNAME,FMT,FM                                         #AB00120
      LOGICAL*4 EX                                                      #AB00130
C                                                                       #AB00140
C                       VARIABLES                                       #AB00150
C                                                                       #AB00160
C  IU = FORTRAN UNIT NUMBER OF FILE                                     #AB00170
C  FNAME = FILE NAME                                                    #AB00180
C  NSTAT = STATUS OF FILE                                               #AB00190
C          1 => OLD FILE                                                #AB00200
C          2 => NEW FILE                                                #AB00210
C          3 => SCRATCH FILE (DELETED AUTOMATICALLY WHEN RUN ENDS)      #AB00220
C          4 => UNDETERMINED STATUS (MAY OR MAY NOT EXIST)              #AB00230
C                 IF IT DOES NOT EXIST, IT IS CREATED BY OPEN STATEMENT #AB00240
C                 IF IT DOES EXIST, IT IS OPENED AS 'OLD' FILE          #AB00250
C                                                                       #AB00260
C  IOUT = UNIT NUMBER FOR OUTPUT FILE TO WRITE ERROR MESSAGE TO         #AB00270
C         IF NECESSARY                                                  #AB00280
C                                                                       #AB00290
C  IBATCH = FLAG INDICATING IF THERE IS INTERACTIVE DIALOGUE AT         #AB00300
C           TERMINAL                                                    #AB00310
C                                                                       #AB00320
C           0 => THERE IS INTERACTIVE DIALOGUE                          #AB00330
C           1 => THERE IS NOT INTERACTIVE DIALOGUE (BATCH MODE)         #AB00340
C                                                                       #AB00350
C  NACT = VARIABLE DENOTING IF A FILE IS READ ONLY, WRITE ONLY, OR      #AB00360
C         READ AND WRITE.                                               #AB00370
C                                                                       #AB00380
C           1 => READ ONLY                                              #AB00390
C           2 => WRITE ONLY                                             #AB00400
C           3 => READ AND WRITE                                         #AB00410
C                                                                       #AB00420
C  "NACT" IS NOT USED IN THIS SUBROUTINE. ALL FILES ARE OPENED FOR      #AB00430
C  READING AND WRITING. OPENING "READ ONLY" AND "WRITE ONLY" FILES      #AB00440
C  IS MACHINE DEPENDENT. THE VARIABLE "NACT" IS PASSED TO THIS          #AB00450
C  ROUTINE TO MAKE IT EASY TO MODIFY THE OPEN STATEMENTS TO ALLOW       #AB00460
C  FOR READ AND WRITE ONLY FILES ON ANY GIVEN MACHINE. THE CALLS TO     #AB00470
C  THIS SUBROUTINE ARE CURRENTLY SET UP SO THAT ANY FILE THAT IS        #AB00480
C  WRITTEN TO IS GIVEN "NACT = 3" AND ANY FILE THAT IS READ FROM BUT    #AB00490
C  NOT WRITTEN TO IS GIVEN "NACT = 1". NO FILES ARE GIVEN "WRITE ONLY"  #AB00500
C  STATUS. "READ ONLY" STATUS IS USEFUL IF A NUMBER OF USERS WANT       #AB00510
C  TO SIMULTANEOUSLY SHARE INPUT FILES.                                 #AB00520
C                                                                       #AB00530
C  IF THE FILE DOES NOT CURRENTLY EXIST, A NEGATIVE UNIT NUMBER IS      #AB00540
C  USED AS A FLAG TO INDICATED THAT A NEW FILE SHOULD BE CREATED AS     #AB00550
C  AN UNFORMATTED (BINARY) FILE.                                        #AB00560
C                                                                       #AB00570
      IBIN=0                                                            #AB00580
      IF (IU.LT.0) THEN                                                 #AB00590
      IU= -IU                                                           #AB00600
      IBIN=1                                                            #AB00610
      END IF                                                            #AB00620
C                                                                       #AB00630
C  CHECK TO SEE IF A FILE IS OPENED TO UNIT=IOUT SO THAT ERROR          #AB00640
C  MESSAGES CAN BE WRITTEN.                                             #AB00650
      INQUIRE (UNIT=IOUT,OPENED=EX)                                     #AB00660
      IO=0                                                              #AB00670
      IF (EX) IO=1                                                      #AB00680
C                                                                       #AB00690
C  OPEN AN EXISTING FILE                                                #AB00700
C                                                                       #AB00710
      IF (NSTAT.EQ.1) THEN                                              #AB00720
```

```
10    INQUIRE (FILE=FNAME,EXIST=EX,UNFORMATTED=FM)                          #AB00730
C  CHECK TO SEE IF FILE EXISTS                                              #AB00740
      IF (EX) GO TO 20                                                      #AB00750
      IF (IBATCH.EQ.0) THEN                                                 #AB00760
      WRITE (*,*) 'FILE DOES NOT EXIST.'                                    #AB00770
      WRITE (*,*) 'ENTER THE NAME OF AN EXISTING FILE (<CR>=QUIT):'         #AB00780
      READ (*,'(A)') FNAME                                                  #AB00790
      IF (FNAME.EQ.' ') STOP                                                #AB00800
      GO TO 10                                                              #AB00810
      ELSE                                                                  #AB00820
      IF (IO.EQ.1) WRITE (IOUT,*) 'FILE DOES NOT EXIST:'                    #AB00830
      IF (IO.EQ.1) WRITE (IOUT,'(A)') FNAME                                 #AB00840
      STOP                                                                  #AB00850
      END IF                                                                #AB00860
20    CONTINUE                                                              #AB00870
      FMT='FORMATTED'                                                       #AB00880
      IF (IBIN.EQ.1.OR.FM.EQ.'YES') FMT='UNFORMATTED'                       #AB00890
      OPEN (IU,FILE=FNAME,STATUS='OLD',FORM=FMT,IOSTAT=IERR)                #AB00900
      IF (IERR.GT.0) GO TO 40                                               #AB00910
      RETURN                                                                #AB00920
      END IF                                                                #AB00930
C                                                                           #AB00940
C  OPEN A NEW FILE                                                          #AB00950
C                                                                           #AB00960
      IF (NSTAT.EQ.2) THEN                                                  #AB00970
30    INQUIRE (FILE=FNAME,EXIST=EX)                                         #AB00980
      IF (EX) THEN                                                          #AB00990
      IF (IBATCH.EQ.0) THEN                                                 #AB01000
      WRITE (*,*) 'FILE ALREADY EXISTS.'                                    #AB01010
      WRITE (*,*) 'ENTER THE NAME OF A NEW FILE (<CR>=QUIT):'               #AB01020
      READ (*,'(A)') FNAME                                                  #AB01030
      IF (FNAME.EQ.' ') STOP                                                #AB01040
      GO TO 30                                                              #AB01050
      ELSE                                                                  #AB01060
      IF (IO.EQ.1) WRITE (IOUT,*) 'FILE ALREADY EXISTS:'                    #AB01070
      IF (IO.EQ.1) WRITE (IOUT,'(A)') FNAME                                 #AB01080
      STOP                                                                  #AB01090
      END IF                                                                #AB01100
      END IF                                                                #AB01110
      FMT='FORMATTED'                                                       #AB01120
      IF (IBIN.EQ.1) FMT='UNFORMATTED'                                      #AB01130
      OPEN (IU,FILE=FNAME,STATUS='NEW',FORM=FMT,IOSTAT=IERR)                #AB01140
      IF (IERR.GT.0) GO TO 40                                               #AB01150
      RETURN                                                                #AB01160
      END IF                                                                #AB01170
C                                                                           #AB01180
C  OPEN A SCRATCH FILE                                                      #AB01190
C                                                                           #AB01200
      IF (NSTAT.EQ.3) THEN                                                  #AB01210
      FMT='FORMATTED'                                                       #AB01220
      IF (IBIN.EQ.1) FMT='UNFORMATTED'                                      #AB01230
      OPEN (IU,STATUS='SCRATCH',FORM=FMT,IOSTAT=IERR)                       #AB01240
C  FOR MICROSOFT FORTRAN USE:                                               #AB01250
C      OPEN (IU,FORM=FMT,IOSTAT=IERR)                                       #AB01260
      IF (IERR.GT.0) GO TO 40                                               #AB01270
      RETURN                                                                #AB01280
      END IF                                                                #AB01290
C                                                                           #AB01300
C  OPEN A FILE OF UNKNOWN STATUS                                            #AB01310
C                                                                           #AB01320
      IF (NSTAT.EQ.4) THEN                                                  #AB01330
      INQUIRE (FILE=FNAME,EXIST=EX,UNFORMATTED=FM)                          #AB01340
      IF (EX) THEN                                                          #AB01350
      FMT='FORMATTED'                                                       #AB01360
      IF (IBIN.EQ.1.OR.FM.EQ.'YES') FMT='UNFORMATTED'                       #AB01370
      OPEN (IU,FILE=FNAME,STATUS='OLD',FORM=FMT,IOSTAT=IERR)                #AB01380
      IF (IERR.GT.0) GO TO 40                                               #AB01390
      ELSE                                                                  #AB01400
      FMT='FORMATTED'                                                       #AB01410
      IF (IBIN.EQ.1) FMT='UNFORMATTED'                                      #AB01420
      OPEN (IU,FILE=FNAME,STATUS='NEW',FORM=FMT,IOSTAT=IERR)                #AB01430
      IF (IERR.GT.0) GO TO 40                                               #AB01440
```

```
      END IF                                                            #AB01450
      RETURN                                                            #AB01460
      END IF                                                            #AB01470
C                                                                       #AB01480
C  WRITE MESSAGE INDICATING PROBLEM OPENING FILE                        #AB01490
C                                                                       #AB01500
40    IF (IBATCH.EQ.0) WRITE (*,5000) IU                                #AB01510
      IF (IO.EQ.1) WRITE (IOUT,5000) IU                                 #AB01520
5000   FORMAT ('ERROR OPENING FILE TO UNIT ',I3)                        #AB01530
      STOP                                                              #AB01540
      END                                                               #AB01550
C-----END OF ROUTINE-------------------------------------------------------#AB01560
```

```
C                                                                       #AC00010
C---Version 17JUL89                                                     #AC00020
C***********************************************************************#AC00030
C                                                                       #AC00040
C                              FACTYP                                   #AC00050
C                                                                       #AC00060
C  THIS ROUTINE DETERMINES WHETHER A CELL FACE IS A BOUNDARY FACE OR    #AC00070
C  A CONNECTION WITH AN ACTIVE CELL. IF A BOUNDARY, NBD=1; IF A         #AC00080
C  CONNECTION TO AN ACTIVE CELL, NBD=0.                                 #AC00090
C                                                                       #AC00100
C***********************************************************************#AC00110
C                                                                       #AC00120
      SUBROUTINE FACTYP (N,NBOUND,JNXT,INXT,KNXT,NBD,IBOUND,NCOL,NROW,  #AC00130
     1 NLAY,J,I,K,IFACE,IPAC,IO)                                        #AC00140
      DIMENSION IBOUND (NCOL,NROW,NLAY)                                 #AC00150
      CHARACTER*3 TEXT(6)                                               #AC00160
      DATA TEXT /'RIV','DRN','GHB','WEL','RCH','EVT'/                   #AC00170
C                                                                       #AC00180
      NBD=0                                                             #AC00190
      IF(N.EQ.NBOUND) THEN                                              #AC00200
      NBD=1                                                             #AC00210
      ELSE                                                              #AC00220
      IF (IBOUND(JNXT,INXT,KNXT).EQ.0) THEN                             #AC00230
      NBD=1                                                             #AC00240
      ELSE                                                              #AC00250
      WRITE (IO,5000) IFACE,J,I,K,TEXT(IPAC)                            #AC00260
5000   FORMAT('FACE',I2,' OF (J,I,K) = ',I4,',',I4,',',I4,              #AC00270
     1' IS NOT A BOUNDARY FACE. ',A3,' FLOW TREATED AS SOURCE/SINK TERM'#AC00280
     2)                                                                 #AC00290
      END IF                                                            #AC00300
      END IF                                                            #AC00310
      RETURN                                                            #AC00320
      END                                                               #AC00330
C-----END OF ROUTINE--------------------------------------------------#AC00340
```

```
C---Version 1.0   July 21, 1989
C*****************************************************************# A00020
C                         MODPATH-PLOT                           *# A00030
C                                                                *# A00040
C          Plotting program for results from the particle        *# A00050
C                tracking program MODPATH                        *# A00060
C                                                                *# A00070
C                               by                               *# A00080
C                                                                *# A00090
C                          David W. Pollock                      *# A00100
C                                                                *# A00110
C*****************************************************************# A00120
C-----------------------------------------------------------------# A00130
C--- REDIMENSION MASTER ARRAY BY CHANGING THE NEXT TWO STATEMENTS ------# A00140
C--- MAKE SURE THAT LENA IS SET EQUAL TO THE LENGTH OF THE A ARRAY -----# A00150
      COMMON A(250000)                                            # A00160
      LENA=250000                                                 # A00170
C-----------------------------------------------------------------# A00180
C  SET UNIT NUMBERS FOR SCREEN I/O AND FILES THAT ARE SPECIFIED BY    # A00190
C  INTERACTIVE I/O                                                 # A00200
C                                                                  # A00210
C  IA = UNIT NUMBER FOR THE TERMINAL SCREEN                        # A00220
      IA=1                                                        # A00230
C  I0 = UNIT NUMBER OF FILE CONTAINING FILE NAMES AND UNIT NUMBERS    # A00240
      I0=101                                                      # A00250
C  I2 = UNIT NUMBER FOR THE "PATHLINE" FILE                        # A00260
      I2=102                                                      # A00270
C  I3 = UNIT NUMBER FOR THE "ENDPOINT" FILE                        # A00280
      I3=103                                                      # A00290
C  I4 = UNIT NUMBER FOR THE "TIMESERS" FILE                        # A00300
      I4=104                                                      # A00310
C  I7 = UNIT NUMBER FOR THE "SUMMARY.PLT" FILE                     # A00320
      I7=107                                                      # A00330
C-----------------------------------------------------------------# A00340
C                                                                  # A00350
C  OPEN STANDARD INPUT AND OUTPUT FILES                            # A00360
C                                                                  # A00370
      CALL FILES (IA,I0,I1,I2,I3,I4,I7)                           # A00380
C                                                                  # A00390
C  ALLOCATE SPACE FOR ARRAYS                                       # A00400
C                                                                  # A00410
      CALL SPACE(LENA,NCOL,NROW,NLAY,NCBL,IGRID,NZDIM,NUNIT,      # A00420
     1LCIBOU,LCXMX,LCXMN,LCDX,LCYMX,LCYMN,LCDY,LCZBOT,LCZTOP,     # A00430
     2LCDZ,LCZMX,LCZMN,LCHEAD,LCBUFF,LCLAYC,LCNCON,LCDZCB,LCIBUF, # A00440
     3LCIUN,LCIBYZ,LCIBXZ,LCIZN,NPART,IA,I1,I7)                   # A00450
C                                                                  # A00460
      NPART2=2*NPART                                              # A00470
C                                                                  # A00480
C  TRANSFER CONTROL TO MAIN SUBROUTINE                             # A00490
C                                                                  # A00500
      CALL DRIVER (A(LCIBOU),A(LCXMX),A(LCXMN),A(LCDX),A(LCYMX),  # A00510
     1A(LCYMN),A(LCDY),A(LCZBOT),A(LCZTOP),A(LCDZ),A(LCZMX),      # A00520
     2A(LCZMN),A(LCHEAD),A(LCBUFF),A(LCLAYC),A(LCNCON),A(LCDZCB), # A00530
     3A(LCIBUF),A(LCIUN),A(LCIBYZ),A(LCIBXZ),NCOL,NROW,NLAY,NZDIM,NUNIT,# A00540
     1NCBL,IGRID,IA,I1,I2,I3,I4,I7,NPART,NPART2,A(LCIZN))         # A00550
C                                                                  # A00560
      STOP                                                        # A00570
      END                                                         # A00580
C-----END OF ROUTINE---------------------------------------------# A00590
```

```
C                                                                       # B00010
C---Version 1.0   July 21, 1989                                         # B00020
C***********************************************************************# B00030
C                                                                       # B00040
C                              DRIVER                                   # B00050
C                                                                       # B00060
C  THIS ROUTINE CONTROLS THE OVERALL SEQUENCE OF OPERATIONS AND         # B00070
C  HANDLES MOST OF THE INTERACTIVE I/O                                  # B00075
C                                                                       # B00080
C***********************************************************************# B00090
C                                                                       # B00100
      SUBROUTINE DRIVER (IBOUND,XMX,XMN,DELR,YMX,YMN,DELC,ZBOT,ZTOP,    # B00110
     1DELZ,ZMX,ZMN,HEAD,BUFF,LAYCON,NCON,DELZCB,IBUFF,IUNIT,IBYZ,IBXZ,  # B00120
     2NCOL,NROW,NLAY,NZDIM,NUNIT,NCBL,IGRID,IA,I1,I2,I3,I4,I7,NPART,    # B00130
     3NPART2,IZPART)                                                    # B00140
C                                                                       # B00150
      DIMENSION LAYCON(NLAY),NCON(NLAY),DELR(NCOL),DELC(NROW),DELZ(NLAY)# B00160
     1,DELZCB(NLAY),XMX(NCOL),YMX(NROW),ZTOP(NZDIM),ZBOT(NZDIM),        # B00170
     2HEAD(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),XMN(NCOL),YMN(NROW),  # B00180
     4IUNIT(NUNIT),BUFF(NCOL,NROW,NLAY),IBUFF(NCOL,NROW,NLAY),ZMN(NLAY),# B00190
     5ZMX(NLAY),IBYZ(NROW,NLAY),IBXZ(NCOL,NLAY),IZPART(NPART2)          # B00200
C                                                                       # B00210
      DIMENSION X(5),Y(5)                                               # B00220
      DIMENSION ISPLOT(51)                                              # B00230
C                                                                       # B00240
      CHARACTER*80 FNAME,MES,TITLE                                      # B00250
C                                                                       # B00260
      DO 10 N=1,51                                                      # B00270
10     ISPLOT(N)=0                                                      # B00280
C                                                                       # B00290
C  READ MODPATH DATA SET                                                # B00300
C                                                                       # B00310
      CALL DATIN (LAYCON,NCON,HEAD,XMX,XMN,YMX,YMN,DELR,                # B00320
     1DELC,DELZ,DELZCB,ZTOP,ZBOT,ZMN,ZMX,IBOUND,IUNIT,NCOL,NROW,NLAY,   # B00330
     2NUNIT,NZDIM,BUFF,IBUFF,IGRID,NCBL,IA,I1,I2,I3,I4,I7,NPART)        # B00340
C                                                                       # B00350
C                                                                       # B00360
      WRITE (*,*)'ENTER TITLE (80 CHARACTERS OR LESS):'                 # B00370
      READ (*,5000) TITLE                                               # B00380
5000  FORMAT(A)                                                         # B00390
C                                                                       # B00400
      WRITE (*,*)'ENTER GRAPHICS DEVICE CODE:'                          # B00410
      WRITE (*,*)'    1 = TEKTRONIX 4010 (GRAPHICS TAB)'                # B00420
      WRITE (*,*)'    2 = TEKTRONIX 4105 OR 4205'                       # B00430
      WRITE (*,*)'    3 = TEKTRONIX 4115 OR 4125'                       # B00440
      WRITE (*,*)'    4 = TEKTRONIX 4107, 4207, OR 4109'                # B00450
      WRITE (*,*)'    5 = CREATE META FILE'                             # B00460
      READ (*,*) MODEL                                                  # B00470
C                                                                       # B00480
      MES= 'DRAW INTERIOR GRID LINES ?'                                 # B00490
      CALL YESNO (MES,IA,IGL)                                           # B00500
C                                                                       # B00510
      WRITE (*,*)'ENTER THE TYPE OF GRAPH:'                             # B00520
      WRITE (*,*)'    1 = FLOW LINE PLOT'                               # B00530
      WRITE (*,*)'    2 = MAP VIEW OF STARTING LOCATIONS (FORWARD TRACKI# B00540
     1NG)'                                                              # B00550
      WRITE (*,*)'    3 = MAP VIEW OF FINAL LOCATIONS (FORWARD TRACKING)# B00560
     1'                                                                 # B00570
      WRITE (*,*)'    4 = MAP VIEW OF FINAL LOCATIONS (BACKWARD TRACKING# B00580
     1)'                                                                # B00590
      WRITE (*,*)'    5 = TIME SERIES PLOT'                             # B00600
      READ (*,*) ITYPE                                                  # B00610
      IVIEW=1                                                           # B00620
      LAYER=1                                                           # B00630
      IF(ITYPE.EQ.1.OR.ITYPE.EQ.5) THEN                                 # B00640
      WRITE (*,*)'WHAT IS THE ORIENTATION OF THE PLOT ?'                # B00650
      WRITE (*,*)'  1 = MAP VIEW'                                       # B00660
      WRITE (*,*)'  2 = CROSS SECTION VIEW ALONG A COLUMN'              # B00670
      WRITE (*,*)'  3 = CROSS SECTION VIEW ALONG A ROW'                 # B00680
      READ (*,*) IVIEW                                                  # B00690
      WRITE (*,*)'WHAT DATA SHOULD BE PLOTTED ?'                        # B00700
      WRITE (*,*)'   0 = PLOT ALL DATA BY PROJECTION ONTO THE 2D SLICE' # B00710
```

