User's Guide for MODPATH/MODPATH-PLOT, Version 3: A particle tracking post-processing package for MODFLOW, the U. S. Geological Survey finite-difference ground-water flow model

by David W. Pollock

U. S. GEOLOGICAL SURVEY
Open-File Report 94-464

Reston, Virginia
September, 1994

U. S. DEPARTMENT OF THE INTERIOR
Bruce Babbitt, Secretary

U. S. GEOLOGICAL SURVEY
Gordon P. Eaton, Director

For additional information, contact:

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

This report can be purchased from:

U. S. Geological Survey
Earth Science Information Center
Box 25286, MS 517
Denver Federal Center
Denver, CO 80225-0046
(303) 236-7476

Preface

This report describes a particle tracking post-processing package, MODPATH/ MODPATH-PLOT, that computes and displays three-dimensional pathlines based on output from MODFLOW, the U. S. Geological Survey finite-difference ground-water flow model. MODPATH is intended for general use and may have to be modified by the user for specific applications. The methods used by MODPATH are based on specific assumptions that result in limitations that must be thoroughly understood to obtain meaningful results. Users are strongly encouraged to carefully read Chapter 2 (Particle Tracking Methodology) before undertaking analyses with MODPATH.

Readers are requested to notify U. S. Geological Survey, Office of Ground Water of any errors in this report or in the computer codes. Updates of both the report and the computer codes may be issued as necessary. To report errors and obtain information about updates, write to:

Office of Ground Water
U. S. Geological Survey
411 National Center
Reston, VA 22092

Copies of the FORTRAN source code for MODPATH/MODPATH-PLOT and sample problem data files are available for the cost of processing from:

U. S. Geological Survey
NWIS Program Office
437 National Center
Reston, VA 22092
(703) 648-5695

Contents

Illustrations

Tables

Conversion Factors

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

Multiply	By	To obtain
foot (ft)	0.3048	meters
day (d)	86,400	seconds
day (d)	1,440	minutes
cubic foot (ft3)	7.48	gallons

User's Guide for MODPATH/MODPATH-PLOT, Version 3: A particle tracking post-processing package for MODFLOW, the U. S. Geological Survey finite-difference ground-water flow model

by

David W. Pollock

Abstract

MODPATH is a particle tracking post-processing package that was developed to compute three-dimensional flow paths using output from steady-state or transient ground-water-flow simulations by MODFLOW, the U. S. Geological Survey finite-difference ground-water-flow model. The particle tracking package consists of two FORTRAN computer codes: (1) MODPATH, which calculates particle paths, and (2) MODPATH-PLOT, which displays results graphically. The current report documents the most recent versions of MODPATH and MODPATH-PLOT, which were originally described in USGS Open-File Reports 89-381 and 89-622.

MODPATH uses a semi-analytical particle tracking scheme that allows an analytical expression of the particle's flow path to be obtained within each finite-difference grid cell. Particle paths are computed by tracking particles from one cell to the next until the particle reaches a boundary, an internal sink/source, or satisfies some other termination criterion.

Data input for MODPATH and MODPATH-PLOT is a combination of data files and interactive keyboard input. Both programs are designed to work with MODFLOW. The number of new data files required by MODPATH is minimized by making use of MODFLOW data files whenever possible.

MODPATH and MODPATH-PLOT are written in standard FORTRAN-77. MODPATH can be compiled and run on any computer system that has a FORTRAN-77 compiler. In addition to a FORTRAN-77 compiler, MODPATH-PLOT requires a graphics subroutine library know as GKS (Graphical Kernel System). GKS is a standardized set of graphics routines that are available commercially for most computer systems.

Chapter 1
Introduction

What is MODPATH?

MODPATH is a particle tracking post-processing program designed to work with the U. S. Geological Survey's finite-difference ground-water-flow model, commonly known as MODFLOW (McDonald and Harbaugh, 1988). Output from steady-state or transient MODFLOW simulations is used in MODPATH to compute paths for imaginary "particles" of water moving through the simulated ground-water system. In addition to computing particle paths, MODPATH keeps track of the time of travel for particles moving through the system. By carefully defining the starting locations of particles, it is possible to perform a wide range of analyses, such as delineating capture and recharge areas or drawing flow nets.

The MODPATH particle tracking package consists of two separate computer programs: (1) MODPATH, which calculates particle paths and travel times, and (2) MODPATH-PLOT, which takes numerical output from MODPATH and displays the results in a variety of graphical formats. Both programs are written in standard FORTRAN-77 (ANSI, 1978). MODPATH can be compiled and run on any computer that has a FORTRAN-77 compiler. In addition to a FORTRAN-77 compiler, MODPATH-PLOT requires access to a standardized set of graphics routines called the Graphical Kernel System (GKS) (ANSI, 1985). GKS is available commercially for most computer systems. By keeping the particle tracking and graphics portions of the MODPATH package separate, users are free to develop customized graphics programs, or other post-processing applications, that make direct use of the numerical output from MODPATH.

This report documents the most recent versions of MODPATH (version 3.0) and MODPATH-PLOT (version 3.0). Previous versions of MODPATH and MODPATH-PLOT are described in Pollock (1989a, 1989b). Several additions and enhancements have been made to both programs. The most significant of these changes are described in the section, **New Features**.

New Features

The current version of the MODPATH particle-tracking package includes many new features that increase the ease of use and the analytical capabilities of the package. The most significant new features are:

- Transient Flow -- Previous versions of MODPATH were restricted to steady-state MODFLOW simulations. The current version of MODPATH allows forward or backward particle tracking analyses to be performed for transient as well as steady-state simulations.
- Multiple Release Times for Particles -- In forward tracking simulations, particles can be re-released at a regular time interval over a specified period of time. This option allows a "plume" of particles to be tracked through the ground-water flow system.
- Contouring -- Head, drawdown, or other gridded data (such as hydraulic conductivity) can be contoured in map view.
- Drawing True Cross-section Grids -- MODPATH-PLOT provides the option of drawing true cross-section grids that accurately show dipping and variable-thickness layers.
- Displaying Spatial Data with Shaded Grid Cells -- Grid cells can be shaded to display spatial information provided in the form of arrays of integer or real-number data. This option can be used to show hydrogeologic units or other types of spatial data, such as hydraulic conductivity.
- Drawing Commands -- MODPATH-PLOT defines a set of commands to perform basic drawing functions, such as draw lines and polygons, shade polygons, and display text. Map view plots can be customized by supplying MODPATH-PLOT with a specific set of drawing instructions in a special file called a "Drawing Commands File".
- Interactive Graphics -- On systems that support interactive GKS graphics, MODPATH-PLOT now supports limited use of a mouse or arrow keys to move a graphics cursor around the screen. The graphics cursor can be used to obtain the x-y and grid-cell indices of specific points on the grid in map view.
- Customized Input using Response Files -- Much of the data for MODPATH and MODPATH-PLOT is entered interactively at the keyboard. Both programs record interactive prompts and responses in a file called a "response file". Data for subsequent runs can be read directly from a response file generated by a previous run. Response files also can be customized by the user to direct MODPATH and MODPATH-PLOT to read some data from the response file and prompt the user to enter other data interactively.
- Free-format Data Files -- Free-format data entry is utilized for all data files. Free-format eliminates the need to place data in specific columns and makes the creation of data files faster and easier.

How to Use this Report

This report serves both as a general reference for the numerical methods used in MODPATH and as a user's guide for MODPATH and MODPATH-PLOT. Numerical methods are described in Chapter 2. Users are strongly advised to read Chapter 2 before undertaking a simulation study with MODPATH. A thorough understanding of how particles are tracked and the limitations of the method greatly reduces the possibility of misusing MODPATH and provides users with broader insight into creative ways to use MODPATH to solve ground-water problems.

Chapters 3 and 4 are user's guides for MODPATH and MODPATH-PLOT, respectively. These chapters are divided into a large number of topics that cover basic features and options of the codes. Each section is relatively self-contained and designed to be referred to, as necessary, during the course of using the codes. Before getting too heavily involved in MODPATH simulations, it is useful to scan chapters 3 and 4 to obtain an overview of the capabilities of the programs. For convenience, input instructions for all data files are provided in Appendix A. Input Files.

Chapter 6 presents two moderately complex sample problems that illustrate most of the basic types of analyses and graphical output generated by MODPATH and MODPATH-PLOT. A complete listing of all the MODFLOW and MODPATH data files needed to run the sample problems is provided in Appendix E. Although it is unwise to rely totally on sample problems as a means of learning how to create data files and run the programs, studying the sample problems in conjunction with the data input instructions in Appendix A should provide an effective introduction to MODPATH and MODPATH-PLOT.

Information related to installing MODPATH and MODPATH-PLOT on computer systems is provided in Appendix D. This report does not include a detailed description of the computer codes from a programming perspective. However, Appendix D does include a discussion of some specific aspects of the MODPATH-PLOT code structure that affect the portability of the GKS-graphics portion of the code among various types of computer systems.

Chapter 2
Particle Tracking Methodology

An Overview of the Particle Tracking Method

The particle tracking algorithm used by MODPATH can be implemented for either steady state or transient flow fields. For simplicity, the algorithm is first described for steady-state flow and then extended to transient flow situations.

Steady State Flow

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

$$\frac{\partial}{\partial x}(nv_x) + \frac{\partial}{\partial y}(nv_y) + \frac{\partial}{\partial z}(nv_z) = W \qquad 1$$

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 2-1 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,

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

2c, 2d

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

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

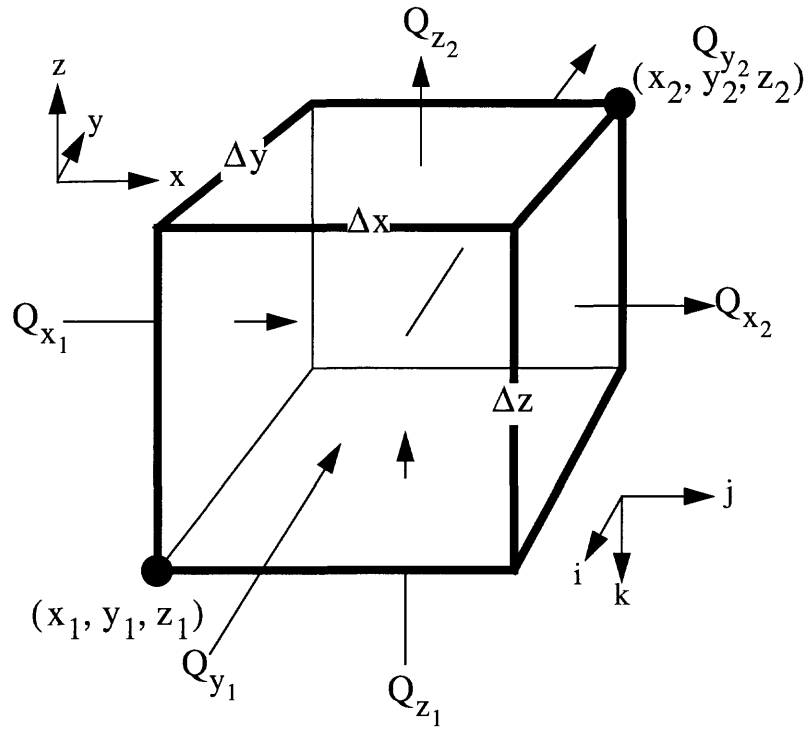


Figure 2-1. Finite-difference cell showing definitions of x-y-z and i-j-k

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,

$$\frac{(nv_{x_2} - nv_{x_1})}{\Delta x} + \frac{(nv_{y_2} - nv_{y_1})}{\Delta y} + \frac{(nv_{z_2} - nv_{z_1})}{\Delta z} = \frac{Q_s}{\Delta x \Delta y \Delta z} \qquad 3$$

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, commonly known as MODFLOW, solves for head and calculates intercell flow rates (McDonald and Harbaugh, 1988).

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,

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

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

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

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

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

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

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

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,

$$\left(\frac{dv_x}{dt}\right)_p = \left(\frac{dv_x}{dx}\right)\left(\frac{dx}{dt}\right)_p \qquad 6$$

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,

$$v_{x_p} = \left(\frac{dx}{dt}\right)_p \qquad 7$$

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

$$\left(\frac{dv_x}{dx}\right) = A_x \qquad 8$$

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

$$\left(\frac{dv_x}{dt}\right)_p = A_x v_{x_p} \qquad 9a$$

Analogous equations are obtained for the y and z directions,

$$\left(\frac{dv_y}{dt}\right)_p = A_y v_{y_p} \qquad 9b$$

$$\left(\frac{dv_z}{dt}\right)_p = A_z v_{z_p} \qquad 9c$$

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

$$\left(\frac{1}{v_{x_p}}\right) d(v_{x_p}) = A_x dt \qquad 10$$

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

$$\ln\left(\frac{v_{x_p}(t_2)}{v_{x_p}(t_1)}\right) = A_x \Delta t \qquad 11$$

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,

$$x_p(t_2) = x_1 + \left(\frac{1}{A_x}\right)\{v_{x_p}(t_1)\exp(A_x\Delta t) - v_{x_1}\} \qquad 12a$$

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

$$y_p(t_2) = y_1 + \left(\frac{1}{A_y}\right)\{v_{y_p}(t_1)\exp(A_y\Delta t) - v_{y_1}\} \qquad 12b$$

$$z_p(t_2) = z_1 + \left(\frac{1}{A_z}\right)\{v_{z_p}(t_1)\exp(A_z\Delta t) - v_{z_1}\} \qquad 12c$$

The velocity components of the particle at time t_1 are known functions of the particle's coordinates; 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 2-2. 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 the particle leaves cell (i,j). For the present example, this is accomplished by noting that the velocity components at the four faces require that the particle leave the cell through either face x_2 or face y_2. Consider the x-direction first. From equation (4a) v_{xp} can be calculated at the point (x_p,y_p). Because 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 the particle to reach face x_2,

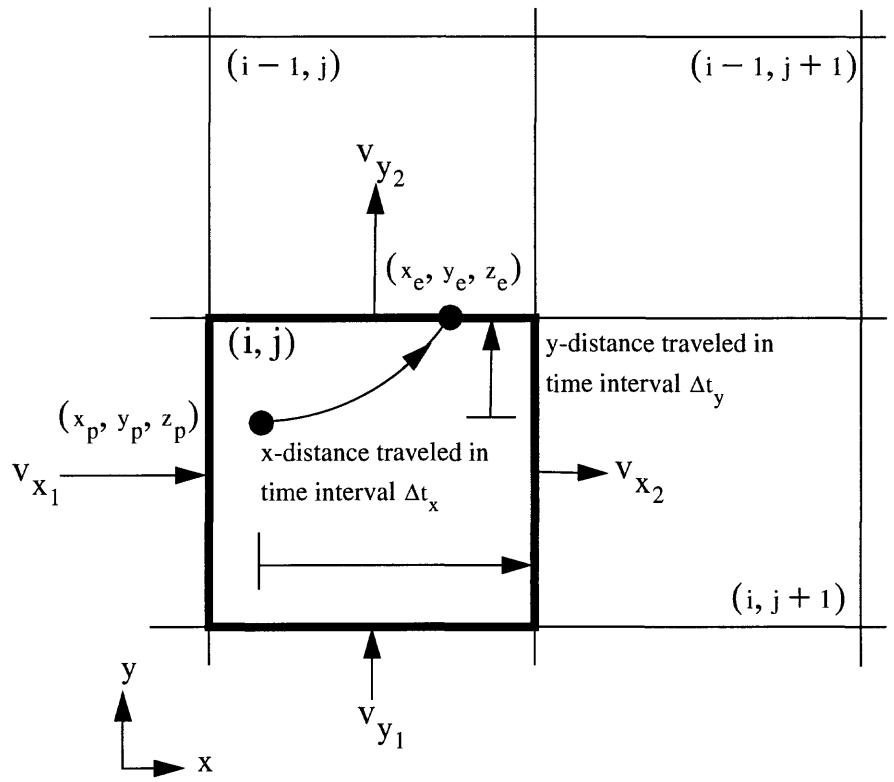


Figure 2-2. Schematic showing the computation of exit point and travel time for the case of two-dimensional flow in the x-y plane.

$$\Delta t_x = \left(\frac{1}{A_x}\right) \ln\left(\frac{v_{x_2}}{v_{x_p}}\right) \qquad 13a$$

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

$$\Delta t_y = \left(\frac{1}{A_y}\right) \ln\left(\frac{v_{y_2}}{v_{y_p}}\right) \qquad 13b$$

where v_{x_p} and v_{y_p} are the x and y components of velocity of the particle at (x_p,y_p). If Δt_x is less than Δt_y, the particle will leave the cell across face x_2 and enter cell (i,j+1). Conversely, if Δt_y is less than Δt_x, the particle 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 the particle would leave through the corner of cell (i,j) and enter cell (i-1,j+1). The particle trajectory shown in figure 2-2 corresponds to a situation where Δt_y is less than Δt_x. The length of time required for the particle 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 the particle as it leaves cell (i,j),

$$x_e = x_1 + \left(\frac{1}{A_x}\right) \{v_{x_p}(t_p) \exp(A_x \Delta t_e) - v_{x_1}\} \quad 14a$$

and

$$y_e = y_1 + \left(\frac{1}{A_y}\right) \{v_{y_p}(t_p) \exp(A_y \Delta t_e) - v_{y_1}\} \quad 14b$$

The time at which the particle 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.

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 example, one might want to know how far a particle would move in a specified time interval. In such 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. The method described in this report does not require time steps smaller than Δt_e because pathlines are integrated analytically with respect to time. The use of time steps that are smaller than Δt_e is solely for convenience. The ultimate discharge point and total time of travel for a given particle will be identical regardless of whether or not time steps smaller than Δt_e are used.

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 2-3 illustrates the other possible situations that can occur in any of the three coordinate directions. Figure 2-3a 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 2-3b 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 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. 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 of a particle in that

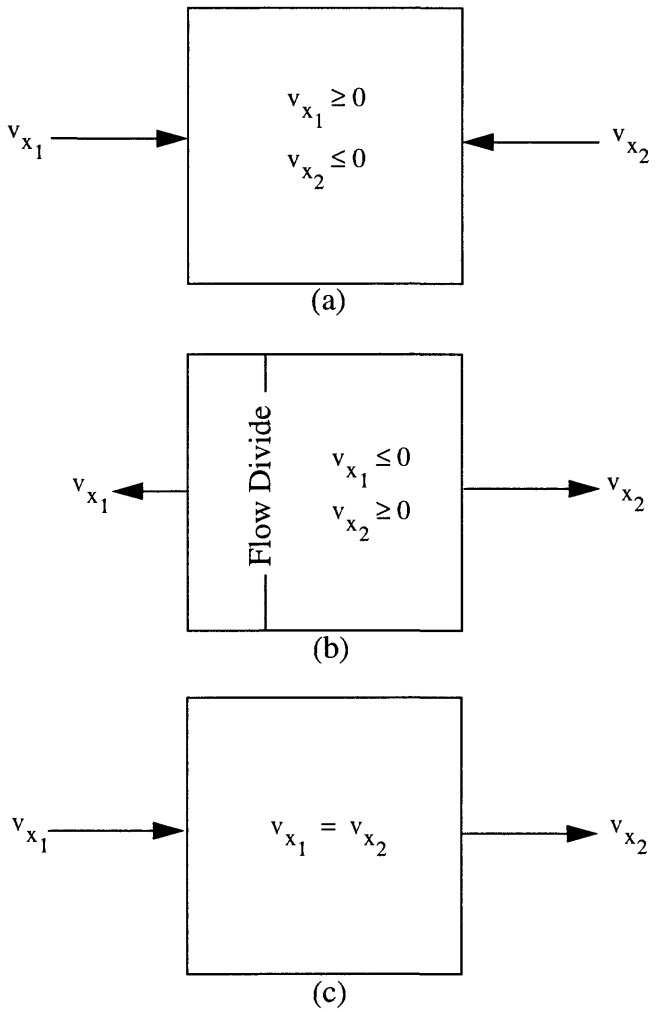


Figure 2-3. Additional combinations of velocities at cell-face pairs.

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 2-3c) 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. In that case, equation (13a) is bypassed and the transit time in the x-direction is computed from the simple relations,

$$\text{for } v_{x_1} > 0 \; : \quad \Delta t_x = \frac{(x_2 - x_p)}{v_{x_1}} \qquad 15a$$

or

$$\text{for } v_{x_1} < 0 \; : \quad \Delta t_x = \frac{(x_1 - x_p)}{v_{x_1}} \qquad 15b$$

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

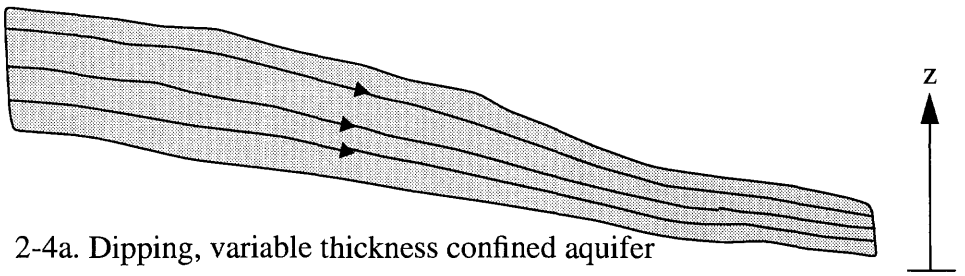
The development presented in the previous section was based on the assumption that the flow domain was discretized into a three-dimensional rectangular grid of horizontal rectangular cells. However, in practice, many three-dimensional finite-difference simulations use a rectangular grid in the horizontal 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. Figures 2-4a, b, and c show how a simple confined aquifer with variable thickness and elevation can be represented by a vertically deformed finite-difference grid.

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,

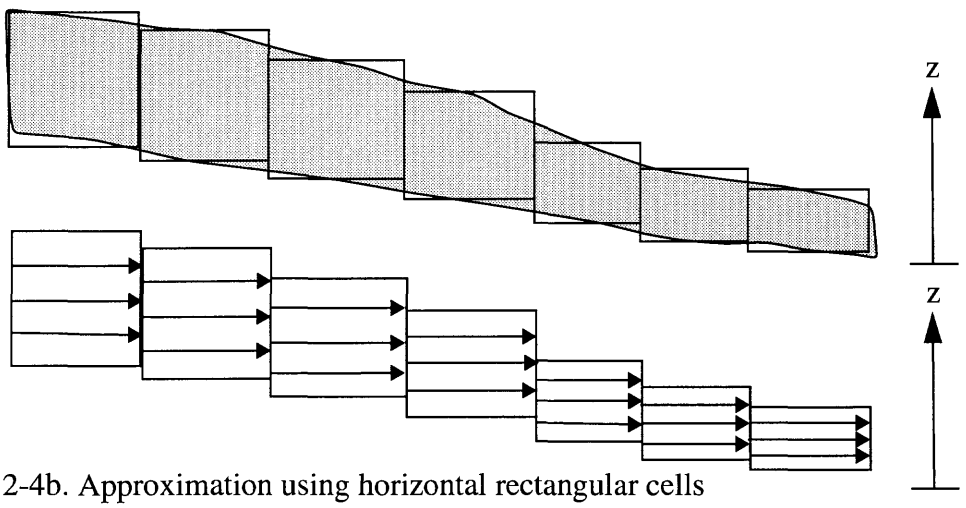
$$z_L = \frac{(z - z_1)}{(z_2 - z_1)} \qquad 16$$

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 2-4 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.

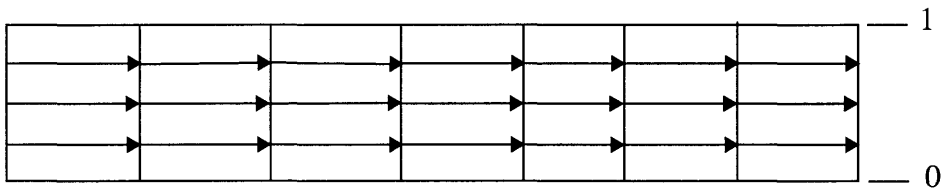
The advantage of a stratigraphic three-dimensional grid is that complex hydrogeologic



2-4a. Dipping, variable thickness confined aquifer



2-4b. Approximation using horizontal rectangular cells



2-4c. Grid represented in local coordinates

Figure 2-4. Schematic siltation of a finite-difference representation of an inclined aquifer with variable thickness.

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

Many ground water systems are characterized by sequences of highly transmissive, near-horizontal aquifers that are 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 assigned to be part of the model layer directly above it. 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 2-5). 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.

Discharge and Recharge 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 is necessary for cells that incorporate discharge or recharge. MODFLOW provides

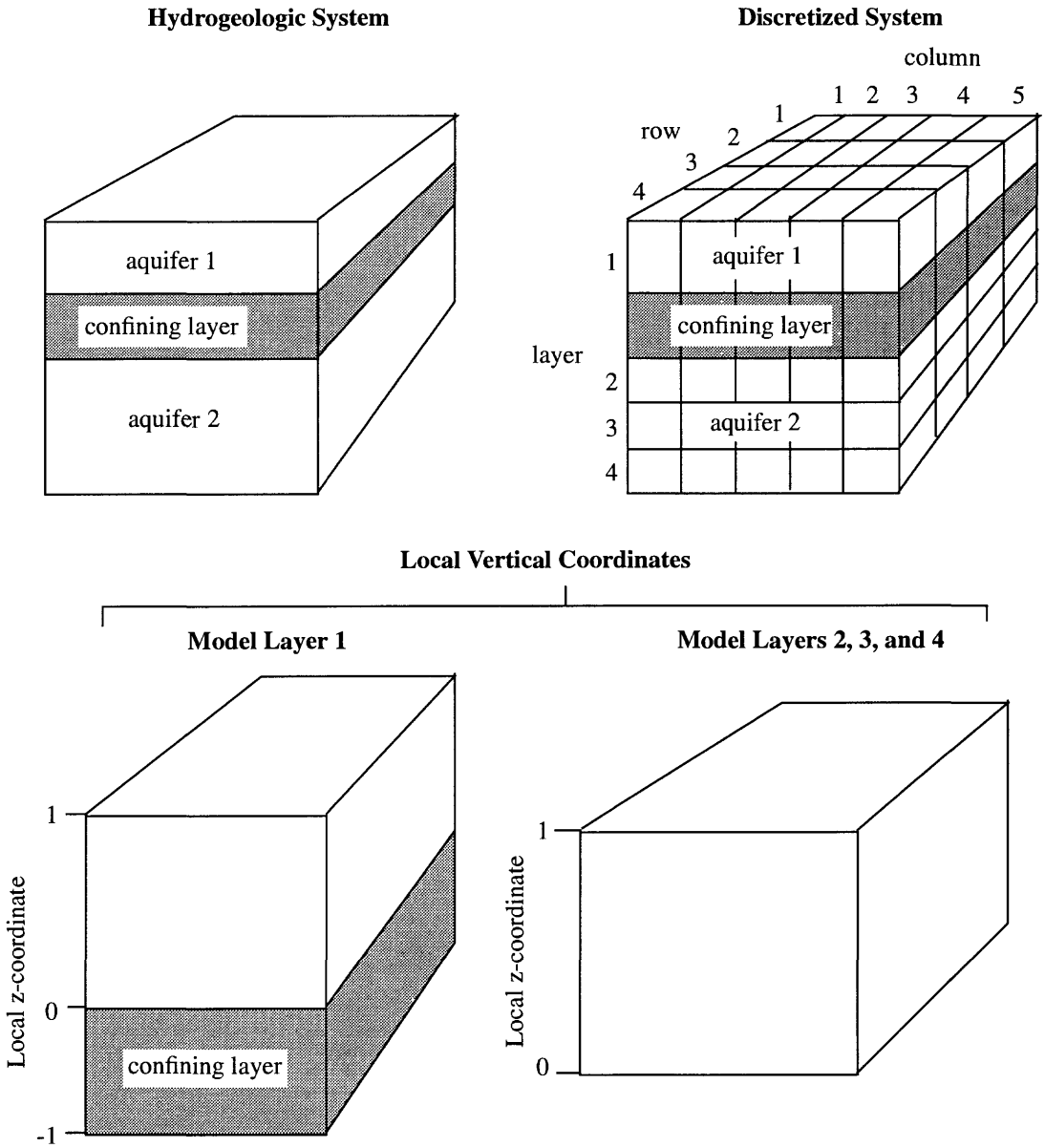


Figure 2-5. Definition of local vertical coordinates for quasi-3-d systems.

for three types of features to account for discharge and recharge:

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. Accordingly, for the purposes of path line computations, specified head cells are treated as active cells that

contain a net source or sink. Pathlines 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 MODFLOW does not compute rates of flow between adjacent specified head cells. Consequently, path lines that pass through 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 are identical for 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 often are accounted for in MODFLOW by using wells.

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 user can specify whether to treat a well as a distributed source or sink or as a flow across a cell face. If specified as a flow across a face, the appropriate velocity component is computed and assigned to that face. In addition to wells, MODFLOW's Recharge Package provides another way to specify flow at a boundary. Therefore, in MODPATH, recharge may be assigned to the top face of a cell, or treated as a distributed source. The distributed source approximation for areal recharge is usually only appropriate for two-dimensional areal flow models.

MODFLOW also includes options to simulate rivers, drains, general-head boundary fluxes, streams, and evapotranspiration as head-dependent fluxes. MODPATH can either assign

each flow to a specific cell face, or treat the flow as a distributed source or sink. When flow is assigned to a face, it can be assigned to any face for rivers, drains, general-head boundaries, and streams. 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 distributed 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, some of the water flowing into the cell discharges to the sink and some passes through the cell. 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) always stopping particles when they enter cells with any amount of discharge to internal sinks, (2) always 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 weak-sink 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. For example, a large number of particles can be distributed on the faces of a cell containing a well and tracked in reverse to their points of recharge, such as the water table. The final locations of these particles will define the capture area for the cell that contains the well.

Transient Flow

Transient finite-difference flow simulations consist of a series of discrete time steps during which flow rates are constant and storage changes within cells contribute an additional component to the internal source/sink term on the right side of equation 3. The particle tracking algorithm described previously for steady-state flow systems can be extended to transient finite-difference simulations by taking advantage of the fact that transient simulations behave as a series of steady-state flow periods. For each time step, particle paths are computed just as for the steady state case until the end of the time step is reached. A new velocity distribution is then calculated for the next time step and the computation of particle paths is resumed. The computation of paths forward or backward, boundary conditions, and the path line termination criteria are handled the same as for steady state flow.

Water table layers represent an additional complication in transient particle tracking analyses because the water table is actually a moving boundary. MODPATH deals with this problem in a simple way by assuming that the water table moves in discrete jumps from one time step to the next. The saturated thickness of a water table cell is assumed to remain constant over the length of a time step and is computed using the head at the end of the time step. This approach is consistent with the fully-implicit transient finite-difference scheme employed by MODFLOW in which inter-cell flow rates are computed using saturated thicknesses and hydraulic gradients derived from heads at the end of the time step. MODPATH computes the vertical velocity component at the water table by dividing the volumetric flux rate across the top of the cell by the porosity. For transient flow, this vertical velocity component is actually the vertical velocity component relative to the velocity of the water table. Consider the case of a falling water table in a situation in which there is no recharge crossing the water table. A particle placed at the water table would move downward at the same speed as the water table and no net flow of water would occur across the water table boundary. At some later time, the particle would still be located at the water table, but its absolute vertical position would be lower. MODPATH accounts for the moving boundary in an approximate way by using the local vertical coordinate of a particle to adjust its absolute vertical coordinate at the beginning of each time step to account for changes in saturated thickness from one time step to the next. Thus, for the example described above, MODPATH would keep the particle at the top of the cell as the water table drops from one time step to the next. In cases where there is a non-zero vertical component of flow at the water table, this approach provides a simple way of approximating the vertical movement of particles relative to the water table that is consistent with the water balances for unconfined cells.

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.

The semi-analytical particle tracking method used in MODPATH is valid only for the simple linear velocity interpolation scheme described in the preceding sections. 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 MODFLOW 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 finite element models.

The accuracy of numerically-generated path lines, and a proper interpretation of what they represent, depends on the extent to which the ground-water system can be realistically represented by a discrete network 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. Time discretization also can be a significant source of error in transient flow simulations.

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 some fraction of the total inflow to the cell eventually flows out of the cell across one or more of the cell's faces. Pathlines 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. This means that 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 illustrates 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 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.

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 the same is true for particle tracking. 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.

Chapter 3

MODPATH User's Guide

Overview of MODPATH

The computer program MODPATH consists of a main program and several subroutines. The primary purpose of the main program is to perform start-up operations such as initializing variables and calling subroutines that open files and allocate space for arrays. Array data is stored in a single, one-dimensional master array that is defined in the main program. The size of the master array can be adjusted simply by changing its dimension in the main program. Once the basic start-up operations are complete, the main program calls subroutine DRIVER, which controls the overall sequence of computations.

MODPATH obtains data from a combination of file and interactive keyboard input. Files that contain basic information about the flow system, such as geometry, boundary conditions, and cell-by-cell budget terms, are referred to as **Flow System Files**. Interactive keyboard input is used primarily to supply information about modeling options that change frequently from run to run. Keyboard input is recorded in a special data file called a **Response File**. Response files generated during one run can be used in place of keyboard input to supply data for subsequent runs. Output from MODPATH consists of a series of files containing information about particle coordinates and time of travel. These files serve as input for MODPATH-PLOT, which displays pathlines and particle locations graphically.

A Quick Guide for Running MODPATH

This section provides a brief outline of the steps involved in making a MODPATH simulation. Detailed descriptions of each of the highlighted topics are provided in later discussions.

Step 1. Run MODFLOW

MODPATH computes velocity components using the cell-by-cell budget information generated by MODFLOW. Before MODPATH can be run, MODFLOW must be run and cell-by-cell output written to a file. MODPATH also requires MODFLOW head output for any layer that can have a water table within it.

Step 2. Assemble MODPATH Data Files

MODPATH always requires the following set of data files:

(1) **Flow System Files** -- MODFLOW and MODPATH data files that contain information about the physical dimensions of the ground-water system, boundary conditions, and hydrologic properties, cell-by-cell budget output, and head.

(2) **Name File** -- A data file containing a list of input and output files required by MODPATH, FORTRAN unit numbers, and file-type descriptions. This information is used by MODPATH to open the files.

(3) **Search Path File** -- If present in the local directory, a file named <mpsearch> is used to specify the full pathname for the MODPATH setup directory. MODPATH's help files normally are located in the setup directory. If <mpsearch>does not exist, help files must be present in the local directory for the on-line help system to function. Some implementations of MODPATH may avoid the need for the search path file by including custom code that permits the search path to be specified in an environmental variable.

Step 3. Run MODPATH

Once the necessary data files have been prepared, MODPATH can be executed using a method appropriate for the specific computer system. On most computer systems, MODPATH is executed by typing the command:

```
mpath3
```

MODPATH responds with the message:

```
 MODPATH Version 3.00 (V3, Release 1, 9-94)
 TO READ INPUT FROM AN EXISTING RESPONSE FILE, ENTER FILE NAME:
 ( <CR> = ENTER DATA INTERACTIVELY )
```

The user can elect to enter responses interactively at the keyboard or to supply the name of a "Response File" containing responses generated from a previous interactive run. If no file name is entered, data is entered interactively in response to prompts, and the results are recorded in a response file named <mpath.rsp>. Response files are described in detail in Chapter 5 -- **Prompt and Response System**.

On some computer systems, it is possible to specify a command-line argument to indicate whether responses are read from a file or from the keyboard. If a command-line argument is provided, MODPATH does not prompt for the name of a response file. To run MODPATH using interactive keyboard input, type the command:

```
mpath3  i
```

The argument "i" indicates that input will be specified interactively at the keyboard. To read data from a response file, type the command:

```
mpath3 filename
```

where *filename* is the name of an existing response file.

The Interactive (or Response File) input provides MODPATH with information about:

1. **Simulation Type** -- steady-state or transient,
2. **Output Mode** options for particle coordinates,
3. **Starting locations** of particles,
4. **Direction of particle tracking computation** -- forward or backward,
5. **Criteria for stopping particles**, and
6. Options for computing and displaying the **volumetric balance check**.

These items are discussed in detail in the sections that follow.

Step 4. Review Output and run MODPATH-PLOT

After the interactive input has been processed, particle paths are computed and several output files containing information about the run and particle coordinate data are produced. The particle coordinate output files generated by MODPATH can be utilized in **MODPATH-PLOT** to generate graphical output.

Definitions of Time Concepts used by MODPATH

The following definitions of time are utilized in MODPATH to track particles in a transient flow field:

Simulation Time--is the value of time associated with the MODFLOW simulation. Simulation time is assigned an arbitrary value at the beginning of a MODFLOW simulation and increases throughout the course of a MODFLOW simulation.

Tracking Time-- is defined in MODPATH relative to a specified reference value of simulation time. Tracking time is defined to be 0 at the specified reference value of simulation time. Tracking time measures the accumulated time during a particle-tracking analysis. The value of tracking time is always positive, regardless of whether particles are tracked forward or backward.

Figure 3-1 illustrates the relation between simulation time and tracking time.

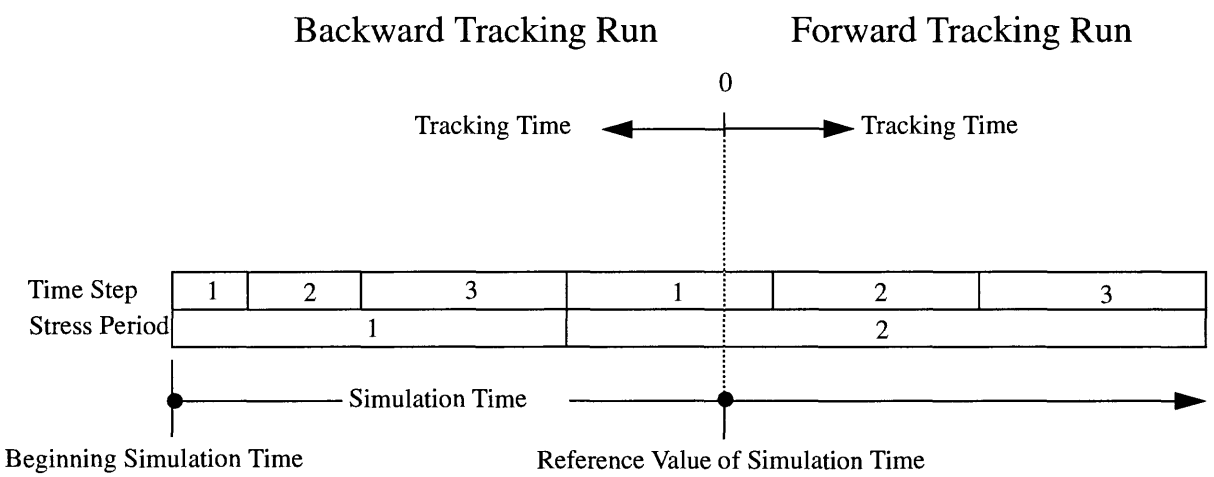


Figure 3-1. Definition of simulation time and tracking time.

For steady state simulations, both the beginning simulation time and the reference value of simulation time are assumed to be 0.

Steady-State or Transient Flow

The option of steady-state or transient flow is designated by the value of NPER specified in the main data file [see Appendix A - Input Files (**Main Data File**)]. For a steady state simulation, set NPER=0. For a transient simulation, set NPER equal to the number of stress periods in the MODFLOW simulation.

If the simulation is transient, MODPATH prompts the user to select a reference time to use when computing particle-tracking time. The reference time can be defined either by (1) specifying a stress period, time step, and relative time within the time step, or by (2) entering the reference time directly. The relative time within the time step is equal to 0 at the beginning of the time step and equal to 1 at the end of the time step. If the reference time is entered directly, the stress period, time step, and relative time within the time step that corresponds to the reference time are computed. For steady state simulations, the reference time always is assumed to equal 0.

For transient flow simulations, a special flow system file called the composite budget file is required. A new composite budget file can be created by MODPATH or an existing file can be used from a previous MODPATH simulation. For more information about the composite budget file, see section **Composite Budget File for Transient Simulations** in this chapter and Appendix A - Input Files(**Composite Budget File**).

Boundary Array and Zone Codes

MODPATH uses a boundary array (IBOUND) similar to that required by MODFLOW. As in MODFLOW, the basic function of IBOUND is to specify the location of constant head cells (IBOUND < 0), no-flow cells (IBOUND=0), and variable head cells (IBOUND>0). In addition, MODPATH uses the absolute numerical values of IBOUND as codes that allow the user to designate blocks of cells as special zones. MODPATH records the zone codes of a particle's initial and final location in the endpoint file. Zone code information can be used in a variety of ways by post-processing programs that use the MODPATH endpoint file. In some situations, MODPATH also uses zone codes to determine whether a particle path computation should be terminated (see section **Criteria for Stopping Particles** in this chapter). The IBOUND array is specified in the MODPATH main data file [Appendix A - Input Files (**Main Data File**)]. Zone codes can be defined by coding IBOUND values directly in the main data file, or they can be assigned "on-the-fly" by specifying values interactively. IBOUND values may be changed interactively for (1) all cells in a layer, (2) a single cell, or (3) all cells in an arbitrary block of grid cells. Zone codes stored in the IBOUND array also are used by MODPATH-PLOT to control a variety of aspects of the graphical output (see section **Zone Codes in MODPATH-PLOT** in chapter 4)

Flow System Files

MODPATH requires the following flow system files:

1) MODPATH's main data file.

2) MODFLOW Stress package data files.

3) Ancillary data files that are referenced by the main data file or any of the stress package data files.

4) Cell-by-cell budget file generated as unformatted output by MODFLOW.

5) Heads generated as output files by MODFLOW.

Main Data File

The main data file is the only flow system file that is not derived from MODFLOW input or output files. It contains information about grid size and geometry, porosity, and stress period data for transient simulations. Array data is input with utility subroutines patterned after those used by MODFLOW. Input instructions for MODPATH's main data file and for the array input utility subroutines are presented in Appendix A - **Input Files**.

Stress Package Data Files

MODPATH requires data files for all of the stress packages used in a simulation analysis. MODPATH reads MODFLOW stress package data files. Input instructions and variable definitions for the stress package data sets are presented in Appendix A - **Input Files**. MODFLOW's stress package data files must be modified to indicate whether individual flow terms are to be treated as distributed internal sources and sinks, or assigned to specific faces of the cell.

In the case of list-oriented stress packages (wells, rivers, drains, streams, and general head boundaries), flow terms either can be treated as internal sources and sinks, or assigned as a directional component of flow to any of the six cell faces. The way in which a flow term is to be treated is designated by entering an additional integer variable, IFACE, at the end of each data record. If IFACE equals 0 or is greater than 6, the flow term is treated as an internal source or sink. If IFACE equals a number 1 through 6, the flow term is assigned to cell face corresponding to a number 1 through 6. The cell faces that correspond to numbers 1 through 6 are shown in Figure 3-2.

If IFACE is less than 0, the source/sink flow term is distributed uniformly across any of the faces 1 through 4 that form boundaries with inactive cells (IBOUND=0). An average velocity across all lateral boundary faces is computed by dividing the total flow rate of the source/sink by the total cross sectional area of the boundary faces and the porosity. The average velocity is then used as the normal component of velocity at each boundary face. If none of the faces are

boundaries with inactive cells, the flow term is treated as an internal source or sink. This approach will never result in flows being assigned to the top or bottom faces of a cell.

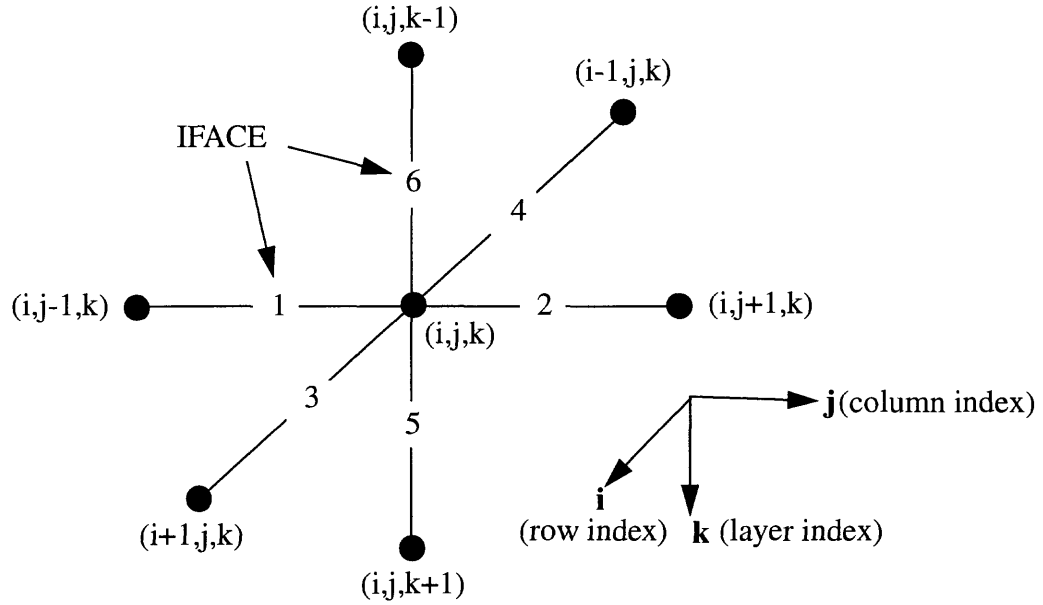


Figure 3-2. Definition of IFACE for a finite-difference cell.

To illustrate the use of IFACE, consider Figure 3-3 which shows a small, one-layer grid with an irregular boundary. The total inflow across the top of the grid in columns 3 and 4 is 200 ft^3/d. A total of 300 ft^3/d enters the system across the upper left part of the grid. Wells are used to account for the boundary fluxes. 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(ft^3/d)	IFACE
1	3	1	50	4
1	2	2	100	1
1	2	2	50	4
1	1	3	100	1
1	1	3	100	4
1	1	4	100	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 the normal velocity components at the specified cell faces.

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

Layer	Row	Column	Q(ft^3/d)	IFACE
1	3	1	50	4
1	2	2	150	-1
1	1	3	100	1
1	1	3	100	4
1	1	4	100	4

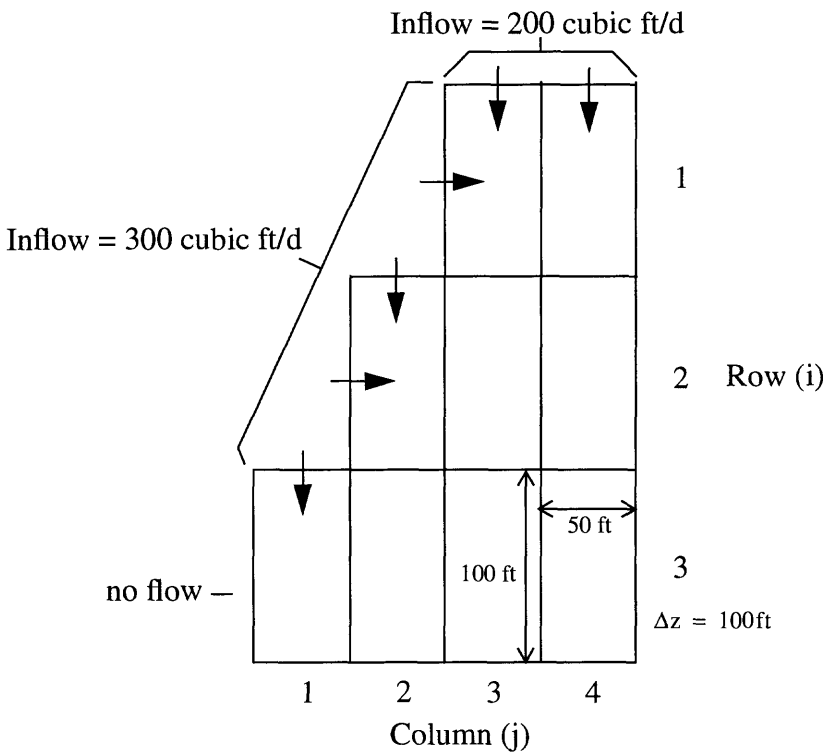


Figure 3-3. Example of how boundary fluxes are assigned to cell faces

In this case, MODPATH determines how to distribute the 150 cubic feet/day among the boundary faces of the cell in layer 1, row 2, column 2. An average velocity is calculated and assigned as the normal velocity component across the boundary faces, 1 and 4.

Often it is convenient for grids with highly irregular boundaries to set IFACE equal to a negative number and let MODPATH 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.

The use of IFACE to designate how river, stream, 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 head-dependent stresses are not specified explicitly in the input data files, but instead are computed as part of the solution to the flow problem and saved as output from MODFLOW. A single flow term is saved for each cell. When more than one data entry is specified for a single cell, the flow term computed and stored by MODFLOW for that stress package represents a composite flow. MODPATH does not redistribute the composite flow among the multiple entries within the cell. Instead, MODPATH distributes the composite flow for that cell according to the IFACE value specified for the first data entry for that cell.

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 A - **Input Files**. MODPATH only reads the first line of the MODFLOW evapotranspiration package file.

Cell-By-Cell Budget File

MODFLOW can write cell-by-cell budget data to binary (unformatted) output files for use by post-processing programs, such as MODPATH. In contrast to previous versions of MODPATH (Pollock, 1989a), version 3 requires that budget data for the block centered flow package and any head-dependent stress packages used in the simulation be recorded in a single, common file. Cell-by-cell budget data for the well and recharge packages need not be included in the budget file because those flow terms are specified explicitly in the stress-package data files. MODPATH will ignore budget data for wells and recharge if present in the budget file. Future versions of MODFLOW may allow the user to write the cell-by-cell budget file using either the standard MODFLOW structure or in a special "compact" structure (Arlen Harbaugh, oral communication, 1994). MODPATH can accept either type of budget file.

Head File

To compute the saturated thickness and vertical coordinates within water-table layers, MODPATH requires that heads for all unconfined and unconfined/confined model layers (LAYCON = 1, 2, or 3) be stored as output from MODFLOW. MODFLOW writes heads in an output file whenever the appropriate flags are specified in MODFLOW's output control data file. Future versions of MODFLOW may allow the user to specify whether head output should be

saved as (1) a binary (unformatted) file, (2) a text file with a header line, or (3) a text file without a header line (Arlen Harbaugh, oral communication, 1994). MODPATH can read heads from either a binary file or a text file with a header line.

MODPATH only uses heads in unconfined or confined/unconfined layers. MODPATH does not require heads for confined model layers. Heads are set equal to zero in layers for which no head data is available.

Composite Budget File for Transient Simulations

For transient flow simulations, MODPATH uses a special binary, direct-access file referred to as a Composite Budget File (CBF). Because data in the CBF can be accessed quickly in any order, transient particle tracking can be implemented efficiently in forward or backward mode just as for steady-state flow systems. The user is given the option of (1) generating a CBF from standard MODFLOW and MODPATH data files, or (2) using an existing CBF. The CBF may be specified in the **name file** as file type **CBF**. If the name file does not contain a file type **CBF**, MODPATH will prompt the user for a file name. A complete description of the contents and structure of the CBF is provided in Appendix A - Input Files (**Composite Budget File**). Composite budget files are not generated for steady-state simulations.

Output Mode

In order to optimize the output for various types of graphical options provided by MODPATH-PLOT, one of three output modes may be selected:

1 = Endpoint Mode
2 = Pathline Mode
3 = Time Series Mode

Endpoint Mode

For endpoint mode, only the particle endpoints (the initial and final locations) are recorded in a file. By default, the endpoint file is named <endpoint> (or <endpoint.bin> if the **BINARY** output option was selected). The endpoint file is generated for all MODPATH runs. Endpoint mode produces the minimum amount of output from MODPATH. It is most useful when the main objective of an analysis is to map recharge locations to discharge locations. A detailed description of the structure and format of the endpoint file is presented in Appendix B - **Particle Coordinate Output Files**.

Pathline Mode

For pathline mode, coordinates along the path of each particle are recorded in a file. By default, the pathline file is named <pathline> (or <pathline.bin> if the **BINARY** output option was selected). 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 (1) a point in time for which a data point was requested by the user, or (2) the end of a MODFLOW time step in a transient simulation. A detailed description of the structure and format of the pathline file is presented in Appendix B - **Particle Coordinate Output Files**.

The user is asked whether particle locations should be computed and recorded at specific values of travel 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) Time points 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 MODFLOW. For example, if the time units in MODFLOW are days, and the MODPATH time step size is specified in years, then the multiplier should be 365.25.

2) Time points can be read from a file.

   When this option is selected, MODPATH checks to see if a file of type **TIME** was

specified in the **name file**. If so, MODPATH reads time data from that file. If not, the user is prompted to enter a file name. The structure of the time data file is described in Appendix A - Input Files (**Time File**).

Time Series Mode

For time series mode, the locations of particles at specified points in time are computed and recorded in a file. By default, the time series file is named <timesers> (or <timesers.bin> if the **BINARY** output option was selected). The locations of all particles are computed for a specified point in time and recorded in the time series file. The procedure is repeated at each point in time for which output is requested. Time series mode produces a series of particle locations in the time series file that are stacked in time. When points in the time series file 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 time series file is presented in Appendix B - **Particle Coordinate Output Files**. For time series mode, MODPATH prompts the user to indicate how the time data will be specified. The options for specifying time points in a time series analysis are the same as described above for pathline mode.

Starting Locations for Particles

Starting locations for particles can be generated by MODPATH for rectangular blocks of cells or can be read directly from a file. In the case of forward tracking analyses, particles also can be assigned a "release time", which allows particles to be released into the flow system over a range of times rather than simply as a single, instantaneous release.

Particle Generation

MODPATH provides a particle generation 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:

(1) distribute particles within a cell.

(2) distribute particles on one or more of the six cell faces.

If the choice is made to distribute particles within cells, the user receives the following prompt:

```
ENTER A 3-D ARRAY OF PARTICLES:  NX  NY  NZ
```

The cell is evenly subdivided in x, y, and z directions according to the values of NX, NY, and NZ to produce (NX x NY x NZ) subvolumes. Particles are placed at the center of each subvolume.

If particles are to be distributed on individual cell faces, MODPATH asks the user to specify which faces will have particles. MODPATH then prompts the user to enter a 2-D array for each of those faces. Using face 1 as an example, the program would issue the prompt:

```
ENTER A 2-D ARRAY OF PARTICLES FOR FACE 1:  NY  NZ
```

Face 1 then is divided in the y and z directions into NY and NZ subdivisions, respectively, to form (NY x NZ) subareas. Particles are placed on face 1 at the center of each subarea. This process is repeated for all of the faces that have particles. Examples of particle placement are shown in Figure 3-4.

The particle generation option can also be used to drape a two-dimensional array of particles over a water-table surface that passes through several layers. To drape particles, select the option to specify a subregion of the grid over which particles will be placed. When MODPATH prompts for the bounding grid cell coordinates that define the subregion, set JMIN,

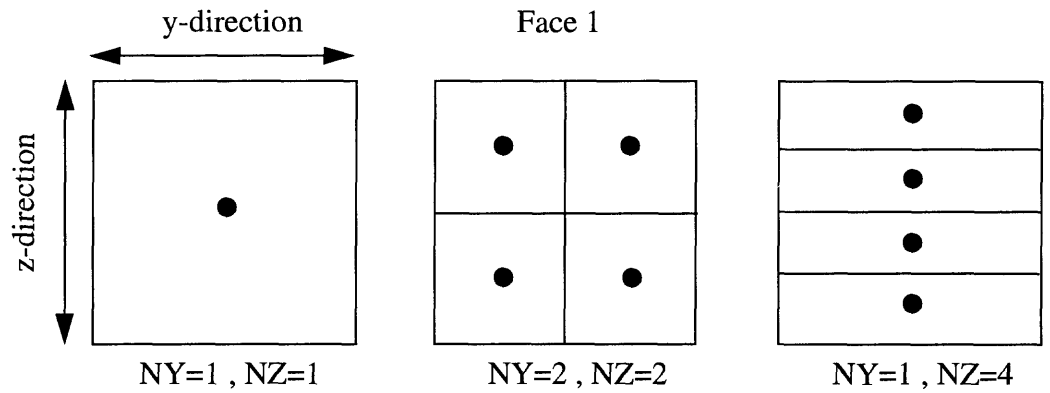


Figure 3-4. Examples of 2-dimensional arrays of particles for face 1.

JMAX, IMIN, and IMAX equal to the values that define the areal extent of the subregion and KMIN and KMAX equal to 0. When KMIN equals zero, particles are placed in the first active layer for each areal cell location within the subregion. Particles can be draped over the water table by placing them on face 6 of all the cells in the subregion. This procedure is analogous to the way in which recharge is assigned to cells when recharge package option 3 is selected.

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. A detailed description of the structure of the starting locations file is presented in Appendix A - Input Files (**Starting Locations File**).

Specifying Starting Locations in a File

Particle coordinates also can be read from a file. If the option to read locations from a file is selected, MODPATH checks to see if a file of type **LOCATIONS** was opened in the **name file**. If so, starting locations are read from that file. If not, MODPATH prompts the user to enter a file name. The file consists of one line of data for each particle. A detailed description of the structure of the starting locations data file and the options for specifying coordinates is presented in Appendix A - Input Files (**Starting Locations File**).

Specifying a Release Time for Particles

In many cases, especially those involving transient flow, it is useful to study the movement of particles released into the flow system over a range of time, rather than as a single, instantaneous release. For forward tracking analyses, MODPATH allows a "release time" to be specified for each particle. If particles are generated by MODPATH, the option is provided to specify an interval of time (tracking time) and a rate of release. MODPATH then generates a distribution of particles that include multiple releases at a regular time interval over the specified

period. Any value of release time can be specified for a particle in a starting locations file.

Particle release time is measured in terms of tracking time. That is, particles released instantaneously at the beginning of a MODPATH analysis have a release time equal to 0. When a particle has a release time greater than 0, the travel time for the particle is no longer equal to the tracking time. Instead,

travel time = tracking time - release time.

Multiple release times are not allowed for backward tracking analyses. Particle release times for backward tracking analyses are automatically set equal to 0.

Reading Starting Locations from an Endpoint File

An endpoint file from a previous MODPATH run may be specified as the starting locations file. This type of starting locations file is specified in exactly the same way as a standard starting locations file. MODPATH automatically detects whether the specified starting locations file is an endpoint file by checking for the endpoint file header record. Either standard or compact text endpoint files can be specified. Binary endpoint files and endpoint files generated by previous versions of MODPATH cannot serve as starting locations files. If MODPATH detects that the starting locations file is an endpoint file, the coordinates of the particle's final location are read and used as starting locations. The release time is set equal to zero for all particles.

By default, MODPATH only generates starting locations for particles that are listed as "active" in the endpoint file. MODPATH can be explicitly instructed to generate starting locations for particles that (1) are active, (2) stopped in an automatic termination zone, or (3) terminated normally by adding the keywords ACTIVE, AUTOMATIC, and TERMINATED to the endpoint file header. For example, the following endpoint file header instructs MODPATH to generate starting locations from an endpoint file for those particles that are active and those that stopped in an automatic termination zone:

```
@ [ MODPATH Version 3.00 (V3, Release 1, 9-94) (TREF= 0.000000E+00 ) ]  active  automatic
```

Criteria for Stopping Particles

Particles are stopped whenever they reach points of termination or whenever the cumulative tracking time equals a maximum value specified by the user.

A particle can terminate when it:

1. reaches an external boundary or internal sink/source cell that captures the particle,
2. enters a cell with a special zone code that is designated as a stopping point,
3. is stranded in a dry cell, or
4. is still active at a user-specified stopping time.

Particle path lines are always 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 cells with weak sinks, an arbitrary decision must be made by the user about whether to stop particles. MODPATH provides three options:

1. particles pass through cells with weak sinks.
2. particles are stopped when they enter cells with weak 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 a specially designated zone code (regardless of whether the cell is a potential discharge point). MODPATH asks if particles should be stopped when they enter a specific zone. If the response is yes, the user is asked to enter the number of the zone. Only one zone number can be specified. If an automatic termination zone is specified, the user is given the chance to modify the IBOUND array interactively. Zone codes are defined originally by the IBOUND array when it is read from the main data file.

In transient simulations, particles also can be stranded in cells that go dry. Stranded particles are considered to be terminated and are labeled as "stranded" in the endpoint file.

For both steady-state and transient systems, an option is provided to specify a maximum value of tracking time at which all particle tracking computations will be stopped. Particles that are still in the flow system when the maximum tracking time is attained are flagged as "active" particles in the endpoint file.

Direction of Tracking Computation

MODPATH provides the option of tracking particles forward in the direction of ground-water flow, or backward 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.

Volumetric Balance Check

Cell-by-Cell Budget Computation

An option is provided to check volumetric balances on a cell-by-cell basis for all active cells in the grid (except constant head cells). If the user chooses to compute volumetric balances, a prompt is issued requesting that an error tolerance be specified. The value should be entered as a percent. Volumetric balances for all cells are computed and a message is printed to the screen and to the file <summary.pth> indicating the number of exceedences. Cell-by-cell volumetric balance checking is very useful as a means of assuring that all of the appropriate stress packages have been included. An effort should be made to investigate those cells that display a large volumetric budget error and identify the cause. Even carefully prepared models with very small global volumetric budget errors commonly contain a few cells in regions with very small ground-water velocities that yield large volumetric budget errors. These localized areas of large volumetric budget error in low-flow regions rarely cause accuracy problems when pathlines are computed by MODPATH. However, occasionally accuracy problems do occur. In those cases, the problem usually can be resolved by rerunning MODFLOW with a smaller closure criterion.

Cell-By-Cell Data Summary for Individual Cells

Cell-by-cell data summaries are useful in the early stage of a ground-water flow simulation study to help find errors in data. 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 volumetric budget 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.

Specifying Input Data Files for MODPATH

A special data file, called the "Name File", is utilized by MODPATH to provide the information it needs to open and manage input data files. MODPATH prompts the user to enter the name of the name file at the beginning of each run. The Name File is described in detail in Appendix A - Input Files (**Name File**).

The following data files must be specified in the Name File if they are required for a MODPATH analysis:

- Main Data File (type **MAIN**)
- MODFLOW budget file (type **BUDGET**)
- MODFLOW head file (type **HEAD(BINARY)** or **HEAD**)
- MODFLOW stress package data files (types **RCH, RIV, DRN, GHB, EVT, WEL, STR**)
- Ancillary data files referenced as external files by an array control record (type **DATA**) Ancillary data files also may be specified directly in the Main Data File with the OPEN/CLOSE option for the array input [see Appendix A - Input Files (**Array Input**)].

The following files may be specified in the name file or as part of the interactive input during a MODPATH run:

- Composite Budget File (type **CBF**)
- Time Data File (type **TIME**)
- Starting Locations File (type **LOCATIONS**)

If any of these three files is required for a MODPATH run, MODPATH checks the Name File to see if a file of the appropriate type has been specified. If so, MODPATH reads the data from that file. If not, MODPATH prompts the user to enter a file name.

MODPATH also allows the endpoint, pathline, time series, and summary files to be specified in the Name File:

- Endpoint File (type **ENDPOINT**)
- Pathline File (type **PATHLINE**)
- Time Series File (type **TIME-SERIES**)
- Summary File (type **LIST**)

If any of these files appear in the Name File, MODPATH uses the file name and unit number provided by the user. If the summary file is specified in the Name File, it must be specified as the first line in the file. When these files are not specified in the Name file, MODPATH uses default file names:

- Endpoint File -- "endpoint" (or "endpoint.bin")
- Pathline File -- "pathline" (or "pathline.bin")
- Time Series File -- "timesers" (or "timesers.bin")
- Summary File -- "summary.pth"

The ".bin" file names are used if the option to generate **BINARY** particle coordinate files was specified in the Main Data File.

MODPATH Output Files

Summary File

A summary of input data and the MODPATH simulation is printed in a file named <summary.pth>. Each major data item is summarized. Arrays can be printed in several formats, or printing can be suppressed (Appendix A - **Input Files**). In addition to summarizing input data, the <summary.pth> file also provides additional information about the MODPATH simulation, including error messages.

Particle Coordinate Files

MODPATH produces the following output files that contain coordinate and travel time data. The combination of files generated by MODPATH is a function of the output mode specified by the user:

Mode	Files
ENDPOINT	endpoint (or, endpoint.bin)
PATHLINE	endpoint (or, endpoint.bin) pathline (or, pathline.bin)
TIME SERIES	endpoint (or, endpoint.bin) timesers (or, timesers.bin)

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. The binary files ending in the ".bin" suffix are produced when the user chooses the keyword, BINARY, in the **main data file**. Detailed descriptions of the structure, contents, and format of the endpoint, pathline, and time series files are presented in Appendix B, **Particle Coordinate Data Files.**

Chapter 4

MODPATH-PLOT User's Guide

An Overview of MODPATH-PLOT

MODPATH-PLOT produces graphical output based on results generated by MODPATH. Input to MODPATH-PLOT consists of (1) particle data stored in endpoint, pathline, and time series files, (2) flow system files, (3) a user-supplied drawing commands file, and (4) interactive keyboard input to select options relating to plot type and appearance.

MODPATH-PLOT requires the following flow system files:

1. **Main Data File**
2. **Head File** (output from MODFLOW)
3. **Well Package File**
4. **River Package File**
5. **Drain Package File**
6. **General Head Boundary Package File**
7. **Stream Package File**

MODPATH-PLOT does not require the MODFLOW binary budget file because no flow calculations are made by MODPATH-PLOT. However, the composite budget file is required for cross sectional plots of transient pathline, endpoint, or time series data. The main data file and the head file are required to draw the grid. The head file is also required for drawing head contours in map view plots. The stress package data files are used only to identify the location of wells, rivers, drains, and general head boundaries for plotting purposes. For convenience, the flow system files used for MODPATH-PLOT are generally the same files used to run MODPATH. However, they can be different for the purpose of customizing a plot. For example, the well and river package data files can be modified so that only some of the wells and river cells used in the MODPATH simulation appear on the plot.