```
      IF(IVIEW.EQ.1) THEN                                                  # B00720
      WRITE (*,*)'   1 = PLOT ONLY THE DATA WITHIN THE LAYER CORRESPONDI# B00730
     1NG TO THE 2D SLICE'                                                  # B00740
      READ (*,*) IPROJ                                                     # B00750
      LAYER=1                                                              # B00760
      IF(IPROJ.GT.0) THEN                                                  # B00770
      WRITE (*,*)'ALONG WHAT LAYER SHOULD THE 2D SLICE BE TAKEN ?'         # B00780
      READ (*,*) LAYER                                                     # B00790
      END IF                                                               # B00800
      ELSE IF (IVIEW.EQ.2) THEN                                            # B00810
      WRITE (*,*)'   1 = PLOT ONLY THE DATA WITHIN THE COLUMN CORRESPOND# B00820
     1ING TO THE CROSS SECTION'                                            # B00830
      READ (*,*) IPROJ                                                     # B00840
      WRITE (*,*)'ENTER THE COLUMN ALONG WHICH CROSS SECTION IS TAKEN:' # B00850
      READ (*,*) JCOL                                                      # B00860
      ELSE IF (IVIEW.EQ.3) THEN                                            # B00870
      WRITE (*,*)'   1 = PLOT ONLY THE DATA WITHIN THE ROW CORRESPONDING# B00880
     1 TO THE CROSS SECTION'                                               # B00890
      READ (*,*) IPROJ                                                     # B00900
      WRITE (*,*)'ALONG WHAT ROW SHOULD THE CROSS SECTION BE TAKEN ?'   # B00910
      READ (*,*) IROW                                                      # B00920
      END IF                                                               # B00930
      END IF                                                               # B00940
      IF(ITYPE.EQ.1) THEN                                                  # B00950
      WRITE (*,*)'WERE THE FLOWLINES GENERATED BY FORWARD OR BACKWARD TR# B00960
     1ACKING ?'                                                            # B00970
      WRITE (*,*)'   0 = FORWARD'                                          # B00980
      WRITE (*,*)'   1 = BACKWARD'                                         # B00990
      READ (*,*) IDIR                                                      # B01000
      IF (IDIR.EQ.0) THEN                                                  # B01010
      MES= 'DO YOU WANT TO SKIP OVER PATH LINES THAT DISCHARGE IN ZONE 1# B01020
     1 ?'                                                                  # B01030
      CALL YESNO (MES,IA,ISKIP)                                            # B01040
      END IF                                                               # B01050
      MES= 'DO YOU WANT TO PLOT POINTS AT SPECIFIED TIME INTERVALS ?'   # B01060
      CALL YESNO (MES,IA,IPTS)                                             # B01070
      MES= 'DO YOU WANT TO STOP DRAWING PATH LINES AT A SPECIFIED TIME?'# B01080
      CALL YESNO (MES,IA,IANS)                                             # B01090
      TMAX= 1.0E+30                                                        # B01100
      IF (IANS.EQ.1) THEN                                                  # B01110
      WRITE (*,*) 'ENTER THE TIME (SAME UNITS AS FLOW MODEL):'             # B01120
      READ (*,*) TMAX                                                      # B01130
      TMAX= 1.00001*TMAX                                                   # B01140
      END IF                                                               # B01150
      END IF                                                               # B01160
5010   FORMAT(A)                                                           # B01170
      IPLOT=0                                                              # B01180
      IF(ITYPE.EQ.3.OR.ITYPE.EQ.4) THEN                                    # B01190
      MES= 'DO YOU WANT TO PLOT ONLY THOSE POINTS THAT TERMINATE IN ONE # B01200
     1SPECIFIC ZONE ?'                                                     # B01210
      CALL YESNO (MES,IA,IANS)                                             # B01220
      IF(IANS.EQ.1) THEN                                                   # B01230
      WRITE (*,*) 'ENTER THE ZONE CODE:'                                   # B01240
      READ (*,*) IPLOT                                                     # B01250
      END IF                                                               # B01260
      END IF                                                               # B01270
      WRITE (*,*)'ENTER NAME OF ENDPOINT FILE (<CR>="ENDPOINT"):'          # B01280
      READ (*,5010) FNAME                                                  # B01290
      IF (FNAME.EQ.' ') FNAME='ENDPOINT'                                   # B01300
      IUCF=I3                                                              # B01310
      CALL OPNFIL (I3,FNAME,1,I7,0,1)                                      # B01320
      IF(ITYPE.EQ.1) THEN                                                  # B01330
      WRITE (*,*)'ENTER NAME OF PATHLINE FILE (<CR>="PATHLINE"):'          # B01340
      READ (*,5010) FNAME                                                  # B01350
      IF (FNAME.EQ.' ') FNAME='PATHLINE'                                   # B01360
      IUL=I2                                                               # B01370
      CALL OPNFIL (I2,FNAME,1,I7,0,1)                                      # B01380
      CALL COUNTP (I3,NPART)                                               # B01390
      END IF                                                               # B01400
      IF(ITYPE.EQ.5) THEN                                                  # B01410
      WRITE (*,*)'ENTER NAME OF TIMESERIES FILE (<CR>="TIMESERS"):'        # B01420
      READ (*,5010) FNAME                                                  # B01430
```

```
      IF (FNAME.EQ.' ') FNAME='TIMESERS'                                # B01440
      CALL OPNFIL (I4,FNAME,1,I7,0,1)                                   # B01450
      WRITE (*,*)'HOW MANY TIME STEPS DO YOU WANT TO PLOT ?'            # B01460
      WRITE (*,*)'  (YOU MAY PLOT UP TO 50 TIME STEPS)'                 # B01470
      WRITE (*,*)'  (TO PLOT ALL OF THE TIME STEPS, ENTER A NEGATIVE NUM# B01480
     1BER)'                                                             # B01490
      READ (*,*) NPSTPS                                                 # B01500
      IF(NPSTPS.GT.0) THEN                                              # B01510
      WRITE (*,*)'ENTER THE TIME STEP NUMBERS THAT YOU WANT TO PLOT:'   # B01520
      READ (*,*) (ISPLOT(N),N=1,NPSTPS)                                 # B01530
      END IF                                                            # B01540
      END IF                                                            # B01550
C                                                                       # B01560
      IF(IVIEW.EQ.1) THEN                                               # B01570
      WRITE (*,*)'ENTER GRID COORDINATES:'                              # B01580
      WRITE (*,*)'     MINIMUM COLUMN VALUE,  MAXIMUM COLUMN VALUE'     # B01590
      READ (*,*) JMIN,JMAX                                              # B01600
      IF (JMAX.LT.1.OR.JMAX.GT.NCOL) JMAX=NCOL                          # B01610
      WRITE (*,*)'     MINIMUM ROW VALUE,  MAXIMUM ROW VALUE'           # B01620
      READ (*,*) IMIN,IMAX                                              # B01630
      IF (IMAX.LT.1.OR.IMAX.GT.NROW) IMAX=NROW                          # B01640
      XMIN= XMN(JMIN)                                                   # B01650
      XMAX= XMX(JMAX)                                                   # B01660
      YMIN= YMN(IMAX)                                                   # B01670
      YMAX= YMX(IMIN)                                                   # B01680
      VEX=1.0E+0                                                        # B01690
      ELSE IF(IVIEW.EQ.2) THEN                                          # B01700
      WRITE (*,*)'ENTER GRID COORDINATES:'                              # B01710
      WRITE (*,*)'     MINIMUM ROW VALUE,  MAXIMUM ROW VALUE'           # B01720
      READ (*,*) IMIN,IMAX                                              # B01730
      IF (IMAX.LT.1.OR.IMAX.GT.NROW) IMAX=NROW                          # B01740
      WRITE (*,*)'     MINIMUM LAYER VALUE,  MAXIMUM LAYER VALUE'       # B01750
      READ (*,*) KMIN,KMAX                                              # B01760
      IF (KMAX.LT.1.OR.KMAX.GT.NLAY) KMAX=NLAY                          # B01770
      WRITE (*,*)'WHAT IS THE VERTICAL EXAGGERATION ?'                  # B01780
      READ (*,*) VEX                                                    # B01790
      DO 20 N=1,NLAY                                                    # B01800
      ZMN(N)=VEX*ZMN(N)                                                 # B01810
20    ZMX(N)=VEX*ZMX(N)                                                 # B01820
      YMIN= YMN(IMAX)                                                   # B01830
      YMAX= YMX(IMIN)                                                   # B01840
      ZMIN= ZMN(KMAX)                                                   # B01850
      ZMAX= ZMX(KMIN)                                                   # B01860
      CALL NEWIB (2,IBOUND,IBYZ,NCOL,NROW,NLAY,NROW,JCOL,HEAD)          # B01870
      ELSE IF(IVIEW.EQ.3) THEN                                          # B01880
      WRITE (*,*)'ENTER GRID COORDINATES:'                              # B01890
      WRITE (*,*)'     MINIMUM COLUMN VALUE,  MAXIMUM COLUMN VALUE'     # B01900
      READ (*,*) JMIN,JMAX                                              # B01910
      IF (JMAX.LT.1.OR.JMAX.GT.NCOL) JMAX=NCOL                          # B01920
      WRITE (*,*)'     MINIMUM LAYER VALUE,  MAXIMUM LAYER VALUE'       # B01930
      READ (*,*) KMIN,KMAX                                              # B01940
      IF (KMAX.LT.1.OR.KMAX.GT.NLAY) KMAX=NLAY                          # B01950
      WRITE (*,*)'WHAT IS THE VERTICAL EXAGGERATION ?'                  # B01960
      READ (*,*) VEX                                                    # B01970
      DO 30 N=1,NLAY                                                    # B01980
      ZMN(N)=VEX*ZMN(N)                                                 # B01990
30    ZMX(N)=VEX*ZMX(N)                                                 # B02000
      XMIN= XMN(JMIN)                                                   # B02010
      XMAX= XMX(JMAX)                                                   # B02020
      ZMIN= ZMN(KMAX)                                                   # B02030
      ZMAX= ZMX(KMIN)                                                   # B02040
      CALL NEWIB (3,IBOUND,IBXZ,NCOL,NROW,NLAY,NCOL,IROW,HEAD)          # B02050
      END IF                                                            # B02060
C                                                                       # B02070
      IF(IVIEW.EQ.1) THEN                                               # B02080
      CALL SCPLOT (PLONG,PL,PSHORT,PS,IPAGE,XMIN,XMAX,YMIN,YMAX,XL,     # B02090
     1XR,YB,YT,MODEL,JUNITS,IA)                                         # B02100
      ELSE IF(IVIEW.EQ.2) THEN                                          # B02110
      CALL SCPLOT (PLONG,PL,PSHORT,PS,IPAGE,YMIN,YMAX,ZMIN,ZMAX,XL,     # B02120
     1XR,YB,YT,MODEL,JUNITS,IA)                                         # B02130
      ELSE IF(IVIEW.EQ.3) THEN                                          # B02140
      CALL SCPLOT (PLONG,PL,PSHORT,PS,IPAGE,XMIN,XMAX,ZMIN,ZMAX,XL,     # B02150
```

```
     1XR,YB,YT,MODEL,JUNITS,IA)                                          # B02160
      END IF                                                             # B02170
C                                                                        # B02180
C  CHANGE ZONE CODES IN IBOUND ARRAY                                     # B02190
C                                                                        # B02200
      IF (ITYPE.NE.5) THEN                                               # B02210
      MES= 'DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND AR# B02220
     1RAY ?'                                                             # B02230
      CALL YESNO (MES,IA,IANS)                                           # B02240
      IF (IANS.EQ.1) THEN                                                # B02250
40    WRITE (*,*) 'WHAT TYPE OF CHANGE DO YOU WANT TO MAKE ?'            # B02260
      WRITE (*,*) '   1 = CHANGE AN ENTIRE LAYER'                        # B02270
      WRITE (*,*) '   2 = CHANGE AN INDIVIDUAL CELL'                     # B02280
      WRITE (*,*) '   3 = CHANGE ALL CELLS IN A BLOCK OF CELLS'          # B02290
      READ (*,*) IANS                                                    # B02300
      IF(IANS.EQ.1) THEN                                                 # B02310
      WRITE (*,*) 'ENTER THE LAYER NUMBER:'                              # B02320
      READ (*,*) NLAYER                                                  # B02330
      WRITE (*,*) 'ENTER THE NEW ZONE CODE:'                             # B02340
      READ (*,*) NUM                                                     # B02350
      DO 50  I=1,NROW                                                    # B02360
      DO 50  J=1,NCOL                                                    # B02370
      ITEMP=IBOUND(J,I,NLAYER)                                           # B02380
      IF(ITEMP.NE.0) THEN                                                # B02390
      IBOUND(J,I,NLAYER)=NUM                                             # B02400
      IF(ITEMP.LT.0) IBOUND(J,I,NLAYER)= -IBOUND(J,I,NLAYER)             # B02410
      END IF                                                             # B02420
50    CONTINUE                                                           # B02430
      ELSE IF (IANS.EQ.2) THEN                                           # B02440
      WRITE (*,*) 'ENTER THE CELL INDICES:  J  I  K'                     # B02450
      READ (*,*) J,I,K                                                   # B02460
      WRITE (*,*) 'ENTER THE NEW ZONE CODE:'                             # B02470
      READ (*,*) NUM                                                     # B02480
      ITEMP=IBOUND(J,I,K)                                                # B02490
      IF(ITEMP.NE.0) THEN                                                # B02500
      IBOUND(J,I,K)=NUM                                                  # B02510
      IF(ITEMP.LT.0) IBOUND(J,I,K)= -IBOUND(J,I,K)                       # B02520
      END IF                                                             # B02530
      ELSE IF (IANS.EQ.3) THEN                                           # B02540
      WRITE (*,*) 'ENTER THE BOUNDARIES OF THE BLOCK OF CELLS:'          # B02550
      WRITE (*,*) '    MIN COLUMN, MAX COLUMN, MIN ROW, MAX ROW, MIN LAY# B02560
     1ER, MAX LAYER'                                                     # B02570
      READ (*,*) JJ1,JJ2,II1,II2,KK1,KK2                                # B02580
      IF (JJ2.LT.1.OR.JJ2.GT.NCOL) JJ2=NCOL                              # B02590
      IF (II2.LT.1.OR.II2.GT.NROW) II2=NROW                              # B02600
      IF (KK2.LT.1.OR.KK2.GT.NLAY) KK2=NLAY                              # B02610
      WRITE (*,*) 'ENTER THE NEW ZONE CODE:'                             # B02620
      READ (*,*) NUM                                                     # B02630
      DO 60  K=KK1,KK2                                                   # B02640
      DO 60  I=II1,II2                                                   # B02650
      DO 60  J=JJ1,JJ2                                                   # B02660
      ITEMP=IBOUND(J,I,K)                                                # B02670
      IF(ITEMP.NE.0) THEN                                                # B02680
      IBOUND(J,I,K)=NUM                                                  # B02690
      IF(ITEMP.LT.0) IBOUND(J,I,K)= -IBOUND(J,I,K)                       # B02700
      END IF                                                             # B02710
60    CONTINUE                                                           # B02720
      END IF                                                             # B02730
      MES= 'DO YOU WANT TO CHANGE SOME MORE ZONE CODES ?'                # B02740
      CALL YESNO (MES,IA,IANS)                                           # B02750
      IF (IANS.EQ.1) GO TO 40                                            # B02760
      END IF                                                             # B02770
      END IF                                                             # B02780
C                                                                        # B02790
C                                                                        # B02800
      MES= 'DO YOU WANT A COLOR PLOT ?'                                  # B02810
      CALL YESNO (MES,IA,IANS)                                           # B02820
      IF (IANS.EQ.0) THEN                                                # B02830
      NCLR= -1                                                           # B02840
      IF (IDIR.NE.0) THEN                                                # B02850
      WRITE (*,*)'WHAT TYPE OF LINE PATTERNS SHOULD BE USED ?'           # B02860
      WRITE (*,*)'   0 = CYCLE THROUGH LINE PATTERNS (SOLID, DASH, DOT)'# B02870
```