MODPATH-PLOT provides the capability of displaying additional, miscellaneous graphics and text on map-view plots through the use of drawing instructions supplied in a drawing commands file [see Appendix A - Input Files (**Drawing Commands File**)]. As an aid in constructing the Drawing Commands File, MODPATH-PLOT provides the option to interactively obtain the x-y coordinates of points from the screen using a graphics input device such as a mouse (see section **Interactive Graphics** in this chapter).

As with MODPATH, interactive keyboard input is recorded in a response file. The file is automatically named <mplot.rsp>. Response files can be used with MODPATH-PLOT to input data from the file instead of typing it at the keyboard. The format of MODPATH-PLOT and MODPATH response files are identical (see Chapter 5 -- **Prompt and Response System**).

MODPATH-PLOT uses the Graphical Kernel System (GKS) subroutine library. GKS is an

industry-standard in graphics and improves the portability of MODPATH-PLOT between different computers and computer operating systems. MODPATH-PLOT is structured to minimize and localize the changes needed to move the program from one GKS installation to another. A detailed description of portability issues related to MODPATH-PLOT is presented in the section, Appendix D - Installation (**Programming and Portability Issues**).

A Quick Guide to Running MODPATH-PLOT

This section provides a brief outline of the steps involved in running MODPATH-PLOT. Detailed descriptions of each highlighted topic are provided in later discussions.

Step 1. Run MODPATH

MODPATH-PLOT reads the numerical coordinate output files from MODPATH to generate several styles of plots. If MODPATH-PLOT is used solely draw the grid or to produce contour plots, MODPATH need not be run first. For a review of how to run MODPATH, see Chapter 3 -- **MODPATH User's Guide**.

Step 2. Assemble MODPATH-PLOT Data Files

MODPATH-PLOT uses most of the same input files as MODPATH. These include (1) most of the **Flow System Files**, (2) the **Name File**, and (3) the **Search Path File** (<mpsearch>).

MODPATH-PLOT also uses the following files in addition to those mentioned above:

1. **Settings File**: an optional file used by MODPATH-PLOT that allows the user to redefine parameters that control characteristics of the graphical output.
2. **Device File**: a file required by MODPATH-PLOT that contains information about the graphics devices displayed in the interactive menu.

The Device File and Settings File normally are located in the MODPATH setup directory. For a detailed discussion of the Settings File, see section **Customizing MODPATH-PLOT** in this chapter. For a detailed discussion of the Device File, see Appendix D - Installation (**Device File**).

Step 3. Run MODPATH-PLOT

Once the necessary data files have been constructed, MODPATH-PLOT can be executed using a method appropriate for the specific computer system. On most systems, MODPATH-PLOT is executed by typing the command:

```
mplot3
```

MODPATH-PLOT responds with the message:

```
MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94)
TO READ INPUT FROM AN EXISTING RESPONSE FILE, ENTER FILE NAME:
( <CR> = ENTER DATA INTERACTIVELY )
```

The user can elect to enter responses interactively or to supply the name of a "Response File" containing responses generated from a previous interactive run. If no file name is entered, data is

entered interactively in response to prompts, and the results are recorded in a response file named <mplot.rsp>. Response files provide the user with a great deal of flexibility to customize the amount and style of interactive input required during a MODPATH-PLOT run. Response files are described in detail in Chapter 5. **Prompt and Response System**.

On some computer systems, it is possible to specify a command-line argument to indicate whether responses are read from a file or from the keyboard. If a command-line argument is provided, MODPATH-PLOT does not prompt for the name of a response file. To run MODPATH-PLOT using interactive keyboard input, type the command:

```
mplot3  i
```

The argument "i" indicates that input will be specified interactively at the keyboard. To read data from a response file, type the command:

```
mplot3 filename
```

where *filename* is the name of an existing response file.

The interactive (or response file) input provides MODPATH-PLOT with information about:

1. which **Settings File** to use for the run,
2. **Selecting a Graphics Output Device**,
3. **Simulation Type** -- steady state or transient
4. **Plot Types**,
5. **Zone Codes**,
6. **Grid Options**,
7. **Contouring**,
9. whether a **Drawing Commands File** should be processed, and
8. **Plot Scaling**.

After the interactive input is processed, MODPATH-PLOT produces a plot on the graphic output device that was specified.

Step 4. Review MODPATH-PLOT Output Files

The primary output from MODPATH-PLOT is the graphic plot. However, MODPATH-PLOT does produce an output file that is named <summary.plt> by default. The summary file provides a record of the input data processed by the program as well as information about the status of the MODPATH-PLOT session. It is good practice to review this file to be sure that the input data is being processed properly.

Selecting a Graphics Output Device

MODPATH-PLOT displays a list of available graphics devices, similar to the example shown below, and prompts the user to select one.

```
ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
     1 = Monitor
     2 = Printer
```

The range of graphics devices supported by MODPATH-PLOT depends on the capabilities of the GKS implementation, which are specified as part of the MODPATH-PLOT installation process in the file, <gks.dat> [See Appendix D - Installation (**GKS File**)]. The graphics devices that appear in MODPATH-PLOT's list of devices is controlled by the Device File, <device.dat> [see Appendix D - Installation (**Device File**)]. In most cases, users will not need to change the GKS setup parameters specified in <gks.dat> once MODPATH-PLOT has been installed. However, users may want to modify <device.dat> to customize the graphics devices presented in the menu. The files <gks.dat> and <device.dat> are normally located in the MODPATH setup directory.

Steady-State or Transient Flow

The use of steady-state or transient flow is designated by the value of NPER specified in the **main data file**. For a steady-state simulation, set NPER=0. For a transient simulation, set NPER equal to the number of stress periods in the MODFLOW simulation.

If the simulation is transient, the user is then prompted to enter a stress period and time step value for which heads will be used to compute the active grid and to draw head and drawdown contours if the contour option is selected.

Heads are read for unconfined and unconfined/confined layers (LAYCON = 1, 2, or 3) for the specified time step from the MODFLOW head output file. MODPATH-PLOT computes the boundaries of the active grid that are drawn on the plot based on heads for the specified time step. Any contours of head or drawdown generated during the run also are restricted to the boundaries for the specified time step.

For true cross section plots of endpoints, pathlines, or time series points, the **composite budget file** also is required for transient simulations. For a discussion of "true cross sections", see section **Cross-Section Grids** in this chapter).

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, the user is given the option to change the zone code of (1) all cells in a layer, (2) a single cell, or (3) all cells in an arbitrary 3-D subregion of the grid. Additional discussion of zone codes and their use in MODPATH can be found in the section **Boundary Array and Zone Codes** in chapter 3.

Plot Types

MODPATH-PLOT provides the option of generating several types of plots:

1 = Pathline plot
2 = Starting locations from a forward tracking analysis
3 = Final locations from a forward tracking analysis
4 = Final locations from a backward tracking analysis
5 = Time series plot
6 = Plot of the grid only
7 = Contour plot only (available only for map view)

Pathline

Plot type 1 draws pathlines using data from a pathline file generated by MODPATH. Pathlines may be plotted in map view or as cross sections along a column or row. An option is provided to (1) draw all pathlines by projection, or (2) draw only those portions of the pathlines that fall within a specified layer, row, or column slice. For complex three-dimensional flow fields, the projection method can lead to 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 a single plot cannot produce clear results for complex 3-D paths. Sometimes it is possible to illustrate 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.

If particle locations were computed at specific points in time during the MODPATH simulation, the user is given the option of plotting those points along the pathlines.

Pathline color can be specified explicitly by the user. The user also can specify that pathline colors be determined by the **zone code** of the discharge cell for the particle. For a forward-tracked

pathline, the color is determined by the final location of the particle. For a backward-tracked pathline, the color is determined by the initial location of the particle. Color options are specified as part of the interactive input (see section **Specifying a Color Cycling Sequence** in this chapter).

When pathlines are drawn in map view, the pathlines are recorded in a **Drawing Commands File** named <paths.dcf>. If the file <paths.dcf> is not renamed, it will be overwritten the next time a map-view pathline plot is generated. The drawing commands file allows the pathlines to be redisplayed in subsequent plots. This capability provides the opportunity to display multiple sets of pathlines, or to overlay pathlines on endpoint and time series plots.

Endpoint

Plot types 2 through 4 are called endpoint plots because they 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. This type of plot can be used to produce a map of source areas 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 zone code of 1 (IBOUND=1) are not plotted.

Plot type 3 displays data points marking the final locations from a forward tracking analysis. This type of plot is useful for showing the distribution of recharge entering a model layer or a designated hydrogeologic zone. The color of a point is determined by the zone code of a 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 data points marking the final locations from a backward tracking analysis. If particles are placed around a discharge point, this type of plot maps source areas for the discharge point. 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.

Time Series

Plot type 5 displays a snapshot of particle locations at specific values of tracking time. Time series plots require a time series file generated by MODPATH. Time series plots may be drawn in map view or as cross sections along a row or column. An option is provided to plot all points by projection or to restrict the plot to points that fall within a specified layer, row, or column

(depending on the orientation of the plot). Points may be drawn in a single color specified by the user, or colors may be allowed to cycle according to the time step number, so that multiple time steps can be distinguished. Color options are specified as part of the interactive input.

Grid Only

This option allows the user to plot the finite-difference grid without any particle tracking or contours. It eliminates the need to provide information about particle tracking options and data or about contouring options.

Contours Only

This type of plot allows the user to plot contours without any particle tracking data. This option allows MODPATH-PLOT to be utilized as a contouring program without the need to run MODPATH first to generate particle tracking output. Contour plots are only available for the map view orientation. The user may select the contour color by setting the color in the MODPATH-PLOT **settings file**.

Grid Options

Drawing the Grid

Three options are provided for drawing the grid boundaries:

1. draw active grid boundary only
2. draw full grid boundary only
3. draw active grid and full grid boundaries

Option 1 produces a grid that shows only the active cells in the model layer or cross section represented by the grid. Option 2 is useful for situations where the user wants to display the entire grid without distinguishing the boundary between active and inactive cells. Often it is desirable to see the boundaries of both the active grid and the full grid. Option 3 provides that capability.

An option is provided to draw the interior grid lines. If the choice was made to draw the boundary of the active grid (options 1 and 3 described above), interior grid lines are drawn only within the active grid. If the full grid or sub-grid is drawn (option 2), interior grid lines are displayed for all cells.

When both the active grid and the full grid boundaries are drawn (option 3), an option is provided to fill the inactive cells of the grid with a solid fill or with a hatch shade pattern. The hatch shade pattern style may be specified in the **settings file**. For cross-sectional plots, an option also is provided to shade confining layers in a quasi-three-dimensional model with solid fill or a hatch pattern. The hatch pattern used to fill quasi-three-dimensional layers may be specified in the **settings file**.

Colors for drawing specific parts of the grid, such as the grid rectangle, the boundaries of the active grid, interior grid lines, and for filling shaded areas can be specified. Setting colors for grid items is discussed in section **Color Options** of this chapter.

Displaying Grid Unit Data

It often is useful to outline or shade portions of the grid to illustrate hydrogeologic boundaries or other important features of the system. MODPATH-PLOT allows this to be done by using a grid unit array, which consists of an integer value, the grid unit number, for each model cell. The grid unit array is similar to the zone code values that are stored in the IBOUND array. Each grid unit number defines a block of cells that can be outlined or shaded with a distinct color or pattern. The option is provided to (1) ignore grid unit data, (2) outline zones based on the grid unit data, (3) shade the grid unit zones with solid fill, or (4) shade the grid unit zones with hatch fill. The color and hatch fill style corresponding to specific grid unit numbers may be specified in the **settings file**. The grid unit array may be specified directly as an array of integer grid unit numbers or indirectly from an array of real-number data (such as hydraulic conductivity) that is then converted to an array of integer grid unit numbers using interval ranges specified by the user. Data for the grid unit array is defined by the grid unit array file, which is described in detail in Appendix A - Input Files (**Grid Unit Array File**). The grid unit array file must be specified in the name file. If it is omitted from the name file, no grid unit array data will be processed.

Cross Section Grids

Cross section plots may be generated either as true cross sections or normalized cross sections. True cross section plots display the actual shape of the model layers, which allows the shape of variable thickness and dipping layers to be viewed. For a normalized cross section, the average thickness for each model layer is computed and used to generate a rectangular cross section with layers of constant thickness. Previous versions of MODPATH-PLOT generated only normalized cross sections. The default style of cross section in the current version of MODPATH-PLOT is a true cross section. The choice to draw either true or normalized cross sections is provided as an option in the interactive input.

True cross sections are plotted using the top and bottom elevation of model layers. Linear interpolation is used between the nodal values to draw the top and bottom surfaces of the model layers. If a quasi-3d confining layer pinches out between two adjacent cells, the point of pinch out is forced to be at the boundary between the two cells. To draw a land-surface profile for the top of the cross section, specify a LAYCON value of 3 for layer 1 and provide a top elevation array for layer 1 that corresponds to land surface. If layer 1 is a water table layer (LAYCON=1), MODPATH-PLOT uses the head in layer 1 as the top of layer 1. For cells in layer 1 that are dry, the top elevation is set equal to the bottom elevation. For inactive cells in layer 1 that are surrounded by other inactive or dry cells, the top elevation is also set equal to the bottom elevation. For inactive cells that are adjacent to active cells within the row or column of the cross section, the top elevation is set equal to the head in the adjacent active cell. By default, MODPATH-PLOT draws the water-table profile as a solid line using the line color specified for

contour lines (see section **Assigning Color to Specific Grid Items** in this chapter). Options provided in the **main data file** allow the line style of the water-table profile to be changed and for the water table profile to be omitted from the plot. Pathline, endpoint, and time series data are plotted on true grids by linearly interpolating the local vertical coordinate of each particle location onto the true cross section grid. In order to draw true grid cross sections that appear realistic, it is essential that reasonable top and bottom elevations be assigned to inactive cells as well as to the active cells in the model.

Users can choose the direction for drawing row or column cross sections. Cross sections drawn with the "regular" option are from left to right in the +x or +y direction. Cross sections drawn with the "reversed" option are from left to right in the -x or -y direction. This option is selected as part of the interactive input.

Symbols for Stress Package Features

The location of cells containing wells, drains, and general-head boundaries can be displayed on map and cross section plots. These features are displayed only when IFACE equals 0. In map view, these features are displayed with small rounded rectangles. In cross section, the outline of the grid cell containing the feature is drawn.

Rivers and streams are displayed in map view as rectangular boxes that outline grid cells containing river or stream reaches. Rivers and streams are not displayed in cross section plots.

The colors of each specific stress package symbol can be selected by specifying the appropriate grid item colors in the settings file. More information is provided in the sections **Assigning Colors to Specific Grid Items** and **Settings File** (Item Colors) in this chapter. The IFACE value is specified as an extra entry for each stress in the MODFLOW stress package files. Instructions for specifying IFACE are provide in the instructions for each stress package file (see Appendix A - **Input Files**).

Color Options

MODPATH-PLOT supports a maximum number of 256 colors. Colors are designated by numbers 0 through 255. Color 0 is the background color. Color 1 is the primary foreground color. These two colors are usually black and white. For output devices such as display monitors, the background color is black and the primary foreground color is white. For hard-copy devices such as printers and plotters, white is the background color and black is the primary foreground color. These two colors are device dependent. Colors 2 through 255 are secondary foreground colors. The maximum number of colors available for a specific graphic output device depends on the capabilities of the device. The number of colors available for each device must be specified in the **GKS file**.

Defining Colors with a Color Table

Colors are defined by three numbers describing the intensities of red (R), green (G), and blue (B) in the color mix. Color intensity is represented by a real number in the range 0 to 1 (full intensity). MODPATH-PLOT sets the following default values for colors 2 through 15:

Color	R	G	B	Description
2	1.00	0.00	0.00	red
3	0.00	1.00	0.00	green
4	0.00	0.00	1.00	blue
5	0.50	0.10	0.67	violet
6	1.00	1.00	0.00	yellow
7	1.00	0.60	0.00	orange
8	1.00	0.00	1.00	magenta
9	0.60	0.30	0.10	brown
10	0.00	1.00	1.00	cyan
11	0.40	0.00	0.00	dark red
12	0.00	0.40	0.00	dark green
13	0.00	0.00	0.40	dark blue
14	0.60	0.60	0.60	light gray
15	0.30	0.30	0.30	dark gray

Colors 16-255 are all set to white (1.00, 1.00, 1.00). Although MODPATH-PLOT allows all colors to be redefined by the user, some GKS implementations do not allow colors to be redefined for certain devices. Color table entries can be redefined in the MODPATH-PLOT **Settings File**. If a specified color number is out of range of the number of colors supported by the selected output device, color number 1 is substituted for the specified color number.

Assigning Colors to Specific Grid Items

The colors used to draw specific items also can be selected by the user through the use of setup files (see section **Settings File** in this chapter). MODPATH-PLOT uses the following default colors for any item not redefined in a setup file:

Item	Color	Description
1	1	boundary of active grid
2	1	interior grid lines
3	1	grid or sub-grid rectangle
4	1	fill for inactive cells
5	0	fill for quasi-three-dimensional layers
6	1	outline quasi-three-dimensional layers
7	1	plot title and scale bar annotation
8	2	wells
9	2	drains
10	2	general-head boundaries
11	4	rivers and streams
12	1	outline color for grid units if shading is solid
13	1	contour line color

The default color scheme corresponds to the color scheme used by previous versions of MODPATH-PLOT.

Specifying a Color Cycling Sequence

Many of the plotting options in MODPATH-PLOT use the concept of color cycling in which the zone codes of initial or final cells, or the time step number, are used to determine the color of the lines and points. The number and order of colors in the cycling sequence can be defined in the MODPATH-PLOT Settings File (see section **Settings File** in this chapter). Unless redefined in the setup file, MODPATH-PLOT cycles through 4 colors in the order: 1, 2, 3, 4. As an example, the setup file could be used to define a new cycling sequence that consisted of 6 colors in the following sequence: 10, 2, 3, 5, 7, and 9. For this cycling sequence, a key zone code of 1 would cause a line or point to be plotted in color 10. A key zone code of 6 would cause lines or points to be drawn with color 9.

Color Menu

MODPATH-PLOT displays a menu of colors from which the user can choose a color number to use when plotting pathlines and particle data points. A maximum of 15 colors can be displayed on the menu even though some graphic devices may support more than 15 colors. The number of colors displayed on the menu can be specified in a **Settings File**.

Contouring

MODPATH-PLOT incorporates a contouring package that is adapted directly from the contouring program developed and described by Harbaugh (1990). The following options are provided for contouring gridded data in map view:

0 = No contouring
1 = Head in a layer
2 = Drawdown in a layer
3 = Other gridded data
4 = Head difference between two layers

Head and Drawdown

Head and drawdown output from MODFLOW must be saved in a file in order to use contouring options 1 or 2. MODPATH and MODPATH-PLOT require that head and drawdown be saved in separate files. For transient simulations, head and drawdown are contoured for the stress period and time step specified by the user. Future versions of MODFLOW may allow head and drawdown to be saved in (1) a binary(unformatted) file, (2) a standard text file without header records, or (3) a standard text file with header records. MODPATH-PLOT can accept either binary files or text files with header records. The head file must be labeled in the Name File as type **HEAD(BINARY)** or type **HEAD**, depending on whether heads are saved using binary or text style. The drawdown file must be labeled either as type **DRAWDOWN(BINARY)** or **DRAWDOWN**.

Other Gridded Data

Option 3 allows miscellaneous gridded model data, such as hydraulic conductivity, to be contoured. The data must be provided in a standard ASCII text file and must be gridded at the same scale as the areal grid of the model (that is, NROW x NCOL). When option 3 is selected, the user is prompted to enter the line number of the first data record and a FORTRAN format code for reading the data. If a blank is entered for the format code, the data is read by row using format-free style. Format-free style requires that data items be separated by a comma or at least one blank space.

This option allows the user to contour any data that can be provided in gridded form. In most cases, the data is associated with a model layer. Consequently, the user is prompted to enter a layer number. The IBOUND values for that layer then are used to blank contours in areas of inactive or dry cells. In some cases, it may not be desirable to blank contours according to the inactive cell pattern in a specific layer. In those situations, contour blanking can be turned off by entering "0" when prompted for the layer number. Contours always are blanked in cells with a

data value equal to HNOFLO. Customized contour blanking can be achieved by entering "0" as the layer number and selectively coding HNOFLO values in the contour-data array.

The contour-data file either can be specified in the Name File or interactively at runtime. If it is placed in the Name File, it must be labeled as file type **CONTOUR-DATA**. If option 3 is selected and the contour-data file is not present in the Name File, MODPATH-PLOT prompts the user to enter a file name.

Head Difference Between Two Layers

Option 4 provides a means of contouring head differences between any two model layers. If this option is selected, the user is prompted to enter two layer numbers. Heads in the second layer specified are subtracted from heads in the first layer specified. If either or both of the layers are inactive or dry at a specific i-j cell location, the head difference for that cell location is set equal to HNOFLO.

Selecting Contour Levels and Labeling Contours

MODPATH-PLOT prompts the user to select one of two options for generating contours:

1. Generate regularly-spaced contours automatically
2. Read contour levels directly from a file.

If the user selects the option to generate contours automatically, MODPATH-PLOT displays the prompt:

```
ENTER: REFERENCE CONTOUR, CONTOUR INTERVAL, LABELING INTERVAL
```

Specifying a reference contour allows the user to control which contours are displayed and labeled. The contour interval and labeling interval are used in conjunction with the reference contour to control the plotting and labelling of regularly spaced contours. As an example, a reference contour value equal to 150, a contour interval of 2, and a labelling interval of 10, would produce the following contours:

... 140 142 144 146 148 150 152 154 156 158 160 ...

The labeled contours would be:

... 140 150 160 ...

When the reference contour falls within the range of contoured values, it always will be plotted. If contours are to be labeled, the reference contour always will be a labeled contour. The reference contour does not need to fall within the range of contoured data. MODPATH-PLOT prompts the user to specify whether contour labels should be displayed or omitted. Additional control over the plotting of contours is provided by allowing the user to specify the range of contours to plot when

using a constant contour interval.

If contour levels are read directly from a file, MODPATH-PLOT will check to see if a file of type **CONTOUR-LEVEL** was specified in the name file. If a file is specified in the name file, contour levels will be read from that file. If a file is not specified in the name file, MODPATH-PLOT will prompt the user to enter the name of the file containing the contour levels. A contour-level data file must be used to draw irregularly-space contours. If contours are generated from a contour-level data file, all contours will be labeled when the contour-labeling option is selected. The **Contour Level File** is described in Appendix A - Input Files.

Contour Input and Output Using a Drawing Commands File

When contouring is selected as an option, contours are plotted and the drawing commands used to generate the contour lines are saved in a file named <contours.dcf>. The file <contours.dcf> is a **Drawing Commands File** that can then be read in by MODPATH-PLOT during future runs to redraw the contour lines. Multiple sets of contours can be superimposed by using Drawing Commands Files in subsequent MODPATH-PLOT runs. The name of <contours.dcf> must be changed in order to avoid being overwritten by a subsequent MODPATH-PLOT run.

Plot Scaling

Plots may be scaled automatically or by entering a specific map scale. The scaling option is selected as part of the interactive input. Automatic scaling is possible on all devices. Exact scaling with a map scale is not possible on all output devices. In MODPATH-PLOT, output to display monitors and metafiles is always scaled automatically regardless of the scaling option that was selected. In the case of hard-copy devices, exact scaling is allowed whenever the GKS system has information about the size of the output display surface in real coordinates (meters). Otherwise, exact scaling is not permitted even if that option was selected. The specific output devices that can be scaled exactly depend on each GKS installation. If the specified map scale results in a plot that is too large for the display surface, MODPATH-PLOT automatically reduces the size of the plot so that it will fit on the display. If the automatic scaling option is selected, the user may specify a scaling factor between 0 and 1 to set the size of the plot. Specifying a scaling factor of 1 results in the largest possible plot that will fit on the display surface.

Interactive Graphics

Miscellaneous graphics and text can be added to map view plots through the use of a special data file called the "drawing commands file"[see Appendix A - Input Files (**Drawing Commands File**)]. The drawing commands file contains x-y coordinates that define the location of text strings and graphics. To aid in the preparation of drawing command files, MODPATH-PLOT can be run in an interactive graphics mode that allows users to determine the x-y location of points using a graphics cursor that can be moved using a graphics input device of some kind, such as a mouse or the keyboard cursor keys.

Not all display monitors or GKS installations support graphics input devices. Consequently, the graphics input mode can be turned on or off by setting the parameter, "KIND", in the file <device.dat> [see Appendix D - Installation (**Device File**)]. When KIND is set to 0, interactive graphics mode is off and no graphics input device is activated. When KIND is set to a negative number, interactive graphics mode is on and the absolute value of KIND indicates the graphics input device that will be used. Because many display monitors support more than one type of graphics input device, MODPATH-PLOT can be set up to allow the user to choose between several input devices by specifying the appropriate value for KIND in <device.dat>. For example, if the cursor keys are defined to be input device 1, the following line would activate graphics mode with cursor keys for graphic input:

```
1  -1  monitor  settings.1
```

If input device 2 is assigned to a mouse, the following line would activate graphics mode with the mouse for graphic input:

```
1  -2  monitor   settings.1
```

MODPATH-PLOT allows a maximum of two input devices. The number and type of input devices supported by display monitors depends on the GKS installation. Input device parameters are defined in the GKS Setup File [(see Appendix D - Installation (**GKS File**)].

When interactive graphics mode is on, a menu appears in the upper right corner of the display along with a graphics cursor (usually a cross or an arrow). A schematic illustration of how the display appears immediately after the interactive graphics mode is initiated is shown in figure 4-1a. The graphics cursor can be moved using the graphics input device. The x-y coordinates of the cursor location can be obtained by entering a point (for example, by pressing one of the mouse buttons). Each time a point is entered, its x-y coordinates appear at the top of the screen along with a sequential number indicating the point number (figure 4-1b). In addition, x-y coordinates of each point are recorded in the file <summary.plt> in the same format required by many of the commands used to process drawing commands files. Recording the coordinates in the file makes it very easy to use the graphics cursor to define complex objects because x-y coordinates can be cut

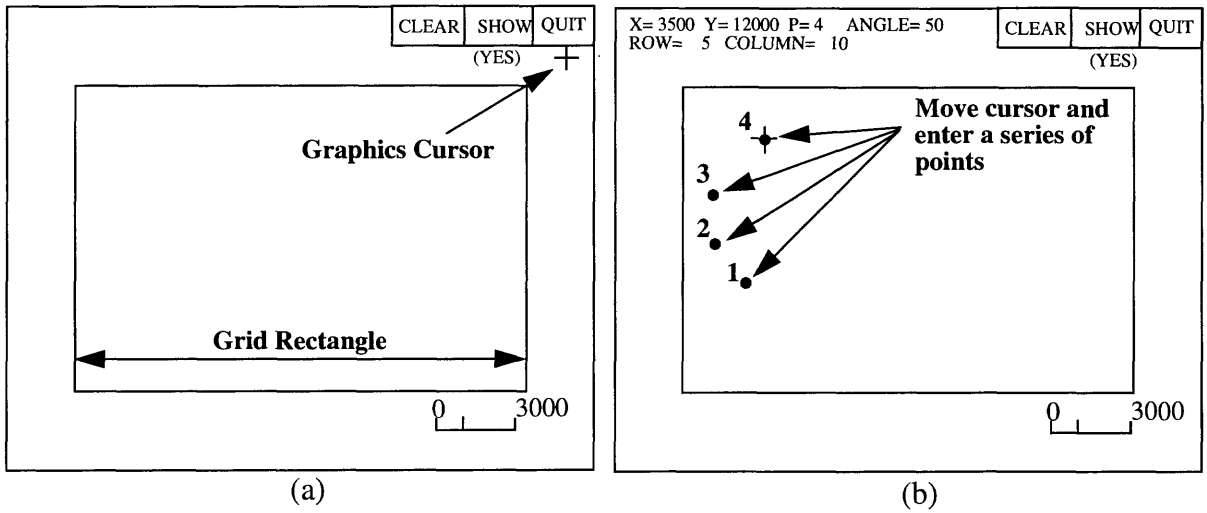


Figure 4-1. General layout of the interactive graphics screen (a) and the information that appears on the status lines when graphic points are entered using a mouse or keyboard (b)

from <summary.plt> and pasted directly into a drawing commands file. The row and column corresponding to each point also is displayed at the top of the screen.

The menu item SHOW provides an option to either show or not show points on the screen when they are entered. When the option is set to show the points, the term "(YES)" appear directly below the menu item SHOW. To turn off the SHOW option, move the graphics cursor to the SHOW box and enter a point. The term "(NO)" then appears directly below SHOW to indicate that the option is set to not show points. The SHOW menu item acts as a toggle switch that can be used to go back and forth between the options as necessary. The menu item CLEAR erases any interactive points and redisplays a fresh picture. Menu item QUIT terminates the plot and quits MODPATH-PLOT.

In addition to displaying the x-y coordinates and point number, the angle formed between a horizontal line and the line segment defined by the current point and the previous point also is computed and displayed. Positive angles are measured counterclockwise from the horizontal. Negative angles are measure clockwise from the horizontal. The angle is provided because it can be used to accurately specify the angle of rotation for text in a drawing commands file. For example, the interactive mode can be used to locate the position and angle of rotation for placement of labels that follow the contours. The labels then could be specified in a drawing commands file.

Specifying Input Data Files for MODPATH-PLOT

MODPATH-PLOT uses a special data file called a name file to provide the information it needs to open and manage input data files. MODPATH-PLOT prompts the user to enter the name of the name file at the beginning of each run. The name file is described in detail in Appendix A - Input Files (**Name File**).

The following data files must be specified in the name file if they are required for a MODPATH-PLOT analysis:

- main data file (type **MAIN**)
- MODFLOW head file (type **HEAD(BINARY)** or **HEAD**)
- MODFLOW stress package data files (types **RIV, DRN, GHB, WEL, STR**)
- ancillary data files referenced as external files by an array control record (type **DATA**)
- Grid Unit Array file (type **GUA**)
- MODFLOW drawdown file (type **DRAWDOWN(BINARY)** or **DRAWDOWN**)

The following files may be specified in the name file or as part of the interactive input during a MODPATH-PLOT run:

- composite budget file (type **CBF**)
- contour data file (type **CONTOUR-DATA**)
- contour level file (type **CONTOUR-LEVEL**)
- endpoint file (type **ENDPOINT**)
- pathline file (type **PATHLINE**)
- time series file (type **TIME-SERIES**)
- drawing commands file (type **DCF**)

If any of these seven files is required for a MODPATH-PLOT run, MODPATH-PLOT checks the name file to see if a file of the appropriate type has been specified. If specified, MODPATH-PLOT reads the data from the file specified in the name file. If not specified, MODPATH-PLOT prompts the user to enter a file name.

MODPATH-PLOT Output Files

MODPATH-PLOT generates the following output files:

- summary.plt -- the primary output file for MODPATH-PLOT.
- contours.dcf -- a drawing command file containing the contours generated during a run that produced contours (see section **Contouring** in this chapter).
- paths.dcf -- a drawing command file containing the pathlines generated during a map-view run that produced pathlines (see section **Pathlines** in this chapter).
- mplot.log -- a log file that contains information about the GKS workstation and system parameters.
- errors.gks -- a file that contains error messages from the GKS system.

The file, summary.plt, contains a summary of the input data as well as information about the status of a plot and errors that may have occurred during a run. In most cases, users will not need to concern themselves with the files, mplot.log and errors.gks.

Customizing MODPATH-PLOT

Major elements of MODPATH-PLOT's graphic environment can be controlled through the use of setup files that are loaded by MODPATH-PLOT at runtime:

(1) **Settings File** -- a configuration file that allows users to control the color and style of many graphic elements in MODPATH-PLOT.

(2) **Device File** -- a file named <device.dat> that allows the user to customize the graphics menu that is displayed by MODPATH-PLOT. The Device File is described in detail in Appendix D - Installation (Device File).

(3) **GKS File** -- a file named <gks.dat> that contains installation-dependent data used to setup the GKS system for the specific computer system. Once it has been set up, users should rarely need to modify the GKS file. A detailed description of the GKS file is provided in Appendix D - Installation (GKS File).

Settings File

The following parameter groups can be specified in the settings file:

1. **Color Cycle**
2. **Item Colors**
3. **Color Table Entries**
4. **Hatch Density**
5. **Window Size**
6. **Window Title**
7. **Grid Unit Color and Hatch Pattern**
8. **Page Layout**
9. **Hatch Pattern for Quasi-Three-Dimensional Confining Beds**
10. **Data Marker Type and Size**
11. **Line Type for Black and White Plots**
12. **Hatch Pattern for Inactive Cells**
13. **Color Menu Items**

The settings file allows the user to redefine the default values set by MODPATH-PLOT. With the exception of a few commands that are installation-dependent, parameters controlled by the settings file have default values defined by MODPATH-PLOT. Consequently, parameters do not have to be specified in the settings file if default values assigned by MODPATH-PLOT are acceptable.