```
      WRITE (*,*)'   1 = USE A SINGLE LINE PATTERN FOR ALL PATH LINES'  # B02880
      READ (*,*) NNN                                                   # B02890
      IF(NNN.EQ.0) THEN                                                # B02900
      NCLR= -4                                                         # B02910
      ELSE                                                             # B02920
      WRITE (*,*)'ENTER THE LINE PATTERN:'                             # B02930
      WRITE (*,*)'   1 = SOLID'                                        # B02940
      WRITE (*,*)'   2 = DASHED'                                       # B02950
      WRITE (*,*)'   3 = DOTTED'                                       # B02960
      READ (*,*) NNN                                                   # B02970
      IF(NNN.LT.1.OR.NNN.GT.3) NNN=1                                   # B02980
      NCLR= -NNN                                                       # B02990
      END IF                                                           # B03000
      END IF                                                           # B03010
      ELSE IF(IANS.EQ.1) THEN                                          # B03020
      WRITE (*,*)'HOW SHOULD COLORS BE CHOSEN FOR DIFFERENT DATA GROUPS # B03030
     1?'                                                               # B03040
      WRITE (*,*)'   0 = CYCLE THROUGH COLORS'                         # B03050
      WRITE (*,*)'   1 = ALL LINES AND (OR) DATA POINTS PLOTTED IN ONE  # B03060
     1COLOR'                                                           # B03070
      READ (*,*) NCLR                                                  # B03080
      IF(NCLR.GT.0) THEN                                               # B03090
      WRITE (*,*)'ENTER THE COLOR:'                                    # B03100
      WRITE (*,*)'   1 = BLACK/WHITE'                                  # B03110
      WRITE (*,*)'   2 = RED'                                          # B03120
      WRITE (*,*)'   3 = GREEN'                                        # B03130
      WRITE (*,*)'   4 = BLUE'                                         # B03140
      READ (*,*) NCLR                                                  # B03150
      IF(NCLR.LT.1.OR.NCLR.GT.4) NCLR=1                                # B03160
      END IF                                                           # B03170
      END IF                                                           # B03180
C                                                                      # B03190
      MES= 'DRAW THE PAGE BORDER ?'                                    # B03200
      CALL YESNO (MES,IA,IBRDR)                                        # B03210
C                                                                      # B03220
      IF(MODEL.EQ.1) CALL TK4010 (960)                                 # B03230
      IF(MODEL.EQ.2) CALL TK41(4105)                                   # B03240
      IF(MODEL.EQ.3) CALL TK41(4115)                                   # B03250
      IF(MODEL.EQ.4) CALL TK41(4107)                                   # B03260
      IF(MODEL.EQ.5) CALL COMPRS                                       # B03270
      IF(IBRDR.EQ.0) CALL NOBRDR                                       # B03280
      CALL PAGE(PLONG,PSHORT)                                          # B03290
      CALL PHYSOR(0.75,1.5)                                            # B03300
      CALL GRACE (1.5)                                                 # B03310
      CALL AREA2D(PL,PS)                                               # B03320
      PX=PL                                                            # B03330
      PY=PX                                                            # B03340
      DX=(XR-XL)/10.                                                   # B03350
      DY=(YT-YB)/10.                                                   # B03360
      CALL GRAF (XL,DX,XR,YB,DY,YT)                                    # B03370
C                                                                      # B03380
      CALL NOTATE (XL,XR,YB,YT,PX,PY,JUNITS,TITLE,VEX,IVIEW)           # B03390
      IF(IVIEW.EQ.1) THEN                                              # B03400
      CALL DGRID (IMIN,IMAX,JMIN,JMAX,XMN,XMX,YMN,YMX,IBOUND(1,1,LAYER),# B03410
     1 NCOL,NROW,IGL,0)                                                # B03420
      IF(IUNIT(6).GT.0) CALL PGHB(IMIN,IMAX,JMIN,JMAX,1,NLAY,          # B03430
     1 XMN,XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(6),NCLR,I7,  # B03440
     2 IPROJ,LAYER)                                                    # B03450
      IF(IUNIT(5).GT.0) CALL PDRAIN(IMIN,IMAX,JMIN,JMAX,1,NLAY,        # B03460
     1 XMN,XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(5),NCLR,I7,  # B03470
     2 IPROJ,LAYER)                                                    # B03480
      IF(IUNIT(4).GT.0) CALL PRIVER(IMIN,IMAX,JMIN,JMAX,1,NLAY,        # B03490
     1 XMN,XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IUNIT(4),NCLR,IVIEW,     # B03500
     2 IBUFF,I7)                                                       # B03510
      IF(IUNIT(2).GT.0) CALL PWELLS(IMIN,IMAX,JMIN,JMAX,1,NLAY,        # B03520
     1 XMN,XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(2),NCLR,I7,  # B03530
     2 IPROJ,LAYER)                                                    # B03540
      ELSE IF(IVIEW.EQ.2) THEN                                         # B03550
      CALL DGRID (KMIN,KMAX,IMIN,IMAX,YMN,YMX,ZMN,ZMX,IBYZ,NROW,NLAY,  # B03560
     1IGL,1)                                                           # B03570
      CALL PCBEDS (NCON,NLAY,ZMN,ZMX,IMIN,IMAX,YMN,YMX,NROW,IBYZ,      # B03580
     1 KMIN,KMAX)                                                      # B03590
```

```
      IF(IUNIT(6).GT.0) CALL PGHB(IMIN,IMAX,JCOL,JCOL,KMIN,KMAX,XMN,     # B03600
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(6),NCLR,I7,        # B03610
     2 IPROJ,JCOL)                                                       # B03620
      IF(IUNIT(5).GT.0) CALL PDRAIN(IMIN,IMAX,JCOL,JCOL,KMIN,KMAX,XMN,  # B03630
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(5),NCLR,I7,        # B03640
     2 IPROJ,JCOL)                                                       # B03650
      IF(IUNIT(4).GT.0) CALL PRIVER(IMIN,IMAX,JCOL,JCOL,KMIN,KMAX,XMN,  # B03660
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IUNIT(4),NCLR,IVIEW,IBUFF,I7) # B03670
      IF(IUNIT(2).GT.0) CALL PWELLS(IMIN,IMAX,JCOL,JCOL,KMIN,KMAX,XMN,  # B03680
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(2),NCLR,I7,        # B03690
     2 IPROJ,JCOL)                                                       # B03700
      ELSE IF(IVIEW.EQ.3) THEN                                           # B03710
      CALL DGRID (KMIN,KMAX,JMIN,JMAX,XMN,XMX,ZMN,ZMX,IBXZ,NCOL,NLAY,    # B03720
     1IGL,0)                                                             # B03730
      CALL PCBEDS (NCON,NLAY,ZMN,ZMX,JMIN,JMAX,XMN,XMX,NCOL,IBXZ,        # B03740
     1 KMIN,KMAX)                                                        # B03750
      IF(IUNIT(6).GT.0) CALL PGHB(IROW,IROW,JMIN,JMAX,KMIN,KMAX,XMN,     # B03760
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(6),NCLR,I7,        # B03770
     5 IPROJ,IROW)                                                       # B03780
      IF(IUNIT(5).GT.0) CALL PDRAIN(IROW,IROW,JMIN,JMAX,KMIN,KMAX,XMN,   # B03790
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(5),NCLR,I7,        # B03800
     5 IPROJ,IROW)                                                       # B03810
      IF(IUNIT(4).GT.0) CALL PRIVER(IROW,IROW,JMIN,JMAX,KMIN,KMAX,XMN,   # B03820
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IUNIT(4),NCLR,IVIEW,IBUFF,I7) # B03830
      IF(IUNIT(2).GT.0) CALL PWELLS(IROW,IROW,JMIN,JMAX,KMIN,KMAX,XMN,   # B03840
     1 XMX,YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUNIT(2),NCLR,I7,        # B03850
     5 IPROJ,IROW)                                                       # B03860
      END IF                                                             # B03870
C                                                                        # B03880
      IF(ITYPE.EQ.1.AND.IVIEW.EQ.1) CALL LINES(IUL,IUCF,XMIN,XMAX,       # B03890
     1YMIN,YMAX,ZMN,ZMX,IDIR,IBOUND,NCOL,NROW,NLAY,IVIEW,VEX,IPTS,       # B03900
     2IPROJ,LAYER,NCLR,ISKIP,TMAX,IZPART,NPART,NPART2)                   # B03910
      IF(ITYPE.EQ.1.AND.IVIEW.EQ.2) CALL LINES(IUL,IUCF,YMIN,YMAX,       # B03920
     1ZMIN,ZMAX,ZMN,ZMX,IDIR,IBOUND,NCOL,NROW,NLAY,IVIEW,VEX,IPTS,       # B03930
     2IPROJ,JCOL,NCLR,ISKIP,TMAX,IZPART,NPART,NPART2)                    # B03940
      IF(ITYPE.EQ.1.AND.IVIEW.EQ.3) CALL LINES(IUL,IUCF,XMIN,XMAX,       # B03950
     1ZMIN,ZMAX,ZMN,ZMX,IDIR,IBOUND,NCOL,NROW,NLAY,IVIEW,VEX,IPTS,       # B03960
     2IPROJ,IROW,NCLR,ISKIP,TMAX,IZPART,NPART,NPART2)                    # B03970
      IF(ITYPE.EQ.2) CALL POINTS (IUCF,1,0,IMIN,IMAX,JMIN,JMAX,XMN,XMX,  # B03980
     1YMN,YMX,IBOUND,NCOL,NROW,NLAY,IPLOT,NCLR)                          # B03990
      IF(ITYPE.EQ.3) CALL POINTS (IUCF,1,1,IMIN,IMAX,JMIN,JMAX,XMN,XMX,  # B04000
     1YMN,YMX,IBOUND,NCOL,NROW,NLAY,IPLOT,NCLR)                          # B04010
      IF(ITYPE.EQ.4) CALL POINTS (IUCF,0,1,IMIN,IMAX,JMIN,JMAX,XMN,XMX,  # B04020
     1YMN,YMX,IBOUND,NCOL,NROW,NLAY,IPLOT,NCLR)                          # B04030
      IF(ITYPE.EQ.5) THEN                                                # B04040
      IF(IVIEW.EQ.1) THEN                                                # B04050
      CALL PTIMS (I4,XMIN,XMAX,YMIN,YMAX,ZMN,ZMX,NCOL,NROW,              # B04060
     1NLAY,IVIEW,VEX,NPSTPS,ISPLOT,51,IPROJ,LAYER,NCLR)                  # B04070
      ELSE IF (IVIEW.EQ.2) THEN                                          # B04080
      CALL PTIMS (I4,YMIN,YMAX,ZMIN,ZMAX,ZMN,ZMX,NCOL,NROW,              # B04090
     1NLAY,IVIEW,VEX,NPSTPS,ISPLOT,51,IPROJ,JCOL,NCLR)                   # B04100
      ELSE IF (IVIEW.EQ.3) THEN                                          # B04110
      CALL PTIMS (I4,XMIN,XMAX,ZMIN,ZMAX,ZMN,ZMX,NCOL,NROW,              # B04120
     1NLAY,IVIEW,VEX,NPSTPS,ISPLOT,51,IPROJ,IROW,NCLR)                   # B04130
      END IF                                                             # B04140
      END IF                                                             # B04150
      CALL ENDPL(0)                                                      # B04160
      CALL DONEPL                                                        # B04170
      RETURN                                                             # B04180
      END                                                                # B04190
C-----END OF ROUTINE-------------------------------------------------------# B04200
```

```
C                                                                       # C00010
C---Version 1.0   July 21, 1989                                         # C00020
C***********************************************************************# C00030
C                                                                       # C00040
C                          POINTS                                       # C00050
C                                                                       # C00060
C  THIS ROUTINE PLOTS POINTS                                            # C00070
C                                                                       # C00080
C***********************************************************************# C00090
C                                                                       # C00100
      SUBROUTINE POINTS (IU,IZN,IEPT,IMIN,IMAX,JMIN,JMAX,XMN,XMX,       # C00110
     1YMN,YMX,IBOUND,NCOL,NROW,NLAY,IPLOT,NCLR)                         # C00120
      DIMENSION XMN(NCOL),XMX(NCOL),YMN(NROW),YMX(NROW),                # C00130
     1 IBOUND(NCOL,NROW,NLAY)                                           # C00140
      IOLD=1                                                            # C00150
      CALL MARKER (3)                                                   # C00160
      CALL SCLPIC (0.05)                                                # C00170
10     READ(IU,*,END=40) IZL,JLAST,ILAST,KLAST,XLAST,YLAST,ZLAST,ZLLAST,# C00180
     1T,XFRST,YFRST,ZLFRST,JFRST,IFRST,KFRST,IZF                        # C00190
      IZL= ABS(IBOUND(JLAST,ILAST,KLAST))                               # C00200
      IZF= ABS(IBOUND(JFRST,IFRST,KFRST))                               # C00210
      X=XLAST                                                           # C00220
      Y=YLAST                                                           # C00230
      J=JLAST                                                           # C00240
      I=ILAST                                                           # C00250
      K=KLAST                                                           # C00260
      IZONE=IZF                                                         # C00270
      IF(IZN.EQ.1) IZONE=IZL                                            # C00280
      DLR=XMX(J)-XMN(J)                                                 # C00290
      DLC=YMX(I)-YMN(I)                                                 # C00300
      IF(IEPT.EQ.0) THEN                                                # C00310
      X=XFRST                                                           # C00320
      Y=YFRST                                                           # C00330
      J=JFRST                                                           # C00340
      I=IFRST                                                           # C00350
      K=KFRST                                                           # C00360
      DLR=(XMX(J)-XMN(J))/6.0                                           # C00370
      DLC=(YMX(I)-YMN(I))/6.0                                           # C00380
      END IF                                                            # C00390
      ISKIP=0                                                           # C00400
      IF(IZONE.LE.1.AND.IEPT.EQ.0) ISKIP=1                              # C00410
      IF(ISKIP.EQ.0) THEN                                               # C00420
      IF(J.LT.JMIN.OR.J.GT.JMAX) GO TO 10                               # C00430
      IF(I.LT.IMIN.OR.I.GT.IMAX) GO TO 10                               # C00440
      IF(NCLR.GE.0) THEN                                                # C00450
      KCLR=IZONE                                                        # C00460
      IF(NCLR.GT.0) KCLR=NCLR                                           # C00470
      IF(IZONE.NE.IOLD) CALL PKCLR (KCLR)                               # C00480
      END IF                                                            # C00490
      IF(IEPT.EQ.0) THEN                                                # C00500
      ISYM=IZONE                                                        # C00510
20    IF(ISYM.LE.4) GO TO 30                                            # C00520
      ISYM=ISYM-3                                                       # C00530
      GO TO 20                                                          # C00540
30    CONTINUE                                                          # C00550
      CALL SYMBL (X,Y,DLR,DLC,ISYM)                                     # C00560
      ELSE                                                              # C00570
      IPLT=1                                                            # C00580
      IF(IPLOT.GT.0) THEN                                               # C00590
      IF(IBOUND(JLAST,ILAST,KLAST).NE.IPLOT) IPLT=0                     # C00600
      END IF                                                            # C00610
      IF(IPLT.EQ.1) CALL CURVE (X,Y,1,-1)                               # C00620
      END IF                                                            # C00630
      IOLD=IZONE                                                        # C00640
      END IF                                                            # C00650
      GO TO 10                                                          # C00660
40    CONTINUE                                                          # C00670
      RETURN                                                            # C00680
      END                                                               # C00690
C-----END OF ROUTINE-------------------------------------------------------# C00700
```

```
C                                                                       # D00010
C---Version 1.0   July 21, 1989                                         # D00020
C***********************************************************************# D00030
C                                                                       # D00040
C                           LINES                                       # D00050
C                                                                       # D00060
C  THIS ROUTINE DRAWS PATH LINES                                        # D00070
C                                                                       # D00080
C***********************************************************************# D00090
C                                                                       # D00100
      SUBROUTINE LINES (IU,IUEP,XMIN,XMAX,YMIN,YMAX,ZMN,ZMX,IDIR,       # D00110
     1IBOUND,NCOL,NROW,NLAY,IVIEW,VEX,IPTS,IPROJ,NSEC,NCLR,ISKIP,TMAX,  # D00120
     2IZPART,NPART,NPART2)                                              # D00130
      DIMENSION IBOUND(NCOL,NROW,NLAY),ZMN(NLAY),ZMX(NLAY),             # D00140
     1IZPART(NPART2)                                                    # D00150
      DIMENSION X(2),Y(2)                                               # D00160
C                                                                       # D00170
      ISKP=0                                                            # D00180
      CALL EZONES (IBOUND,IZPART,NCOL,NROW,NLAY,NPART2,IUEP)            # D00190
      IF (IDIR.EQ.0.AND.ISKIP.EQ.1) ISKP=1                              # D00200
      CALL HEIGHT (0.08)                                                # D00210
      CALL MARKER (16)                                                  # D00220
      IOLD=1                                                            # D00230
      IF(NCLR.GE.0) THEN                                                # D00240
      KCLR=NCLR                                                         # D00250
      IF(KCLR.EQ.0) KCLR=1                                              # D00260
      CALL PKCLR (KCLR)                                                 # D00270
      END IF                                                            # D00280
      IF (IDIR.EQ.1.AND.NCLR.LT.-1.AND.NCLR.GT.-4) THEN                 # D00290
      NNN= -NCLR                                                        # D00300
      CALL PKPAT (NNN)                                                  # D00310
      END IF                                                            # D00320
      IC=0                                                              # D00330
      IPC=1                                                             # D00340
C      WRITE(6,100) IDIR                                                # D00350
10     CONTINUE                                                         # D00360
      IF(IVIEW.EQ.1) THEN                                               # D00370
      READ(IU,*,END=30) IP,X(2),Y(2),ZZL,ZZ,TT,J,I,K                    # D00380
      LL=K                                                              # D00390
      ELSE IF (IVIEW.EQ.2) THEN                                         # D00400
      READ(IU,*,END=30) IP,XX,X(2),ZZL,ZZ,TT,J,I,K                      # D00410
      LL=J                                                              # D00420
      IF(ZZL.GE.0.0) Y(2)= (1.0-ZZL)*ZMN(K) + ZZL*ZMX(K)                # D00430
      IF(ZZL.LT.0.0) Y(2)= (1.0E+0+ZZL)*ZMN(K) - ZZL*ZMX(K+1)           # D00440
      ELSE IF (IVIEW.EQ.3) THEN                                         # D00450
      READ(IU,*,END=30) IP,X(2),YY,ZZL,ZZ,TT,J,I,K                      # D00460
      LL=I                                                              # D00470
      IF(ZZL.GE.0.0) Y(2)= (1.0-ZZL)*ZMN(K) + ZZL*ZMX(K)                # D00480
      IF(ZZL.LT.0.0) Y(2)= (1.0E+0+ZZL)*ZMN(K) - ZZL*ZMX(K+1)           # D00490
      END IF                                                            # D00500
      IF(IC.EQ.0.OR.IP.NE.IPC) THEN                                     # D00510
      IF(IDIR.EQ.1) THEN                                                # D00520
      IZONE= IZPART(2*IP-1)                                             # D00530
      ELSE                                                              # D00540
      IZONE= IZPART(2*IP)                                               # D00550
      END IF                                                            # D00560
      IZONE= ABS(IZONE)                                                 # D00570
      IF(NCLR.EQ.0) THEN                                                # D00580
      IF(IZONE.NE.IOLD.OR.IC.EQ.0) CALL PKCLR (IZONE)                   # D00590
      ELSE IF (NCLR.EQ.-4) THEN                                         # D00600
      IF(IZONE.NE.IOLD.OR.IC.EQ.0) CALL PKPAT (IZONE)                   # D00610
      END IF                                                            # D00620
      IOLD=IZONE                                                        # D00630
      X(1)=X(2)                                                         # D00640
      Y(1)=Y(2)                                                         # D00650
      LSEC=LL                                                           # D00660
      IC=1                                                              # D00670
      IPC=IP                                                            # D00680
      GO TO 10                                                          # D00690
      END IF                                                            # D00700
      IF(X(2).GT.XMAX.OR.X(1).GT.XMAX) GO TO 20                         # D00710
      IF(X(2).LT.XMIN.OR.X(1).LT.XMIN) GO TO 20                         # D00720
```