It often is convenient to create multiple settings files for differing situations. A hierarchy of options is available for specifying which settings to use during a run:

1. No settings file is specified. Default settings hard-coded into the MODPATH program are used for all parameters.
2. Customized "default" settings files can be specified for individual devices in the Device File, <device.dat> [see Appendix D - Installation (**Device File**)]
3. A settings file can be specified interactively at run time.

The effect of specifying settings files in the order outline above is cumulative: (1) If no settings files are specified in <device.dat> or at runtime, the default settings in MODPATH-PLOT are used; (2) If a settings file is specified in <device.dat>, that file is used to change the value of any parameters specified within the file; and (3) If an additional settings file is specified at runtime, it is used to further change parameter values. This approach allows the user to define several parameter values using a settings file specified in <device.dat> and then redefine one or two values by specifying another settings file at runtime.

All data in the settings file is free-format. The settings file is scanned from beginning to end for each of the parameter groups. Parameters are specified with command lines that have the syntax:

```
COMMAND :  Pvalues
```

COMMAND-- is a character string command.
Pvalues-- are the parameter values. Pvalues may consist of numerical or character data.

The character string, COMMAND, must be terminated by a ":". The character string may be upper or lower case. Parameter values for the command are listed to the right of the ":" on the remainder of the line. Lines that contain unrecognized commands are ignored. Command lines for the various parameter groups may appear in any order in the settings file. Comment lines and in-line comments can be added anywhere in the file. Comment lines are denoted by an "@" in column 1. In-line comments are set off by inserting an "@" at the appropriate place in a line. The "@" and anything to the right are ignored when the line is processed. Blank lines also are ignored.

Example:

```
color cycle:  8    1 2 3 4 5 6 7 8   @ cycle through 8 colors (numbers 1-8)
item color:   2  15                 @interior grid lines(item 2) set to color 15
item color:  11  10                   @river symbol(item 11) set to color 10
color table: 14   0.50 0.50 0.50   'Medium Gray'
hatch density:  2.00  1.00
```

Color Cycle

Defines the number of colors in the cycling sequence and the color numbers that compose the sequence. Color cycling is described in the section, **Color Options**, in this chapter.

COLOR CYCLE : Ncycle Icycle(Ncycle)

Ncycle-- number of colors in the color cycling sequence.

Icycle-- color numbers that define the cycling sequence.

Default cycling sequence: 4 colors, color numbers 1 through 4.

Item Colors

Defines the colors used to draw specific features, or "items", on the plot.

ITEM COLOR : Itemnumber Itemcolor

Itemnumber-- indicates the number of the item being defined.

Itemcolor-- color number assigned to the item.

Plot items and the default colors assigned to them are defined in the section, **Color Options**.

Color Table Entries

This command defines colors assigned to specific color numbers in the color table by setting the red-green-blue intensities. All color numbers can be redefined. Color intensities can range from 0.0 to 1.0.

COLOR TABLE : ColorNumber Red Green Blue Name

ColorNumber--color number being defined.

Red-- red intensity.

Green-- green intensity.

Blue-- blue intensity.

Name-- character string specifying the name of the color that will appear in the menu. If the character string contains blanks, the string must be enclosed by apostrophes.

The color name should be 40 characters or less and must be enclosed by apostrophes only if it contains one or more blank spaces or a comma. The specified color name will appear in the color choices menu when running MODPATH-PLOT. Default color table entries are defined in the section, **Color Options**.

Hatch Density

Hatch density defines the density of hatch shade lines relative to the default density.

HATCH DENSITY : `Hvscale Dscale`

Hvscale-- is a scaling factor to adjust the line spacing for horizontal/vertical cross hatch shading. Values < 1 give shading that is less dense than the default density. Values > 1 give shading that is more dense than the default density.

Dscale-- is a scaling factor to adjust the line spacing for diagonal cross hatch shading. Values < 1 give shading that is less dense than the default density. Values > 1 give shading that is more dense than the default density.

Defaults: Hvscale = 1.0
Dscale = 1.0

Window Size

This command provides the option of customizing the size of the plot window for screen plots if the GKS implementation supports a multiple, or variable-size window environment. This command allows the user to enter a character string containing the data needed to set the size of the graphics window on GKS implementations that allow variable size or multiple windows. For each GKS implementation that supports variable window size, a special version of subroutine SWSIZE must be supplied that will decode the data in the character string and set the window size. The default subroutine SWSIZE supplied with MODPATH-PLOT does not perform any action, therefore this command has no effect if it appears in the Settings file. The specific data required for this command is a function of the GKS implementation.

WINDOW SIZE : `String`

String -- is a character string of up to 80 characters that contains data for setting the window size.

Defaults: MODPATH-PLOT does not specify a default condition for window size. The default window size for most GKS implementations in windowing environments is a full-screen plot.

Implementation with Prior GKS on a Data General workstation: The WINDOW SIZE command controls the location, size, and aspect ratio of an X-windows plot. The x and y coordinates are specified in pixels.

WINDOW SIZE : `x(left) y(top) x(right) y(bottom)`

x(left)-- x-coordinate of left edge of window.
y(top)-- y-coordinate of top edge of window.
x(right)-- x-coordinate of right edge of window.
y(bottom)-- y-coordinate of bottom edge of window.

Window Title

This command allows the user to specify a title in the window title bar of the plot window on workstations that support multiple windows. The title is passed to the user-defined subroutine SWSIZE. Subroutine SWSIZE is responsible for setting the window size and title for windowing workstations. The default subroutine SWSIZE supplied with MODPATH-PLOT does not perform any action, and the Window Title command is ignored if it appears in the Settings file.

WINDOW TITLE : Wtitle

Wtitle-- a character string of up to 80 characters defining the window title. In this case, the character string does not have to be enclosed in apostrophes even if it contains blanks.

Defaults: MODPATH-PLOT does not specify a default condition for window title.

Implementation with Prior GKS on a Data General workstation: The WINDOW TITLE controls the title that will appear in the title bar of an X-windows plot. If the user does not set the title, MODPATH-PLOT substitutes the MODPATH-PLOT program name and version number.

Grid Unit Color and Hatch Pattern

This command defines the color used to shade data from the grid unit array. It also defines the hatch pattern that will be used when the option to shade grid unit data with a hatch fill is used.

GRID UNIT : GUnumber GUcolor GUhatch

GUnumber-- is the grid unit number.

GUcolor-- is the color index number associated with the specified grid unit number.

GUhatch-- is the index defining the hatch style to use if hatch shading is in effect.

Available patterns:
1 = +45 degree diagonal lines
2 = -45 degree diagonal lines
3 = diagonal cross hatch
4 = horizontal lines
5 = vertical lines
6 = horizontal/vertical cross hatch

Defaults: MODPATH-PLOT defines 15 grid unit numbers, where GUcolor=GUnumber. Beginning with grid unit number 1, the hatch styles cycle 1 to 6.

Page Layout

The command, LAYOUT, allows the user to specify a character string of up to 80 characters that contain instructions for defining the page size. Special page layout instructions can be created for specific graphic devices. These instructions are not pre-defined by MODPATH-PLOT. Custom code must be added to MODPATH-PLOT that will interpret the instructions in the LAYOUT command for specific devices supported by a GKS system. The standard version of MODPATH-PLOT ignores the LAYOUT command.

LAYOUT : `Playout`

Playout-- is a character string of up to 80 characters.

Defaults: MODPATH-PLOT does not specify a default condition for window size.

Implementation with Prior GKS on a Data General workstation: The LAYOUT command controls the page orientation for postscript output files. Playout can have the following values:

LANDSCAPE-- generates an 11 inch x 8.5 inch plot.

PORTRAIT-- generates an 8.5 inch x 11 inch plot.

Hatch Pattern for Quasi-3d Confining Beds

The command, CONFINING BED PATTERN, sets the hatch pattern that is used to fill quasi-3d confining beds when the option to shade with a hatch pattern is selected.

CONFINING BED PATTERN : `CBpattern`

CBpattern-- is the index number of the hatch pattern used to fill quasi-3d confining beds.

Available patterns:
1 = +45 degree diagonal lines
2 = -45 degree diagonal lines
3 = diagonal cross hatch
4 = horizontal lines
5 = vertical lines
6 = horizontal/vertical cross hatch

Defaults: CBpattern = 1

Data Marker Type and Size

The command, DATA MARKER, sets the marker type and size scaling factor that is used to draw the point markers for time series plots and for endpoint plots that do not cycle through marker types.

DATA MARKER : `DMtype DMsize DMkind`

DMtype-- is the data marker type. The following five marker types are available:

Available types: 1 = •
2 = +
3 = *
4 = o
5 = x

DMsize-- a scale factor for setting the marker type. When DMsize = 1, the marker size is equal to the default size, which is a function of the specific GKS implementation.

DMkind-- a flag that indicates how the markers are constructed. If DMkind=0, the program uses the GKS system to construct the markers. If DMkind=1, MODPATH-PLOT constructs markers with GKS line-drawing commands. It is usually most efficient to let the GKS system generate the markers (DMkind=0). However, markers constructed entirely from lines (DMkind=1) often are more portable when the output from MODPATH-PLOT is in the form of a computer graphics metafile (CGM). When DMkind=1, marker number 4 is displayed as a square rather than a circle.

Defaults: DMtype = 1 DMsize = 1 DMkind = 0

Line Type for Black and White Plots

The command, LINE TYPE, controls the line style used to draw pathlines when a black and white plot is drawn.

LINE TYPE : `Ltype`

Ltype-- is the line type. The following options are available:

0 = cycle through line types according to zone code value
1 = solid
2 = dashed
3 = dotted

Default: Ltype = 1

Hatch Pattern for Inactive Cells

The command, INACTIVE CELL PATTERN, specifies the hatch pattern used to fill inactive cells when the option to shade inactive cells with hatch fill is selected.

INACTIVE CELL PATTERN : IACpattern

IACpattern-- is the index of the hatch pattern used to fill inactive cells.

Available patterns:
1 = +45 degree diagonal lines
2 = -45 degree diagonal lines
3 = diagonal cross hatch
4 = horizontal lines
5 = vertical lines
6 = horizontal/vertical cross hatch

Defaults: IACpattern = 1

Color Menu Items

The command COLOR MENU ITEMS controls the number of colors that will be included when the menu of colors is displayed on the screen. A maximum of 15 colors can be displayed on the menu even though some graphic devices may support more than 15 colors. The number of colors displayed on the menu does not limit the ability of the user to select color numbers supported by the device but not displayed on the menu.

COLOR MENU ITEMS : NumItems

NumItems-- is the number of colors listed on the menu

Defaults: NumItems = 4

Chapter 5

Prompt and Response System

Overview of the Prompt and Response System

In interactive mode, MODPATH and MODPATH-PLOT obtain much of their data by displaying prompts on the screen and reading responses from the keyboard. The prompts and responses are recorded in a special output file called a **Response File** that can be used in several ways:

(1) to provide a record of an interactive run,

(2) to run in "batch" mode by reading all responses directly from the response file, and

(3) to design customized interactive input schemes. Custom input schemes are developed by inserting special "flags" in the response file that indicate whether specific data items should be entered interactively or read directly from the response file.

An on-line **help system** also is provided for data items that are input interactively. The help system is designed so that it can be extended and customized without modifying the FORTRAN source code.

Keyboard and Response File Input

At the beginning of each run, the user is asked to enter the name of a response file. If no file name is entered, data is input interactively from the keyboard and the interactive input is recorded in a file named <mpath.rsp> (for MODPATH) or <mplot.rsp> (for MODPATH-PLOT). Response files can be renamed to avoid being overwritten during subsequent runs. Prompts are signified by an "*" in column 1. The end of a sequence of prompts is denoted by a "prompt termination line" that begins with the character string "@RESPONSE". One or more response lines then are recorded in the file exactly as they were entered at the keyboard. A response file provides a complete, sequential record of all the prompts and responses used to make an interactive run.

Both programs can be run in batch mode with a response file generated from a previous interactive run. A batch run is made by entering the name of a response file when prompted for a response file name. In batch mode, input is read directly from the response file and prompts are not displayed on the screen. Batch mode runs can be further streamlined by forcing the program to read from the default response file (<mpath.rsp> or <mplot.rsp>) without prompting the user to enter a name for the response file. To implement this option, the response file must have the appropriate default file name (<mpath.rsp> or <mplot.rsp>) and the word "batch" must be added to the comment header line. The header line is always the first line in the response file. For example, MODPATH produces a header line of the form:

@[MODPATH Version 3.00 (V3, Release 1, 9-94)]

A batch version of the response file is produced by changing this line to read:

@[MODPATH Version 3.00 (V3, Release 1, 9-94)] batch

At the beginning of each run, the programs check to see if a file with default response file name is present in the local directory. If the file is present, the first line is read to see if a batch run has been specified. If a batch run was specified, the programs read data directly from <mpath.rsp> or <mplot.rsp> without prompting for a response file name. Although batch mode suppresses prompts and keyboard input, some summary information and error messages are written to the screen. All screen output can be suppressed by adding the word "silent" to the response file header line:

@[MODPATH Version 3.00 (V3, Release 1, 9-94)] batch silent

The "batch silent" designation is useful for running the programs repeatedly as part of an automated sequence, such as a Monte Carlo analysis based on MODPATH simulations.

Response files can be modified so that some input is obtained from the keyboard and some is read directly from the response file. In batch mode, the programs will redirect a prompt sequence to the screen and wait for keyboard input if it finds the character string "(?)" at the end of the prompt termination line (the line that begins with "@RESPONSE"). If desired, a text editor

can be used to customize the prompts that are redirected to the screen. This approach allows both the quantity and style of interactive input to be tailored to produce "templates" for the most frequently-used combinations of simulation options.

Comment lines and in-line comments can be added to annotate a response file. Comment lines begin with "@" in column 1 and can appear anywhere in the file. Comment lines are ignored by MODPATH and MODPATH-PLOT. In-line comments can be added to any line in a response file by inserting an "@" to mark the beginning of the comment. The "@" and anything to its right are ignored. For an in-line comment, the "@" must appear in column 2 or beyond.

Care should be taken when a response file is edited and reused because some changes may result in the need for additional lines of data that were not included in the original response file. The response file will not provide the proper sequence of responses if the additional lines of data are not present.

Help File System

For keyboard input, on-line help is available for an item whenever the message [?=Help] is displayed at the end of a prompt sequence. The help information is accessed by entering the response, ?. One or more lines of help text will be displayed on the screen. MODPATH then waits for the user to re-enter a response. If additional help is available, the program displays the message, [??=More Help]. The additional help can be displayed by entering the character string, ??. The help information for MODPATH is stored in a text file named <mpath.hlp> located in the directory containing MODPATH and MODPATH-PLOT setup files. Help information for MODPATH-PLOT is contained in a file named <mplot.hlp>. Users also can create a third level of supplemental help by generating an additional help text file named <mpath.shp>, for MODPATH, or <mplot.shp>, for MODPATH-PLOT. The third level of supplementary help is accessed by entering the response, ???.

The information displayed by the help system is stored in standard ASCII text files. These files normally are located in the MODPATH setup directory. The help files provide two levels of help that can be accessed by entering "?" or "??" in response to an interactive prompt. The structure of the help files consists a series of help "items". Each help item consists of a one-line control record followed by the text for that help item. Control records are flagged by placing a ":" in column 1. Data is free-format. The control record structure is:

```
: Label  Nhelp1  Nhelp2
```

Label-- a character string identifying the help item. This text label for each prompt sequence also is recorded in the response file on the prompt termination line. Recording the label in the response file makes it easy to associate help items in the help files with specific input items.

Nhelp1-- the number of lines of help text that will be displayed in response to a "?" inquiry.

Nhelp2-- the number of lines of help text that will be displayed in response to a "??" inquiry.

The example presented on the next page demonstrates how the help system works.

As part of its interactive input, MODPATH-PLOT prompts the user to enter the name of the Name File:

```
ENTER THE NAME FILE:
[?=HELP]
```

If the user enters a "?", the help system responds:

```
> * The "Name File" supplies MODPATH-PLOT with the information it needs to
>   manage input data files. It includes (1) data file names, (2) file unit
>   numbers, and (3) file types.
>
> See Chapter 4: MODPATH-PLOT User's Guide
>                Section: "Specifying Input Data Files for MODPATH-PLOT".
>
>    Appendix A: Input Files
>                 Section: "Name File"
--> ENTER A RESPONSE:
```

The help file entry corresponding to this input item is:

```
: 5.2.1   9   0
> * The "Name File" supplies MODPATH-PLOT with the information it needs to
>   manage input data files. It includes (1) data file names, (2) file unit
>   numbers, and (3) file types.
>
> See Chapter 4: MODPATH-PLOT User's Guide
>                Section: "Specifying Input Data Files for MODPATH-PLOT".
>
>    Appendix A: Input Files
>                 Section: "Name File"
```

The help label "5.2.1" identifies the input item. The symbol ">" at the beginning of each line is not required, but is included for all help items to clearly identify the blocks of help text. In this example, no second level of help is available. Consequently, if a user requested second-level help by entering "??", a message would be displayed on the screen that no additional help is available. The prompt sequence for this item would be recorded in the response file as:

```
* ENTER THE NAME FILE:
@RESPONSE:  HELP LABEL = 5.2.1
```

Users can define a third level of help by creating supplemental help files named <mpath.shp> and <mplot.shp>. The supplemental help is accessed by entering "???". The structure of the supplemental help files is the same as for <mpath.hlp> and <mplot.hlp>. Only one level of help is available from the supplemental help file. Variable Nhelp2 either can be omitted from control records or set equal to zero. A supplemental help file can be created relatively easily by using the response file to identify the help label associated with a specific prompt. That help label then can be used to mark a block of help text in the help file. It is not necessary to refer to the source code.

Chapter 6

Examples

Description

Two sample problems are presented to illustrate the input and output data from MODPATH and MODPATH-PLOT. Both problems are based on the same hypothetical flow system illustrated in figure 6-1. The flow system consists of two aquifers separated by a 20-foot-thick confining layer. Recharge to the system is uniformly distributed over the water table at a rate of 0.0045 feet per day. Discharge occurs to a partially-penetrating river along the right side of the flow system in the upper aquifer. Some water also discharges to wells. In problem 1, the system is at steady state and contains one well, which is located in the lower aquifer and discharges at 80,000 cubic feet per day. In problem 2, a second well, located in the upper aquifer immediately above the well in the lower aquifer, is added to the system described for problem 1. Transient flow resulting from the addition of the second well is simulated.

The system is simulated using a finite-difference grid containing 27 rows, 27 columns, and 5 layers (**figure 6-2**). Horizontal grid spacing varies from 40 foot by 40 foot squares at the wells to 400 foot by 400 foot squares away from the wells. The upper unconfined aquifer is simulated with one finite-difference layer. A quasi-three-dimensional representation is used for the confining layer. The lower aquifer is represented by four 50-foot-thick layers. Both wells are located in row 14, column 14. The well in the lower aquifer is located in model layer 4.

A complete listing of the MODFLOW and MODPATH input data files for these two problems is provided in Appendix E - **Data Files for Example Problems.**

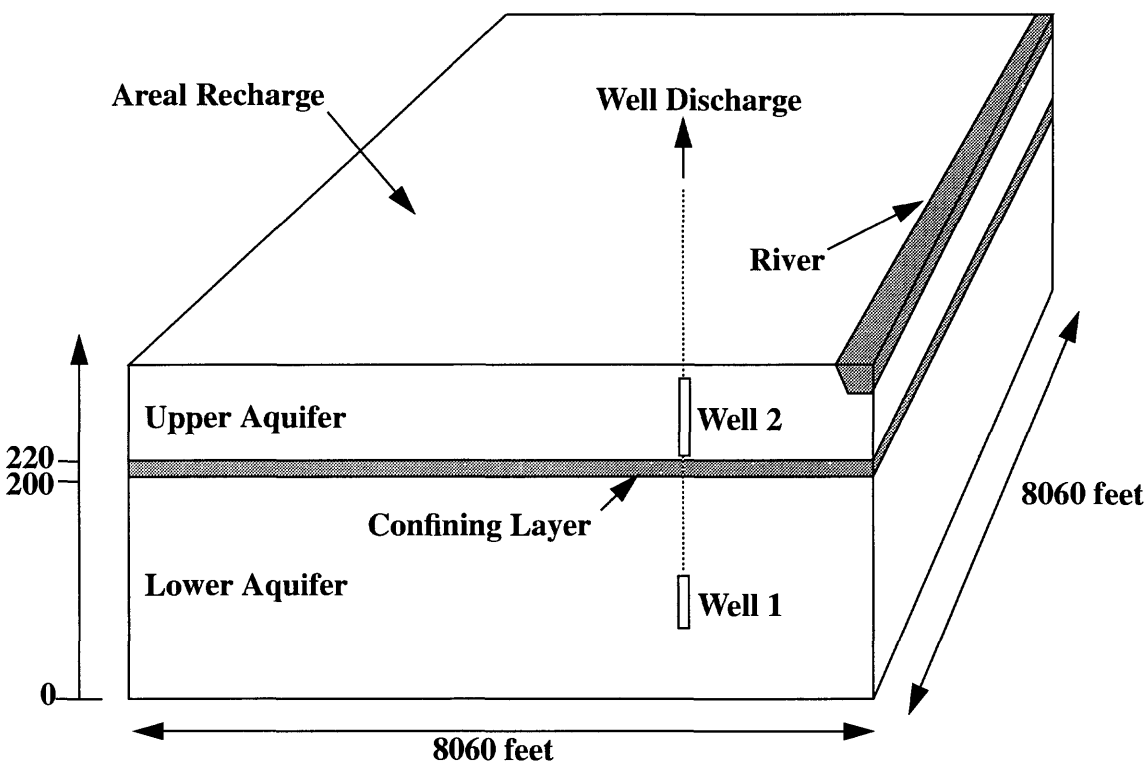


Figure 6-1. Conceptual model for hypothetical sample problems.

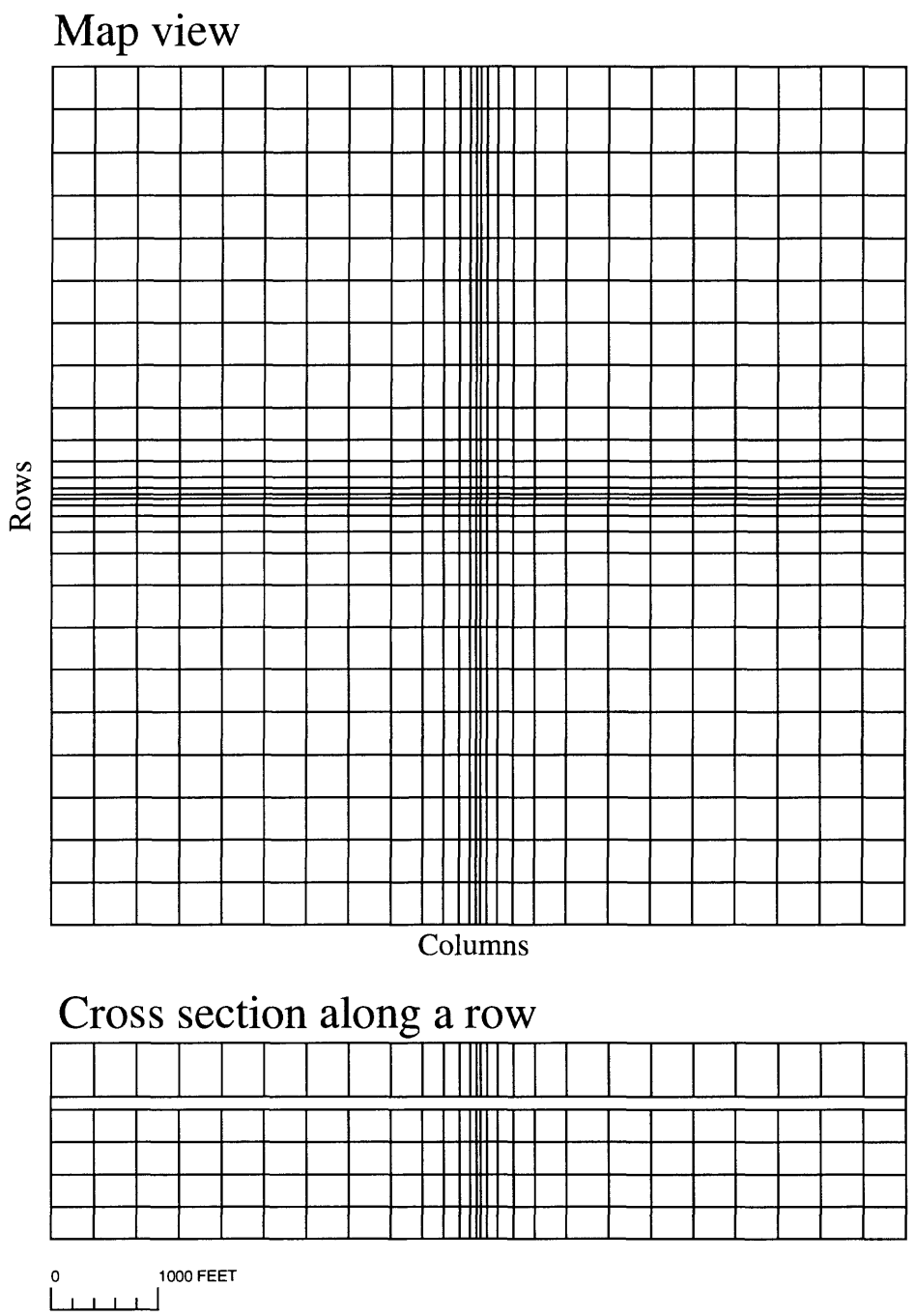


Figure 6-2. Finite-difference grid of hypothetical flow system of problems 1 and 2.

Problem 1 -- steady-state flow

Once the MODFLOW simulation has been completed, the next step in a MODPATH analysis is to construct the MODPATH main data file. Figure 6-3 shows the main data file that is used in all of the steady state runs of problem 1. Detailed instructions for constructing the main data file are provided in Appendix A - **Input Files**. Note that steady state is specified by setting the variable NPER equal to 0 on the first line of the main data file. Also, the structure of the file is simplified considerably by the use of free-format input and keyword-indicators in the array control records. Free-format input and array-data input is discussed in detail in Appendix A - Input Files (**General Structure of Input Data**).

```
27 27 5 1 1 0 0 999.9 1.0E+30 0                    [Item 1: Grid Dimensions and Specifications]
compact                                            [Item 2: Options]
1 0 0 0 0                                          [Item 3: LAYCON]
1 0 0 0 0                                          [Item 4: NCON]
internal 1.0 (free) 0                              [Item 5a: DELR control record]
 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.
internal 1.0 (free) 0                              [Item 5b: DELC control record]
 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.
internal 1.0 (free) 0                              [Item 6a: DELZ control record]
 100. 50. 50. 50. 50.
internal 1.0 (free) 0                              [Item 6b: DELZCB control record]
 20. 0. 0. 0. 0.
220.                                               [Item 6c: ZBL1, bottom elevation of layer 1]
open/close ibound.1 1 (free) 0                     [Item 8: IBOUND control record for layer 1]
constant 1                                         [Item 8: IBOUND control record for layer 2]
constant 1                                         [Item 8: IBOUND control record for layer 3]
open/close ibound.4 1 (free) 0                     [Item 8: IBOUND control record for layer 4]
constant 1                                         [Item 8: IBOUND control record for layer 5]
constant 0.3                                       [Item 9a: POROSITY control record for layer 1]
constant 0.3                                       [Item 9b: POROSITY control record for cb]
constant 0.3                                       [Item 9a: POROSITY control record for layer 2]
constant 0.3                                       [Item 9a: POROSITY control record for layer 3]
constant 0.3                                       [Item 9a: POROSITY control record for layer 4]
constant 0.3                                       [Item 9a: POROSITY control record for layer 5]
```

Figure 6-3. MODPATH main data file for problem 1.

In addition to the main data file, MODPATH reads modified versions of the stress package data files that were prepared for the MODFLOW simulation. Stress package files must be modified to add the IFACE and ITOP data. IFACE and ITOP are numeric codes that tell MODPATH how to distribute the stress flow within the cell. **Figure 6-4** shows the stress package data files used in problem 1. Note that the IFACE has been added to the river and well packages. The IFACE value of 6 in the river package tells MODPATH to treat flow to or from rivers as if it comes across the top

face of the cell. The IFACE value of 0 in the well package tells MODPATH to treat wells as internal sinks. The ITOP value in the recharge package is set equal to 1, which tells MODPATH to treat recharge as if it enters the aquifer across the top face of those cells that contain recharge.

River Package File

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

Well Package File

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

Recharge Package File

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

Figure 6-4. Stress package data files for problem 1.

The final step in preparing the data is to construct the name file, which provides a list of data files needed by MODPATH and MODPATH-PLOT. Figure 6-5 shows the name file used to make one of the MODPATH runs for problem 1 (Run 1: Forward Tracking Endpoint Analysis).

```
main           10  demo-s.mdf
wel            11  demo-s.wel
riv            12  demo-s.riv
rch            13  demo-s.rch
budget         50  demo-s.bud
head(binary)   60  demo-s.hed
endpoint       75  endpoint.s1
```

Figure 6-5. Name file for problem 1.

The name file supplies MODPATH and MODPATH-PLOT with three pieces of information for each data file: (1) type of data, (2) FORTRAN unit number, and (3) file name. A detailed description of the name file is provided in Appendix A - Input Files **(Name File)**. In this case, the main data file named "demo-s.mdf" is designated as type MAIN and assigned to FORTRAN unit 10. The stress package data files are designated in a similar fashion. The file "demo-s.bud", designated as type "budget", contains the cell-by-cell budget data output by MODFLOW. The binary head data output by MODFLOW is contained in file "demo-s.hed", which is designated as type "head(binary)".

The last line in the name file shown in figure 6-5 designates the file named "endpoint.s1" as the endpoint file. In previous versions of MODPATH and MODPATH-PLOT, particle coordinate output files (endpoint, pathline, and time series files) were automatically given the default names "endpoint", "pathline", and "timesers". The current versions of MODPATH and MODPATH-PLOT provide the option of specifying those file names directly in the name file. If endpoint, pathline, and time series files are not specified directly in the name file, MODPATH automatically uses the default names for the files.

Run 1: Forward Tracking Endpoint Analysis

One of the most common uses of particle tracking models is to delineate areas of recharge that contribute water to aquifers or to major discharge features, such as wells. The endpoint analysis option in MODPATH provides a convenient way of mapping recharge areas by recording the initial and final locations of particles in an endpoint file. MODPATH-PLOT reads the endpoint file and plots the points. Endpoint analyses can be carried out either in forward or backtracking mode. Endpoint analyses are efficient because MODPATH only records information about the initial location, final location, and tracking time. In this example, a forward tracking endpoint analysis is performed to map out the recharge area at the water table that contributes water to the well in layer 4. The automatic particle generation option is utilized to place a 2x2 array of particles on the top face of every cell in layer 1 (a total of 2,916 particles). MODPATH records the endpoint data in a file; MODPATH-PLOT reads the data from the file and plots the points.

Upon execution, MODPATH issues the prompt:

```
MODPATH Version 3.00 (V3, Release 1, 9-94)
TO READ INPUT FROM AN EXISTING RESPONSE FILE, ENTER FILE NAME:
( <CR> = ENTER DATA INTERACTIVELY )
 [ ? = Help ]
```

The user has the choice of entering data interactively or specifying a response file that contains input responses from a previous run. Press ENTER to supply data interactively for this run. MODPATH proceeds to prompt for the name file and other data to define the type of analysis, style of output, and particle starting locations. The prompts and user responses are recorded in a response file named "mpath.rsp". The response file for this MODPATH run is listed in Appendix E (**path-s1.rsp**). MODPATH can read data directly from a response file. This run could be duplicated by executing MODPATH again and then entering the file name, "mpath.rsp". Because each interactive run overwrites the file "mpath.rsp" if it exists, response files must be renamed if they are to be saved.

When MODPATH finishes, the following summary information is displayed on the screen:

```
MINIMUM TRAVEL TIME =    3.07695E-02
MAXIMUM TRAVEL TIME =    2.82989E+05
AVERAGE TRAVEL TIME =    1.46916E+04
  78.5% OF THE PARTICLES HAD TRAVEL TIMES LESS THAN THE AVERAGE TRAVEL TIME

      0 PARTICLES REMAIN ACTIVE
   2916 PARTICLES STOPPED AT INTERNAL SINKS/SOURCES OR BOUNDARIES
      0 PARTICLES STOPPED IN AN AUTOMATIC TERMINATION ZONE
      0 PARTICLES WERE STRANDED IN INACTIVE CELLS
      0 PARTICLES WERE NOT RELEASED
-----
   2916 PARTICLES ACCOUNTED FOR OUT OF A TOTAL OF    2916
```

The first four lines of summary information provide some basic statistics about travel time for the entire distribution of particles. Travel time is defined as TRACKING TIME - RELEASE TIME. In this case, the release time for all particles was 0. The second five lines summarize the ultimate disposition of each particle in the system. In this case, all 2,916 particles terminated normally at internal sinks (the well in layer 4) or boundaries (the top faces of cells that contain rivers). For this run, a stopping time for particles was not specified. Consequently, all of the particles were tracked until they reached a point of termination. As an alternative, we could have specified a time at which to stop the particle tracking computation. In that case, the number of particles that remained within the flow system at the end of the run would be listed as "active". It also is possible to specify a zone code value that will cause MODPATH to terminate a particle whenever it enters a cell with an IBOUND value equal to that zone code. Such particles would be listed as terminating in an "automatic termination zone". The final two categories ("stranded" and "not released") should only occur in certain transient flow simulations. Stranded particles are those that are left in cells that go dry during the course of a time step in a transient simulation. Particles can fall in the category "not released" for some transient simulations in which particles are released using the multiple release option. For steady-state simulations, particles should never be stranded or unreleased. Legitimate situations can occur for which a small number of particles can be stranded or unreleased in transient simulations. However, when a large percentage of the total number of particles fall into either of these two categories, the model's spatial or time step discretization probably is not adequate to yield reliable results with MODPATH, or errors may exist in particle starting locations.