```
      IF(Y(2).GT.YMAX.OR.Y(1).GT.YMAX) GO TO 20                          # D00730
      IF(Y(2).LT.YMIN.OR.Y(1).LT.YMIN) GO TO 20                          # D00740
      IF (ISKP.EQ.0.OR.IZONE.GT.1) THEN                                  # D00750
      IF (IPROJ.EQ.1) THEN                                               # D00760
      IF (LSEC.EQ.NSEC) THEN                                             # D00770
      IF (ABS(TT).LE.TMAX) THEN                                          # D00780
      CALL CURVE (X,Y,2,0)                                               # D00790
      IF(TT.LT.0.0.AND.IPTS.EQ.1) CALL CURVE (X(2),Y(2),1,-1)            # D00800
      END IF                                                             # D00810
      END IF                                                             # D00820
      ELSE IF(IPROJ.EQ.0) THEN                                           # D00830
      IF (ABS(TT).LE.TMAX) THEN                                          # D00840
      CALL CURVE (X,Y,2,0)                                               # D00850
      IF(TT.LT.0.0.AND.IPTS.EQ.1) CALL CURVE (X(2),Y(2),1,-1)            # D00860
      END IF                                                             # D00870
      END IF                                                             # D00880
      END IF                                                             # D00890
20    X(1)=X(2)                                                          # D00900
      Y(1)=Y(2)                                                          # D00910
      LSEC=LL                                                            # D00920
      GO TO 10                                                           # D00930
30    CONTINUE                                                           # D00940
      IF (NCLR.LT.0) THEN                                                # D00950
      CALL RESET ('DASH')                                                # D00960
      CALL RESET ('DOT')                                                 # D00970
      END IF                                                             # D00980
      CALL RESET ('HEIGHT')                                              # D00990
      RETURN                                                             # D01000
      END                                                                # D01010
C-----END OF ROUTINE-------------------------------------------------------# D01020
```

```
C                                                                       # E00010
C---Version 1.0   July 21, 1989                                         # E00020
C***********************************************************************# E00030
C                                                                       # E00040
C                              EZONES                                   # E00050
C                                                                       # E00060
C  THIS ROUTINE READS THE ENDPOINT FILE TO GET THE ZONE CODES           # E00070
C  FOR THE INITIAL AND FINAL PARTICLE LOCATIONS. IT PUTS THE ZONE       # E00080
C  CODES IN THE ARRAY "IZPART"                                          # E00090
C                                                                       # E00100
C*********************************************************************** # E00110
C                                                                       # E00120
      SUBROUTINE EZONES (IBOUND,IZPART,NCOL,NROW,NLAY,NPART2,IUEP)      # E00130
      DIMENSION IBOUND(NCOL,NROW,NLAY),IZPART(NPART2)                   # E00140
C                                                                       # E00150
      KOUNT=0                                                           # E00160
10    READ(IUEP,*,END=20 ) IZL,JLAST,ILAST,KLAST,XLAST,YLAST,ZLAST,     # E00170
     1ZLLAST,T,XFRST,YFRST,ZLFRST,JFRST,IFRST,KFRST,IZF                 # E00180
      KOUNT=KOUNT+1                                                     # E00190
      N= 2*KOUNT - 1                                                    # E00200
      IZPART(N)= IBOUND (JFRST,IFRST,KFRST)                             # E00210
      IZPART(N+1)= IBOUND (JLAST,ILAST,KLAST)                           # E00220
      GO TO 10                                                          # E00230
20    CONTINUE                                                          # E00240
      RETURN                                                            # E00250
      END                                                               # E00260
C-----END OF ROUTINE-------------------------------------------------- # E00270
```

```
C                                                                       # F00010
C***********************************************************************# F00020
C                                                                       # F00030
C                          PWELLS                                       # F00040
C                                                                       # F00050
C  THIS ROUTINE READS WELL DATA AND PLOTS WELLS                         # F00060
C                                                                       # F00070
C***********************************************************************# F00080
C                                                                       # F00090
      SUBROUTINE PWELLS (IMIN,IMAX,JMIN,JMAX,KMIN,KMAX,XMN,XMX,        # F00100
     1YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUWELL,NCLR,I7,IPROJ,ISLICE) # F00110
      DIMENSION XMN(NCOL),XMX(NCOL),YMN(NROW),YMX(NROW),ZMN(NLAY),      # F00120
     1ZMX(NLAY)                                                         # F00130
C                                                                       # F00140
      IV= IVIEW*IPROJ                                                   # F00150
      IF(NCLR.GE.0) THEN                                                # F00160
      CALL PKCLR (2)                                                    # F00170
      END IF                                                            # F00180
      WRITE(I7,5000)                                                    # F00190
5000  FORMAT('-------------------------------------------------------------# F00200
     1----------------------')                                          # F00210
      WRITE(I7,*) 'WELL DATA BEING READ...'                             # F00220
      READ(IUWELL,5010) MXW,ID                                          # F00230
      READ(IUWELL,5010) NWELL                                           # F00240
5010   FORMAT(2I10)                                                     # F00250
      IF(IVIEW.GT.1) CALL DASH                                          # F00260
      DO 10 N=1,NWELL                                                   # F00270
      READ(IUWELL,5020) K,I,J,Q,IFACE                                   # F00280
      IF(IFACE.NE.0) GO TO 10                                           # F00290
      IF(I.LT.IMIN.OR.I.GT.IMAX) GO TO 10                               # F00300
      IF(J.LT.JMIN.OR.J.GT.JMAX) GO TO 10                               # F00310
      IF(K.LT.KMIN.OR.K.GT.KMAX) GO TO 10                               # F00320
      IF (IV.EQ.1.AND.K.NE.ISLICE) GO TO 10                             # F00330
      IF (IVIEW.EQ.2.AND.J.NE.ISLICE) GO TO 10                          # F00340
      IF (IVIEW.EQ.3.AND.I.NE.ISLICE) GO TO 10                          # F00350
      IF(IVIEW.EQ.1) THEN                                               # F00360
      X=(XMX(J)+XMN(J))/2.0                                             # F00370
      Y=(YMX(I)+YMN(I))/2.0                                             # F00380
      DX=XMX(J)-XMN(J)                                                  # F00390
      DY=YMX(I)-YMN(I)                                                  # F00400
      ELSE IF(IVIEW.EQ.2) THEN                                          # F00410
      X=(YMX(I)+YMN(I))/2.0                                             # F00420
      Y=(ZMX(K)+ZMN(K))/2.0                                             # F00430
      DX=YMX(I)-YMN(I)                                                  # F00440
      DY=ZMX(K)-ZMN(K)                                                  # F00450
      ELSE IF(IVIEW.EQ.3) THEN                                          # F00460
      X=(XMX(J)+XMN(J))/2.0                                             # F00470
      Y=(ZMX(K)+ZMN(K))/2.0                                             # F00480
      DX=XMX(J)-XMN(J)                                                  # F00490
      DY=ZMX(K)-ZMN(K)                                                  # F00500
      END IF                                                            # F00510
      IF (IVIEW.EQ.1) THEN                                              # F00520
      CALL SYMBL (X,Y,DX,DY,1)                                          # F00530
      ELSE                                                              # F00540
      CALL SYMBL (X,Y,DX,DY,0)                                          # F00550
      END IF                                                            # F00560
10    CONTINUE                                                          # F00570
      IF (IVIEW.GT.1) CALL RESET ('DASH')                               # F00580
5020   FORMAT(3I10,F10.0,I10)                                           # F00590
      WRITE(I7,*) '  WELL DATA HAS BEEN READ...'                        # F00600
      IF (NCLR.GE.0) CALL PKCLR (1)                                     # F00610
      RETURN                                                            # F00620
      END                                                               # F00630
C-----END OF ROUTINE-------------------------------------------------------# F00640
```

```
C                                                                       # G00010
C---Version 1.0   July 21, 1989                                         # G00020
C***********************************************************************# G00030
C                                                                       # G00040
C                              SYMBL                                    # G00050
C                                                                       # G00060
C  THIS ROUTINE DRAWS SYMBOLS                                           # G00070
C                                                                       # G00080
C***********************************************************************# G00090
C                                                                       # G00100
      SUBROUTINE SYMBL (XC,YC,DELR,DELC,ISYM)                           # G00110
      DIMENSION X(20),Y(20),XX(2),YY(2)                                 # G00120
      DMIN=DELR                                                         # G00130
      IF(DELC.LT.DELR) DMIN=DELC                                        # G00140
      IF(ISYM.EQ.0) THEN                                                # G00150
      X(1)= XC-0.45*DELR                                                # G00160
      Y(1)= YC-0.45*DELC                                                # G00170
      X(2)= XC+0.45*DELR                                                # G00180
      Y(2)= Y(1)                                                        # G00190
      X(3)= X(2)                                                        # G00200
      Y(3)= YC+0.45*DELC                                                # G00210
      X(4)= X(1)                                                        # G00220
      Y(4)= Y(3)                                                        # G00230
      X(5)= X(1)                                                        # G00240
      Y(5)= Y(1)                                                        # G00250
      CALL CURVE (X,Y,5,0)                                              # G00260
      ELSE IF(ISYM.EQ.1) THEN                                           # G00270
      X(1)= XC-0.1625*DMIN                                              # G00280
      Y(1)= YC-0.25*DMIN                                                # G00290
      X(2)= XC+0.1625*DMIN                                              # G00300
      Y(2)= Y(1)                                                        # G00310
      X(3)= XC+0.25*DMIN                                                # G00320
      Y(3)= YC-0.1625*DMIN                                              # G00330
      X(4)= X(3)                                                        # G00340
      Y(4)= YC+0.1625*DMIN                                              # G00350
      X(5)= XC+0.1625*DMIN                                              # G00360
      Y(5)= YC+0.25*DMIN                                                # G00370
      X(6)= XC-0.1625*DMIN                                              # G00380
      Y(6)= Y(5)                                                        # G00390
      X(7)= XC-0.25*DMIN                                                # G00400
      Y(7)= YC+0.1625*DMIN                                              # G00410
      X(8)= X(7)                                                        # G00420
      Y(8)= YC-0.1625*DMIN                                              # G00430
      X(9)= X(1)                                                        # G00440
      Y(9)= Y(1)                                                        # G00450
      CALL CURVE (X,Y,9,0)                                              # G00460
      ELSE IF(ISYM.EQ.2) THEN                                           # G00470
      X(1)= XC-0.5*DMIN                                                 # G00480
      Y(1)= YC+0.5*DMIN                                                 # G00490
      X(2)= XC+0.5*DMIN                                                 # G00500
      Y(2)= YC-0.5*DMIN                                                 # G00510
      CALL CURVE (X,Y,2,0)                                              # G00520
      YTEMP=Y(1)                                                        # G00530
      Y(1)=Y(2)                                                         # G00540
      Y(2)=YTEMP                                                        # G00550
      CALL CURVE (X,Y,2,0)                                              # G00560
      ELSE IF(ISYM.EQ.3) THEN                                           # G00570
      X(1)= XC-0.3*DMIN                                                 # G00580
      Y(1)= YC-0.2598*DMIN                                              # G00590
      X(2)= XC+0.3*DMIN                                                 # G00600
      Y(2)= Y(1)                                                        # G00610
      X(3)= XC                                                          # G00620
      Y(3)= YC+0.2598*DMIN                                              # G00630
      X(4)= X(1)                                                        # G00640
      Y(4)= Y(1)                                                        # G00650
      CALL CURVE (X,Y,4,0)                                              # G00660
      ELSE IF (ISYM.EQ.4) THEN                                          # G00670
      X(1)=XC                                                           # G00680
      X(2)=XC                                                           # G00690
      Y(1)=YC-0.5*DMIN                                                  # G00700
      Y(2)=YC+0.5*DMIN                                                  # G00710
      CALL CURVE (X,Y,2,0)                                              # G00720
```

```
      X(1)=XC-0.5*DMIN                                                  # G00730
      X(2)=XC+0.5*DMIN                                                  # G00740
      Y(1)=YC                                                           # G00750
      Y(2)=YC                                                           # G00760
      CALL CURVE (X,Y,2,0)                                              # G00770
      ELSE IF (ISYM.EQ.5) THEN                                          # G00780
      X(1)= XC + 0.4*DMIN                                               # G00790
      Y(1)= YC                                                          # G00800
      DA=6.28319/100.                                                   # G00810
      A=0.0                                                             # G00820
      DO 10 N=1,99                                                      # G00830
      A=A+DA                                                            # G00840
      X(2)= XC + 0.4*DMIN*COS(A)                                        # G00850
      Y(2)= YC + 0.4*DMIN*SIN(A)                                        # G00860
      CALL CURVE (X,Y,2,0)                                              # G00870
      X(1)=X(2)                                                         # G00880
      Y(1)=Y(2)                                                         # G00890
10    CONTINUE                                                          # G00900
      X(2)= XC + 0.4*DMIN                                               # G00910
      Y(2)= YC                                                          # G00920
      CALL CURVE (X,Y,2,0)                                              # G00930
      END IF                                                            # G00940
      RETURN                                                            # G00950
      END                                                               # G00960
C-----END OF ROUTINE----------------------------------------------------# G00970
```

```
C                                                                       # H00010
C---Version 1.0   July 21, 1989                                         # H00020
C***********************************************************************# H00030
C                                                                       # H00040
C                              YESNO                                    # H00050
C                                                                       # H00060
C  THIS ROUTINE PRINTS QUESTION PROMPT REQUIRING Y OR N ANSWER. Y AND N # H00070
C  ARE CONVERTED TO NUMERICAL 0 AND 1 RESPONSE                          # H00080
C                                                                       # H00090
C***********************************************************************# H00100
C                                                                       # H00110
      SUBROUTINE YESNO(MES,IU,IANS)                                     # H00120
      CHARACTER*80 MES                                                  # H00130
      CHARACTER*1 IA                                                    # H00140
      CHARACTER*10 ANSWR                                                # H00150
10    WRITE(IU,5000) MES                                                # H00160
      WRITE(IU,*) '    [ Y = YES;      N OR <CR> = NO ]'                # H00170
      READ(IU,'(A10)') ANSWR                                            # H00180
5000   FORMAT(A)                                                        # H00190
      IA= ' '                                                           # H00200
      DO 20 N=1,10                                                      # H00210
      IF (ANSWR(N:N).NE.' ') THEN                                       # H00220
      IA= ANSWR(N:N)                                                    # H00230
      GO TO 30                                                          # H00240
      END IF                                                            # H00250
20    CONTINUE                                                          # H00260
30    CONTINUE                                                          # H00270
      IANS= -1                                                          # H00280
      IF(IA.EQ.'Y'.OR.IA.EQ.'y') IANS=1                                 # H00290
      IF(IA.EQ.'N'.OR.IA.EQ.'n'.OR.IA.EQ.' ') IANS=0                    # H00300
      IF(IANS.EQ.-1) GO TO 10                                           # H00310
      RETURN                                                            # H00320
      END                                                               # H00330
C-----END OF ROUTINE-------------------------------------------------- # H00340
```

```
C                                                                       # I00010
C---Version 1.0   July 21, 1989                                         # I00020
C***********************************************************************# I00030
C                                                                       # I00040
C                          DGRID                                        # I00050
C                                                                       # I00060
C  THIS ROUTINE DRAWS THE GRID                                          # I00070
C                                                                       # I00080
C***********************************************************************# I00090
C                                                                       # I00100
      SUBROUTINE DGRID(IMIN,IMAX,JMIN,JMAX,XMN,XMX,YMN,YMX,IBOUND,NCOL, # I00110
     1NROW,IGL,IXS)                                                     # I00120
      DIMENSION IBOUND(NCOL,NROW),XMN(NCOL),XMX(NCOL),YMN(NROW),        # I00130
     1YMX(NROW)                                                         # I00140
      DIMENSION XS(2),YS(2)                                             # I00150
C                                                                       # I00160
      DO 10 I=IMIN,IMAX                                                 # I00170
      DO 10 J=JMIN,JMAX                                                 # I00180
      ITOP=0                                                            # I00190
      IF(I.GT.1) ITOP=IBOUND(J,I-1)                                     # I00200
      XS(1)= XMN(J)                                                     # I00210
      YS(1)= YMX(I)                                                     # I00220
      XS(2)= XMX(J)                                                     # I00230
      YS(2)=YS(1)                                                       # I00240
      IF (ITOP.EQ.0.AND.IBOUND(J,I).EQ.0) GO TO 10                      # I00250
      IF(ITOP.EQ.0.OR.IBOUND(J,I).EQ.0) THEN                            # I00260
      CALL RESET ('DASH')                                               # I00270
      CALL CURVE (XS,YS,2,0)                                            # I00280
      ELSE IF(IGL.EQ.1.OR.I.EQ.IMIN) THEN                               # I00290
      IF(I.NE.IMIN) CALL RESET ('DASH')                                 # I00300
      IF(I.EQ.IMIN) CALL DASH                                           # I00310
      CALL CURVE (XS,YS,2,0)                                            # I00320
      END IF                                                            # I00330
10     CONTINUE                                                         # I00340
      DO 20 J=JMIN,JMAX                                                 # I00350
      IBOT=0                                                            # I00360
      IF(IMAX.LT.NROW) IBOT=IBOUND(J,IMAX+1)                            # I00370
      XS(1)= XMN(J)                                                     # I00380
      YS(1)= YMN(IMAX)                                                  # I00390
      XS(2)= XMX(J)                                                     # I00400
      YS(2)=YS(1)                                                       # I00410
      IF(IBOT.EQ.0.AND.IBOUND(J,IMAX).EQ.0) GO TO 20                    # I00420
      IF(IBOT.EQ.0.OR.IBOUND(J,IMAX).EQ.0) THEN                         # I00430
      CALL RESET ('DASH')                                               # I00440
      CALL CURVE(XS,YS,2,0)                                             # I00450
      ELSE                                                              # I00460
      CALL DASH                                                         # I00470
      CALL CURVE(XS,YS,2,0)                                             # I00480
      END IF                                                            # I00490
20     CONTINUE                                                         # I00500
      DO 30 J=JMIN,JMAX                                                 # I00510
      DO 30 I=IMIN,IMAX                                                 # I00520
      JLEFT=0                                                           # I00530
      IF(J.GT.1) JLEFT=IBOUND(J-1,I)                                    # I00540
      XS(1)= XMN(J)                                                     # I00550
      IF(IXS.EQ.1) XS(1)=XMX(J)                                         # I00560
      YS(1)= YMX(I)                                                     # I00570
      XS(2)=XS(1)                                                       # I00580
      YS(2)=YMN(I)                                                      # I00590
      IF(JLEFT.EQ.0.AND.IBOUND(J,I).EQ.0) GO TO 30                      # I00600
      IF(JLEFT.EQ.0.OR.IBOUND(J,I).EQ.0) THEN                           # I00610
      CALL RESET ('DASH')                                               # I00620
      CALL CURVE (XS,YS,2,0)                                            # I00630
      ELSE IF(IGL.EQ.1.OR.J.EQ.JMIN) THEN                               # I00640
      IF(J.NE.JMIN) CALL RESET ('DASH')                                 # I00650
      IF(J.EQ.JMIN) CALL DASH                                           # I00660
      CALL CURVE (XS,YS,2,0)                                            # I00670
      END IF                                                            # I00680
30    CONTINUE                                                          # I00690
      DO 40 I=IMIN,IMAX                                                 # I00700
      JRIGHT=0                                                          # I00710
      IF(JMAX.LT.NCOL) JRIGHT=IBOUND(JMAX+1,I)                          # I00720
```

```
      XS(1)= XMX(JMAX)                                                  # I00730
      IF(IXS.EQ.1) XS(1)=XMN(JMAX)                                      # I00740
      YS(1)= YMX(I)                                                     # I00750
      XS(2)=XS(1)                                                       # I00760
      YS(2)=YMN(I)                                                      # I00770
      IF(JRIGHT.EQ.0.AND.IBOUND(JMAX,I).EQ.0) GO TO 40                  # I00780
      IF(JRIGHT.EQ.0.OR.IBOUND(JMAX,I).EQ.0) THEN                       # I00790
      CALL RESET ('DASH')                                               # I00800
      CALL CURVE (XS,YS,2,0)                                            # I00810
      ELSE                                                              # I00820
      CALL DASH                                                         # I00830
      CALL CURVE(XS,YS,2,0)                                             # I00840
      END IF                                                            # I00850
40    CONTINUE                                                          # I00860
      CALL RESET ('DASH')                                               # I00870
      CALL PKCLR (1)                                                    # I00880
      RETURN                                                            # I00890
      END                                                               # I00900
C-----END OF ROUTINE----------------------------------------------------# I00910
```