The results from the MODPATH run can be plotted using MODPATH-PLOT. MODPATH-PLOT reads the same data files as MODPATH. MODPATH-PLOT also allows data to be input interactively or with a response file. The response file used to plot the endpoints from run 1 is listed in Appendix E (**plot-s1.rsp**). The plot is shown in **figure 6-6**. The starting locations of particles that discharge to the well in layer 4 are displayed. Starting locations of particles that discharge to the river are not plotted. Control over the color of the points and which points are

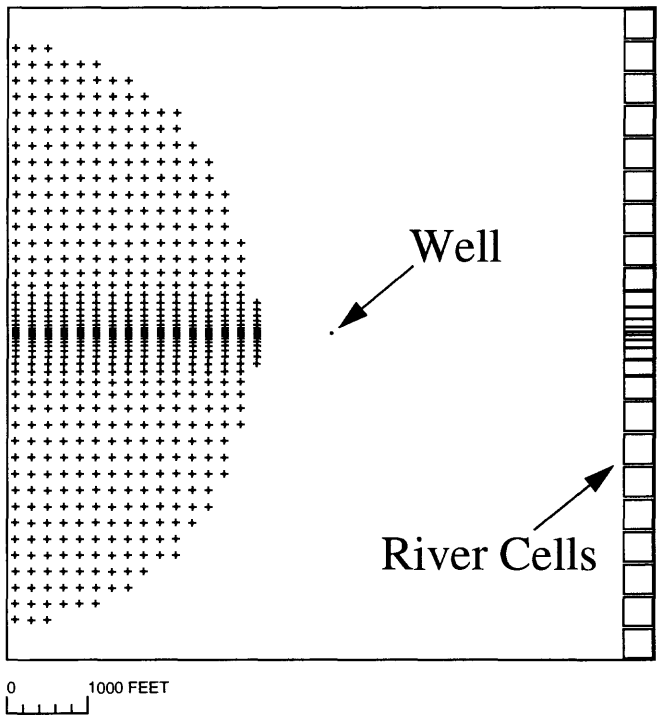


Figure 6-6. Capture area for well in layer 4 delineated by tracking 2916 particles forward from the water table.

plotted is accomplished by the zone codes stored in the IBOUND array. The IBOUND value for the cell containing the well in layer 4 is set equal to 2, which corresponds to the color red. The IBOUND values for the river cells are equal to 1. In the case of a forward tracking endpoint plot, starting locations are not plotted for particles that discharge in cells with an IBOUND value of 1. Starting locations for the river discharge could be displayed by changing the IBOUND zone codes for the river cells to a number other than 1.

The particle locations in figure 6-6 map out the capture area for the well. Note that the capture area does not immediately overly the well. The reason for the shape of the capture area will become clearer later when we look at pathlines in a cross section plot perpendicular to the river.

Run 2: Backward Tracking Endpoint Analysis

An alternative way of mapping the capture area for the well is to track particles backward from the well cell and plot the final locations of the particles at the water table. Backward tracking often allows capture areas to be delineated with fewer particles than would be required for an equivalent forward tracking analysis. In this run, a total of 200 particles are placed on the six faces of the well cell (row 14, column 14, layer 4). When backtracking particles from a cell containing a well, it usually is most efficient to place particles on the faces of the well cell rather than distribute them internally within the cell.

The response files for the MODPATH and MODPATH-PLOT runs are listed in Appendix E (**path-s2.rsp** and **plot-s2.rsp**). **Figure 6-7** shows the result of this simulation. Note that the capture area mapped out by backward tracking coincides with that produced by forward tracking. In this case, the final locations of particles are plotted. The rings of particles reflect the regular pattern of the two-dimensional arrays of particles placed on each of the well cell faces.

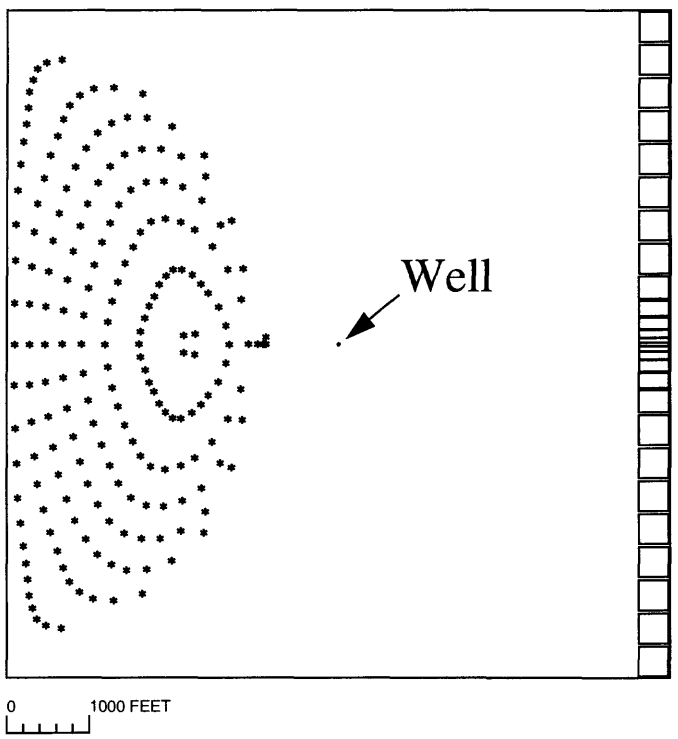


Figure 6-7. Capture area for well in layer 4 delineated by tracking 200 particles backward from the cell containing the well.

Run 3: Forward Tracking Pathline Analysis in Cross Section

For this analysis, it is instructive to examine pathlines in a cross section along row 14 that is perpendicular to the river and passes through the cell containing the well. Twenty uniformly-spaced particles are placed at the water table and tracked forward to their points of discharge. In the previous two runs, MODPATH's automatic particle generation option was used to generate regular arrays of particles. For this run, particle locations will be read from the data file shown in figure 6-8.

```
1 14 1   383 0.5 1.0   2 0 0   0.0
1 14 1   766 0.5 1.0   2 0 0   0.0
1 14 1  1149 0.5 1.0   2 0 0   0.0
1 14 1  1532 0.5 1.0   2 0 0   0.0
1 14 1  1915 0.5 1.0   2 0 0   0.0
1 14 1  2298 0.5 1.0   2 0 0   0.0
1 14 1  2681 0.5 1.0   2 0 0   0.0
1 14 1  3064 0.5 1.0   2 0 0   0.0
1 14 1  3447 0.5 1.0   2 0 0   0.0
1 14 1  3830 0.5 1.0   2 0 0   0.0
1 14 1  4213 0.5 1.0   2 0 0   0.0
1 14 1  4596 0.5 1.0   2 0 0   0.0
1 14 1  4979 0.5 1.0   2 0 0   0.0
1 14 1  5362 0.5 1.0   2 0 0   0.0
1 14 1  5745 0.5 1.0   2 0 0   0.0
1 14 1  6128 0.5 1.0   2 0 0   0.0
1 14 1  6511 0.5 1.0   2 0 0   0.0
1 14 1  6894 0.5 1.0   2 0 0   0.0
1 14 1  7277 0.5 1.0   2 0 0   0.0
1 14 1  7660 0.5 1.0   2 0 0   0.0
```

Figure 6-8. Starting locations file for run 3 of problem 1 (demo-s3.loc).

Each line in figure 6-8 represents starting location data for one particle. The first three numbers give the column, row, and layer of the cell that contains the particle; the second three numbers give the x, y, and z locations of the particle; the third three numbers are coordinate-type codes that indicate whether the x-y-z coordinates represent local or global coordinates; the last number on each line is the release time for the particle. A detailed description of the starting locations file is provided in Appendix A - Input Files (**Starting Locations File**). In this case, the x coordinates are input as global coordinates (code 2). The y and z coordinates are specified as local coordinates (code 0). The local y equals 0.5, the mid-point in the cell; the local z equals 1.0, the top of the cell. When the coordinate-type code is 2, MODPATH reads the global coordinate and determines the corresponding column, row, or layer. A value of 1 was given for the starting column location of each particle, but the value is ignored in this case because MODPATH calculates the column location automatically. It would have been possible to use the automatic particle generation option to create particles for this pathline run. However, automatic particle generation would produce irregularly spaced starting locations because of the variable grid spacing. The starting

locations file allows the starting locations to be customized to produce uniform spacing in the x direction.

Response files for MODPATH and MODPATH-PLOT are listed in Appendix E (**path-s3.rsp** and **plot-s3.rsp**). **Figure 6-9** shows the pathlines generated by this run. The colors of the pathlines are determined by the zone codes contained in the IBOUND array for the well cell and the river cells, just as in runs 1 and 2. In this case, the pathlines show a clear picture of vertical flow in the system because the cross section was taken along a line of symmetry in the flow system where flow is truly two-dimensional in vertical cross section. Most three-dimensional flow systems are asymmetric and it usually is not possible to see clear pathline plots because the three-dimensional lines overlap when projected onto a cross section or a map view. In cases where flow patterns are complex and three-dimensional, plots containing a small number of well-chosen pathlines usually result in the clearest figures.

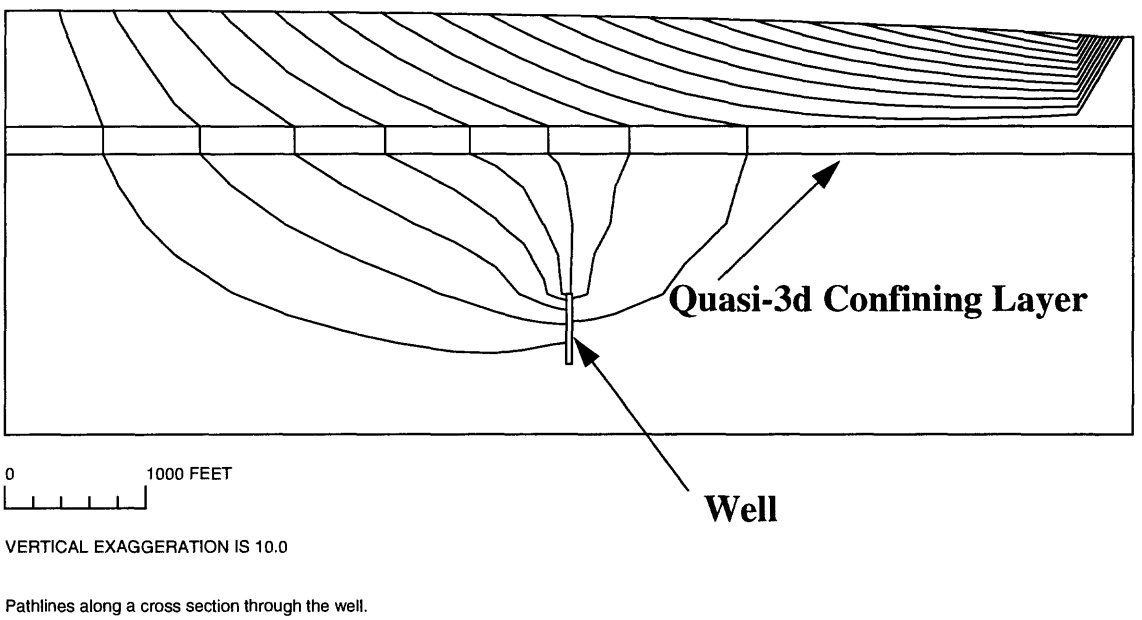


Figure 6-9. Pathlines tracked forward from the water table along row 14.

Run 4: Time Series Analysis with Multiple Release Times

In addition to endpoint and pathline analyses, MODPATH provides the option to record and display the location of particles at specific points in time. That option is referred to as a time series analysis. The current version of MODPATH also adds the capability to release a stream of particles over a specified period of time. Combining multiple particle release with time series analysis allows the user to generate a "plume" of particles that can be displayed at selected points in time. This approach is extremely useful in general, and is especially valuable for transient simulations.

A line of particles is released along a portion of the water table in row 14 (columns 8 and 9). The particles are released every 0.5 years for a period of 10 years. That 10-year plume of particles is then followed through the system. The particles are placed within the same line-of-symmetry cross section used in run 3. The plume is viewed by looking at cross section plots along row 14 for a number of points in time. A series of four plots were generated with MODPATH-PLOT. Each plot shows the locations of particles at a specific point in time: 10, 20, 25, and 35 years. These points in time correspond to time steps 2, 4, 5, and 7. A composite of these four plots is shown in **figure 6-10**. The response files for MODPATH and MODPATH-PLOT are listed in Appendix E (**path-s4.rsp** and **plot-s4.rsp**). Only the response file for generating the 10-year time plot is provided in Appendix E. The other plots can be generated simply by changing the time step number specified in the response file. These plots give a visual representation of the plume as it moves vertically and laterally through the system. Note how the plume eventually splits into two parts, with one part discharging to the well and the other to the stream.

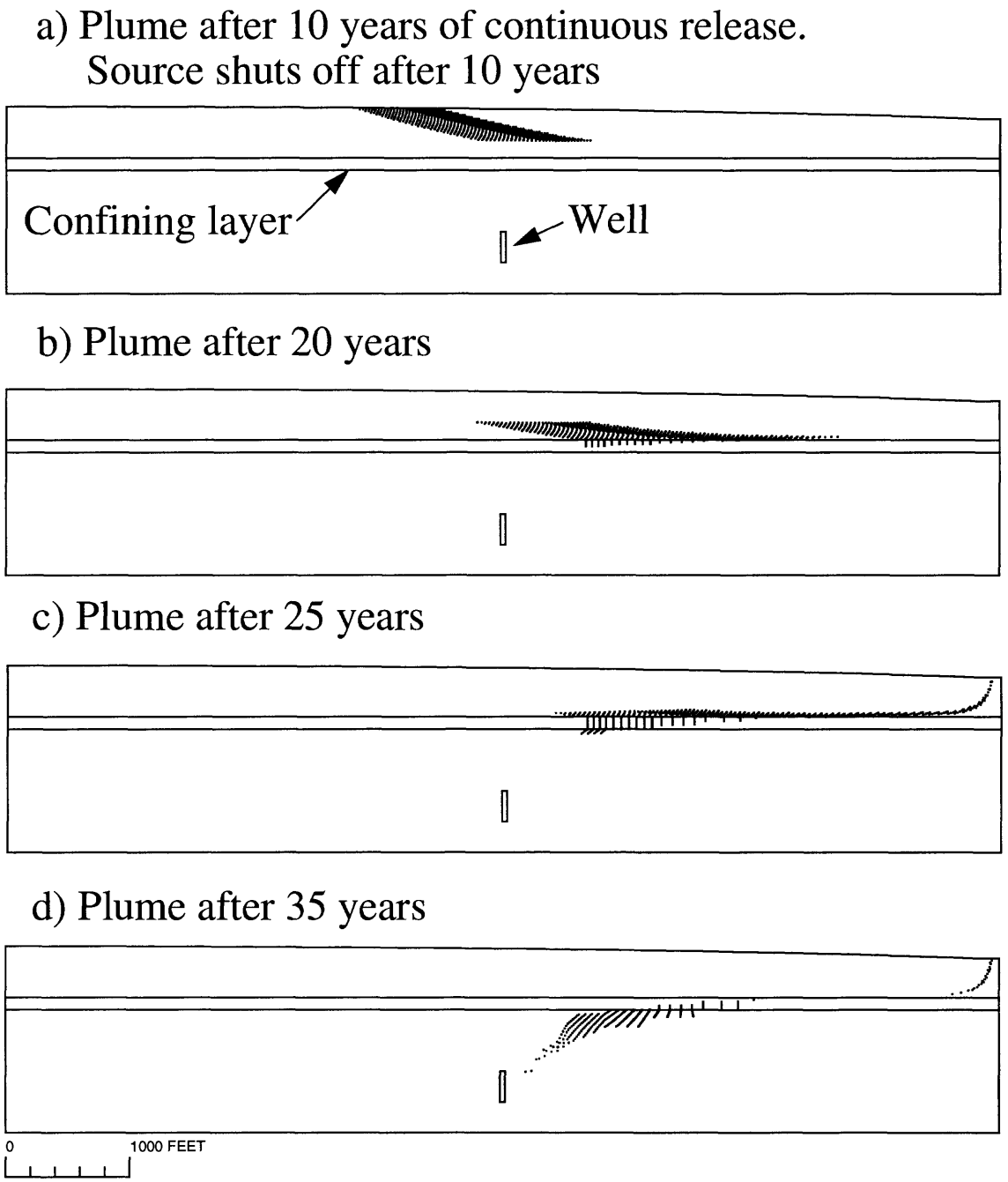


Figure 6-10. Time series plots showing dissipation of a plume formed by 10 years of continuous release of particles at the water table.

Problem 2 -- transient flow

Transient flow simulation with MODPATH requires relatively few changes to the main data file. Stress package data files do not require any special changes for transient MODPATH simulations beyond those already described for steady-state simulations. The primary difference between steady state and transient MODPATH simulations is the addition of a special binary data file called the "composite budget file", or CBF. The CBF is a random-access file constructed only for transient MODPATH simulations using output from the MODFLOW cell-by-cell budget file, the head output file, and stress package files. The principal advantage of the CBF is to allow MODPATH to efficiently read through transient flow and head data in any order for a series of time steps. Once the CBF has been created, it is possible to do backward tracking for transient simulations.

Problem 2 is an extension of problem 1. A second well is added in layer 1 directly above the well in layer 4. A transient MODFLOW and MODPATH simulation is performed to examine the change in capture area with time for the two wells. All of the MODFLOW and MODPATH data files for problem 2 are listed in Appendix E - **Data Files for Example Problems**.

The MODFLOW simulation is divided into three stress periods. Stress period 1 is 500 years long with one time step. During stress period 1, only one well is present in layer 4 and all stresses are identical to those of the steady state simulation in problem 1. Stress period 2 is 30 years long and consists of eight time steps. During stress period 2, wells are present in layers 1 and 4, each discharging at 80,000 cubic feet per day. Stress period 3 is 1,250 years long and contains one time step. Both wells continue to discharge at 80,000 cubic feet per day during stress period 3. The head distribution from the steady state simulation of problem 1 is used as the initial condition for the transient simulation. An examination of the MODFLOW budget shows that, for practical purposes, the system has attained a new hydraulic steady state 30 years after the onset of pumping in layer 1 (the end of stress period 2). As a result, the transient simulation of problem 2 includes a period of transient flow (stress period 2) bracketed by two periods of steady state flow (stress periods 1 and 3). This approach may seem unusual and unnecessary. Indeed, it would be unnecessary if the sole purpose of the MODFLOW simulation was to perform an hydraulic analysis, or if particle paths were to be computed only over a short interval of time within the period of transient flow. However, in many cases, the objective is to examine capture areas in a transient flow system. To accomplish that, it is necessary to track particles completely through the system. Most of the water discharging to the well in layer 4 after ten years of pumping in layer 1 will have entered the flow system as recharge long before pumping began in layer 1. Adding a long steady-state time step at the beginning of the transient flow period allows MODPATH to track particles completely through the system. Adding a long steady-state time step at the end of the transient flow period is necessary because capture areas respond slowing to changes in the flow system and evolve over a much longer period of time than is required to reach a new

hydraulic steady state.

Figure 6-11 shows the main data file for the transient MODPATH simulation. The number of stress periods (NPER) has been set equal to 3 and stress period data has been added at the end of the file.

```
27 27 5 1 1 3 0 999.9 1.0E+30 0                    [Item 1: Grid Dimensions and Specifications]
compact                                            [Item 2: Options]
1 0 0 0 0                                          [Item 3: LAYCON]
1 0 0 0 0                                          [Item 4: NCON]
internal 1.0 (free) 0                              [Item 5a: DELR control record]
 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.
internal 1.0 (free) 0                              [Item 5b: DELC control record]
 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.
internal 1.0 (free) 0                              [Item 6a: DELZ control record]
 100. 50. 50. 50. 50.
internal 1.0 (free) 0                              [Item 6b: DELZCB control record]
 20. 0. 0. 0. 0.
220.                                               [Item 6c: ZBL1, bottom elevation of layer 1]
open/close ibound.1 1 (free) 0                     [Item 8: IBOUND control record for layer 1]
constant 1                                         [Item 8: IBOUND control record for layer 2]
constant 1                                         [Item 8: IBOUND control record for layer 3]
open/close ibound.4 1 (free) 0                     [Item 8: IBOUND control record for layer 4]
constant 1                                         [Item 8: IBOUND control record for layer 5]
constant 0.3                                       [Item 9: POROSITY control record for layer 1]
constant 0.3                                       [Item 9: POROSITY control record for cb]
constant 0.3                                       [Item 9: POROSITY control record for layer 2]
constant 0.3                                       [Item 9: POROSITY control record for layer 3]
constant 0.3                                       [Item 9: POROSITY control record for layer 4]
constant 0.3                                       [Item 9: POROSITY control record for layer 5]
0.0                                                [Item 10a: TBEGIN]
  182500.0          1        1.0                   [Item 10b: period 1: 500 years, 1 time step]
   10950.0          8        2.0                   [Item 10b: period 2: 30 years, 8 time steps]
  456250.0          1        1.0                   [Item 10b: period 3: 1250 years, 1 time step]
1 1  3 1                                           [Item 10c: range of time steps in MODFLOW budget file]
```

Figure 6-11. MODPATH main data file for problem 2.

The beginning simulation time for the MODFLOW run is defined by the variable TBEGIN, which is set equal to 0 in this case. Therefore, the well in layer 1 begins pumping at a simulation time of 500 years.

Run 1: Map View Backward Tracking Pathline Analysis for Well 2

The beginning of pumping from the well in layer 1 results in transient flow. The system rapidly approaches a new hydraulic steady state. Even after only one year of pumping the head distribution is very close to the new steady state. However, changes in the well capture areas occur much more slowly than changes in the hydraulic head distribution. The slow response of capture areas occurs because capture areas represent the net effect of ground water moving through the system over a long period of time. The MODPATH analyses in the next few sections will illustrate this point. First, the response of two particle paths computed by tracking backward from the cell in layer 1 that contains well 2 is considered. Pairs of particles will be started at four points in time: the start of pumping in layer 1; after 1 year of pumping; after 3 years of pumping; and, after 10 years of pumping. Because the particles are tracked backward, these lines represent the paths taken by water discharging to the well at those four points in time. In this analysis, a different reference time was used for each of these MODPATH simulations. Because well 2 begins pumping at a simulation time of 500 years, reference times of 500, 501, 503, and 510 years were used in the four simulations.

Response files are listed in Appendix E (**path-t1.rsp** and **plot-t1.rsp**). Results are plotted in **figure 6-12**. Figure 6-12a shows the path of particles discharging to the well at the beginning of pumping. Contours show the head distribution in layer 1. Those pathlines represent water that moved through the system prior to the effect of pumping in layer 1. Figures 6-12b, c, and d show the progressive change in pathlines as the well begins to have more and more effect on particle paths. The sharp bend in pathlines corresponds to the time at which pumping began in layer 1. At 10 years, both pathlines represent water that originated as recharge after the start of pumping. The sharp change in direction of pathlines at 1 and 3 years reflects a rapid adjustment in head distribution relative to the ground-water flow velocity.

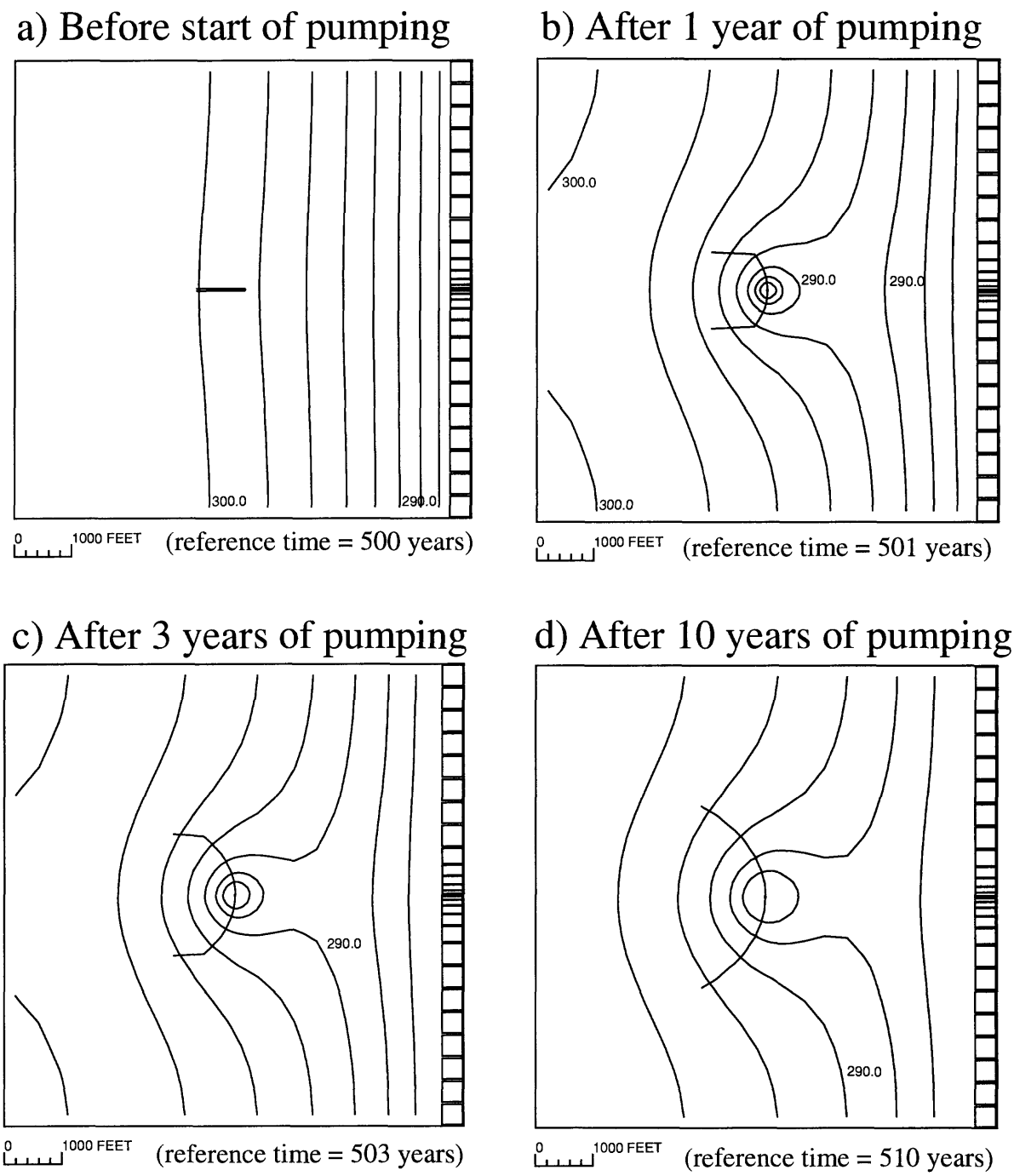


Figure 6-12. Evolution in time of two sample path lines that discharge to the well cell in layer 1.

Run 2: Backward Tracking Endpoint Analysis for Well 2

The evolution of the capture area for well 2 in layer 1 can be examined by placing approximately 200 particles on the faces of the cell containing well 2 and tracking them backward to their recharge locations. Four separate MODPATH simulations were made. These four runs started particles: after 3 years of pumping; after 10 years of pumping; after 30 years of pumping; and, after 1,000 years of pumping (new steady state capture area). Response files are listed in Appendix E (**path-t2.rsp** and **plot-t2.rsp**). Results are shown in **Figure 6-13**.

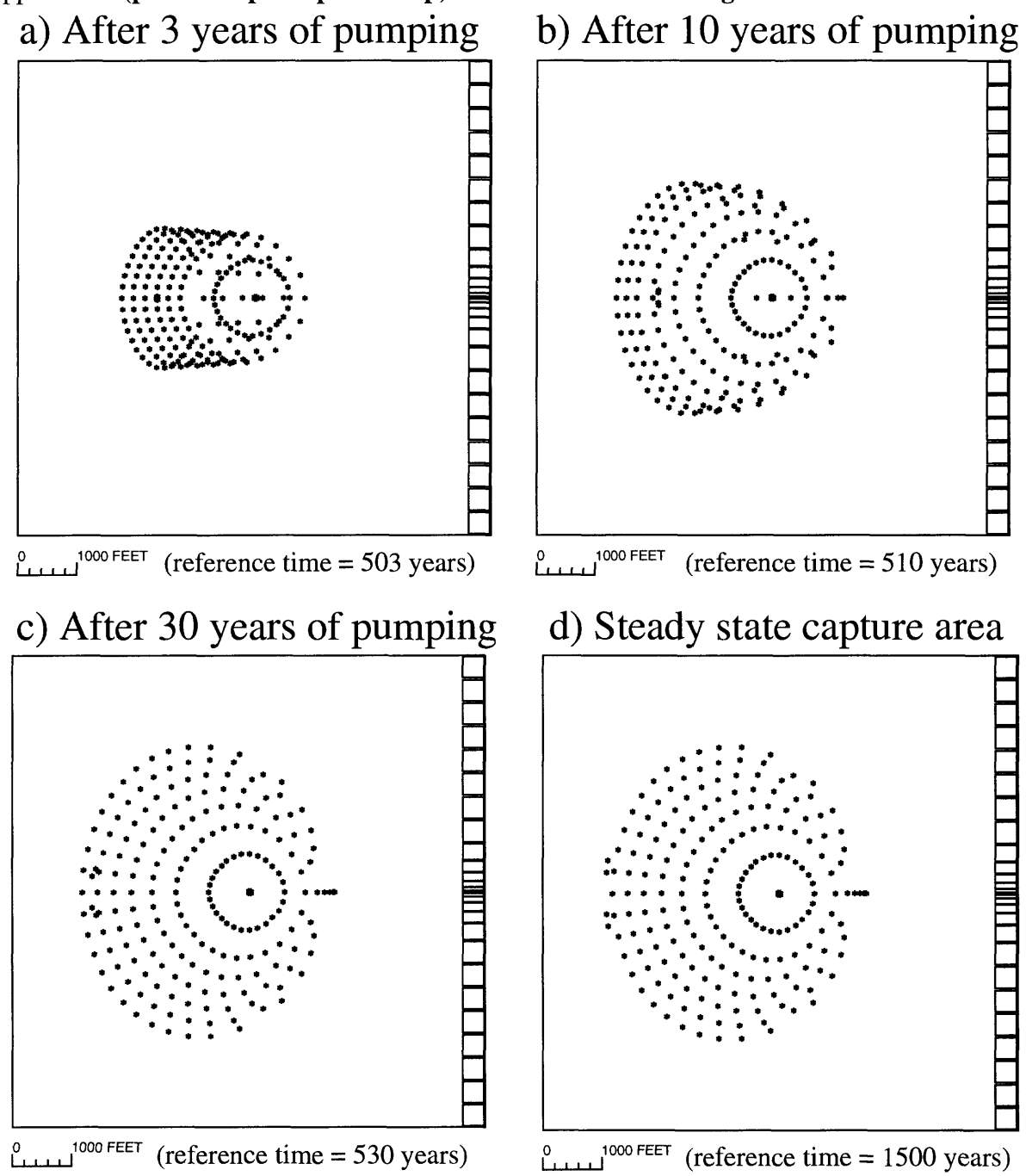


Figure 6-13. Evolution in time of the capture area for the well located in layer 1.

The capture area after 3 years is considerably smaller than the steady-state capture area for well 2 when both wells are pumping. The size and shape of the 3-year capture area reflects that most of the water discharging to well 2 after 3 years of pumping entered as recharge long before the beginning of pumping in layer 1. The new steady-state capture area is nearly fully developed after 30 years.

Run 3: Forward Tracking Endpoint Analysis for Wells 1 and 2

This MODPATH run will delineate the new steady state capture areas for both wells. The capture areas are most easily delineated with forward tracking. Because the system has attained a new hydraulic steady state by the end of stress period 2, a 2x2 array of particles are placed at the water table in all cells in layer 1 at the beginning of stress period 3 (simulation time = 530 years). These particles can be tracked completely through the system because the 1,250 year length of stress period 3 is much longer than the residence time of any of the particles in the system. Response files for this run are listed in Appendix E (**path-t3.rsp** and **plot-t3.rsp**). Results are plotted in **Figure 6-14**. Simulated pumping at well 2 has a major effect on the shape of the capture area for well 1. Because these capture areas reflect steady-state conditions, the total area of captured recharge for well 1 under the new conditions is the same as when only well 1 was pumping, although the location of the capture area is different.