```
C                                                                       # J00010
C---Version 1.0   July 21, 1989                                         # J00020
C***********************************************************************# J00030
C                                                                       # J00040
C                          SCPLOT                                       # J00050
C                                                                       # J00060
C  THIS ROUTINE DETERMINES THE SIZE OF THE PLOT AND COMPUTES THE SCALE  # J00070
C  OF THE PLOT                                                          # J00080
C                                                                       # J00090
C***********************************************************************# J00100
C                                                                       # J00110
      SUBROUTINE SCPLOT (PLONG,PL,PSHORT,PS,IPAGE,XMIN,XMAX,YMIN,YMAX,  # J00120
     1XL,XR,YB,YT,MODEL,JUNITS,IA)                                      # J00130
      CHARACTER*80 MES                                                  # J00140
      IPAGE=1                                                           # J00150
      IF(MODEL.GT.4) THEN                                               # J00160
      WRITE (*,*)'ENTER PAGE SIZE:'                                     # J00170
      WRITE (*,*)'   1 = 8.5 X 11'                                      # J00180
      WRITE (*,*)'   2 = 11 X 17'                                       # J00190
      READ (*,*) IPAGE                                                  # J00200
      END IF                                                            # J00210
      WRITE (*,*)'WHAT ARE THE UNITS OF DISTANCE IN THE MODEL ?'        # J00220
      WRITE (*,*)'   1 = FEET'                                          # J00230
      WRITE (*,*)'   2 = METERS'                                        # J00240
      READ (*,*) JUNITS                                                 # J00250
      UNITS=12.0                                                        # J00260
      IF(JUNITS.EQ.2) UNITS=39.0                                        # J00270
      IF (IPAGE.EQ.1) THEN                                              # J00280
      PLONG=11.0                                                        # J00290
      PSHORT=8.5                                                        # J00300
      PL=9.5                                                            # J00310
      PS=6.0                                                            # J00320
      ELSE                                                              # J00330
      PLONG=17.                                                         # J00340
      PSHORT=11.                                                        # J00350
      PL=15.5                                                           # J00360
      PS=8.5                                                            # J00370
      END IF                                                            # J00380
      XLNG=XMAX-XMIN                                                    # J00390
      YLNG=YMAX-YMIN                                                    # J00400
        R=YLNG/XLNG                                                     # J00410
        PSPL=PS/PL                                                      # J00420
        IF(R.LE.PSPL) THEN                                              # J00430
          SCA=XLNG*UNITS/PL                                             # J00440
          YLNG=PSPL*XLNG                                                # J00450
        ELSE                                                            # J00460
          SCA=YLNG*UNITS/PS                                             # J00470
          XLNG=YLNG/PSPL                                                # J00480
        END IF                                                          # J00490
      SCMAX=SCA                                                         # J00500
      WRITE (*,5000) SCMAX                                              # J00510
5000   FORMAT('MAXIMUM SIZE PLOT IS AT SCALE OF (1:',F9.0,')')          # J00520
      MES= 'IS THIS SCALE ACCEPTABLE ?'                                 # J00530
      CALL YESNO(MES,IA,IANS)                                           # J00540
      IF(IANS.EQ.0) THEN                                                # J00550
10    WRITE (*,*)'ENTER NEW SCALE -- (1:M)'                             # J00560
      WRITE (*,*)'  ENTER: M'                                           # J00570
      READ (*,*) SCA                                                    # J00580
      IF(SCA.LT.SCMAX) THEN                                             # J00590
      WRITE (*,5010) SCMAX                                              # J00600
5010   FORMAT('VALUE OF M MUST BE LARGER THAN ',F6.0)                   # J00610
      GO TO 10                                                          # J00620
      END IF                                                            # J00630
      XLNG=XLNG*SCA/SCMAX                                               # J00640
      YLNG=YLNG*SCA/SCMAX                                               # J00650
      END IF                                                            # J00660
      XL= ((XMAX+XMIN)/2.) - XLNG/2.0                                   # J00670
      XR= XL + XLNG                                                     # J00680
      YB= ((YMAX+YMIN)/2.) - YLNG/2.0                                   # J00690
      YT= YB +YLNG                                                      # J00700
      RETURN                                                            # J00710
      END                                                               # J00720
```

```
C-----END OF ROUTINE----------------------------------------------------# J00730
```

```
C                                                                       # K00010
C---Version 1.0   July 21, 1989                                         # K00020
C***********************************************************************# K00030
C                                                                       # K00040
C                          PRIVER                                       # K00050
C                                                                       # K00060
C  THIS ROUTINE READS RIVER DATA AND PLOTS RIVER CELL LOCATIONS         # K00070
C                                                                       # K00080
C***********************************************************************# K00090
C                                                                       # K00100
      SUBROUTINE PRIVER (IMIN,IMAX,JMIN,JMAX,KMIN,KMAX,XMN,XMX,YMN,     # K00110
     1 YMX,ZMN,ZMX,NCOL,NROW,NLAY,IURIV,NCLR,IVIEW,IBUFF,I7)            # K00120
      DIMENSION XMN(NCOL),XMX(NCOL),YMX(NROW),YMN(NROW),ZMN(NLAY),      # K00130
     1 ZMX(NLAY),IBUFF(NCOL,NROW,NLAY)                                  # K00140
      DIMENSION XX(2),YY(2)                                             # K00150
C                                                                       # K00160
      DO 10 K=1,NLAY                                                    # K00170
      DO 10 I=1,NROW                                                    # K00180
      DO 10 J=1,NCOL                                                    # K00190
10     IBUFF(J,I,K)=0                                                   # K00200
      IF (NCLR.GE.0) THEN                                               # K00210
      CALL PKCLR (4)                                                    # K00220
      END IF                                                            # K00230
      WRITE(I7,5000)                                                    # K00240
5000  FORMAT('-------------------------------------------------------------# K00250
     1----------------------')                                          # K00260
      WRITE(I7,*) 'RIVER DATA NOW BEING READ...'                        # K00270
      READ(IURIV,5010) MX,IR                                            # K00280
      READ(IURIV,5010) ITMP                                             # K00290
5010   FORMAT(3I10,3F10.0,I10)                                          # K00300
      IF(ITMP.EQ.0) GO TO 30                                            # K00310
      DO 20 N=1,ITMP                                                    # K00320
      READ(IURIV,5010) K,I,J,H,C,RB,IFACE                               # K00330
      IF(I.LT.IMIN.OR.I.GT.IMAX) GO TO 20                               # K00340
      IF(J.LT.JMIN.OR.J.GT.JMAX) GO TO 20                               # K00350
      IF(K.LT.KMIN.OR.K.GT.KMAX) GO TO 20                               # K00360
      IF (IBUFF(J,I,K).EQ.1) GO TO 20                                   # K00370
      IBUFF(J,I,K)=1                                                    # K00380
      IF (IFACE.GT.6) GO TO 20                                          # K00390
      IF (IVIEW.EQ.1) THEN                                              # K00400
      DX=XMX(J)-XMN(J)                                                  # K00410
      DY=YMX(I)-YMN(I)                                                  # K00420
      X= (XMX(J)+XMN(J))/2.0                                            # K00430
      Y= (YMX(I)+YMN(I))/2.0                                            # K00440
      CALL SYMBL (X,Y,DX,DY,0)                                          # K00450
      END IF                                                            # K00460
20    CONTINUE                                                          # K00470
30    CONTINUE                                                          # K00480
      WRITE(I7,*) '  RIVER DATA HAS BEEN READ'                          # K00490
      IF (NCLR.GE.0) CALL PKCLR (1)                                     # K00500
      RETURN                                                            # K00510
      END                                                               # K00520
C-----END OF ROUTINE-------------------------------------------------------# K00530
```

```
C                                                                       # L00010
C---Version 1.0   July 21, 1989                                         # L00020
C***********************************************************************# L00030
C                                                                      *# L00040
C                            SPACE                                     *# L00050
C                                                                      *# L00060
C  THIS ROUTINE ALOCATES SPACE IN THE MASTER ARRAY AND COMPUTES         # L00070
C  POINTERS FOR ARRAY LOCATIONS WITHIN THE MASTER ARRAY                 # L00080
C                                                                       # L00090
C***********************************************************************# L00100
C                                                                       # L00110
      SUBROUTINE SPACE(LENA,NCOL,NROW,NLAY,NCBL,IGRID,NZDIM,NUNIT,      # L00120
     1LCIBOU,LCXMX,LCXMN,LCDX,LCYMX,LCYMN,LCDY,LCZBOT,LCZTOP,           # L00130
     2LCDZ,LCZMX,LCZMN,LCHEAD,LCBUFF,LCLAYC,LCNCON,LCDZCB,LCIBUF,       # L00140
     3LCIUN,LCIBYZ,LCIBXZ,LCIZN,NPART,IA,IU,I7)                         # L00150
C                                                                       # L00160
      NUNIT=8                                                           # L00170
C                                                                       # L00180
C  ENTER DIMENSION DATA                                                 # L00190
C                                                                       # L00200
      READ(IU,5000) NCOL,NROW,NLAY,NCBL,IGRID                           # L00210
5000  FORMAT(8I10)                                                      # L00220
C                                                                       # L00230
      NCRL=NCOL*NROW*NLAY                                               # L00240
C                                                                       # L00250
      IF(IGRID.EQ.1) THEN                                               # L00260
      NZDIM=NLAY                                                        # L00270
      ELSE                                                              # L00280
      NZDIM=NCRL                                                        # L00290
      END IF                                                            # L00300
C                                                                       # L00310
      ISUM=1                                                            # L00320
      LCIBOU=ISUM                                                       # L00330
      ISUM=ISUM+NCRL                                                    # L00340
      LCXMX=ISUM                                                        # L00350
      ISUM=ISUM+NCOL                                                    # L00360
      LCXMN=ISUM                                                        # L00370
      ISUM=ISUM+NCOL                                                    # L00380
      LCDX=ISUM                                                         # L00390
      ISUM=ISUM+NCOL                                                    # L00400
      LCYMX=ISUM                                                        # L00410
      ISUM=ISUM+NROW                                                    # L00420
      LCYMN=ISUM                                                        # L00430
      ISUM=ISUM+NROW                                                    # L00440
      LCDY=ISUM                                                         # L00450
      ISUM=ISUM+NROW                                                    # L00460
      LCZBOT=ISUM                                                       # L00470
      ISUM=ISUM+NZDIM                                                   # L00480
      LCZTOP=ISUM                                                       # L00490
      ISUM=ISUM+NZDIM                                                   # L00500
      LCDZ=ISUM                                                         # L00510
      ISUM=ISUM+NLAY                                                    # L00520
      LCZMX=ISUM                                                        # L00530
      ISUM=ISUM+NLAY                                                    # L00540
      LCZMN=ISUM                                                        # L00550
      ISUM=ISUM+NLAY                                                    # L00560
      LCHEAD=ISUM                                                       # L00570
      ISUM=ISUM+NCRL                                                    # L00580
      LCBUFF=ISUM                                                       # L00590
      ISUM=ISUM+NCRL                                                    # L00600
      LCLAYC=ISUM                                                       # L00610
      ISUM=ISUM+NLAY                                                    # L00620
      LCNCON=ISUM                                                       # L00630
      ISUM=ISUM+NLAY                                                    # L00640
      LCDZCB=ISUM                                                       # L00650
      ISUM=ISUM+NLAY                                                    # L00660
      LCIBUF=ISUM                                                       # L00670
      ISUM=ISUM + NCRL                                                  # L00680
      LCIUN=ISUM                                                        # L00690
      ISUM=ISUM + NUNIT                                                 # L00700
      LCIBYZ=ISUM                                                       # L00710
      ISUM=ISUM + (NROW*NLAY)                                           # L00720
```

```
      LCIBXZ=ISUM                                                       # L00730
      ISUM=ISUM + (NCOL*NLAY)                                           # L00740
      ISM1=ISUM-1                                                       # L00750
      NDIF=LENA-ISM1                                                    # L00760
      NPART= NDIF/2                                                     # L00770
      IF (NPART.LT.1) THEN                                              # L00780
      WRITE (*,*)                                                       # L00790
     1'THE SIZE OF THE MASTER ARRAY IS NOT LARGE ENOUGH.'               # L00800
      WRITE (*,*) 'DIMENSION THE MASTER ARRAY:   A(LENA)'               # L00810
      WRITE (*,5010) ISM1                                               # L00820
      WRITE (I7,*)                                                      # L00830
     1'THE SIZE OF THE MASTER ARRAY IS NOT LARGE ENOUGH.'               # L00840
      WRITE (I7,*) 'DIMENSION THE MASTER ARRAY:   A(LENA)'              # L00850
      WRITE (I7,5010) ISM1                                              # L00860
5010   FORMAT ('WITH   LENA =',I8,' + 2 x (NUMBER OF PARTICLES)')       # L00870
      STOP                                                              # L00880
      END IF                                                            # L00890
      LCIZN=ISUM                                                        # L00900
      ISUM=ISUM + (2*NPART)                                             # L00910
      ISM1=ISUM-1                                                       # L00920
      WRITE (*,5020) NPART                                              # L00930
5020  FORMAT ('THE MAXIMUM NUMBER OF PARTICLES FOR THIS RUN IS ',I8)    # L00940
      WRITE (I7,5030) NPART                                             # L00950
5030  FORMAT('MAXIMUM NUMBER OF PARTICLES IS ',I8)                      # L00960
      WRITE (I7,5040) ISM1,LENA                                         # L00970
5040  FORMAT(I8,' ELEMENTS OUT OF ',I8,' USED IN "A" ARRAY')            # L00980
      IF(ISM1.GT.LENA) STOP                                             # L00990
      RETURN                                                            # L01000
      END                                                               # L01010
C-----END OF ROUTINE----------------------------------------------------# L01020
```

```
C                                                                       # M00010
C---Version 1.0   July 21, 1989                                         # M00020
C***********************************************************************# M00030
C                                                                      *# M00040
C                              DATIN                                   *# M00050
C                                                                      *# M00060
C  THIS ROUTINE READS IN DATA                                           # M00070
C                                                                       # M00080
C***********************************************************************# M00090
C                                                                       # M00100
      SUBROUTINE DATIN (LAYCON,NCON,HEAD,XMX,XMN,YMX,YMN,DELR,          # M00110
     1DELC,DELZ,DELZCB,ZTOP,ZBOT,ZMN,ZMX,IBOUND,IUNIT,NCOL,NROW,NLAY,   # M00120
     2NUNIT,NZDIM,BUFF,IBUFF,IGRID,NCBL,IA,I1,I2,I3,I4,I7,NPART)        # M00130
                                                                        # M00140
C                                                                       # M00150
      CHARACTER*80 FLHEAD,FLFLO1,FLFLO2,FLPART,MES,SUFFIX,FLOUT1,       # M00160
     1FLOUT2,FLOUT3,FLOUT4                                              # M00170
      CHARACTER*16 TEXT                                                 # M00180
C                                                                       # M00190
      DIMENSION LAYCON(NLAY),NCON(NLAY),DELR(NCOL),DELC(NROW),DELZ(NLAY)# M00200
     1,DELZCB(NLAY),XMX(NCOL),YMX(NROW),ZTOP(NZDIM),ZBOT(NZDIM),        # M00210
     2HEAD(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),XMN(NCOL),YMN(NROW),  # M00220
     4IUNIT(NUNIT),BUFF(NCOL,NROW,NLAY),IBUFF(NCOL,NROW,NLAY),ZMN(NLAY),# M00230
     5ZMX(NLAY)                                                         # M00240
C                                                                       # M00250
      IU=I1                                                             # M00260
      WRITE (*,*)'READING MODPATH DATA FILES...'                        # M00270
      WRITE(I7,5000)                                                    # M00280
5000  FORMAT('-------------------------------------------------------'# M00290
     1----------------------')                                          # M00300
      WRITE(I7,5010) NCOL,NROW,NLAY,NCBL                                # M00310
5010  FORMAT(I5,' COLUMNS',I5,' ROWS',I5,' LAYERS',I5,' CONFINING LAYERS'# M00320
     1')                                                                # M00330
      WRITE(I7,5020) IGRID                                              # M00340
5020  FORMAT('IGRID (GRID TYPE CODE) IS',I2)                            # M00350
C                                                                       # M00360
C  READ UNIT NUMBERS FOR INPUT FILES                                    # M00370
C                                                                       # M00380
      READ(IU,5030) (IUNIT(N),N=1,NUNIT)                                # M00390
5030  FORMAT(16I5)                                                      # M00400
      WRITE(I7,5000)                                                    # M00410
      WRITE(I7,5040) (IUNIT(N),N=1,8)                                   # M00420
5040  FORMAT('IUNIT ARRAY: ',8I4)                                       # M00430
C                                                                       # M00440
C  LAYER TYPE CODES                                                     # M00450
C                                                                       # M00460
      READ(I1,5050) (LAYCON(N),N=1,NLAY)                                # M00470
5050  FORMAT(40I2)                                                      # M00480
      WRITE(I7,5000)                                                    # M00490
      WRITE(I7,*) 'LAYCON (LAYER TYPE CODES):'                          # M00500
      WRITE(I7,5060) (LAYCON(N),N=1,NLAY)                               # M00510
5060   FORMAT(25I3)                                                     # M00520
C                                                                       # M00530
C  CONFINING BED CODES                                                  # M00540
C                                                                       # M00550
      IF(NCBL.GT.0) THEN                                                # M00560
      READ(I1,5050) (NCON(N),N=1,NLAY)                                  # M00570
      ELSE                                                              # M00580
      DO 10 N=1,NLAY                                                    # M00590
10    NCON(N)=0                                                         # M00600
      END IF                                                            # M00610
      IF(NCBL.EQ.0) THEN                                                # M00620
      WRITE(I7,5000)                                                    # M00630
      WRITE(I7,*) 'NO CONFINING LAYERS.  NCON = 0 FOR ALL LAYERS.'      # M00640
      ELSE                                                              # M00650
      WRITE(I7,5000)                                                    # M00660
      WRITE(I7,*) 'NCON (CONFINING LAYER CODES):'                       # M00670
      WRITE(I7,5060) (NCON(N),N=1,NLAY)                                 # M00680
      END IF                                                            # M00690
      IF(NCBL.GT.0) THEN                                                # M00700
      NN=0                                                              # M00710
      DO 20 N=1,NLAY                                                    # M00720
```