Run 4: Backward Tracking Endpoint Analysis for Well 1

The capture area for well 2 reached steady state size and shape after about 30 years of pumping. A backward tracking endpoint analysis of well 1 was performed to determine the time to reach the steady-state capture area. Approximately 200 particles are placed on the faces of the cell containing well 1. The particles are tracked backward to their recharge locations. Four MODPATH simulations are performed to map out the capture area at four specific points in time: after 10 years of pumping in layer 1; after 30 years of pumping; after 60 years of pumping; and, after 100 years of pumping. Response files are listed in Appendix E (**path-t4.rsp** and **plot-t4.rsp**). Results are plotted in **figure 6-15**. Because well 1 is located deep in a confined aquifer, the effect of pumping in layer 1 on the capture area takes much longer to develop than was the case for well 2. After 30 years of pumping, very little change in the recharge locations of water entering the well has occurred. Only after approximately 100 years does the capture area for well 1 begin to approach the ultimate steady-state capture area.

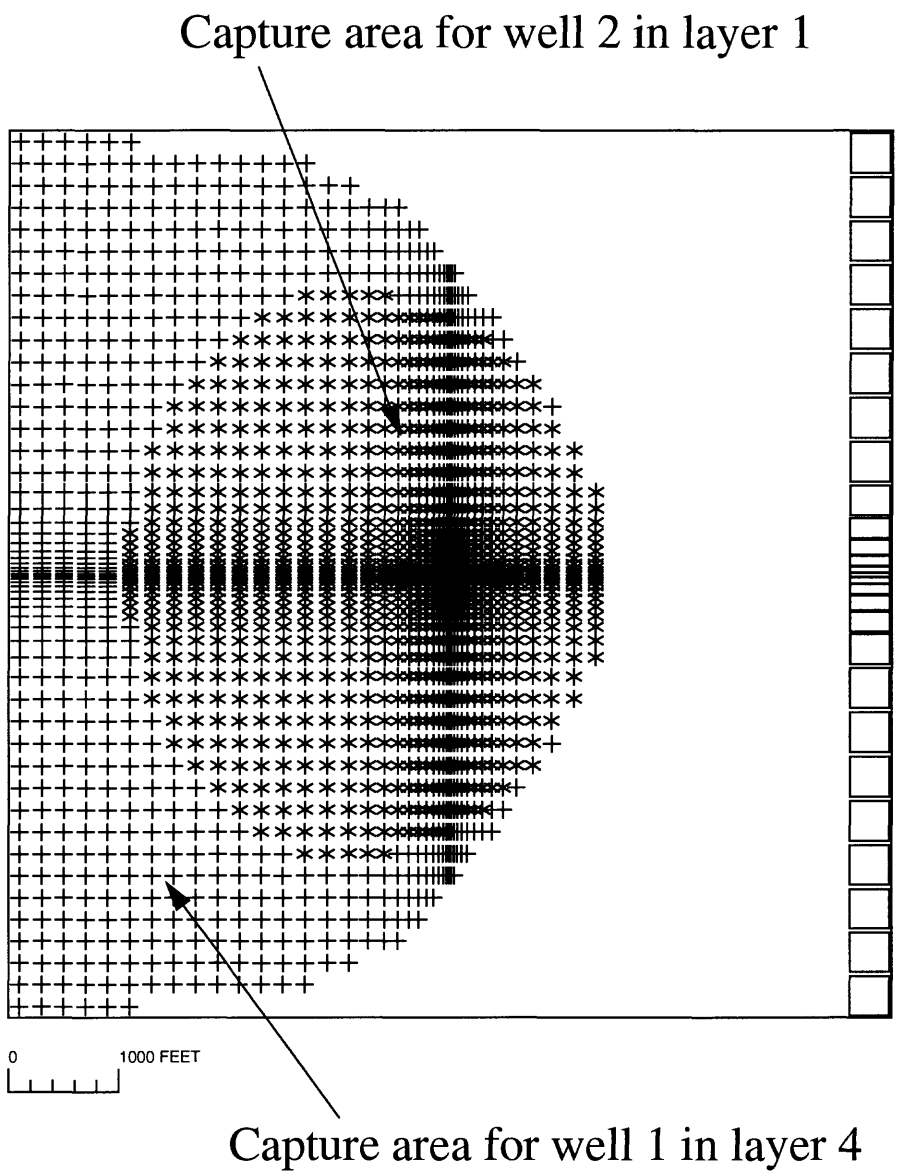


Figure 6-14. Steady-state capture areas for pumping wells in layer 1 and layer 4.

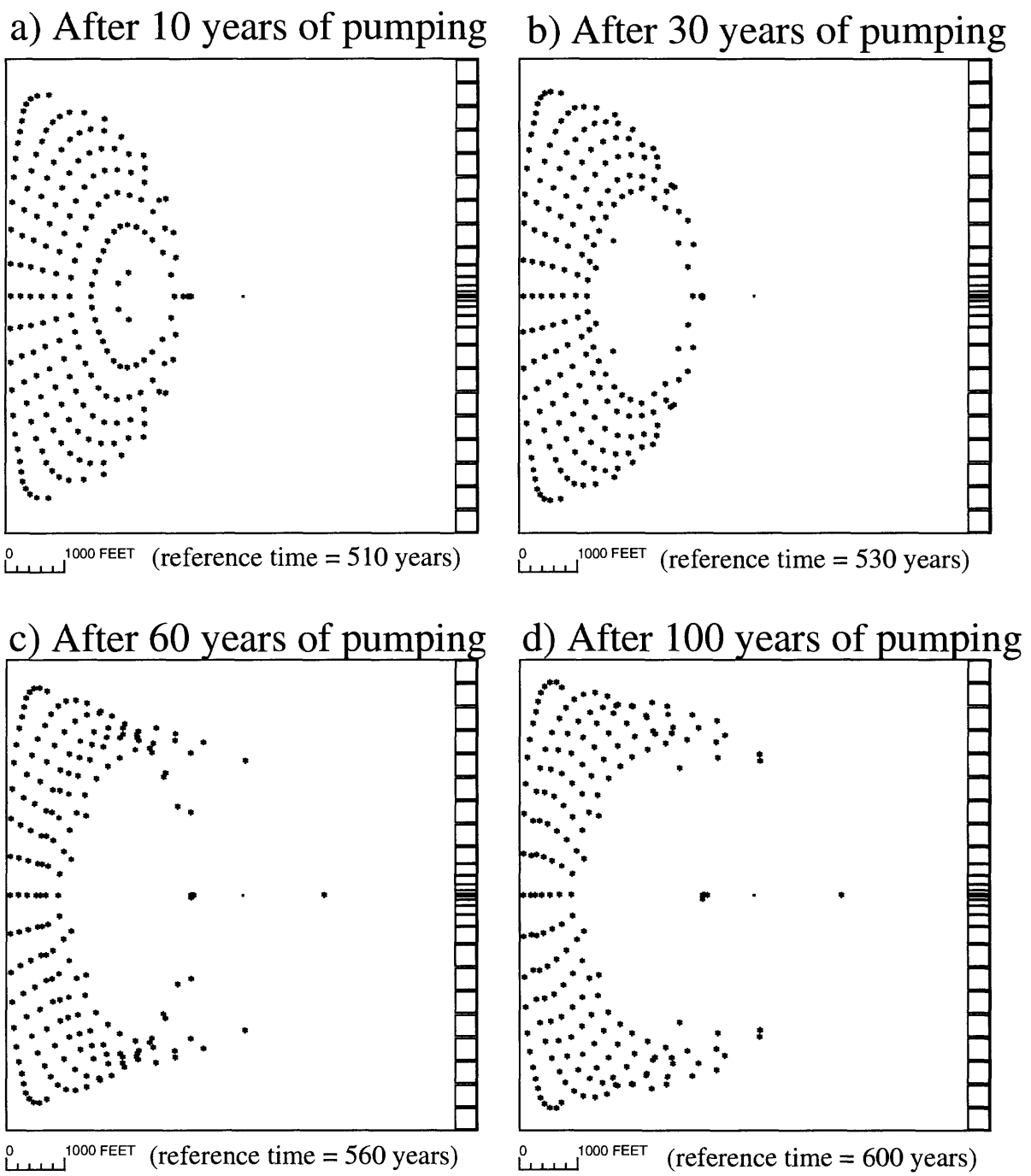


Figure 6-15. Evolution in time of the capture area of well in layer 4 in response to the onset of pumping in layer 1.

Appendix A

Input Files

General Structure of Input Data

This section provides detailed descriptions of the structure of the data files used by MODPATH and MODPATH-PLOT. Input instructions for the MODFLOW stress package data files emphasize those parameters used by MODPATH and MODPATH-PLOT. More detailed descriptions of the MODFLOW data files are presented in McDonald and Harbaugh (1988).

Free-Format Input

Free format is used unless otherwise noted. Free format is similar to FORTRAN's list directed input. With free format, the spacing of values within a record is not fixed. Each value can consist of any number of characters. One or more spaces, or a single comma optionally combined with spaces, must separate adjacent values. A value of 0 must be explicitly represented with 0 and not by one or more spaces because there is no way to detect the difference between a space that represents a 0 value and one that represents a value separator. Two capabilities included in FORTRAN's list directed input are not included. First, null values are not allowed in an input list. Second, a "/" cannot be used to terminate an input record; data values for all input items must be explicitly specified. For character data, MODPATH's free format implementation is less stringent than FORTRAN's list directed input. Fortran requires character data to be delineated by apostrophes. MODPATH does not require apostrophes if the character data item contains no spaces. If the character data item contains spaces, the data item must be enclosed in apostrophes.

Array Input

The real two-dimensional array reader (U2DREL), the integer two-dimensional array reader (U2DINT), and the real one-dimensional array reader (U1DREL) read one array-control record and, optionally, a data array in a format specified on the array-control record. Five alternate structures for the control record are provided. The original fixed-format control records work as documented (McDonald and Harbaugh, 1988). Four free-format versions of the control record have been added in addition to the original fixed-format control record.

FIXED FORMAT CONTROL RECORD FOR REAL AND INTEGER ARRAY READERS:

1.	LOCAT	CNSTNT	FMTIN	IPRN
Format:	I10	F10.0	A20	I10

For U2DINT, CNSTNT is an integer that is read with an I10 format.

FREE-FORMAT CONTROL RECORD FOR REAL ARRAY READERS

Values in bold italics are key words that can be specified as uppercase or lowercase. Each control record is limited to a length of 79 characters.

1. ***CONSTANT*** CNSTNT

All values in the array are set equal to CNSTNT.

2. ***INTERNAL*** CNSTNT FMTIN IPRN

The individual array elements will be read from the same file that contains the control record.

3. ***EXTERNAL*** Nunit CNSTNT FMTIN IPRN

The individual array elements will be read from the file unit number specified by Nunit. The name of the file associated with this file unit is contained in the name file.

4. ***OPEN/CLOSE*** FNAME CNSTNT FMTIN IPRN

The array will be read from the file whose name is specified by FNAME. This file will be opened on unit 99 just prior to reading the array and closed immediately after the array is read. Thus, a file that is read using this control record can contain only a single array. Files opened using the OPEN/CLOSE option should not be included in the name file.

Explanation of Parameters in the Array-Control Records

Nunit-- is the unit for reading the array when the EXTERNAL free-format control record is used.

CNSTNT-- is a real-number constant for U2DREL and U1DREL, and an integer constant for U2DINT. If the array is being defined as a constant, CNSTNT is the constant value. If individual elements of the array are being read, the values are multiplied by CNSTNT after they are read. When CNSTNT is used as a multiplier and if it is specified as 0, it is changed to 1.

FMTIN-- is the format for reading array elements. The format must contain 20 characters or less. The format must either be (1) a standard Fortran format that is enclosed in parentheses, (2)

"(FREE)", which indicates free format, or (3) "(BINARY)", which indicates binary (unformatted) data. When using a free-format control record, the format must be enclosed in apostrophes if it contains one or more blanks or commas. The binary files that can be read by the array readers must have either been created by MODFLOW or by another program capable of generating binary files with the appropriate structure. "(FREE)" and "(BINARY)" can only be specified in free-format control records. Also, "(BINARY)" can only be specified when using U2DREL or U2DINT, and only when the control record is EXTERNAL or OPEN/CLOSE.

LOCAT-- indicates the location of the array values for a fixed-format array control record. If LOCAT=0, all elements are set equal to CNSTNT. If LOCAT>0, it is the unit number for reading formatted records using FMTIN as the format. If LOCAT<0, it is the unit number for binary(unformatted) records, and FMTIN is ignored. Also, when LOCAT is not 0, the array values are multiplied by CNSTNT after they are read.

IPRN-- is a flag indicating that the array being read should be printed and a code for indicating the format that should be used. The format codes are different for each of the three array-reader modules as shown below. IPRN is set to zero when the specified value exceeds those defined. If IPRN is less than zero, the array will not be printed.

IPRN	U2DREL	U2DINT	U1DREL
0	10G11.4	10I11	10G12.5
1	11G10.3	60I1	5G12.5
2	9G13.6	40I2	
3	15F7.1	30I3	
4	15F7.2	25I4	
5	15F7.3	20I5	
6	15F7.4	10I11	
7	20F5.0	20I2	
8	20F5.1	15I4	
9	20F5.2	10I6	
10	20F5.3		
11	20F5.4		
12	10G11.4		
13	10F6.0		
14	10F6.1		
15	10F6.2		
16	10F6.3		
17	10F6.4		
18	10F6.5		

Several examples of free-format control records used to read a real array consisting of 4 rows with 7 values per row are shown on the next page.

```
CONSTANT  5.7      This sets an entire array to the value "5.7".

INTERNAL  1.0  (7F4.0)  3     This reads the array values from the
 1.2 3.7 9.3 4.2 2.2 9.9 1.0  file that contains the control record.
 3.3 4.9 7.3 7.5 8.2 8.7 6.6  Thus, the values immediately follow the
 4.5 5.7 2.2 1.1 1.7 6.7 6.9  control record.
 7.4 3.5 7.8 8.5 7.4 6.8 8.8

EXTERNAL 52  1.0 (7F4.0)  3   This reads the array from the file
opened on unit 52.

EXTERNAL 47  1.0 (BINARY) 3   This reads the array from the binary
file opened on unit 47.

OPEN/CLOSE  test.dat  1.0  (7F4.0)  3   This reads the array from the
file named "test.dat".
```

Name File

The name file contains one line for each input data file. This file provides the information needed to manage the input data files. MODPATH and MODPATH-PLOT prompt the user to specify the name file. The file then is opened internally within the program. Each line consists of 3 items: (1) a character string signifying the type of data file, (2) an integer file unit number, and (3) the file name. The data file is read using free format input. Any valid unit number for a given operating system may be specified. However, MODPATH reserves several numbers in the range 80 to 99 for internal use. Files can be specified in any order. For a typical simulation, the name file might look like:

```
main          11  main.dat
rch           12  recharge.dat
wel           13  well.dat
data          31  ibound.1
budget        50  budget.out
head(binary)  60  head.out
```

The first data item on each line is a string of up to 15 characters that identifies the type of data file. The character string identifier is not case sensitive. MODPATH and MODPATH-PLOT reserve several special character strings to indicate specific data files:

MAIN	=	main data file
RCH	=	recharge package
WEL	=	well package
RIV	=	river package
STR	=	stream package
DRN	=	drain package
GHB	=	general head boundary package
EVT	=	evapotranspiration package
CBF	=	composite budget file
DCF	=	drawing commands file
ENDPOINT	=	endpoint file
PATHLINE	=	pathline file
TIME-SERIES	=	time series file
TIME	=	time data file
LOCATIONS	=	starting locations file
BUDGET	=	binary (unformatted) budget file produced by MODFLOW
HEAD(BINARY)	=	binary (unformatted) head file produced by MODFLOW
HEAD	=	text head file produced by MODFLOW
DRAWDOWN(BINARY)	=	binary (unformatted) drawdown file produced by MODFLOW
DRAWDOWN	=	text drawdown file produced by MODFLOW
CONTOUR-DATA	=	text file containing 2-D data to contour
CONTOUR-LEVEL	=	text file containing contour levels
DATA	=	ancillary text input data files
LIST	=	summary output file
GUA	=	grid unit array file

Ancillary data files usually contain large arrays that are referenced by array control records in other data files. Ancillary data files always must be declared as type DATA. MODPATH's data file requirements are summarized in **Table A-1**.

Table A-1. Summary of Data File Requirements

Description	File Type Keyword	Needed for MODPATH?	Needed for MODPATH PLOT?	Include in Name file?
Main data file	MAIN	yes	yes	yes
Recharge package	RCH	yes[1]	no	yes
Well package	WEL	yes[1]	yes	yes
River package	RIV	yes[1]	yes	yes
Stream package	STR	yes[1]	yes	yes
Drain package	DRN	yes[1]	yes	yes
GHB package	GHB	yes[1]	yes	yes
ET package	EVT	yes[1]	no	yes
Composite budget file	CBF	transient only	transient cross sections only	optional[3]
Drawing commands file	DCF	no	user's option for map view plots	optional[3]
Endpoint file	ENDPOINT	yes	yes	optional[3]
Pathline file	PATHLINE	pathline mode	pathline mode	optional[3]
Time series file	TIME-SERIES	time series mode	time series mode	optional[3]
Time data file	TIME	user's option	no	optional[3]
Starting locations file	LOCATIONS	user's option	no	optional[3]
Binary budget file	BUDGET	yes1	no	yes
Head file (text or binary)	HEAD HEAD(BINARY)	yes[1]	yes[2]	yes
Drawdown file (text or binary)	DRAWDOWN DRAWDOWN(BINARY)	no	user's option for map view plots	yes
Contour data file	CONTOUR-DATA	no	user's option for map view plots	optional[3]
Contour level file	CONTOUR-LEVEL	no	user's option for map view plots	optional[3]
Ancillary text data file	DATA	user's option	user's option	yes
Summary output file	LIST	yes	yes	optional[3]
Grid unit array file	GUA	no	users option	yes

Notes:
1. MODFLOW stress package, budget, and head files are not used by MODPATH in transient simulations that read data directly from a composite budget file.
2. The MODFLOW head file is not required by MODPATH-PLOT for some plot types. See section Plot Types in Chapter 4.
3. Users have the option of specifying these files in the name file or allowing MODPATH to prompt for file names or, in some cases, assign default file names.

Main Data File

The main data file is used by MODPATH and MODPATH-PLOT to define most of the basic elements of the MODPATH simulation. Although most of the data in this file is used by both MODPATH and MODPATH-PLOT, some data is specific to one program or the other. MODPATH and MODPATH-PLOT are designed to read data from a single main data file. Each of the programs ignores those data items that are specific to the other program. The MODPATH main data file is opened in the **Name file** as type MAIN.

The main data file consists of 10 data items:

- **Item 1 -- Grid Dimensions and Specifications**
- **Item 2 -- Options**
- **Item 3 -- Layer Type Codes**
- **Item 4 -- Confining Bed Codes**
- **Item 5 -- Horizontal Grid Spacing**
- **Item 6 -- Model Layer and Confining Layer Thickness**
- **Item 7 -- Top and Bottom Elevations for Model Layers**
- **Item 8 -- Boundary Array**
- **Item 9 -- Porosity**
- **Item 10 -- Stress Period and Time Step Data**

Data items 6 and 7 provide alternate ways of specifying vertical discretization. Each MODPATH simulation uses either data item 6 or data item 7, but not both. Data item 10 is omitted for steady-state simulations. Detailed instructions for these data items are provided on the pages that follow.

Item 1 -- Grid Dimensions and Specifications

```
1.  NCOL NROW NLAY NCBL IGRID NPER MAXSIZ HNOFLO HDRY NPART
```

NCOL-- is the number of columns
NROW-- is the number of rows
NLAY-- is the number of layers

NCBL-- is the number of quasi 3-D confining layers

IGRID-- is the 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.)

NPER-- is the number of stress periods. NPER also indicates whether the simulation is steady-state or transient. To specify a steady-state simulation, set NPER=0.

MAXSIZ-- is the maximum allowed size (in bytes) of the **Composite Budget File**.
If MAXSIZ = 0, the program uses a default value that is set in the MODPATH main program. The default value can be changed by resetting the MAXSIZ value in the MODPATH main program and then recompiling the source code. If a transient MODPATH run will generate a composite budget file larger than MAXSIZ, a warning message is issued and an option is provided to either quit or proceed.

HNOFLO-- is the value specified in MODFLOW to represent head in inactive cells.

HDRY-- is the head assigned by MODFLOW to cells that have gone dry. If the original version of MODFLOW was used with the BCF1 package, HDRY should be set to "1.0E+30". If the BCF2 package was used in the MODFLOW simulation, HDRY must equal the value specified in the MODFLOW simulation.

NPART-- is the maximum number of particles allowed for a MODPATH run. The value of NPART has no effect in MODPATH-PLOT. If NPART is omitted or set equal to 0, MODPATH automatically resets NPART to a default value that is defined in the MODPATH main program. For the standard version of the MODPATH source code, there is no advantage to setting NPART to a value other than 0 because array memory is not allocated dynamically. However, providing the variable NPART makes it relatively easy to take advantage of FORTRAN compilers that allow arrays to be dimensioned dynamically at runtime. The use of dynamic memory allocation requires that MODPATH's main source program be modified as appropriate for each specific FORTRAN compiler.

Item 2 -- Options

```
2.  Options
```

Options-- is a 1-line character record that may contain one or more keyword codes that control options in MODPATH and MODPATH-PLOT. A blank line must be included for this data item even if none of the keyword options is used. Keywords that only affect MODPATH-PLOT are ignored by MODPATH.

XSECTION indicates that the model is a 1-row cross section for which IBOUND and the Grid Unit Array (GUA) should each be read as a single, 2-dimensional array with dimensions of NCOL and NLAY.

COMPACT indicates that MODPATH should generate endpoint, pathline, and time series files as text files using the global node number to indicate cell location. If COMPACT is omitted, the cell location is designated using the row-column-layer grid indices (as in previous versions of MODPATH).

BINARY indicates that the endpoint, pathline, and time series files will be generated by MODPATH in binary form. This keyword also is required by MODPATH-PLOT in order to correctly read binary versions of these files.

METERS indicates to MODPATH-PLOT that distances are in meters. The METERS parameter only affects the label on the distance scale produced by MODPATH-PLOT. It has no affect on computations produced by MODPATH. If the option METERS is not specified, MODPATH assumes that length units are in feet. If a length unit other than feet or meters is used in MODFLOW, the scaling performed by MODPATH-PLOT will not be correct.

WT(OFF) indicates that the water table profile will not be drawn on true cross section plots. If none of the three "WT" options is specified, MODPATH-PLOT draws the water-table profile as a solid line with the same color used for drawing contour lines.

WT(DOTTED) indicates that the water-table profile will be drawn as a dotted line on true cross section plots.

WT(DASHED) indicates that the water-table profile will be drawn as a dotted line on true cross section plots.

Item 3 -- Layer Type Codes

```
3.  LAYCON(NLAY)
```

LAYCON-- is the layer type code.

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

Item 4 -- Confining Bed Codes

*****INCLUDE ITEM 4 ONLY IF NCBL > 0

```
4.  NCON(NLAY)
```

NCON-- is the array containing the confining layer code for each model layer.

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

Item 5 -- Horizontal Grid Spacing

```
5A.  DELR(NCOL)
     FORMAT: 1-D real array reader (U1DREL)

5B.  DELC(NROW)
     FORMAT: 1-D real array reader (U1DREL)
```

DELR-- is the grid spacing along rows (x-direction in MODPATH)
DELC-- is the grid spacing along columns (y-direction in MODPATH)

Recharge Package File

Input to the Recharge Package is read from the file that is opened as type **RCH**. The recharge package data file is required by MODPATH but is not used by MODPATH-PLOT.

FOR EACH SIMULATION

```
1.  NRCHOP  IRCHCB  ITOP
```

FOR EACH STRESS PERIOD

```
2.  INRECH  INIRCH
3.  RECH(NCOL,NROW)
    FORMAT: 2-D real array reader (U2DREL)
4.  IRCH(NCOL,NROW)    If NRCHOP is 2
    FORMAT: 2-D integer array reader (U2DINT)
```

NRCHOP--is the recharge option code.

1 - Recharge is only to the top grid layer.

2 - Vertical distribution of recharge is specified in array IRCH.

3 - Recharge is applied to the highest active cell in each vertical column.

IRCHCB--is a flag and a unit number.

If IRCHCB > 0, unit number for cell-by-cell flow terms.

If IRCHCB ≤ 0, cell-by-cell flow terms will not be printed or recorded.

MODPATH does not use cell-by-cell flow terms for the recharge package. This variable is read in but not used.

ITOP--is a flag indicating whether recharge is assigned to the top face.

If ITOP=0, recharge is treated as an internal source.

If ITOP=1, recharge is assigned as a vertical component of flow to the top face.

INRECH--is the RECH read flag.

If INRECH < 0, recharge fluxes from the preceding stress period are used.

If INRECH ≥ 0, an array of recharge fluxes, RECH (Lt^{-1}), is read.

INIRCH--is the IRCH read flag, which is similar to INRECH.

Well Package File

Input for the Well Package is read from the file that is opened as type **WEL**.

FOR EACH SIMULATION

```
1. MXWELL  IWELCB
```

FOR EACH STRESS PERIOD

```
2. ITMP
3. Layer  Row  Column  Q  IFACE
```

(Item 3 normally consists of one record for each well. If ITMP is negative or zero, item 3 is not read.)

MXWELL--is the maximum number of wells used at any time.

IWELCB--is a flag and a unit number.

If IWELCB > 0, unit number for cell-by-cell flow terms.

If IWELCB = 0, cell-by-cell flow terms will not be printed or recorded.

If IWELCB < 0, well recharge will be printed whenever ICBCFL is set.

MODPATH does not use cell-by-cell flow terms for the well package. This variable is read in but not used.

ITMP--is a flag and a counter.

If ITMP < 0, well data from the last stress period will be reused.

If ITMP ≥ 0, ITMP will be the number of wells active during the current stress period.

IFACE--flag indicating whether well flow is assigned to a face

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

If IFACE=1 through 6, flow term is assigned to the cell face corresponding to the value of IFACE.

If IFACE<0, flow term is apportioned uniformly among all faces perpendicular to the horizontal direction (faces 1-4) that are boundaries with inactive cells.

Evapotranspiration Package File

Input to the Evapotranspiration Package is read from the file that is opened as type **EVT**. MODPATH only reads the first line of this file.

FOR EACH SIMULATION

```
1.  NEVTOP  IEVTCB  ITOP
```

FOR EACH STRESS PERIOD

```
2.  INSURF  INEVTR  INEXDP  INIEVT
3.  SURF(NCOL,NROW)
    FORMAT: 2-D real array reader (U2DREL)
4.  EVTR(NCOL,NROW)
    FORMAT: 2-D real array reader (U2DREL)
5.  EXDP(NCOL,NROW)
    FORMAT: 2-D real array reader (U2DREL)
```

IF THE ET OPTION IS EQUAL TO TWO

```
6.  IEVT(NCOL,NROW)
    FORMAT: 2-D integer array reader (U2DINT)
```

NEVTOP--is the evapotranspiration (ET) option code.
1 - ET is calculated only for cells in the layer 1.
2 - The cell for each vertical column is specified by the user in array IEVT.

IEVTCB--is a flag and a unit number.
If IEVTCB > 0, unit number for cell-by-cell flow terms. The value of IEVTCB must correspond to the file unit number for the file containing the cell-by-cell flow rates for the ET package.

ITOP--is a flag indicating if evapotranspiration is assigned to the top face.
If ITOP=0, ET is treated as an internal sink.
If ITOP=1, ET is assigned as a vertical component of flow to the top face.

INSURF--is the ET surface (SURF) read flag.
If INSURF ≥ 0, an array containing the ET surface elevation will be read.
If INSURF < 0, the ET surface from the preceding stress period will be reused.

INEVTR--is the EVTR read flag, which is similar to INSURF.

INEXDP--is the EXDP read flag, which is similar to INSURF.

INIEVT--is the IEVT read flag, which is similar to INSURF.

River Package File

Input to the River Package is read from the file that is opened as type **RIV**.

FOR EACH SIMULATION

1. MXRIVR IRIVCB

FOR EACH STRESS PERIOD

2. ITMP
3. Layer Row Column Stage Cond Rbot IFACE

(Item 3 normally consists of one record for each river reach. If ITMP is negative or zero, item 3 is not read.)

IRIVCB--is a flag and a unit number.

If IRIVCB > 0, cell-by-cell flow terms will be recorded. The value of IRIVCB must correspond to the file unit number for the file containing the cell-by-cell flow rates for the river package.

ITMP--is a flag and a counter.

If ITMP < 0, river data from the last stress period will be reused.

If ITMP ≥ 0, ITMP will be the number of reaches active during the current stress period.

IFACE--flag indicating whether river leakage is assigned to a face

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

If IFACE=1 through 6, flow term is assigned to the cell face corresponding to the value of IFACE.

If IFACE<0, flow term is apportioned uniformly among all faces perpendicular to the horizontal direction (faces 1-4) that are boundaries with inactive cells.

Stream Package File

Input to the Stream Package (Prudic, 1989) is read from the file that is opened as type **STR**.

FOR EACH SIMULATION

```
1.  MXSTRM  NSS NTRIB NDIV ICALC CONST ISTCB1 ISTCB2
```

FOR EACH STRESS PERIOD

```
2.  ITMP IRDFLG IPTFLG
3.  Layer Row Col Seg Reach Flow Stage Cond Sbot Stop IFACE
```

(Item 3 normally consists of one record for each reach. Records are read in sequential order from upstream to downstream, first by segments, and then by reaches. The downstream ordering and reading of segments and reaches are important, as the order determines the connection of inflows and outflows. If ITMP is negative or zero, items 3-6 will not be read.)

If stream stages for each reach are to be calculated (ICALC>0), then the following data set is read in sequential order of segment and reach.

```
4.  Width Slope Rough
```

If the maximum number of tributaries (NTRIB) that can join a segment is greater than zero, then the following data set is read. One record for each segment is read in sequential order. A record is necessary even for segments that do not have tributaries. In this case a blank record or a record with all zeros is read.

```
5.  Itrib(1) Itrib(2) ... Itrib(NTRIB)
```

If diversions are specified (NDIV>0), then the following data set is read. One record is read for each segment in sequential order. For segments that are not diversions, zeros or blanks are specified for each input item.

```
6.  Iupseg
```

MXSTRM--is the maximum number of stream reaches that can be active during the simulation.
NSS--is the maximum number of segments that can be used during the simulation.
NTRIB--is the maximum number of tributary segments that can join during a simulation.
NDIV--is a flag, which when positive, specifies that diversions from segments are simulated.
ICALC--is a flag, which when positive, specifies that stream stages in reaches are computed.
ITMP--is a flag and counter.

- If ITMP < 0, stream data from last stress period will be reused.
- If ITMP > or = 0, ITMP will be the number of reaches active during the current stress period.

IFACE--flag indicating whether stream leakage is assigned to a face

- If IFACE=0 or IFACE>6, flow term is treated as an internal source/sink.
- If IFACE=1 through 6, flow term is assigned to the cell face corresponding to the value of IFACE.
- If IFACE<0, flow term is apportioned uniformly among all faces perpendicular to the horizontal direction (faces 1-4) that are boundaries with inactive cells.

CONST, ISTCB1, ISTCB2-- these parameters are not used by MODPATH.

Drain Package File

Input to the Drain Package is read from the file that is opened as type **DRN**.

FOR EACH SIMULATION

```
1.  MXDRN  IDRNCB
```

FOR EACH STRESS PERIOD

```
2.  ITMP
3.  Layer  Row  Column  Elevation  Conductance  IFACE
```

(Item 3 normally consists of one record for each drain. If ITMP is negative or zero, item 3 will not be read.)

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

IDRNCB--is a flag and a unit number.

If IDRNCB > 0, unit number for cell-by-cell flow terms. The value of IDRNCB must correspond to the file unit number for the file containing the cell-by-cell flow rates for the drain package.

ITMP--is a flag and a counter.

If ITMP < 0, drain data from the last stress period will be reused.

If ITMP ≥ 0, ITMP will be the number of drains active during the current stress period.

IFACE--flag indicating whether drain flow is assigned to a face

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

If IFACE=1 through 6, flow term is assigned to the cell face corresponding to the value of IFACE.

If IFACE<0, flow term is apportioned uniformly among all faces perpendicular to the horizontal direction (faces 1-4) that are boundaries with inactive cells.

General-Head Boundary Package File

Input to the General-Head Boundary Package is read from the file that is opened as type **GHB**.

FOR EACH SIMULATION

```
1. MXBND  IGHBCB
```

FOR EACH STRESS PERIOD

```
2. ITMP
3. Layer  Row  Column  Head  Conductance  IFACE
```

(Item 3 normally consists of one record for each GHB. If ITMP is negative or zero, item 3 is not read.)

MXBND--is the maximum number of general-head boundary cells at one time.

IGHBCB--is a flag and a unit number.