```
      IF(NCON(N).EQ.0) GO TO 20                                          # M00730
      NN=NN+1                                                            # M00740
      NCON(N)=NN                                                         # M00750
20    CONTINUE                                                           # M00760
      END IF                                                             # M00770
C                                                                        # M00780
C  DELR, GRID SPACING ALONG A ROW (X-DIRECTION)                          # M00790
C                                                                        # M00800
      WRITE(I7,5000)                                                     # M00810
      WRITE(I7,*) 'DELR ARRAY NOW BEING READ...'                         # M00820
      CALL IN1DR (IU,DELR,NCOL,I7)                                       # M00830
C                                                                        # M00840
C     COMPUTE XMAX ARRAY                                                 # M00850
C                                                                        # M00860
      XMX(1)=DELR(1)                                                     # M00870
      XMN(1)=0.0                                                         # M00880
      DO 30 J=2,NCOL                                                     # M00890
      XMN(J)=XMX(J-1)                                                    # M00900
30     XMX(J)=XMX(J-1)+DELR(J)                                           # M00910
C                                                                        # M00920
C  DELC, GRID SPACING ALONG A COLUMN (Y-DIRECTION)                       # M00930
C                                                                        # M00940
      WRITE(I7,5000)                                                     # M00950
      WRITE(I7,*) 'DELC ARRAY NOW BEING READ...'                         # M00960
      CALL IN1DR (IU,DELC,NROW,I7)                                       # M00970
C                                                                        # M00980
C     COMPUTE YMAX ARRAY                                                 # M00990
C                                                                        # M01000
      YMX(NROW)=DELC(NROW)                                               # M01010
      YMN(NROW)=0.0                                                      # M01020
      DO 40 I=2,NROW                                                     # M01030
      II=NROW+1-I                                                        # M01040
      YMN(II)=YMX(II+1)                                                  # M01050
40     YMX(II)=YMX(II+1)+DELC(II)                                        # M01060
C                                                                        # M01070
C  READ VERTICAL COORDINATE DATA                                         # M01080
C                                                                        # M01090
      IF (IGRID.EQ.1) THEN                                               # M01100
      WRITE(I7,5000)                                                     # M01110
      WRITE(I7,*) 'DELZ ARRAY NOW BEING READ...'                         # M01120
      CALL IN1DR (IU,DELZ,NLAY,I7)                                       # M01130
      IF(NCBL.GT.0) THEN                                                 # M01140
      WRITE(I7,5000)                                                     # M01150
      WRITE(I7,*) 'DELZCB ARRAY NOW BEING READ...'                       # M01160
      CALL IN1DR (IU,DELZCB,NLAY,I7)                                     # M01170
      ELSE                                                               # M01180
      DO 50 N=1,NLAY                                                     # M01190
50    DELZCB(N)=0.0                                                      # M01200
      END IF                                                             # M01210
      READ(IU,5070) ZBL1                                                 # M01220
5070  FORMAT(F10.0)                                                      # M01230
      ZBOT(1)=ZBL1                                                       # M01240
      WRITE(I7,5000)                                                     # M01250
      WRITE(I7,5080) ZBOT(1)                                             # M01260
5080  FORMAT('BOTTOM ELEVATION OF LAYER 1 IS',E13.5)                     # M01270
      ZTOP(1)= ZBOT(1) + DELZ(1)                                         # M01280
      DO 60 K=2,NLAY                                                     # M01290
      ZTOP(K)= ZBOT(K-1) - DELZCB(K-1)                                   # M01300
      ZBOT(K)= ZTOP(K) - DELZ(K)                                         # M01310
60    CONTINUE                                                           # M01320
      ELSE                                                               # M01330
      DO 70 K=1,NLAY                                                     # M01340
      KP= 1 + (K-1)*NCOL*NROW                                            # M01350
      IF(LAYCON(K).NE.1) THEN                                            # M01360
      WRITE(I7,5000)                                                     # M01370
      WRITE(I7,5090) K                                                   # M01380
5090  FORMAT('TOP ELEVATION OF LAYER',I4,' NOW BEING READ...')           # M01390
      CALL IN2DR (IU,ZTOP(KP),NCOL,NROW,I7)                              # M01400
      END IF                                                             # M01410
      WRITE(I7,5000)                                                     # M01420
      WRITE(I7,5100) K                                                   # M01430
5100  FORMAT('BOTTOM ELEVATION OF LAYER',I4,' NOW BEING READ...')        # M01440
```

```
      CALL IN2DR (IU,ZBOT(KP),NCOL,NROW,I7)                              # M01450
70    CONTINUE                                                           # M01460
      END IF                                                             # M01470
C                                                                        # M01480
C  IBOUND DATA                                                           # M01490
C                                                                        # M01500
      DO 80 K=1,NLAY                                                     # M01510
      WRITE(I7,5000)                                                     # M01520
      WRITE(I7,5110) K                                                   # M01530
5110  FORMAT('IBOUND ARRAY FOR LAYER',I4,' NOW BEING READ...')           # M01540
      CALL IN2DI (IU,IBOUND(1,1,K),NCOL,NROW,I7)                         # M01550
80    CONTINUE                                                           # M01560
C                                                                        # M01570
C  READ HEADS                                                            # M01580
C                                                                        # M01590
      DO 90 K=1,NLAY                                                     # M01600
      DO 90 I=1,NROW                                                     # M01610
      DO 90 J=1,NCOL                                                     # M01620
90    HEAD(J,I,K)=0.                                                     # M01630
      IUHED=IUNIT(8)                                                     # M01640
      IF (IUHED.NE.0) THEN                                               # M01650
100    CONTINUE                                                          # M01660
      READ (IUHED,END=110) KSTP,KPER,PERTIM,TOTIM,TEXT,NC,NR,K           # M01670
      IF (TEXT.NE.'            HEAD') THEN                               # M01680
      WRITE (*,*) 'HEAD FILE DOES NOT CONTAIN HEAD DATA.'                # M01690
      WRITE (I7,*) 'HEAD FILE DOES NOT CONTAIN HEAD DATA.'               # M01700
      STOP                                                               # M01710
      END IF                                                             # M01720
      WRITE(I7,5000)                                                     # M01730
      WRITE(I7,5120) K                                                   # M01740
5120  FORMAT('HEADS NOW BEING READ FOR LAYER',I4)                        # M01750
      READ(IUHED) ((HEAD(J,I,K),J=1,NCOL),I=1,NROW)                      # M01760
      GO TO 100                                                          # M01770
      END IF                                                             # M01780
110    CONTINUE                                                          # M01790
      WRITE(I7,*) 'HEADS HAVE BEEN READ'                                 # M01800
      IF(IGRID.EQ.0) THEN                                                # M01810
      DO 140 K=1,NLAY                                                    # M01820
      DELZ(K)=0.0                                                        # M01830
      COUNT=0.0                                                          # M01840
      DO 120 I=1,NROW                                                    # M01850
      DO 120 J=1,NCOL                                                    # M01860
      IF(IBOUND(J,I,K).EQ.0.OR.HEAD(J,I,K).GT.1.0E+29) GO TO 120         # M01870
      COUNT=COUNT+1.0E+0                                                 # M01880
      JIK= (K-1)*NCOL*NROW + (I-1)*NCOL + J                              # M01890
      IF(LAYCON(K).NE.1) THEN                                            # M01900
      DELZ(K)=DELZ(K) + (ZTOP(JIK)-ZBOT(JIK))                            # M01910
      ELSE                                                               # M01920
      DELZ(K)=DELZ(K) + (HEAD(J,I,K)-ZBOT(JIK))                          # M01930
      END IF                                                             # M01940
120    CONTINUE                                                          # M01950
      DELZ(K)=DELZ(K)/COUNT                                              # M01960
      DELZCB(K)=0.0                                                      # M01970
      COUNT=0.0                                                          # M01980
      IF(NCON(K).EQ.0) GO TO 140                                         # M01990
      DO 130 I=1,NROW                                                    # M02000
      DO 130 J=1,NCOL                                                    # M02010
      IF(IBOUND(J,I,K).EQ.0) GO TO 130                                   # M02020
      COUNT=COUNT+1.0E+0                                                 # M02030
      JIK= (K-1)*NCOL*NROW + (I-1)*NCOL + J                              # M02040
      JIK1=JIK+(NCOL*NROW)                                               # M02050
      DELZCB(K)= DELZCB(K) + (ZBOT(JIK)-ZTOP(JIK1))                      # M02060
130    CONTINUE                                                          # M02070
      DELZCB(K)=DELZCB(K)/COUNT                                          # M02080
140    CONTINUE                                                          # M02090
      ZMX(NLAY)=DELZ(NLAY)                                               # M02100
      ZMN(NLAY)=0.0                                                      # M02110
      DO 150 K=2,NLAY                                                    # M02120
      KK=NLAY+1-K                                                        # M02130
      ZMN(KK)=ZMX(KK+1)+DELZCB(KK)                                       # M02140
      ZMX(KK)=ZMN(KK)+DELZ(KK)                                           # M02150
150    CONTINUE                                                          # M02160
```

```
      END IF                                                                 # M02170
      IF (IGRID.EQ.1) THEN                                                   # M02180
      IF(LAYCON(1).EQ.1) THEN                                                # M02190
      DELZ(1)=0.0                                                            # M02200
      COUNT=0.0                                                              # M02210
      DO 160 I=1,NROW                                                        # M02220
      DO 160 J=1,NCOL                                                        # M02230
      IF(IBOUND(J,I,1).EQ.0.OR.HEAD(J,I,1).GT.1.0E+29) GO TO 160             # M02240
      COUNT=COUNT+1.0E+0                                                     # M02250
      DELZ(1)=DELZ(1) + (HEAD(J,I,1)-ZBOT(1))                                # M02260
160   CONTINUE                                                               # M02270
      DELZ(1)=DELZ(1)/COUNT                                                  # M02280
      END IF                                                                 # M02290
      ZMX(1)=ZBOT(1) + DELZ(1)                                               # M02300
      ZMN(1)=ZBOT(1)                                                         # M02310
      DO 170 K=2,NLAY                                                        # M02320
      ZMX(K)=ZTOP(K)                                                         # M02330
      ZMN(K)=ZBOT(K)                                                         # M02340
170   CONTINUE                                                               # M02350
      END IF                                                                 # M02360
      RETURN                                                                 # M02370
      END                                                                    # M02380
C-----END OF ROUTINE-------------------------------------------------------# M02390
```

```
C                                                                       # N00010
C---Version 1.0   July 21, 1989                                         # N00020
C***********************************************************************# N00030
C                                                                       # N00040
C                              IN2DR                                    # N00050
C                                                                       # N00060
C  THIS SUBROUTINE READS A TWO DIMENSIONAL ARRAY OF REAL NUMBERS.       # N00070
C                                                                       # N00080
C***********************************************************************# N00090
C                                                                       # N00100
      SUBROUTINE IN2DR (IU,X,JJ,II,I7)                                  # N00110
      DIMENSION X(JJ,II)                                                # N00120
      CHARACTER*20 FMT                                                  # N00130
C                                                                       # N00140
      READ(IU,5000) LOCAT,CNSTNT,FMT,IPRN                               # N00150
5000  FORMAT(I10,F10.0,A20,I10)                                         # N00160
      IF(LOCAT.LE.0) THEN                                               # N00170
      DO 10 I=1,II                                                      # N00180
      DO 10 J=1,JJ                                                      # N00190
10    X(J,I)= CNSTNT                                                    # N00200
      WRITE(I7,5010) CNSTNT                                             # N00210
5010  FORMAT('  CONSTANT VALUE OF',E13.5)                              # N00220
      RETURN                                                            # N00230
      END IF                                                            # N00240
C                                                                       # N00250
      IF(FMT.EQ.'                    ') THEN                            # N00260
      DO 20 I=1,II                                                      # N00270
      READ(LOCAT,*) (X(J,I),J=1,JJ)                                     # N00280
20    CONTINUE                                                          # N00290
      ELSE                                                              # N00300
      DO 30 I=1,II                                                      # N00310
      READ(LOCAT,FMT) (X(J,I),J=1,JJ)                                   # N00320
30    CONTINUE                                                          # N00330
      END IF                                                            # N00340
      DO 40 I=1,II                                                      # N00350
      DO 40 J=1,JJ                                                      # N00360
40    X(J,I)= CNSTNT*X(J,I)                                             # N00370
      IF(IPRN.GT.0) THEN                                                # N00380
      DO 50 I=1,II                                                      # N00390
      WRITE(I7,5020) (X(J,I),J=1,JJ)                                    # N00400
50    CONTINUE                                                          # N00410
      ELSE                                                              # N00420
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # N00430
      END IF                                                            # N00440
5020  FORMAT(10E13.5)                                                   # N00450
C                                                                       # N00460
      RETURN                                                            # N00470
      END                                                               # N00480
C-----END OF ROUTINE-------------------------------------------------- # N00490
```

```
C                                                                       # O00010
C---Version 1.0   July 21, 1989                                         # O00020
C***********************************************************************# O00030
C                                                                       # O00040
C                              IN1DR                                    # O00050
C                                                                       # O00060
C  THIS SUBROUTINE READS A ONE DIMENSIONAL ARRAY OF REAL NUMBERS.       # O00070
C                                                                       # O00080
C***********************************************************************# O00090
C                                                                       # O00100
      SUBROUTINE IN1DR (IU,X,JJ,I7)                                     # O00110
      DIMENSION X(JJ)                                                   # O00120
      CHARACTER*20 FMT                                                  # O00130
C                                                                       # O00140
      READ(IU,5000) LOCAT,CNSTNT,FMT,IPRN                               # O00150
5000  FORMAT(I10,F10.0,A20,I10)                                         # O00160
      IF(LOCAT.LE.0) THEN                                               # O00170
      DO 10 J=1,JJ                                                      # O00180
10    X(J)= CNSTNT                                                      # O00190
      WRITE(I7,5010) CNSTNT                                             # O00200
5010  FORMAT('  CONSTANT VALUE OF',E13.5)                              # O00210
      RETURN                                                            # O00220
      END IF                                                            # O00230
C                                                                       # O00240
      IF(FMT.EQ.'                    ') THEN                            # O00250
      READ(LOCAT,*) (X(J),J=1,JJ)                                       # O00260
      ELSE                                                              # O00270
      READ(LOCAT,FMT) (X(J),J=1,JJ)                                     # O00280
      END IF                                                            # O00290
      DO 20 J=1,JJ                                                      # O00300
20    X(J)= CNSTNT*X(J)                                                 # O00310
C                                                                       # O00320
      IF(IPRN.GT.0) THEN                                                # O00330
      WRITE(I7,5020) (X(J),J=1,JJ)                                      # O00340
5020  FORMAT(10E13.5)                                                   # O00350
      ELSE                                                              # O00360
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # O00370
      END IF                                                            # O00380
      RETURN                                                            # O00390
      END                                                               # O00400
C-----END OF ROUTINE-------------------------------------------------- # O00410
```

```
C                                                                       # P00010
C---Version 1.0   July 21, 1989                                         # P00020
C***********************************************************************# P00030
C                                                                       # P00040
C                              IN2DI                                    # P00050
C                                                                       # P00060
C  THIS SUBROUTINE READS A TWO DIMENSIONAL ARRAY OF INTEGERS.           # P00070
C                                                                       # P00080
C***********************************************************************# P00090
C                                                                       # P00100
      SUBROUTINE IN2DI (IU,N,JJ,II,I7)                                  # P00110
      DIMENSION N(JJ,II)                                                # P00120
      CHARACTER*20 FMT                                                  # P00130
C                                                                       # P00140
      READ(IU,5000) LOCAT,ICON,FMT,IPRN                                 # P00150
5000  FORMAT(2I10,A20,I10)                                              # P00160
      IF(LOCAT.LE.0) THEN                                               # P00170
      DO 10 I=1,II                                                      # P00180
      DO 10 J=1,JJ                                                      # P00190
10    N(J,I)= ICON                                                      # P00200
      WRITE(I7,5010) ICON                                               # P00210
5010  FORMAT('  CONSTANT VALUE OF',I10)                                 # P00220
      RETURN                                                            # P00230
      END IF                                                            # P00240
C                                                                       # P00250
      IF(FMT.EQ.'                    ') THEN                            # P00260
      DO 20 I=1,II                                                      # P00270
      READ(LOCAT,*) (N(J,I),J=1,JJ)                                     # P00280
20    CONTINUE                                                          # P00290
      ELSE                                                              # P00300
      DO 30 I=1,II                                                      # P00310
      READ(LOCAT,FMT) (N(J,I),J=1,JJ)                                   # P00320
30    CONTINUE                                                          # P00330
      END IF                                                            # P00340
      DO 40 I=1,II                                                      # P00350
      DO 40 J=1,JJ                                                      # P00360
40    N(J,I)= ICON*N(J,I)                                               # P00370
      IF(IPRN.GT.0) THEN                                                # P00380
      DO 50 I=1,II                                                      # P00390
      WRITE(I7,5020) (N(J,I),J=1,JJ)                                    # P00400
50    CONTINUE                                                          # P00410
      ELSE                                                              # P00420
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # P00430
      END IF                                                            # P00440
5020  FORMAT(25I5)                                                      # P00450
C                                                                       # P00460
      RETURN                                                            # P00470
      END                                                               # P00480
C-----END OF ROUTINE-------------------------------------------------- # P00490
```

```
C                                                                       # Q00010
C---Version 1.0   July 21, 1989                                         # Q00020
C***********************************************************************# Q00030
C                                                                       # Q00040
C                              IN1DI                                    # Q00050
C                                                                       # Q00060
C  THIS SUBROUTINE READS A ONE DIMENSIONAL ARRAY OF INTEGERS.           # Q00070
C                                                                       # Q00080
C***********************************************************************# Q00090
C                                                                       # Q00100
      SUBROUTINE IN1DI (IU,N,JJ,I7)                                     # Q00110
      DIMENSION N(JJ)                                                   # Q00120
      CHARACTER*20 FMT                                                  # Q00130
C                                                                       # Q00140
      READ(IU,5000) LOCAT,ICON,FMT,IPRN                                 # Q00150
5000  FORMAT(2I10,A20,I10)                                              # Q00160
      IF(LOCAT.LE.0) THEN                                               # Q00170
      DO 10 J=1,JJ                                                      # Q00180
10    N(J)= ICON                                                        # Q00190
      WRITE(I7,5010) ICON                                               # Q00200
5010  FORMAT('  CONSTANT VALUE OF',I10)                                 # Q00210
      RETURN                                                            # Q00220
      END IF                                                            # Q00230
C                                                                       # Q00240
      IF(FMT.EQ.'                    ') THEN                            # Q00250
      READ(LOCAT,*) (N(J),J=1,JJ)                                       # Q00260
      ELSE                                                              # Q00270
      READ(LOCAT,FMT) (N(J),J=1,JJ)                                     # Q00280
      END IF                                                            # Q00290
      DO 20 J=1,JJ                                                      # Q00300
20    N(J)= ICON*N(J)                                                   # Q00310
      IF(IPRN.GT.0) THEN                                                # Q00320
      WRITE(I7,5020) (N(J),J=1,JJ)                                      # Q00330
      ELSE                                                              # Q00340
      WRITE(I7,*) '  DATA WAS READ BUT NOT PRINTED'                     # Q00350
      END IF                                                            # Q00360
5020  FORMAT(25I5)                                                      # Q00370
C                                                                       # Q00380
      RETURN                                                            # Q00390
      END                                                               # Q00400
C-----END OF ROUTINE-------------------------------------------------- # Q00410
```

```
C                                                                       # R00010
C---Version 1.0   July 21, 1989                                         # R00020
C***********************************************************************# R00030
C                                                                       # R00040
C                          NEWIB                                        # R00050
C                                                                       # R00060
C  THIS ROUTINE FILLS A NEW TWO DIMENSIONAL "IBOUND" ARRAY FOR CROSS    # R00070
C  SECTIONAL SLICE THROUGH THE 3-D GRID                                 # R00080
C                                                                       # R00090
C***********************************************************************# R00100
C                                                                       # R00110
      SUBROUTINE NEWIB (IVIEW,IBOUND,IB,NCOL,NROW,NLAY,NH,L,HEAD)       # R00120
      DIMENSION IBOUND(NCOL,NROW,NLAY),IB(NH,NLAY),                     # R00130
     1 HEAD(NCOL,NROW,NLAY)                                             # R00140
C                                                                       # R00150
      IF(IVIEW.EQ.2) THEN                                               # R00160
      DO 10 K=1,NLAY                                                    # R00170
      DO 10 I=1,NROW                                                    # R00180
      IB(I,K)= IBOUND(L,I,K)                                            # R00190
      IF(HEAD(L,I,K).GT.1.0E+29) IB(I,K)=0                              # R00200
10    CONTINUE                                                          # R00210
      ELSE IF(IVIEW.EQ.3) THEN                                          # R00220
      DO 20 K=1,NLAY                                                    # R00230
      DO 20 J=1,NCOL                                                    # R00240
      IB(J,K)=IBOUND(J,L,K)                                             # R00250
      IF(HEAD(J,L,K).GT.1.0E+29) IB(J,K)=0                              # R00260
20    CONTINUE                                                          # R00270
      END IF                                                            # R00280
      RETURN                                                            # R00290
      END                                                               # R00300
C-----END OF ROUTINE-------------------------------------------------- # R00310
```

```
C                                                                       # S00010
C---Version 1.0   July 21, 1989                                         # S00020
C***********************************************************************# S00030
C                                                                       # S00040
C                              PCBEDS                                   # S00050
C                                                                       # S00060
C  THIS ROUTINE DRAWS CONFINING LAYERS                                  # S00070
C                                                                       # S00080
C***********************************************************************# S00090
C                                                                       # S00100
      SUBROUTINE PCBEDS (NCON,NLAY,ZMN,ZMX,JMIN,JMAX,XMN,XMX,NH,IB,KMIN,# S00110
     1 KMAX)                                                            # S00120
      DIMENSION ZMN(NLAY),ZMX(NLAY),NCON(NLAY),XMN(NH),XMX(NH),         # S00130
     1 IB(NH,NLAY)                                                      # S00140
      DIMENSION X(2),Y(2)                                               # S00150
C                                                                       # S00160
      DO 20 K=KMIN,KMAX                                                 # S00170
      IF(NCON(K).EQ.0) GO TO 20                                         # S00180
      DO 10 J=JMIN,JMAX                                                 # S00190
      IF(IB(J,K).NE.0) THEN                                             # S00200
      X(1)=XMN(J)                                                       # S00210
      X(2)=XMX(J)                                                       # S00220
      Y(1)=ZMN(K)                                                       # S00230
      Y(2)=ZMN(K)                                                       # S00240
      CALL CURVE (X,Y,2,0)                                              # S00250
      END IF                                                            # S00260
      IF(K.EQ.NLAY) GO TO 20                                            # S00270
      IF(IB(J,K+1).NE.0) THEN                                           # S00280
      X(1)=XMN(J)                                                       # S00290
      X(2)=XMX(J)                                                       # S00300
      Y(1)=ZMX(K+1)                                                     # S00310
      Y(2)=ZMX(K+1)                                                     # S00320
      CALL CURVE (X,Y,2,0)                                              # S00330
      END IF                                                            # S00340
10    CONTINUE                                                          # S00350
20    CONTINUE                                                          # S00360
      RETURN                                                            # S00370
      END                                                               # S00380
C-----END OF ROUTINE-------------------------------------------------------# S00390
```

```
C                                                                       # T00010
C---Version 1.0   July 21, 1989                                         # T00020
C***********************************************************************# T00030
C                                                                       # T00040
C                            NOTATE                                     # T00050
C                                                                       # T00060
C  THIS ROUTINE DRAWS THE SCALE BAR AND WRITES PLOT ANNOTATION          # T00070
C                                                                       # T00080
C***********************************************************************# T00090
C                                                                       # T00100
      SUBROUTINE NOTATE (XMIN,XMAX,YMIN,YMAX,PX,PY,JUNITS,TITLE,VX,     # T00110
     1IVIEW)                                                            # T00120
      DIMENSION XA(3),YA(3)                                             # T00130
      CHARACTER*5 JU1                                                   # T00140
      CHARACTER*7 JU2                                                   # T00150
      CHARACTER*80 TITLE                                                # T00160
      CHARACTER*25 EXAG                                                 # T00170
C                                                                       # T00180
      JU1= ' FEET'                                                      # T00190
      JU2= ' METERS'                                                    # T00200
      EXAG= 'VERTICAL EXAGGERATION IS '                                 # T00210
      CALL HEIGHT (0.10)                                                # T00220
      YPY= (YMAX-YMIN)/PY                                               # T00230
      XPX= (XMAX-XMIN)/PX                                               # T00240
      XMID= XMIN + 0.6*(XMAX-XMIN)                                      # T00250
      X=2.0*(XMAX-XMIN)/PX                                              # T00260
      XL10= ALOG10(X)                                                   # T00270
      IXL10= IFIX(XL10)                                                 # T00280
      XX= FLOAT(IXL10)                                                  # T00290
      XMAN= XL10-XX                                                     # T00300
      XXX= 10.0**(XMAN)                                                 # T00310
      IXXX= IFIX(XXX)                                                   # T00320
      XINC= 10.0**(IXL10)                                               # T00330
      NINC= IXXX                                                        # T00340
C                                                                       # T00350
      YP5= YMIN-(YPY*1.20)                                              # T00360
      YP4= YMIN-(YPY*0.30)                                              # T00370
      YP3= YMIN-(YPY*0.35)                                              # T00380
      YP2= YMIN-(YPY*0.50)                                              # T00390
      YP1= YMIN-(YPY*0.80)                                              # T00400
      CALL RLMESS (TITLE,80,XMIN,YP5)                                   # T00410
      XA(1)= XMID                                                       # T00420
      XA(2)=XMID                                                        # T00430
      XA(3)=XMID                                                        # T00440
      YA(1)=YP3                                                         # T00450
      YA(2)=YP2                                                         # T00460
      YA(3)=YP2                                                         # T00470
      CALL CURVE(XA,YA,3,0)                                             # T00480
      XA(1)=XMID                                                        # T00490
      YA(1)=YP2                                                         # T00500
      YA(2)=YP2                                                         # T00510
      YA(3)=YP3                                                         # T00520
      DO 10  N=1,NINC                                                   # T00530
      XA(2)=XMID + (XINC*N)                                             # T00540
      XA(3)=XA(2)                                                       # T00550
      CALL CURVE(XA,YA,3,0)                                             # T00560
      XA(1)=XA(2)                                                       # T00570
10    CONTINUE                                                          # T00580
      INUM=0                                                            # T00590
      CALL RLINT (INUM,XMID,YP4)                                        # T00600
      INUM= IFIX(XINC)                                                  # T00610
      XVAL= XMID + XINC                                                 # T00620
      CALL RLINT (INUM,XVAL,YP4)                                        # T00630
      XVAL= XINC*NINC                                                   # T00640
      INUM= IFIX(XVAL)                                                  # T00650
      XVAL=XVAL + XMID                                                  # T00660
      CALL RLINT (INUM,XVAL,YP4)                                        # T00670
      IF(JUNITS.EQ.1) THEN                                              # T00680
      CALL RLMESS (JU1,5,'ABUT','ABUT')                                 # T00690
      ELSE                                                              # T00700
      CALL RLMESS (JU2,7,'ABUT','ABUT')                                 # T00710
      END IF                                                            # T00720
```

```
      CALL RLMESS (EXAG,25,XMID,YP1)                                    # T00730
      IF(VX.GE.1.0) CALL RLREAL (VX,1,'ABUT','ABUT')                    # T00740
      IF(VX.LT.1.0) CALL RLREAL (VX,4,'ABUT','ABUT')                    # T00750
      YP3S= YP2 + (YP3-YP2)/2.0                                         # T00760
      IF(NINC.LE.3) THEN                                                # T00770
      XINC=XINC/5.0                                                     # T00780
      XA(1)=XMID                                                        # T00790
      YA(1)= YP2                                                        # T00800
      YA(2)=YP2                                                         # T00810
      YA(3)= YP3S                                                       # T00820
      DO 20  N=1,4                                                      # T00830
      XA(2)= XMID + (XINC*N)                                            # T00840
      XA(3)= XA(2)                                                      # T00850
      CALL CURVE(XA,YA,3,0)                                             # T00860
      XA(1)=XA(2)                                                       # T00870
20    CONTINUE                                                          # T00880
      END IF                                                            # T00890
      CALL RESET ('HEIGHT')                                             # T00900
      RETURN                                                            # T00910
      END                                                               # T00920
C-----END OF ROUTINE-----------------------------------------------------# T00930
```

```
C                                                                       # U00010
C---Version 1.0   July 21, 1989                                         # U00020
C***********************************************************************# U00030
C                                                                       # U00040
C                          PTIMS                                        # U00050
C                                                                       # U00060
C  THIS ROUTINE PLOTS TIMESERIES DATA                                   # U00070
C                                                                       # U00080
C***********************************************************************# U00090
C                                                                       # U00100
      SUBROUTINE PTIMS (IU,XMIN,XMAX,YMIN,YMAX,ZMN,ZMX,NCOL,NROW,NLAY,  # U00110
     1 IVIEW,VEX,NPSTPS,ISPLOT,NISP,IPROJ,NSEC,NCLR)                    # U00120
      DIMENSION ZMN(NLAY),ZMX(NLAY),ISPLOT(NISP)                        # U00130
C                                                                       # U00140
      NN=1                                                              # U00150
      IF(NPSTPS.LT.0) THEN                                              # U00160
      KSTP=0                                                            # U00170
      NEXT=1                                                            # U00180
      ELSE                                                              # U00190
      KSTP=ISPLOT(NN)                                                   # U00200
      NEXT= ISPLOT(NN+1)                                                # U00210
      END IF                                                            # U00220
      CALL MARKER (3)                                                   # U00230
      CALL SCLPIC (0.05)                                                # U00240
      IF(NCLR.GE.0) THEN                                                # U00250
      KCLR=NCLR                                                         # U00260
      IF(KCLR.EQ.0) KCLR=1                                              # U00270
      CALL PKCLR (KCLR)                                                 # U00280
      END IF                                                            # U00290
10    CONTINUE                                                          # U00300
      IF (IVIEW.EQ.1) THEN                                              # U00310
      READ(IU,*,END=20) KNT,N,J,I,K,XPT,YPT,ZL,ZLL,TIME                 # U00320
      LSEC=K                                                            # U00330
      ELSE IF (IVIEW.EQ.2) THEN                                         # U00340
      READ(IU,*,END=20) KNT,N,J,I,K,XX,XPT,ZL,ZLL,TIME                  # U00350
      LSEC=J                                                            # U00360
      IF(ZLL.GE.0.0) YPT= (1.0-ZLL)*ZMN(K) + ZLL*ZMX(K)                 # U00370
      IF(ZLL.LT.0.0) YPT= (1.0E+0+ZLL)*ZMN(K) - ZLL*ZMX(K+1)            # U00380
      ELSE IF (IVIEW.EQ.3) THEN                                         # U00390
      READ(IU,*,END=20) KNT,N,J,I,K,XPT,YY,ZL,ZLL,TIME                  # U00400
      LSEC=I                                                            # U00410
      IF(ZLL.GE.0.0) YPT= (1.0-ZLL)*ZMN(K) + ZLL*ZMX(K)                 # U00420
      IF(ZLL.LT.0.0) YPT= (1.0E+0+ZLL)*ZMN(K) - ZLL*ZMX(K+1)            # U00430
      END IF                                                            # U00440
      ICK=0                                                             # U00450
      IF(XPT.GT.XMAX.OR.XPT.LT.XMIN) ICK=1                              # U00460
      IF(YPT.GT.YMAX.OR.YPT.LT.YMIN) ICK=1                              # U00470
      IF(IPROJ.EQ.1.AND.LSEC.NE.NSEC) ICK=1                             # U00480
      IF (KNT.EQ.KSTP) THEN                                             # U00490
      IF(ICK.EQ.0) CALL CURVE (XPT,YPT,1,-1)                            # U00500
      GO TO 10                                                          # U00510
      ELSE IF (KNT.EQ.NEXT) THEN                                        # U00520
      KSTP=NEXT                                                         # U00530
      IF(NCLR.EQ.0) THEN                                                # U00540
      KCLR=KCLR+1                                                       # U00550
      CALL PKCLR (KCLR)                                                 # U00560
      END IF                                                            # U00570
      IF(NPSTPS.LT.0) THEN                                              # U00580
      NEXT=KSTP+1                                                       # U00590
      IF(ICK.EQ.0) CALL CURVE (XPT,YPT,1,-1)                            # U00600
      ELSE                                                              # U00610
      NN=NN+1                                                           # U00620
      NEXT= ISPLOT(NN+1)                                                # U00630
      IF(ICK.EQ.0) CALL CURVE (XPT,YPT,1,-1)                            # U00640
      END IF                                                            # U00650
      ELSE IF (KNT.GT.KSTP.AND.NEXT.EQ.0) THEN                          # U00660
      GO TO 20                                                          # U00670
      END IF                                                            # U00680
      GO TO 10                                                          # U00690
20    CONTINUE                                                          # U00700
      RETURN                                                            # U00710
      END                                                               # U00720
```

```
C-----END OF ROUTINE----------------------------------------------------# U00730
```

```
C                                                                       # V00010
C---Version 1.0   July 21, 1989                                         # V00020
C***********************************************************************# V00030
C                                                                       # V00040
C                            PKCLR                                      # V00050
C                                                                       # V00060
C  THIS ROUTINE SELECTS A PEN COLOR                                     # V00070
C                                                                       # V00080
C***********************************************************************# V00090
C                                                                       # V00100
      SUBROUTINE PKCLR (KCLR)                                           # V00110
      IF(KCLR.LE.4) GO TO 20                                            # V00120
10    KCLR=KCLR-4                                                       # V00130
      IF(KCLR.LE.4) GO TO 20                                            # V00140
      GO TO 10                                                          # V00150
20    CONTINUE                                                          # V00160
      IF (KCLR.EQ.1) THEN                                               # V00170
      CALL RESET ('SETCLR')                                             # V00180
      ELSE IF (KCLR.EQ.2) THEN                                          # V00190
      CALL SETCLR ('RED')                                               # V00200
      ELSE IF (KCLR.EQ.3) THEN                                          # V00210
      CALL SETCLR ('GREEN')                                             # V00220
      ELSE IF (KCLR.EQ.4) THEN                                          # V00230
      CALL SETCLR ('BLUE')                                              # V00240
      END IF                                                            # V00250
      RETURN                                                            # V00260
      END                                                               # V00270
C-----END OF ROUTINE-------------------------------------------------- # V00280
```

```
C                                                                       # W00010
C---Version 1.0   July 21, 1989                                         # W00020
C***********************************************************************# W00030
C                                                                       # W00040
C                          PKPAT                                        # W00050
C                                                                       # W00060
C  THIS ROUTINE SELECTS A PEN PATTERN                                   # W00070
C                                                                       # W00080
C***********************************************************************# W00090
C                                                                       # W00100
      SUBROUTINE PKPAT (IPAT)                                           # W00110
      IF(IPAT.LE.3) GO TO 20                                            # W00120
10    IPAT=IPAT-3                                                       # W00130
      IF(IPAT.LE.3) GO TO 20                                            # W00140
      GO TO 10                                                          # W00150
20    CONTINUE                                                          # W00160
      IF (IPAT.EQ.1) THEN                                               # W00170
      CALL RESET ('DASH')                                               # W00180
      CALL RESET ('DOT')                                                # W00190
      ELSE IF (IPAT.EQ.2) THEN                                          # W00200
      CALL DASH                                                         # W00210
      ELSE IF (IPAT.EQ.3) THEN                                          # W00220
      CALL DOT                                                          # W00230
      END IF                                                            # W00240
      RETURN                                                            # W00250
      END                                                               # W00260
C-----END OF ROUTINE-------------------------------------------------- # W00270
```

```
C                                                                       # X00010
C---Version 1.0   July 21, 1989                                         # X00020
C***********************************************************************# X00030
C                                                                       # X00040
C                          FILES                                        # X00050
C                                                                       # X00060
C  THIS ROUTINE OPENS FILES                                             # X00070
C                                                                       # X00080
C***********************************************************************# X00090
C                                                                       # X00100
      SUBROUTINE FILES (IA,I0,I1,I2,I3,I4,I7)                           # X00110
      CHARACTER*80 FNAME                                                # X00120
C                                                                       # X00130
C  CREATE AND OPEN STANDARD OUTPUT AND SCRATCH FILES                    # X00140
C                                                                       # X00150
      FNAME= 'SUMMARY.PLT'                                              # X00160
      CALL OPNFIL (I7,FNAME,4,I7,0,3)                                   # X00170
C                                                                       # X00180
C  OPEN FILE CONTAINING FILE NAMES AND FORTRAN UNITS                    # X00190
C                                                                       # X00200
      WRITE  (*,*) 'ENTER NAME OF FILE CONTAINING NAMES AND UNITS OF DAT# X00210
     1A FILES:'                                                         # X00220
      READ  (*,5000) FNAME                                              # X00230
5000   FORMAT (A)                                                       # X00240
C                                                                       # X00250
      CALL OPNFIL (I0,FNAME,1,I7,0,1)                                   # X00260
C                                                                       # X00270
      KFIRST=0                                                          # X00280
10    READ (I0,*,END=20 ) IU,FNAME                                      # X00290
      IF (KFIRST.EQ.0) I1=IU                                            # X00300
      KFIRST=1                                                          # X00310
C                                                                       # X00320
C  OPEN DATA FILES                                                      # X00330
C                                                                       # X00340
      CALL OPNFIL (IU,FNAME,1,I7,0,1)                                   # X00350
C                                                                       # X00360
      WRITE(I7,5010) IU,FNAME                                           # X00370
5010   FORMAT('FORTRAN UNIT ',I3,' ASSIGNED TO ',A60)                   # X00380
      GO TO 10                                                          # X00390
20    CONTINUE                                                          # X00400
      RETURN                                                            # X00410
      END                                                               # X00420
C-----END OF ROUTINE-------------------------------------------------- # X00430
```

```
C                                                                       # Y00010
C---Version 1.0   July 21, 1989                                         # Y00020
C*********************************************************************** # Y00030
C                                                                       # Y00040
C                           COUNTP                                      # Y00050
C                                                                       # Y00060
C  THIS ROUTINE READS THE ENDPOINT FILE AND COUNTS THE NUMBER OF        # Y00070
C  PARTICLES RECORDED IN THE FILE. IT REWINDS THE FILE WHEN THE         # Y00080
C  COUNT IS FINISHED                                                    # Y00090
C                                                                       # Y00100
C***********************************************************************# Y00110
C                                                                       # Y00120
      SUBROUTINE COUNTP (IU,NPART)                                      # Y00130
      N=0                                                               # Y00140
10    READ (IU,5000,END=20 )                                            # Y00150
5000   FORMAT(I10)                                                      # Y00160
      N=N+1                                                             # Y00170
      GO TO 10                                                          # Y00180
20    CONTINUE                                                          # Y00190
      REWIND (IU)                                                       # Y00200
      IF (N.GT.NPART) THEN                                              # Y00210
      WRITE (*,5010) N,NPART                                            # Y00220
5010   FORMAT (I6,' PARTICLES IN ENDPOINT FILE (MAXIMUM ALLOWED IS',I6, # Y00230
     1')')                                                              # Y00240
      WRITE (*,5020) NPART                                              # Y00250
5020   FORMAT ('RERUN PROGRAM AND SPECIFY NUMBER OF PARTICLES >= ',I6)  # Y00260
      STOP                                                              # Y00270
      END IF                                                            # Y00280
      RETURN                                                            # Y00290
      END                                                               # Y00300
C-----END OF ROUTINE----------------------------------------------------# Y00310
```