If IGHBCB > 0, unit number for cell-by-cell flow terms. The value of IGHBCB must correspond to the file unit number for the file containing the cell-by-cell flow rates for the general head boundary package.

ITMP--is a flag and a counter.

If ITMP < 0, GHB data from the preceding stress period will be reused.

If ITMP ≥ 0, ITMP is the number of general-head boundaries during the current stress period.

IFACE--flag indicating whether GHB flow is assigned to a face

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

If IFACE=1 through 6, flow term is assigned to the cell face corresponding to the value of IFACE.

If IFACE<0, flow term is apportioned uniformly among all faces perpendicular to the horizontal direction (faces 1-4) that are boundaries with inactive cells.

Starting Locations File

The starting locations file consists of one line of data for each particle:

```
1.  J  I  K  X  Y  Z  JCODE  ICODE  KCODE  TRELEAS
```

J-- is the column index of the cell containing the particle.
I-- is the row index of the cell containing the particle.
K-- is the layer index of the cell containing the particle. If K=0, K is set equal to the top-most active model layer at the specified row and column location.

X-- is the x-coordinate of the particle.
Y-- is the y-coordinate of the particle.
Z-- is the z-coordinate of the particle.

JCODE-- indicates whether X is a global or local coordinate.
- JCODE=0; X is a local coordinate within the range 0 to 1 in a cell located in column J. The global value of the particle's x-coordinate is computed by MODPATH.
- JCODE=1; X is a global coordinate and the particle is located in a cell in column J. If the value of X lies outside the range of cells in column J, the particle is discarded.
- JCODE=2; X is a global coordinate. The column number, J, that contains X is computed by MODPATH from the specified global value of X and grid spacing data. The value of J read as input is ignored.

ICODE-- indicates whether Y is a global or local coordinate.
- ICODE=0; Y is a local coordinate within the range 0 to 1 in a cell located in row I. The global value of the particle's y-coordinate is computed by MODPATH.
- ICODE=1; Y is a global coordinate and the particle is located in a cell in row I. If the value of Y lies outside the range of cells in row I, the particle is discarded.
- ICODE=2; Y is a global coordinate. The row number, I, that contains Y is computed by MODPATH from the specified global value of Y and grid spacing data. The value of I read as input is ignored.

KCODE-- indicates whether Z is a global or local coordinate.
- KCODE=0; Z is a local coordinate within the range 0 to 1 in a cell located in layer K. The global value of the particle's z-coordinate is computed by MODPATH. If the layer includes an underlying quasi-3D confining layer, starting locations within the confining layer are specified using values between -1 (bottom of the confining layer) to 0 (top of confining layer and bottom of model layer).
- KCODE=1; Z is a global coordinate and the particle is located in a cell in layer K. If the value of Z lies outside the range of cells in layer K, the particle is discarded. This option is only available for true rectangular grids (IGRID=1). If KCODE=1 and IGRID=0, the particle is discarded.
- KCODE=2; Z is a global coordinate. The layer number, K, that contains Z is computed by MODPATH from the specified global value of Z and grid spacing data. The value of K

read as input is ignored. This option is only available for true rectangular grids (IGRID=1). If KCODE=2 and IGRID=0, the particle is discarded.

TRELEAS-- is the release time for the particle. The release time is measured relative to the reference value of simulation time defined by the user (see section **Definitions of Time Concepts used by MODPATH** in chapter 3). Particles released at the reference simulation time have a release time equal to 0. The release time has the same time units as used in the MODFLOW simulation. Release times greater than 0 are only allowed for forward tracking runs. For backward tracking runs, it is assumed that all particles have a release time of zero.

If JCODE, ICODE, KCODE, and TRELEAS are all omitted, they are all set equal to 0 and the particle coordinates are assumed to be specified in terms of local coordinates. This feature allows starting location files from previous versions of MODPATH to be used by the current version of the program. However, if any of these four parameters are non-zero, all of the values must be specified explicitly.

Contour Level File

The contour level file can be used to specify contours to plot in MODPATH-PLOT. The structure of the file is free-format and consists of one contour level per line:

```
1. Clevel
```

Clevel-- value of a contour level.

Any number of contours can be specified in the file, however MODPATH-PLOT has a pre-set limit of 1000 contours. The preset value can be changed by resetting the value of the variable NLVMAX in the MODPATH-PLOT main program. MODPATH-PLOT reads contour levels until it reaches the end of the file.

Grid Unit Array File

Grid unit data can be provided in two forms: (1) as an integer array of grid unit numbers, or (2) as an array of real numbers that MODPATH-PLOT converts to an integer array of grid unit numbers using interval ranges specified by the user. An interactive option is provided to display a legend for grid unit array data. The grid unit array file has the following structure:

```
1. Option
```

Option-- is a keyword code:

ZONES indicates that an integer array of grid unit numbers will be read.

RANGES indicates that a real-number array will be read.

```
2. Title
```

Title-- is a text string that will be used as the legend title when a legend is displayed

```
3. GUarray
   Format: 2-D array reader (U2DINT or U2DREL)
```

GUarray-- is an integer array of grid unit numbers if the ZONES option is in effect, or an array of real numbers if the RANGES option is in effect. For a standard 3-D model, GUarray is read as a series of 2-D arrays, one layer at a time. If the model is a 1-row cross section specified with the XSECTION option, GUarray is read as a single 2-D array for the cross section. When an integer array is entered, a value of 0 designates a grid cell that should not be shaded.

If the RANGES option is in effect, include the following data items that define the ranges used to compute the integer grid unit array used by MODPATH-PLOT:

```
4. GUnumber  MinValue  MaxValue
```

GUnumber-- is the grid unit number (> 0) assigned to values of GUarray that fall in this range.
MinValue-- defines the minimum value of the range.
MaxValue-- defines the maximum value of the range.

Open-ended ranges can be specified using the symbols < and > in place of the minimum and maximum values, respectively. Specifying "<" for the minimum value sets the minimum = -10^{30}. Specifying ">" for the maximum value sets the maximum = 10^{30}. The minimum and maximum values specified for each grid unit number will be used to generate legend entries when a legend is displayed.

Or, if the ZONES option is in effect, include the following data items that define the legend entries that will be displayed for the grid unit data when a legend is displayed:

```
5. GUnumber  LegendText
```

GUnumber-- is the grid unit number (> 0) that corresponds to the legend entry.
LegendText-- text string for the legend entry.

The number of legend entries defined can be less than the total number of zones in the grid unit data.

The color and hatch style associated with each grid unit number is set in the **settings file**.

Drawing Commands File

Miscellaneous graphics and text can be added to map-view plots through the use of a special data file called the Drawing Commands File. This file contains a series of one-line drawing commands that are executed sequentially as they are encountered in the file. A total of 18 drawing commands provide the means of plotting lines, markers, polygons, and text in a variety of styles and colors. Detailed instructions for each command are provided in Appendix C - **Drawing Commands**.

Coordinate System

All coordinates are specified in the same units as used in the flow model computations (feet or meters, in most cases). The coordinate system is defined so that the origin is located at the lower left corner of the grid in map view. The coordinate system is assumed to extend over the entire page or plot surface, including those areas outside the grid region. As an example, for a grid with 30 rows and 50 columns the origin of the coordinate system would be in the lower left corner of row 30, column 1.

Drawing Level

The order in which objects are drawn can have a significant effect on the legibility of complex drawings when plotted on devices such as monitors and printers. MODPATH-PLOT constructs the plot in the following order:

1. writes the title and draws the scale
2. draws the grid
3. draws particle data (lines and points) and stress locations (rivers, etc.)
4. draws contours

The order in which objects specified in the Drawing Commands File are drawn during the construction of the plot can be controlled by specifying one of the following drawing "levels" for each object:

level = 0; the object is not drawn
level = 1; drawn immediately after the title and scale
level = 2; drawn immediately after the grid
level = 3; drawn immediately after particle data and stress locations
level = or > 4; drawn last, after the contours

Objects within the same drawing level are drawn in the order in which they occur in the Drawing Commands File.

Clipping

Map view plots generated by MODPATH-PLOT consist of a grid rectangle that is defined by the dimensions of the grid or sub-grid that is plotted. The plot also includes border areas on all four sides of the grid rectangle. The plot title and scale appear in the bottom border area. The drawing commands provide the option of drawing objects and text anywhere on the page or to "clip" output at the edges of the grid rectangle. If the clipping indicator is on, those parts of the object that extend beyond the grid rectangle will not be drawn. In some cases, annotating a plot with text or lines that fall outside the grid rectangle can be accomplished by turning the clipping indicator off for those items.

Command Syntax

The Drawing Commands file contains a series of one line commands that are denoted by "key words" that appear as the first entry on a line. The format of a command name is a two-letter prefix that describes the type of object affected by the command:

PL = polyline
PF = polygon fill
MK = marker
TX = text

In most cases, the prefix is followed by a "." and a suffix that describes the specific function performed by the command. Most commands contain one or more items of data that follow the command on the same line. All data in the drawing commands file is free-format. The command

```
PL.COLOR 6
```

sets the color of polylines to color number 6.

Comment lines can be added at any point in the file and are denoted by an "@" in column 1. Labeling objects with comment lines makes a file easier to understand and edit.

Some commands are followed by a block of (x-y) data. The following sequence of commands draws a box with its lower left corner at (0, 0) and its upper right corner at (50,100):

```
@ begin a polyline
PL.BEGIN  1 1 0 1
0 0
50 0
50 100
0 100
PL.END
```

The command PL.BEGIN signals the beginning of a block of (x-y) data. In this case, the data block consists of four points that define the x-y coordinates of the four corners of the box. The command PL.END signals the end of the x-y data block. The command PL.BEGIN contains four additional data items. The first item indicates that the box will be drawn at drawing level 1. The second item indicates that the box should be clipped at the edges of the grid rectangle. The third item specifies that the box should not be filled. The final item indicates that a closed polygon should be formed by drawing the line connecting the last data point to the first data point. Note that the data items were labeled using a comment line. The command PL.END does not contain any additional data items.

Time File

This file specifies the values of tracking time at which output is required for the pathline and time series modes. The file has the structure:

```
1. Nvalues  TCfactor
```

Nvalues-- is the number of time values specified in the file.

TCfactor-- is a time units conversion factor. Each time value is multiplied by TCfactor to obtain the time value in the time units used in the MODFLOW simulation.

Data item 2 should be repeated Nvalues times:

```
2. Tvalue
```

Tvalue-- is a time value

Composite Budget File

The Composite Budget File (CBF) is a binary, direct access file constructed by MODPATH for transient flow simulations. The CBF permits MODPATH to retrieve budget and head data quickly and efficiently. Once constructed, transient MODPATH simulations can obtain budget and head data for any time step, in any order, directly from the CBF without the need to reprocess the data from MODFLOW data files.

Binary, direct access files require all data records to be the same length. The length of CBF data records is 4*(NCOL + 1) bytes, where NCOL is the number of columns in the grid. If NCOL is less than 4, the record length is set to 20 bytes. All data in the CBF is numerical with a precision of 4 bytes. The CBF contains two types of records: (1) header records and (2) array data records. Three-dimensional arrays are written as a series of records, with each record corresponding to one row of data. Records are written in the order of increasing rows, and the arrays are stacked by layer, beginning with layer 1.

The structure of the CBF is as follows:

Main Header Record

1. NHLAY NSTPSV

   NHLAY = number of layers for which head is recorded in the CBF
   NSTPSV = total number of time steps recorded in the CBF

Time Step Records

This sequence of records is repeated for each time step:

2. Time Step Header Record
   Number of records in group = 1
   Record Data: KKPER KKSTP DELT TOTSIM NNSTP

   KKPER = stress period
   KKSTP = time step
   DELT = time step length
   TOTSIM = accumulated simulation time (as computed by MODFLOW)
   NNSTP = total number of time steps in current stress period

3. QX(NCOL+1, NROW, NLAY)
   Number of records in group = NLAY*NROW
   Record Data: (QX(J,I,K), J=1,NCOL+1)

   QX = Array of x-face flows
   (The structure of Record Group 3 is illustrated in **figure A-1**.)

4. QY(NCOL, NROW+1, NLAY)
   Number of records in group = NLAY*(NROW + 1)
   Record Data: (QY(J,I,K), J=1,NCOL)

   QY = Array of y-face flows
   (The structure of Record Group 4 is illustrated in **figure A-2.**)

5. QZ(NCOL, NROW, NLAY+1)
   Number of records in group = (NLAY + 1)*NROW
   Record Data: (QZ(J,I,K), J=1,NCOL)

   QZ = Array of z-face flows
   (The structure of Record Group 5 is illustrated in **figure A-3.**)

6. QSS(NCOL, NROW, NLAY)
   Number of records in group = NLAY* NROW
   Record Data: (QSS(J,I,K), J=1,NCOL)

   QSS = Array of internal source/sink flow rates

7. QSTO(NCOL, NROW, NLAY)
   Number of records in group = NLAY*NROW
   Record Data: (QSTO(J,I,K), J=1,NCOL)

   QSTO = Array of storage flow rates

8. IBOUND(NCOL, NROW, NLAY)
   Number of records in group = NLAY*NCOL
   Record Data: (IBOUND(J,I,K), J=1,NCOL)

   IBOUND = Array of boundary codes

9. HEAD(NCOL, NROW, NLAY)
   Number of records in group = NHLAY*NROW
   Record Data: (HEAD(J,I,K), J=1,NCOL) , LAYER

   HEAD = Array of heads
   LAYER = layer number

   Head is not recorded for confined layers (LAYCON=0).

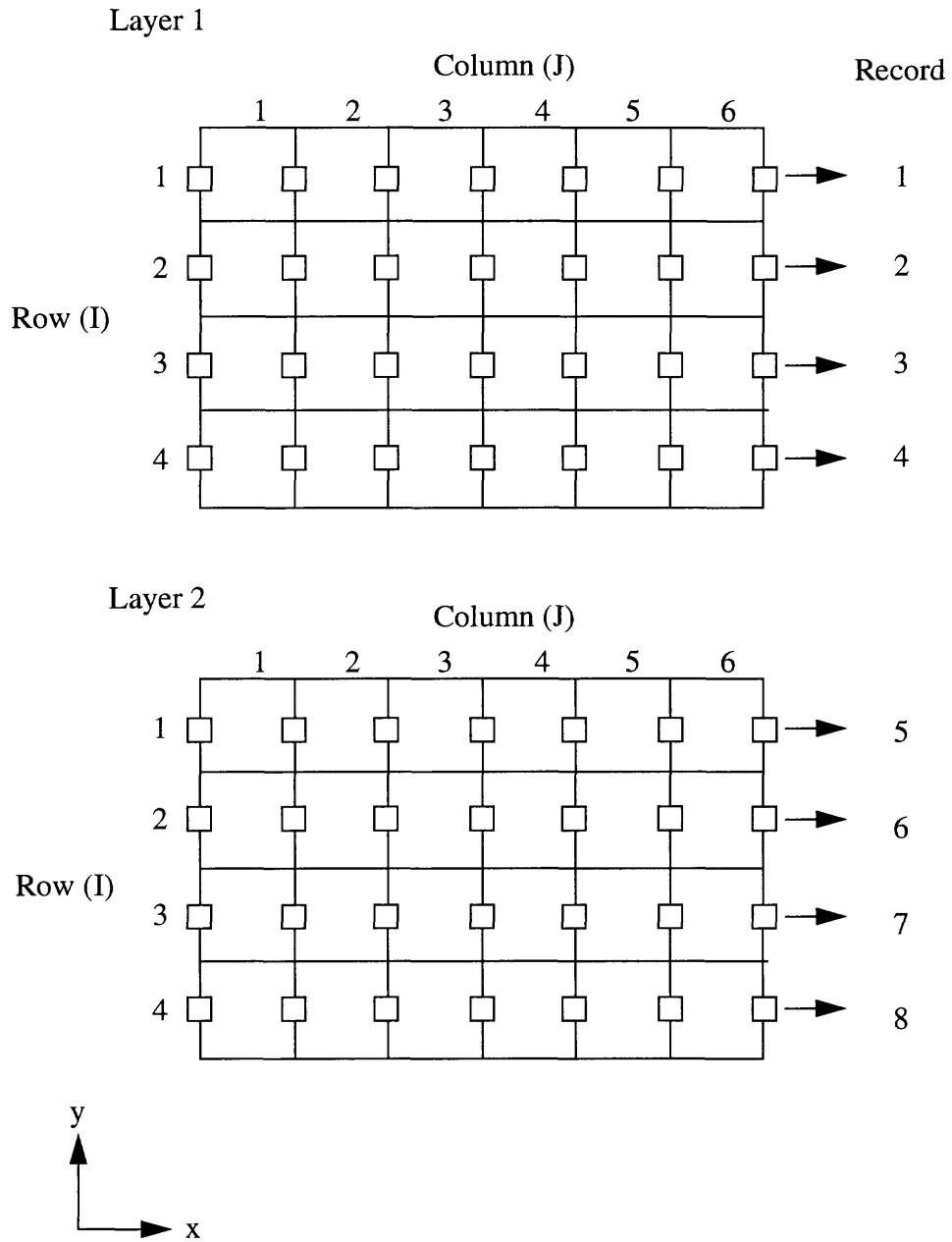


Figure A-1. Record structure for x-face flow rates for a 6-column, 4-row, 2-layer sample grid.

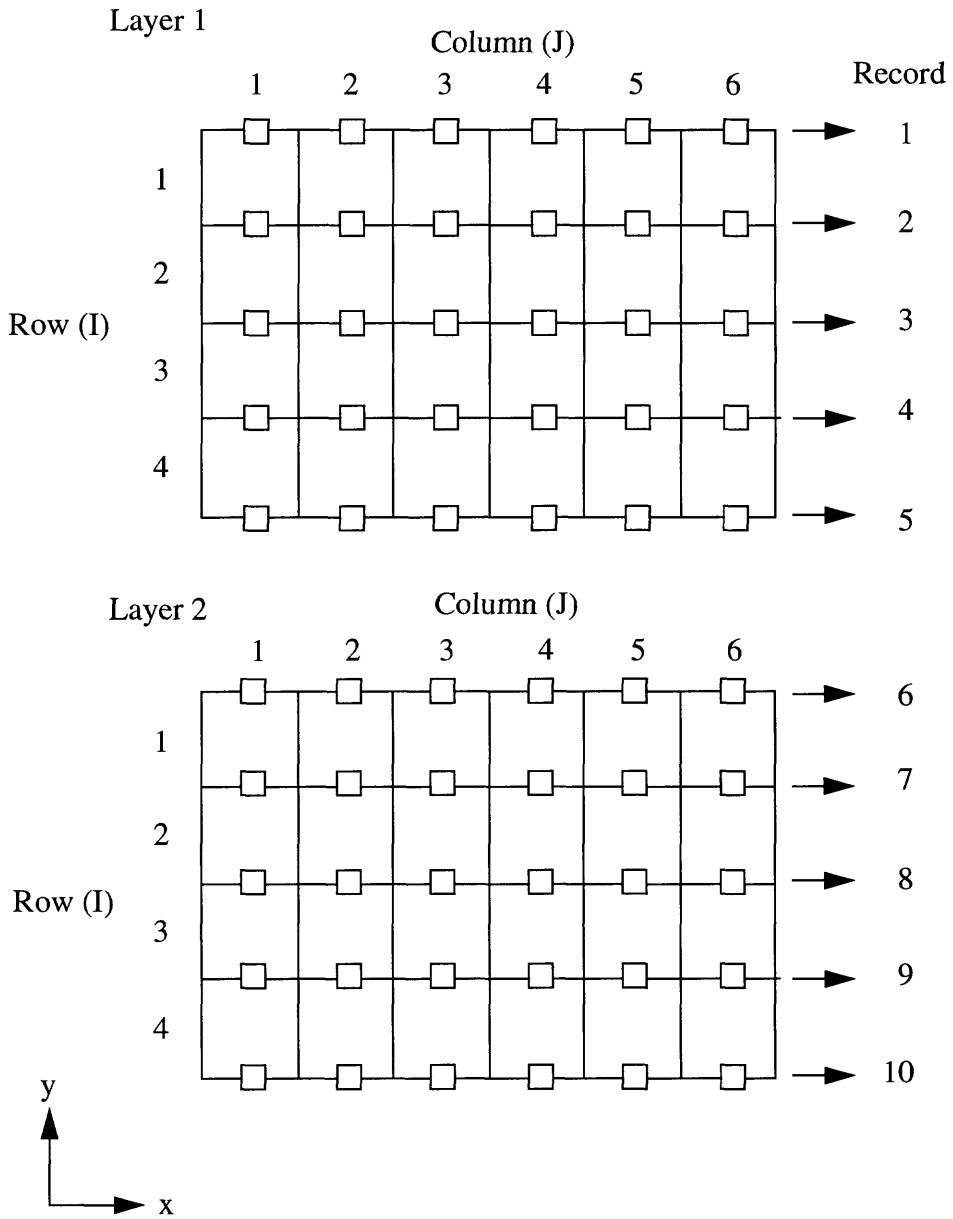


Figure A-2. Record structure for y-face flow rates for a 6-column, 4-row, 2- layer sample grid.

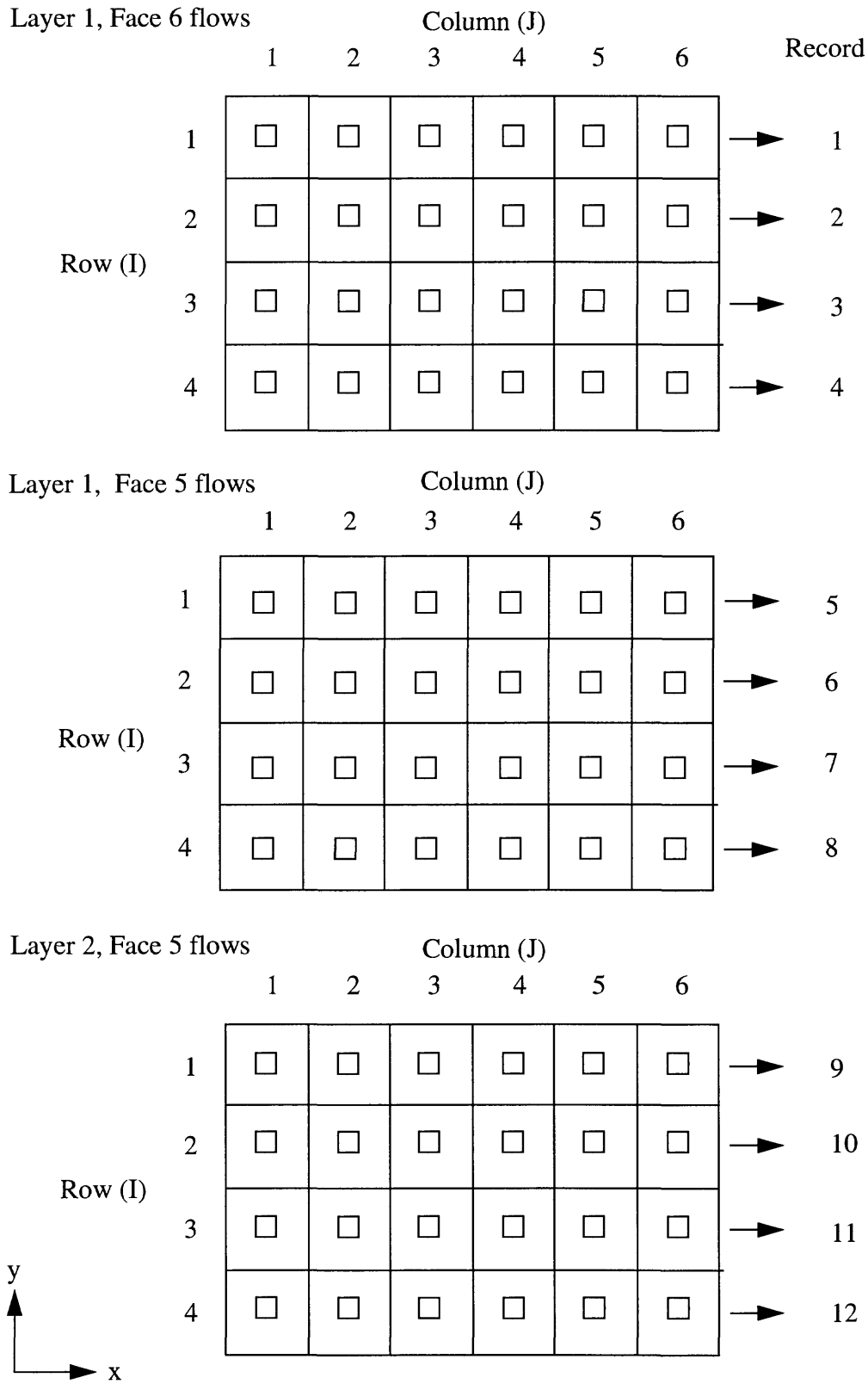


Figure A-3. Record structure for z-face flow rates for a 6-column, 4-row, 2-layer sample grid.

Appendix B

Particle Coordinate Output Files

The particle termination code, IPCODE, is a composite value that represents two parameters, (1) the particle discharge code (IDCODE), and (2) the cumulative time step number corresponding to the particle's final location (NSLAST).

When IPCODE is negative, IDCODE = IPCODE. When IPCODE is positive, IDCODE is given by the first digit of IPCODE. The possible values of IDCODE are:

IDCODE = -2; particle is unreleased.
IDCODE = -1; particle stranded in inactive (dry) cell.
IDCODE = 0; particle remains active.
IDCODE = 1; particle discharged normally.
IDCODE = 2; particle stopped in a specified zone.

The remaining digits of IPCODE give the value of NSLAST. EXAMPLES: IPCODE = 211 means IDCODE = 1 and NSLAST = 21. IPCODE = -1 means IDCODE = -1 and NSLAST is undefined.

MODPATH generates a standard text endpoint file by default.

Compact Text Endpoint File

An alternative, compact text form of the endpoint file can be generated in which grid cell locations are specified in terms of global node numbers rather than by column/row/layer indices. The global node number is defined as: N = (K-1)*NCOL*NROW + (I-1)*NCOL + J. The compact text endpoint file also omits the global z coordinate of the final location.

The compact text file contains the header record of the form:

@ [MODPATH Version 3.00 (V3, Release 1, 9-94) (COMPACT) (TREF= 0.000000E+00)]

Comment lines can be added between the header record and the first particle data record, just as in the **standard text endpoint file**.

The particle data record contains the following 14 items:

1. Zone code for the cell containing the final location of the particle
2. Global node number for the cell containing the final location
3. Global coordinate in the x-direction (J index direction) for the final location
4. Global coordinate in the y-direction (I index direction) for the final location
5. Local coordinate for the z-direction within the grid cell
6. Total tracking time
7. Global coordinate in the x-direction for starting location
8. Global coordinate in the y-direction for starting location
9. Local coordinate in the z-direction within the cell for starting location
10. Global node number for the cell containing starting location
11. Zone code for cell containing starting location
12. Cumulative MODFLOW time step number corresponding to the time of release
13. Particle termination code, IPCODE
14. Release time

MODPATH generates the compact text endpoint file whenever the keyword **COMPACT** is present in data item 2 of the **main data file**.

Binary Endpoint File

A third option is provided to save the endpoint file in binary form in a file with the default name "endpoint.bin". The binary version of the endpoint file contains a header record which includes an 80-character text string describing the MODPATH version number followed by the reference simulation time used in the MODPATH run. The following segment of code shows how to read the header record in a FORTRAN program:

```
CHARACTER*80 TEXT
READ(50) TEXT,TREF
```

The header record is followed by the particle data records. The particle data records are stored by global node number, which is computed as:

$$N = (K-1)*NCOL*NROW + (I-1)*NCOL + J$$

The binary endpoint file also omits the global z coordinate of the final location.

Particle data records contain the following items:

1. Zone code for the cell containing the final location of the particle
2. Global node number for the cell containing the final location
3. Global coordinate in the x-direction (J index direction) for the final location
4. Global coordinate in the y-direction (I index direction) for the final location
5. Local coordinate for the z-direction within the grid cell
6. Total tracking time
7. Global coordinate in the x-direction for starting location
8. Global coordinate in the y-direction for starting location
9. Local coordinate in the z-direction within the cell for starting location
10. Global node number for the cell containing starting location
11. Zone code for cell containing starting location
12. Cumulative MODFLOW time step number corresponding to the time of release
13. Particle termination code, IPCODE
14. Release time

MODPATH generates a binary endpoint file whenever the keyword **BINARY** appears in data item 2 of the **main data file**. The binary form of the endpoint file produces the smallest file but has the disadvantage of not being accessible to standard text editors or viewing programs.

Endpoint File

The ENDPOINT file contains a one-line header followed by one line of data for each particle. Each data record contains information about the initial and final status of a particle. The endpoint file may be saved in a **standard text**, **compact text**, or **binary** structure.

Standard Text Endpoint File

The standard form of the endpoint file generated by MODPATH is a text file with the default name "endpoint". The first line of the endpoint file is a header record indicating the MODPATH version number and the reference simulation time value used in the MODPATH run. The header record has the form:

@ [MODPATH Version 3.00 (V3, Release 1, 9-94) (TREF= 0.000000E+00)]

The header record must be the first line in the file. Users may add any number of comment lines between the header line and the first particle data record. Comment lines must contain the "@" symbol in column 1. Comment lines may not be interspersed with the particle data records.

The header record and any additional comment lines are followed by particle data records that contain the following data items in the order listed:

1. Zone code 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 tracking time
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. Zone code for cell containing starting location
17. Cumulative MODFLOW time step number corresponding to the time of release
18. Particle termination code, IPCODE
19. Release time

Compact Text Pathline File

An alternative, compact text form of the pathline file can be generated in which grid cell locations are specified in terms of global node numbers rather than by column/row/layer indices. The global node number is defined as: N = (K-1)*NCOL*NROW + (I-1)*NCOL + J.

The compact text file contains the header record of the form:

@ [MODPATH Version 3.00 (V3, Release 1, 9-94) (COMPACT) (TREF= 0.000000E+00)]

Comment lines can be added between the header record and the first particle data record, just as in the **standard text pathline file**.

The particle location records contain the following data items:

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 tracking time
7. Global node number of cell containing the point
8. Cumulative MODFLOW time step number

MODPATH generates the compact text pathline file whenever the keyword **COMPACT** is present in data item 2 of the **main data file**.

Pathline File

If pathline output mode is selected, coordinates along the path of each particle are recorded in the pathline file. The file contains information about particle coordinates and accumulated tracking time at every point where a particle enters a new cell or quasi-3d confining layer. In addition, coordinates of intermediate points are recorded whenever the cumulative tracking 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 tracking time by -1 and storing the negative value. The pathline file also contains particle locations at the end of MODFLOW time steps for transient flow simulations. The pathline file consists of a header record followed by a sequence of one-line records, each line containing coordinate time information for one point on a path line. Pathline files can be generated in either (1) **standard text**, (2) **compact text**, or (3) **binary** structure. Standard text and compact text pathline files are given the default name, "pathline". Binary pathline files are given the default name, "pathline.bin".

Standard Text Pathline File

The standard text pathline file is a text file that begins with the header of the form:

@ [MODPATH Version 3.00 (V3, Release 1, 9-94) (TREF= 0.000000E+00)]

Users may add any number of comment lines following the header line and before the particle data records. Comment lines must contain the "@" symbol in column 1. Comment lines may not be interspersed with the particle data records.

The header and comment lines are followed by a sequence of particle location records that contain the following data items in the order specified:

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 tracking 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
10. Cumulative MODFLOW time step number

MODPATH generates a standard text pathline file by default.

Binary Pathline File

A third option is provided to save the pathline file in binary form. The binary version of the pathline file contains a header record which includes an 80-character text string describing the MODPATH version number followed by the reference simulation time used in the MODPATH run. The following segment of code shows how to read the header record in a FORTRAN program:

```
CHARACTER*80 TEXT
READ(50) TEXT,TREF
```

The header record is followed by the particle location data records. The particle location records are stored by global node number, which is computed as:

$$N = (K-1)*NCOL*NROW + (I-1)*NCOL + J$$

Particle location records contain the following items:

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 tracking time
7. Global node number of cell containing the point
8. Cumulative MODFLOW time step number

MODPATH generates a binary pathline file whenever the keyword **BINARY** is present in data item 2 of the **main data file**. The binary form of the pathline file produces the smallest file but cannot be accessed by standard text editors or viewing programs.

Time Series File

If time series output mode is selected, the locations of particles at specified points in time are computed and recorded in a time series file. The locations of all particles are computed for a specified point in time and recorded in sequence. The procedure is repeated at each point in time for which output is specified. The time series file contains a header record followed by a series of particle data records that contain coordinate and time information for each particle. Initial locations are always recorded first in the time series file and are given a time step index number of 0. Time series files can be generated in either (1) **standard text**, (2) **compact text**, or (3) **binary** structure. Standard text and compact text time series files are given the default name, "timesers". Binary time series files are given the default name, "timesers.bin".

Standard Text Time Series File

The standard text time series file is a text file that begins with the header record of the form:

@ [MODPATH Version 3.00 (V3, Release 1, 9-94) (TREF= 0.000000E+00)]

Users may add any number of comment lines following the header line and before the particle data records. Comment lines must contain the "@" symbol in column 1. Comment lines may not be interspersed with the particle data records.