```
C                                                                       # Z00010
C---Version 1.0   July 21, 1989                                         # Z00020
C***********************************************************************# Z00030
C                                                                       # Z00040
C                          PDRAIN                                       # Z00050
C                                                                       # Z00060
C  THIS ROUTINE READS DRAIN DATA AND PLOTS DRAINS                       # Z00070
C                                                                       # Z00080
C***********************************************************************# Z00090
C                                                                       # Z00100
      SUBROUTINE PDRAIN (IMIN,IMAX,JMIN,JMAX,KMIN,KMAX,XMN,XMX,        # Z00110
     1YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUDRAI,NCLR,I7,IPROJ,ISLICE) # Z00120
      DIMENSION XMN(NCOL),XMX(NCOL),YMN(NROW),YMX(NROW),ZMN(NLAY),      # Z00130
     1ZMX(NLAY)                                                         # Z00140
C                                                                       # Z00150
      IV= IVIEW*IPROJ                                                   # Z00160
      IF(NCLR.GE.0) THEN                                                # Z00170
      CALL PKCLR (2)                                                    # Z00180
      END IF                                                            # Z00190
      WRITE(I7,5000)                                                    # Z00200
5000  FORMAT('-------------------------------------------------------------# Z00210
     1----------------------')                                          # Z00220
      WRITE(I7,*) 'DRAIN DATA BEING READ...'                            # Z00230
      READ(IUDRAI,5010) MXW,ID                                          # Z00240
      READ(IUDRAI,5010) NDRAI                                           # Z00250
5010   FORMAT(2I10)                                                     # Z00260
      IF(IVIEW.GT.1) CALL DASH                                          # Z00270
      DO 10 N=1,NDRAI                                                   # Z00280
      READ(IUDRAI,5020) K,I,J,H,C,IFACE                                 # Z00290
      IF(IFACE.NE.0) GO TO 10                                           # Z00300
      IF(I.LT.IMIN.OR.I.GT.IMAX) GO TO 10                               # Z00310
      IF(J.LT.JMIN.OR.J.GT.JMAX) GO TO 10                               # Z00320
      IF(K.LT.KMIN.OR.K.GT.KMAX) GO TO 10                               # Z00330
      IF (IV.EQ.1.AND.K.NE.ISLICE) GO TO 10                             # Z00340
      IF (IVIEW.EQ.2.AND.J.NE.ISLICE) GO TO 10                          # Z00350
      IF (IVIEW.EQ.3.AND.I.NE.ISLICE) GO TO 10                          # Z00360
      IF(IVIEW.EQ.1) THEN                                               # Z00370
      X=(XMX(J)+XMN(J))/2.0                                             # Z00380
      Y=(YMX(I)+YMN(I))/2.0                                             # Z00390
      DX=XMX(J)-XMN(J)                                                  # Z00400
      DY=YMX(I)-YMN(I)                                                  # Z00410
      ELSE IF(IVIEW.EQ.2) THEN                                          # Z00420
      X=(YMX(I)+YMN(I))/2.0                                             # Z00430
      Y=(ZMX(K)+ZMN(K))/2.0                                             # Z00440
      DX=YMX(I)-YMN(I)                                                  # Z00450
      DY=ZMX(K)-ZMN(K)                                                  # Z00460
      ELSE IF(IVIEW.EQ.3) THEN                                          # Z00470
      X=(XMX(J)+XMN(J))/2.0                                             # Z00480
      Y=(ZMX(K)+ZMN(K))/2.0                                             # Z00490
      DX=XMX(J)-XMN(J)                                                  # Z00500
      DY=ZMX(K)-ZMN(K)                                                  # Z00510
      END IF                                                            # Z00520
      IF (IVIEW.EQ.1) THEN                                              # Z00530
      CALL SYMBL (X,Y,DX,DY,1)                                          # Z00540
      ELSE                                                              # Z00550
      CALL SYMBL (X,Y,DX,DY,0)                                          # Z00560
      END IF                                                            # Z00570
10    CONTINUE                                                          # Z00580
      IF (IVIEW.GT.1) CALL RESET ('DASH')                               # Z00590
5020   FORMAT(3I10,2F10.0,I10)                                          # Z00600
      WRITE(I7,*) '  DRAIN DATA HAS BEEN READ...'                       # Z00610
      IF (NCLR.GE.0) CALL PKCLR (1)                                     # Z00620
      RETURN                                                            # Z00630
      END                                                               # Z00640
C-----END OF ROUTINE-------------------------------------------------------# Z00650
```

```
C                                                                       #AA00010
C---Version 1.0   July 21, 1989                                         #AA00020
C***********************************************************************#AA00030
C                                                                       #AA00040
C                                  PGHB                                 #AA00050
C                                                                       #AA00060
C  THIS ROUTINE READS GHB DATA AND PLOTS GENERAL HEAD BOUNDARIES        #AA00070
C                                                                       #AA00080
C***********************************************************************#AA00090
C                                                                       #AA00100
      SUBROUTINE PGHB (IMIN,IMAX,JMIN,JMAX,KMIN,KMAX,XMN,XMX,          #AA00110
     1YMN,YMX,ZMN,ZMX,NCOL,NROW,NLAY,IVIEW,IUGHB,NCLR,I7,IPROJ,ISLICE)  #AA00120
      DIMENSION XMN(NCOL),XMX(NCOL),YMN(NROW),YMX(NROW),ZMN(NLAY),      #AA00130
     1ZMX(NLAY)                                                         #AA00140
C                                                                       #AA00150
      IV= IVIEW*IPROJ                                                   #AA00160
      IF(NCLR.GE.0) THEN                                                #AA00170
      CALL PKCLR (2)                                                    #AA00180
      END IF                                                            #AA00190
      WRITE(I7,5000)                                                    #AA00200
5000  FORMAT('-----------------------------------------------------------#AA00210
     1----------------------')                                          #AA00220
      WRITE(I7,*) 'GHB DATA BEING READ...'                              #AA00230
      READ(IUGHB,5010) MXW,ID                                           #AA00240
      READ(IUGHB,5010) NGHB                                             #AA00250
5010   FORMAT(2I10)                                                     #AA00260
      IF(IVIEW.GT.1) CALL DASH                                          #AA00270
      DO 10 N=1,NGHB                                                    #AA00280
      READ(IUGHB,5020) K,I,J,H,C,IFACE                                  #AA00290
      IF(IFACE.NE.0) GO TO 10                                           #AA00300
      IF(I.LT.IMIN.OR.I.GT.IMAX) GO TO 10                               #AA00310
      IF(J.LT.JMIN.OR.J.GT.JMAX) GO TO 10                               #AA00320
      IF(K.LT.KMIN.OR.K.GT.KMAX) GO TO 10                               #AA00330
      IF (IV.EQ.1.AND.K.NE.ISLICE) GO TO 10                             #AA00340
      IF (IVIEW.EQ.2.AND.J.NE.ISLICE) GO TO 10                          #AA00350
      IF (IVIEW.EQ.3.AND.I.NE.ISLICE) GO TO 10                          #AA00360
      IF(IVIEW.EQ.1) THEN                                               #AA00370
      X=(XMX(J)+XMN(J))/2.0                                             #AA00380
      Y=(YMX(I)+YMN(I))/2.0                                             #AA00390
      DX=XMX(J)-XMN(J)                                                  #AA00400
      DY=YMX(I)-YMN(I)                                                  #AA00410
      ELSE IF(IVIEW.EQ.2) THEN                                          #AA00420
      X=(YMX(I)+YMN(I))/2.0                                             #AA00430
      Y=(ZMX(K)+ZMN(K))/2.0                                             #AA00440
      DX=YMX(I)-YMN(I)                                                  #AA00450
      DY=ZMX(K)-ZMN(K)                                                  #AA00460
      ELSE IF(IVIEW.EQ.3) THEN                                          #AA00470
      X=(XMX(J)+XMN(J))/2.0                                             #AA00480
      Y=(ZMX(K)+ZMN(K))/2.0                                             #AA00490
      DX=XMX(J)-XMN(J)                                                  #AA00500
      DY=ZMX(K)-ZMN(K)                                                  #AA00510
      END IF                                                            #AA00520
      IF (IVIEW.EQ.1) THEN                                              #AA00530
      CALL SYMBL (X,Y,DX,DY,1)                                          #AA00540
      ELSE                                                              #AA00550
      CALL SYMBL (X,Y,DX,DY,0)                                          #AA00560
      END IF                                                            #AA00570
10    CONTINUE                                                          #AA00580
      IF (IVIEW.GT.1) CALL RESET ('DASH')                               #AA00590
5020   FORMAT(3I10,2F10.0,I10)                                          #AA00600
      WRITE(I7,*) '  GHB DATA HAS BEEN READ...'                         #AA00610
      IF (NCLR.GE.0) CALL PKCLR (1)                                     #AA00620
      RETURN                                                            #AA00630
      END                                                               #AA00640
C-----END OF ROUTINE-------------------------------------------------------#AA00650
```

```
C                                                                       #AB00010
C---Version 1.0   July 21, 1989                                         #AB00020
C***********************************************************************#AB00030
C                                                                       #AB00040
C                           OPNFIL                                      #AB00050
C                                                                       #AB00060
C  THIS ROUTINE OPENS A SINGLE FILE.                                    #AB00070
C                                                                       #AB00080
C***********************************************************************#AB00090
C                                                                       #AB00100
      SUBROUTINE OPNFIL (IU,FNAME,NSTAT,IOUT,IBATCH,NACT)               #AB00110
      CHARACTER*80 FNAME,FMT,FM                                         #AB00120
      LOGICAL*4 EX                                                      #AB00130
C                                                                       #AB00140
C                       VARIABLES                                       #AB00150
C                                                                       #AB00160
C  IU = FORTRAN UNIT NUMBER OF FILE                                     #AB00170
C  FNAME = FILE NAME                                                    #AB00180
C  NSTAT = STATUS OF FILE                                               #AB00190
C          1 => OLD FILE                                                #AB00200
C          2 => NEW FILE                                                #AB00210
C          3 => SCRATCH FILE (DELETED AUTOMATICALLY WHEN RUN ENDS)      #AB00220
C          4 => UNDETERMINED STATUS (MAY OR MAY NOT EXIST)              #AB00230
C                 IF IT DOES NOT EXIST, IT IS CREATED BY OPEN STATEMENT #AB00240
C                 IF IT DOES EXIST, IT IS OPENED AS 'OLD' FILE          #AB00250
C                                                                       #AB00260
C  IOUT = UNIT NUMBER FOR OUTPUT FILE TO WRITE ERROR MESSAGE TO         #AB00270
C         IF NECESSARY                                                  #AB00280
C                                                                       #AB00290
C  IBATCH = FLAG INDICATING IF THERE IS INTERACTIVE DIALOGUE AT         #AB00300
C           TERMINAL                                                    #AB00310
C                                                                       #AB00320
C           0 => THERE IS INTERACTIVE DIALOGUE                          #AB00330
C           1 => THERE IS NOT INTERACTIVE DIALOGUE (BATCH MODE)         #AB00340
C                                                                       #AB00350
C  NACT = VARIABLE DENOTING IF A FILE IS READ ONLY, WRITE ONLY, OR      #AB00360
C         READ AND WRITE.                                               #AB00370
C                                                                       #AB00380
C           1 => READ ONLY                                              #AB00390
C           2 => WRITE ONLY                                             #AB00400
C           3 => READ AND WRITE                                         #AB00410
C                                                                       #AB00420
C  "NACT" IS NOT USED IN THIS SUBROUTINE. ALL FILES ARE OPENED FOR      #AB00430
C  READING AND WRITING. OPENING "READ ONLY" AND "WRITE ONLY" FILES      #AB00440
C  IS MACHINE DEPENDENT. THE VARIABLE "NACT" IS PASSED TO THIS          #AB00450
C  ROUTINE TO MAKE IT EASY TO MODIFY THE OPEN STATEMENTS TO ALLOW       #AB00460
C  FOR READ AND WRITE ONLY FILES ON ANY GIVEN MACHINE. THE CALLS TO     #AB00470
C  THIS SUBROUTINE ARE CURRENTLY SET UP SO THAT ANY FILE THAT IS        #AB00480
C  WRITTEN TO IS GIVEN "NACT = 3" AND ANY FILE THAT IS READ FROM BUT    #AB00490
C  NOT WRITTEN TO IS GIVEN "NACT = 1". NO FILES ARE GIVEN "WRITE ONLY"  #AB00500
C  STATUS. "READ ONLY" STATUS IS USEFUL IF A NUMBER OF USERS WANT       #AB00510
C  TO SIMULTANEOUSLY SHARE INPUT FILES.                                 #AB00520
C                                                                       #AB00530
C  IF THE FILE DOES NOT CURRENTLY EXIST, A NEGATIVE UNIT NUMBER IS      #AB00540
C  USED AS A FLAG TO INDICATED THAT A NEW FILE SHOULD BE CREATED AS     #AB00550
C  AN UNFORMATTED (BINARY) FILE.                                        #AB00560
C                                                                       #AB00570
      IBIN=0                                                            #AB00580
      IF (IU.LT.0) THEN                                                 #AB00590
      IU= -IU                                                           #AB00600
      IBIN=1                                                            #AB00610
      END IF                                                            #AB00620
C                                                                       #AB00630
C  CHECK TO SEE IF A FILE IS OPENED TO UNIT=IOUT SO THAT ERROR          #AB00640
C  MESSAGES CAN BE WRITTEN.                                             #AB00650
      INQUIRE (UNIT=IOUT,OPENED=EX)                                     #AB00660
      IO=0                                                              #AB00670
      IF (EX) IO=1                                                      #AB00680
C                                                                       #AB00690
C  OPEN AN EXISTING FILE                                                #AB00700
C                                                                       #AB00710
      IF (NSTAT.EQ.1) THEN                                              #AB00720
```

```
10    INQUIRE (FILE=FNAME,EXIST=EX,UNFORMATTED=FM)                          #AB00730
C  CHECK TO SEE IF FILE EXISTS                                              #AB00740
      IF (EX) GO TO 20                                                      #AB00750
      IF (IBATCH.EQ.0) THEN                                                 #AB00760
      WRITE (*,*) 'FILE DOES NOT EXIST.'                                    #AB00770
      WRITE (*,*) 'ENTER THE NAME OF AN EXISTING FILE (<CR>=QUIT):'         #AB00780
      READ (*,'(A)') FNAME                                                  #AB00790
      IF (FNAME.EQ.' ') STOP                                                #AB00800
      GO TO 10                                                              #AB00810
      ELSE                                                                  #AB00820
      IF (IO.EQ.1) WRITE (IOUT,*) 'FILE DOES NOT EXIST:'                    #AB00830
      IF (IO.EQ.1) WRITE (IOUT,'(A)') FNAME                                 #AB00840
      STOP                                                                  #AB00850
      END IF                                                                #AB00860
20    CONTINUE                                                              #AB00870
      FMT='FORMATTED'                                                       #AB00880
      IF (IBIN.EQ.1.OR.FM.EQ.'YES') FMT='UNFORMATTED'                       #AB00890
      OPEN (IU,FILE=FNAME,STATUS='OLD',FORM=FMT,IOSTAT=IERR)                #AB00900
      IF (IERR.GT.0) GO TO 40                                               #AB00910
      RETURN                                                                #AB00920
      END IF                                                                #AB00930
C                                                                           #AB00940
C  OPEN A NEW FILE                                                          #AB00950
C                                                                           #AB00960
      IF (NSTAT.EQ.2) THEN                                                  #AB00970
30    INQUIRE (FILE=FNAME,EXIST=EX)                                         #AB00980
      IF (EX) THEN                                                          #AB00990
      IF (IBATCH.EQ.0) THEN                                                 #AB01000
      WRITE (*,*) 'FILE ALREADY EXISTS.'                                    #AB01010
      WRITE (*,*) 'ENTER THE NAME OF A NEW FILE (<CR>=QUIT):'               #AB01020
      READ (*,'(A)') FNAME                                                  #AB01030
      IF (FNAME.EQ.' ') STOP                                                #AB01040
      GO TO 30                                                              #AB01050
      ELSE                                                                  #AB01060
      IF (IO.EQ.1) WRITE (IOUT,*) 'FILE ALREADY EXISTS:'                    #AB01070
      IF (IO.EQ.1) WRITE (IOUT,'(A)') FNAME                                 #AB01080
      STOP                                                                  #AB01090
      END IF                                                                #AB01100
      END IF                                                                #AB01110
      FMT='FORMATTED'                                                       #AB01120
      IF (IBIN.EQ.1) FMT='UNFORMATTED'                                      #AB01130
      OPEN (IU,FILE=FNAME,STATUS='NEW',FORM=FMT,IOSTAT=IERR)                #AB01140
      IF (IERR.GT.0) GO TO 40                                               #AB01150
      RETURN                                                                #AB01160
      END IF                                                                #AB01170
C                                                                           #AB01180
C  OPEN A SCRATCH FILE                                                      #AB01190
C                                                                           #AB01200
      IF (NSTAT.EQ.3) THEN                                                  #AB01210
      FMT='FORMATTED'                                                       #AB01220
      IF (IBIN.EQ.1) FMT='UNFORMATTED'                                      #AB01230
      OPEN (IU,STATUS='SCRATCH',FORM=FMT,IOSTAT=IERR)                       #AB01240
C  FOR MICROSOFT FORTRAN USE:                                               #AB01250
C      OPEN (IU,FORM=FMT,IOSTAT=IERR)                                       #AB01260
      IF (IERR.GT.0) GO TO 40                                               #AB01270
      RETURN                                                                #AB01280
      END IF                                                                #AB01290
C                                                                           #AB01300
C  OPEN A FILE OF UNKNOWN STATUS                                            #AB01310
C                                                                           #AB01320
      IF (NSTAT.EQ.4) THEN                                                  #AB01330
      INQUIRE (FILE=FNAME,EXIST=EX,UNFORMATTED=FM)                          #AB01340
      IF (EX) THEN                                                          #AB01350
      FMT='FORMATTED'                                                       #AB01360
      IF (IBIN.EQ.1.OR.FM.EQ.'YES') FMT='UNFORMATTED'                       #AB01370
      OPEN (IU,FILE=FNAME,STATUS='OLD',FORM=FMT,IOSTAT=IERR)                #AB01380
      IF (IERR.GT.0) GO TO 40                                               #AB01390
      ELSE                                                                  #AB01400
      FMT='FORMATTED'                                                       #AB01410
      IF (IBIN.EQ.1) FMT='UNFORMATTED'                                      #AB01420
      OPEN (IU,FILE=FNAME,STATUS='NEW',FORM=FMT,IOSTAT=IERR)                #AB01430
      IF (IERR.GT.0) GO TO 40                                               #AB01440
```

```
      END IF                                                          #AB01450
      RETURN                                                          #AB01460
      END IF                                                          #AB01470
C                                                                     #AB01480
C  WRITE MESSAGE INDICATING PROBLEM OPENING FILE                      #AB01490
C                                                                     #AB01500
40    IF (IBATCH.EQ.0) WRITE (*,5000) IU                              #AB01510
      IF (IO.EQ.1) WRITE (IOUT,5000) IU                               #AB01520
5000   FORMAT ('ERROR OPENING FILE TO UNIT ',I3)                      #AB01530
      STOP                                                            #AB01540
      END                                                             #AB01550
C-----END OF ROUTINE----------------------------------------------------#AB01560
```