The header record and comment lines are followed by the particle data records, which contain the following data items in the order specified:

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 within the cell
10. Cumulative tracking time
11. Cumulative MODFLOW time step number

MODPATH generates a standard text time series file by default.

Compact Text Time Series File

An alternative, compact text form of the time series file can be generated in which grid cell locations are specified in terms of global node numbers rather than by column/row/layer indices. The global node number is defined as: N = (K-1)*NCOL*NROW + (I-1)*NCOL + J.

The compact text file contains the header record of the form:

@ [MODPATH Version 3.00 (V3, Release 1, 9-94) (COMPACT) (TREF= 0.000000E+00)]

Comment lines can be added between the header record and the first particle data record, just as in the **standard text time series file**.

The header record and comment lines are followed by the particle data records, which contain the following data items in the order specified:

1. Time step index number
2. Particle index number
3. Global node number of cell containing particle
4. Global coordinate in x-direction
5. Global coordinate in y-direction
6. Global coordinate in z-direction
7. Local coordinate in z-direction within the cell
8. Cumulative tracking time
9. Cumulative MODFLOW time step number

MODPATH generates the compact text time series file whenever the keyword **COMPACT** is present in data item 2 of the **main data file**.

Binary Time Series File

A third option is provided to save the time series file in binary form. The binary version of the time series file contains a header record which includes an 80-character text string describing the MODPATH version number followed by the reference simulation time used in the MODPATH run. The following segment of code shows how to read the header record in a FORTRAN program:

```
CHARACTER*80 TEXT
READ(50) TEXT,TREF
```

The header record is followed by the particle location data records. The particle data records are stored by global node number, which is computed as:

$$N = (K-1)*NCOL*NROW + (I-1)*NCOL + J$$

Particle data records contain the following items:

1. Time step index number
2. Particle index number
3. Global node number of cell containing particle
4. Global coordinate in x-direction
5. Global coordinate in y-direction
6. Global coordinate in z-direction
7. Local coordinate in z-direction within the cell
8. Cumulative tracking time
9. Cumulative MODFLOW time step number

MODPATH generates a binary time series file whenever the keyword **BINARY** is present in data item 2 of the **main data file**. The binary form of the time series file produces the smallest file but cannot be accessed by standard text editors or viewing programs.

Importing Old Versions of Particle Coordinate Files

Old versions of the endpoint, pathline, and time series files are automatically detected and read by MODPATH-PLOT. When old particle coordinate files are read, all time step values are automatically set equal to 1 and the particle release time is set equal to 0. The discharge code (IDCODE) for all particles is set to 1 (normally terminated).

Appendix C

DRAWING COMMANDS

Polyline and Polygon Fill Commands

A polyline is a series of connected line segments. A polygon is a two-dimensional shape that is bounded by a closed polyline.

Attribute commands:These commands set characteristics of polylines and polygons. These commands remain in effect until they are reset.

Command	Function
PL.COLOR	Sets the line color.
PL.STYLE	Sets the line style.
PF.COLOR	Sets the fill color for polygons.
PF.DENSE	Sets the density of hatch shading relative to the nominal default hatch line density.
PF.STYLE	Sets polygon shading style.

Drawing commands:

Command	Function
PL.BEGIN	Marks the beginning of a block of (x,y) coordinates that define a polyline or polygon.
PL.END	Marks the end of a block of (x,y) coordinates that define a polyline or polygon. The polyline/polygon is drawn after this command is issued.

PL.COLOR -- Set Polyline Color

Parameter	Description
1	PL.COLOR
2	Color index number

Color index numbers range from 0 to 255. Most devices do not support 256 colors. If the color selected is out of range for the device, color number 1 is selected.

default: 1

PL.STYLE -- Set Polyline Style

Parameter	Description
1	PL.STYLE
2	Line style index number

Available Styles:
1 = solid
2 = dashed
3 = dotted
4 = dash-dot

default: 1

PF.COLOR -- Set Polygon Fill Color

Parameter	Description
1	PF.COLOR
2	Color index number

Color index numbers range from 0 to 255. Most devices do not support 256 colors. If the color selected is out of range for the device, color number 1 is selected.

default: 1

PF.DENSE -- Set Hatch Fill Line Density

Parameter	Description
1	PF.DENSE
2	Scale factor for horizontal and vertical shade lines
3	Scale factor for diagonal shade lines

Scale factors > 1 increase line density. Scale factors < 1 reduce line density. Both scale factors must be set when this command is used.

default: Both scale factors = 1.0

The default shade density is dependent on the output device and can also be redefined in the setup file. Scale factors entered with this command adjust the spacing relative to the existing default shade density.

PF.STYLE -- Set Polygon Fill Style

Parameter	Description
1	PF.STYLE
2	Fill style index number

Available styles:

1 = +45 degree diagonal lines

2 = -45 degree diagonal lines

3 = diagonal cross hatch

4 = horizontal lines

5 = vertical lines

6 = horizontal/vertical cross hatch

7 = solid

default: 1

PL.BEGIN -- Begin Polyline Data Block

Parameter	Description
1	PL.BEGIN
2	Drawing level Available Options: drawing levels 0 through 4
3	Clipping indicator 0 = no clipping 1 = clipping at the boundaries of the grid rectangle
4	Polygon fill indicator 0 = polygon is not filled. 1 = polygon is filled over existing background. 2 = existing background is erased and then polygon is filled. -1 = polygon is filled over existing background, but the outline of the polygon is not drawn. -2 = existing background is erased and then polygon is filled, but the outline of the polygon is not drawn.
5	Close polygon 0 = do not close polyline to form polygon. 1 = close polyline by connecting first and last points. (Polylines are automatically closed when the polygon fill indicator is set to a value other than 0).

This command must be followed by a block of x-y coordinate data. Data blocks consist of one line for each (x,y) location. The coordinates are entered in the order: x, y. Comment lines can be placed within the data block, but no other commands are allowed within the data block. Data blocks must be followed immediately by the command PL.END to terminate the data block.

There is no limit on the number of points that define an open polyline or an unfilled, closed polyline. However, filled polygons are limited to 200 vertices.

PL.END -- End Polyline Data Block

Parameter	Description
1	PL.END

Marker Commands

Markers are symbols used to mark the location of individual points.

Attribute commands:These commands set characteristics of markers. These commands remain in effect until they are reset.

Command	Function
MK.COLOR	Sets the marker color.
MK.SIZE	Sets the marker size relative to the default size.
MK.TYPE	Sets marker type.

Drawing commands:

Command	Function
MK.BEGIN	Marks the beginning of a block of (x,y) coordinates that define a group of markers to be plotted.
MK.END	Marks the end of a block of (x,y) coordinates that define a group of markers. This command causes the group of markers to be plotted.
MK	Specifies the (x,y) coordinates and defines the plotting level and clipping indicator for a single marker. This command causes a single marker to be plotted.

MK.COLOR -- Set Marker Color

Parameter	Description
1	MK.COLOR
2	Color index number

Color index numbers range from 0 to 255. Most devices do not support 256 colors. If the color selected is out of range for the device, color number 1 is selected.

default: 1

MK.SIZE -- Set Marker Size

Parameter	Description
1	MK.SIZE
2	Marker scale factor

A scale factor > 1 increases marker size relative to the default size. A scale factor < 1 reduces marker size relative to the default.

default: 1.0 (The default marker size is device dependent.)

MK.TYPE -- Set Marker Type

Parameter	Description
1	MK.TYPE
2	Marker type index

Available types:

1 = •

2 = +

3 = *

4 = o

5 = x

default: 3

MK.BEGIN -- Begin Marker Data Block

Parameter	Description
1	MK.BEGIN
2	Drawing level Available options: drawing levels 0 through 4.
3	Clipping indicator 0 = no clipping 1 = clipping at the boundaries of the grid rectangle

This command must be followed by a block of x-y coordinate data. Data blocks consist of one line for each (x,y) location. The coordinates are entered in the order: x,y. Comment lines can be placed within the data block, but no other commands are allowed within the data block. Data blocks must be followed immediately by the command MK.END to terminate the data block.

There is no limit on the number of markers that can be plotted using this command.

MK.END -- End Marker Data Block

Parameter	Description
1	MK.END

MK -- Plot Single Marker

Parameter	Description
1	MK
2	x-coordinate of point
3	y-coordinate of point
4	Drawing level Available options: drawing levels 0 through 4.
5	Clipping indicator 0 = no clipping 1 = clipping at the boundaries of the grid rectangle

A single point is plotted after this command is issued. The same result can be obtained with the MK.BEGIN-MK.END sequence of commands by specifying only one data point. The MK command is more compact for plotting single points. Both commands are provided as a matter of convenience and preference for the user.

Text Commands

Generating text output is somewhat more complicated than generating polylines and markers. Basic text attributes such as size and color can be set in the same way as for polylines and markers. Text strings are positioned by specifying the x-y coordinates of a single point called the "text reference point" along with parameters that indicate how the string should be aligned relative to the reference point. Text also can be displayed at any angle by specifying an angle of rotation about the text reference point.

Attribute commands:These commands set characteristics of text strings. These commands remain in effect until they are reset.

Command	Function
TX.ALIGN	Sets text alignment relative to text reference point.
TX.COLOR	Sets the text color.
TX.FILL	Sets text fill color indicator to show what color will be used to fill the background of text rectangles.
TX.SIZE	Sets the text size relative to the default size.

Drawing commands:

Command	Function
TX	Marks the beginning of a text string. This command defines the (x,y) coordinates of the text reference point, drawing level, clipping indicator, angle of rotation, and text fill indicator. The line immediately following this command is assumed to be the text string.

TX.ALIGN -- Set Text Alignment Relative to Text Reference Point

Parameter	Description
1	TX.ALIGN
2	Horizontal alignment
	Available options: 1 = left
	2 = center
	3 = right
	default: 1
3	Vertical alignment
	Available options: 1 = bottom
	2 = center
	3 = top
	default: 1

TX.COLOR -- Set Text Color

Parameter	Description
1	TX.COLOR
2	Color index number
	Color index numbers range from 0 to 255. Most devices do not support 256 colors. If the color selected is out of range for the device, color number 1 is selected.
	default: 1

TX.FILL -- Set Fill Color Flag

Parameter	Description
1	TX.FILL
2	Text fill color
	Available options: value < 0 : text rectangle is not filled with a solid color.
	value > 0 : text rectangle is filled. The fill color number is given by the value of the fill index color flag.
	default: -1

TX.SIZE -- Scale Text Size Relative to Default Size

Parameter	Description
1	TX.SIZE
2	Text size scaling factor Values > 1 increase text size relative to default. Values < 1 decrease text size relative to default. default: 1.0

The default text size is device dependent. The default text size is defined to be equal to that used to label the scale bar.

TX -- Generate a Text String

Parameter	Description
1	TX
2	x-coordinate of text reference point
3	y-coordinate of text reference point
4	Drawing level Available options: drawing levels 0 through 4.
5	Clipping indicator 0 = no clipping 1 = clipping at the boundaries of the grid rectangle
6	Angle of rotation Available options: -180 to +180 degrees The angle of rotation is measured from the horizontal. Positive angles indicate counter-clockwise rotation from the horizontal. Negative angles indicate clockwise rotation from the horizontal. Horizontal text has an angle of rotation = 0. Text strings are rotated around the text reference point as illustrated in **Figure C-1**.
7	Text fill indicator Available options: 0 = text rectangle is not filled. 1 = text rectangle is filled. The value of this flag has no effect if the Text Fill Color indicator is set to a negative value (see TX.FILL)

The TX command must be followed by a line containing the text string. The text string can be up to 80 characters in length beginning in column 1.

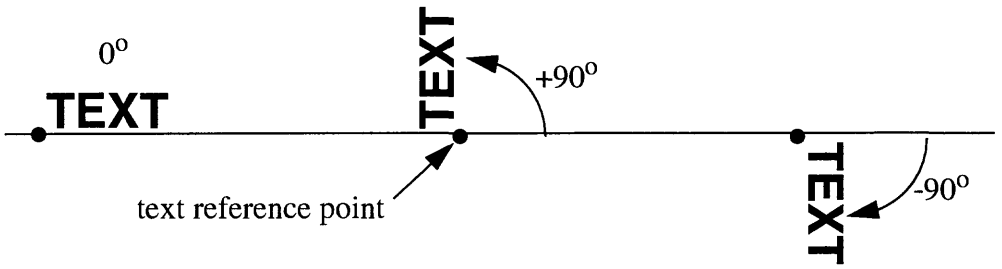


Figure C-1. Definition of angle of rotation about a reference point for rotated text

Appendix D
Installation

Setup Directory

MODPATH and MODPATH-PLOT require several special data files that define values of key parameters and provide general information needed at runtime. These files, which are referred to as "setup files", are:

1. **Device File**: a file required by MODPATH-PLOT that contains information about the graphics devices displayed in the interactive menu.
2. **Settings File**: a file that allows the user to redefine parameters that control characteristics of the graphical output from MODPATH-PLOT. A Settings File can be specified for each graphic output device listed in the Device File. The settings file is defined in the section "Customizing MODPATH-PLOT" (Chapter 4).
3. **Help Files**: files required by MODPATH and MODPATH-PLOT that contain the information displayed by the on-line help system.
4. **GKS File**: a file required by MODPATH-PLOT that contains data for defining GKS parameters and other installation-dependent parameters.

The settings files and the device file allow users to easily customize features that control the appearance of a MODPATH-PLOT run. Users generally should not need to modify the GKS Setup file or the Help files once MODPATH and MODPATH-PLOT have been installed on a computer system.

For convenience, the setup files should be located in a single directory. MODPATH and MODPATH-PLOT allow the user to specify the pathname of the setup directory so that the files do not have to be present in the local directory from which MODPATH and MODPATH-PLOT are executed. The standard versions of MODPATH and MODPATH-PLOT obtain the pathname from a special file, called the "**Search Path File**", that contains the pathname to the MODPATH setup directory. On some computer systems, MODPATH and MODPATH-PLOT can be modified so that the pathname is obtained directly from the operating system through the definition of a system environment variable. In such cases, the search path file is not needed by MODPATH and MODPATH-PLOT.

Search Path File

The user may specify a pathname to a directory that contains setup files to avoid having to copy all of the setup files into each local directory containing MODPATH data files. The pathname is specified in a special data file named <mpsearch> that must be present in the local directory when MODPATH-PLOT is run. The file <mpsearch> contains one line of data specifying the pathname to a directory containing the setup files.

At the beginning of each run, MODPATH-PLOT looks in the local directory for the file <device.dat>. If <device.dat> is not in the local directory, MODPATH-PLOT checks if the file <mpsearch> is in the local directory. If it is, the directory specified in <mpsearch> is searched for <device.dat>. If <device.dat> is not in either directory, an error message is displayed and the run is stopped. If <device.dat> is in both directories, the file in the local directory is utilized by MODPATH-PLOT.

If a settings file is specified as a simple file name, the search process works exactly as with <device.dat>. The local directory is searched for the settings file. If it is present, that file is used and the file <mpsearch> is ignored. If the settings file is not present, the search path in <mpsearch> is then used to try to locate the file. If the file does not exist in either location, an error message is printed and the run is stopped.

If a settings file is specified as a pathname (either absolute or relative), that directory is searched first (not the local directory). If the setting file is not found, then the pathname specified by the user is appended to the search directory pathname listed in <mpsearch> and the resulting directory is searched for the settings file. If no settings file is found, or if the resulting pathname is illegal, an error message is printed and the run is stopped.

Help files must be present either in the setup directory specified in <mpsearch> or in the local directory. MODPATH and MODPATH-PLOT will run even if the help files are not found, but no help will be available when the on-line help system is accessed.

The pathname specified in <mpsearch> on an IBM PC-compatible computer utilizing the DOS operating system might be:

```
c:\modpath\setup\
```

This pathname indicates that the setup files are in a directory named "setup" that is a subdirectory of directory "modpath" on disk drive "c". Note that the pathname must be terminated with the delimiter character, "\", that is used in the pathname. The delimiter character depends on the computer system. UNIX operating systems use a "/".

<mpsearch> can be omitted if the pathname of the setup directory is obtained directly from an operating system environment variable. In other situations, <mpsearch> also can be omitted provided that <device.dat>, <gks.dat>, and any settings files are present in the local directory.

Programming and Portability Issues

MODPATH conforms to the FORTRAN-77 programming standard and, therefore, can be compiled on virtually any computer with a FORTRAN-77 compiler. MODPATH-PLOT also conforms to the FORTRAN-77 standard. MODPATH-PLOT produces graphics using the Graphical Kernel System (GKS). GKS is a set of standardized graphics functions adopted by the American National Standards Institute (ANSI). Libraries of GKS functions are available for most types of computer systems and can be obtained from many sources. Although the use of GKS improves the portability of MODPATH-PLOT between computers, the implementation of graphics is complex, and even standardized graphics systems such as GKS usually require some degree of customization for each GKS installation. In addition to the computer-dependent aspects of GKS that affect MODPATH-PLOT, both programs include two subroutines that may contain compiler-dependent FORTRAN extensions to obtain data from the operating system. These extensions allow MODPATH and MODPATH-PLOT to obtain command-line arguments and a search path from an environment variable.

GKS Modules

An attempt has been made to isolate those aspects of the GKS implementation in MODPATH-PLOT that are most likely to require modifications for each specific GKS implementation. The following modules contain most of the implementation-dependent aspects of the GKS system for MODPATH-PLOT:

UTILGKS -- This module includes subroutines to initiate and close the GKS system.

GKSLEV -- This module includes subroutines that can be customized to accommodate a range of levels of GKS implementation. For example, subroutines in this module can be set up so that MODPATH-PLOT can (1) use a graphic input device and graphic segments (GKS level 2b or 2c), or (2) produce a simple plot with no graphic input device and no segment storage (GKS level 0a). The source code provided for MODPATH-PLOT includes both the GKS-2b and GKS-0a level versions of module GKSLEV.

GKSCUS -- This module includes subroutines that implement extensions of GKS that take advantage of special capabilities of computer hardware or special software capabilities provided by a GKS vendor. For example, some GKS vendors provide a way of resizing the plot window on computer systems that support resizable windows. Many specialized graphics tasks, such as resizing a window, are not part of the GKS standard and therefore must be implemented as an extension to GKS by each vendor.

Tables D-1, **D-2**, and **D-3** contain a summary of the subroutines in each of these modules.

Figures D-1 and **D-2** are highly schematic flow charts that show the basic structure of the GKS implementation within MODPATH-PLOT. In most cases, the modifications needed to implement MODPATH-PLOT on a specific computer will be confined to the subroutines shown in figures D-1 and D-2 and the additional routines present in modules GKSLEV and GKSCUS.

In order to minimize the need to modify the FORTRAN code to implement MODPATH-PLOT for each GKS implementation, many of the installation-dependent parameters are specified in a special **GKS File** named <gks.dat>. The GKS data file is set up during installation and usually should not need to be modified by users. The graphic device menu also must be customized for each GKS installation by setting up the **Device File**, <device.dat>.

Table D-1. Subroutines in Module UTILGKS

Subroutine	Descriptions of UTILGKS Routines
OPGKS	Subroutine OPGKS initiates GKS. OPGKS calls subroutines to (1) set GKS parameters, (2) open the GKS kernel, (3) activate the workstation, (4) set up the workstation-independent segment storage system, and (5) obtain workstation display attributes.
CLGKS	Subroutine CLGKS is called after the plot has been generated. It transfers control to subroutine WAIT, which waits for the user to terminate the plot. When control is returned to CLGKS by WAIT, workstations are deactivated and closed, and the GKS kernel is closed.
SETGKS	Subroutine SETGKS sets key GKS system parameters.
GKSCFG	Subroutine GKSCFG reads GKS parameter settings from the GKS configuration file, <gks.dat>.
SETTXT	Subroutine SETTXT sets text parameters such as font type, character height, and character spacing.

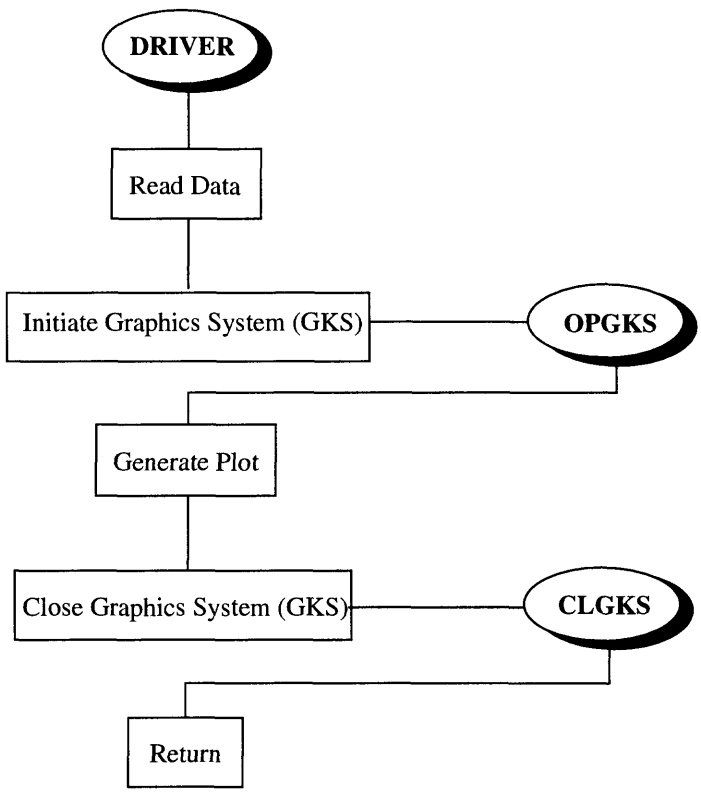


Figure D-1. Schematic flow chart showing basic routines to initiate and close GKS

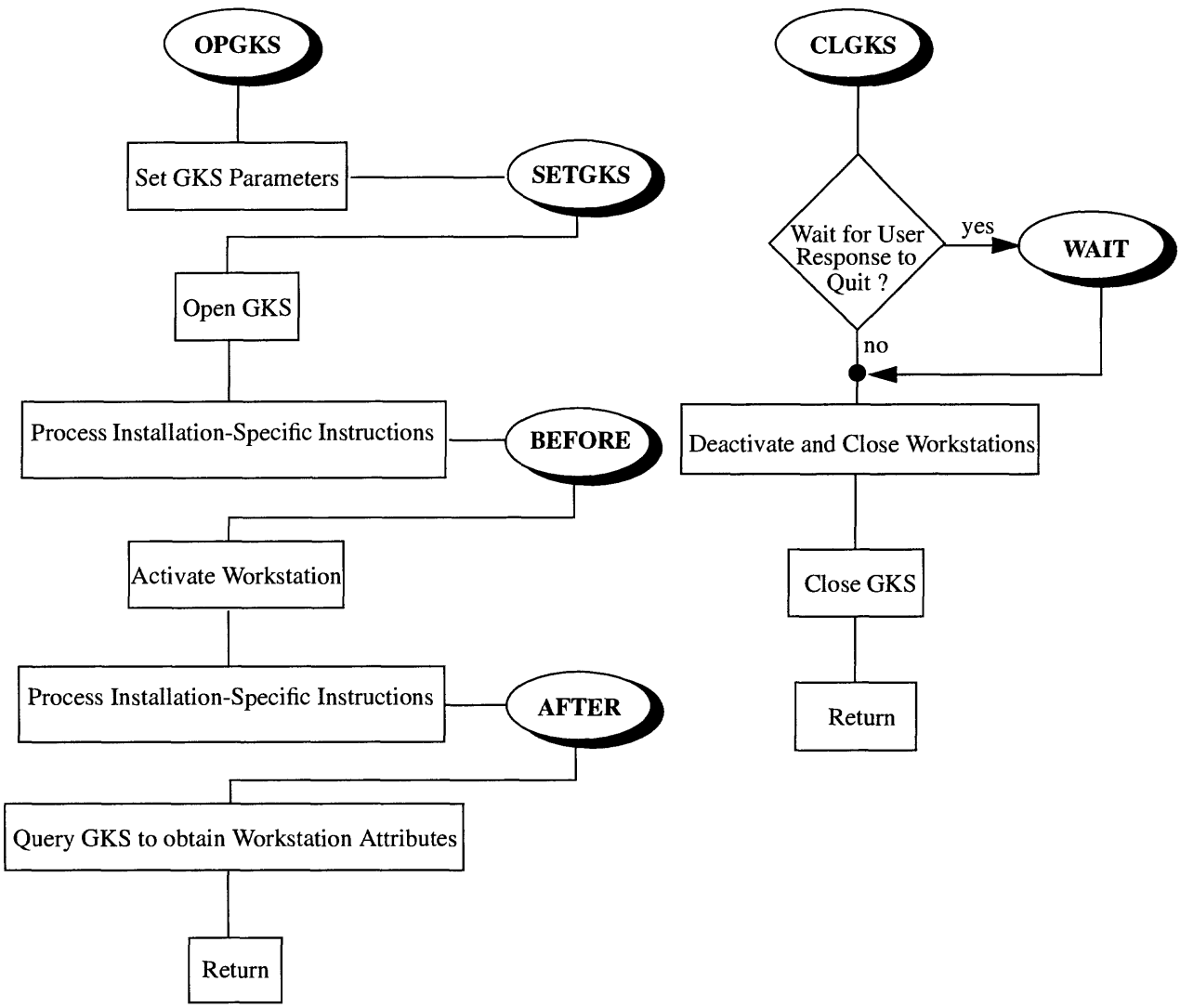


Figure D-2. Schematic flow charts showing structure of subroutines OPGKS and CLGKS.

Table D-2. Subroutines in Module GKSLEV

	Descriptions of GKSLEV2B Routines
WAIT	Subroutine WAIT is called by subroutine CLGKS. For GKS level 2B, this subroutine displays an interactive menu and status line if the graphic device is a display monitor that supports interactive input. The routine monitors the plot and processes graphic input commands. Control is returned to CLGKS when the user enters a "quit" command.
SEGMNT	Subroutine SEGMNT opens and closes graphic segments. Level 2B of GKS allows graphic primitives to be saved in "segments". Programs can manipulate segments to redisplay plots.
THETA	Subroutine THETA computes the angle between the line formed by two consecutive user-defined points and the horizontal. The value of the angle is displayed on the status line.
INSIDE	Subroutine INSIDE determines whether the user has entered an interactive point within a menu box.
GETRC	Subroutine GETRC calculates the model grid cell (row and column) that contains the interactive point entered by the user.
	Descriptions of GKSLEV0A Routines
WAIT	Subroutine WAIT is called by subroutine CLGKS. For GKS level 0A, this subroutine simply waits for the user to press ENTER and then returns control to CLGKS.
SEGMNT	GKS level 0A does not support the use of graphic segments. For the GKS level 0A implementation, subroutine SEGMNT is non-functional; when SEGMNT is called, control is immediately returned to the calling routine.

Table D-3. Subroutines in Module GKSCUS

	Descriptions of GKSCUS Routines
RWSIZE	Subroutine RWSIZE checks to see if the current window dimensions have changed since the last call to RWSIZE. If they have, RWSIZE rescales the plot to fit within the new window dimensions.
SWSIZE	Subroutine SWSIZE sets the initial window dimensions.
QWSIZE	Subroutine QWSIZE obtains the current window dimensions from GKS.
BEFORE	Subroutine BEFORE is a user-defined routine that performs installation-dependent tasks that must be accomplished before the workstation is activated.
AFTER	Subroutine AFTER is a user-defined routine that performs installation-dependent tasks that must be accomplished after the workstation is activated.

Compiler-Dependent FORTRAN Extensions Module

The source code for MODPATH and MODPATH-PLOT includes a module named SYS that contains two subroutines that can be modified with compiler-dependent code to access system environmental variables and command-line arguments. **Table D-4** summarizes the subroutines in module SYS.

Table D-4. Subroutines in Module SYS

	Descriptions of SYS Routines
MPSRCH	Subroutine MPSRCH obtains the text string corresponding to the pathname of the MODPATH setup directory. As currently programmed, this routine contains two optional blocks of code. Only one of these blocks can be active. The unused block must be removed or commented out. Additional blocks of code may be added as necessary. • Block 1 is the default code, which is compiler- and system-independent. This block of code checks to see if a file named "mpsearch" is present in the local directory. If it is, the search path is read directly from file "mpsearch". If "mpsearch is not present, the MODPATH setup files are assumed to be present in the local directory. • Block 2 contains code that is designed to work on an IBM PC-compatible computer using the Lahey EM32 FORTRAN compiler and the Spindrift Utilities package. This code obtains the pathname for MODPATH setup directory by checking the value of an environmental variable named "MODPATH". If the environmental variable does not exist or is set equal to a blank string, the MODPATH setup files are assumed to be present in the local directory.

	Descriptions of SYS Routines
NAMARG	Subroutine NAMARG is designed to obtain a command-line argument from the MODPATH or MODPATH-PLOT command-line. Users may specify a command-line argument to tell the programs whether input will be from an existing response file or from the keyboard. Users enter a filename to specify response file input or the letter "i" to indicate that input will be from the keyboard. If a blank character string is returned as the command-line argument, the programs will prompt the user to specify how data will be entered. As currently programmed, this subroutine contains three optional blocks of code. Only one of these blocks can be active. The unused blocks must be removed or commented out. Additional blocks of code may be added as necessary. • Block 1 is the default code, which is compiler- and system-independent. This block of code simply returns a blank for the string variable containing the command-line argument. Any arguments present on the command line are ignored. • Block 2 obtains the command-line argument for MODPATH and MODPATH-PLOT implemented on Data General Aviion workstations using the Greenhills FORTRAN compiler. • Block 3 obtains the command-line argument for MODPATH and MODPATH-PLOT implemented on an IBM PC-compatible computer using the Lahey EM32 FORTRAN compiler and the Spindrift utilities package.

GKS File

The GKS file specifies values for parameters that are not likely to change once MODPATH-PLOT has been installed on a computer system. The GKS file must be named <gks.dat>. Most of the data specified in the GKS file relates to installation-dependent parameters that are a function of the specific GKS package implemented on the computer system. The style of input is nearly identical to that of the MODPATH-PLOT Settings file. Data is input by a series of format-free command lines that can appear in any order in the file. Default values are supplied by MODPATH-PLOT for most commands. Comment lines can be added by placing an "@" in column 1. In-line comments are marked by an "@" placed within a line. Blank lines are ignored. The following commands are supported:

1. **Font**
2 **Character Spacing**
3. **Bell**
4. **Segments**
5. **Buffer**
6. **Interactive Marker**
7. **WISS (Workstation-Independent Segment Storage)**
8. **Locator Devices**
9. **Device**

Font

This command sets the font number used by the GKS system to write text.

```
FONT : FontNumber
```

FontNumber-- GKS font number
Defaults: FontNumber = 1

Character Spacing

This command specifies the amount of extra space to place between characters. It is expressed as a fraction of the nominal font width.

```
CHARACTER SPACING : Spacing
```

Spacing-- extra space between characters
Defaults: Spacing = 0.0

Bell

An option is provided to ring a bell when the a screen plot is finished.

```
BELL : Status
```

Status-- text string "on" or "off" indicating the status of the bell.
Default: Status = off

Segments

Interactive graphics and re-sizing windows involve redrawing the plot to refresh the screen. GKS requires graphics commands to be stored in "segments" in order to support redrawing. This command allows segment support to be turned on or off. As implemented in MODPATH-PLOT, segment support requires a "level 2" implementation of GKS. Segments should be turned off for systems that have only level 0 or level 1 GKS implementations.

```
SEGMENTS : Status
```

Status-- text string "on" or "off" indicating whether segments will be used.
Default: Status = off

Buffer

The GKS subroutine GOPKS, which initiates the GKS system, requires that the size of a buffer array be specified. The buffer array is used as a scratch work space by some GKS systems.

```
BUFFER : Size
```

Size-- size of the buffer array
Default: Size = 0

Interactive Marker

MODPATH-PLOT provides the option of activating a graphical input cursor that can be moved using either a mouse or the keyboard cursor keys, depending on the specific capabilities of the GKS installation and output device (see section **Interactive Graphics** in chapter 4). The graphic input device can be used to move the cursor and obtain the location of points on the screen. The position of those points can be displayed on the screen with a marker. This data set allows the user to define the marker color, type, and size used to plot the points.

```
INTERACTIVE MARKER : ColorNumber  Type  Scale
```

ColorNumber--is the marker color.

Type-- is the marker type.

Options: 1 = •
2 = +
3 = *
4 = o
5 = x

Scale-- is a scale factor to adjust the size of the marker relative to the default marker size. Values < 1 give marker sizes smaller than the default. Values > 1 give marker sizes larger than the default.

Defaults: ColorNumber = 1
Type = 1
Scale = 1

Workstation-Independent Segment Storage (WISS)

This command specifies the workstation type and connection identifier for the WISS workstation. The WISS workstation is managed internally by the GKS system to manage and manipulate graphic segments.

```
WISS : Type   ID
```

Type-- integer value defining the GKS workstation type

ID-- GKS connection identifier for the WISS workstation

Defaults: None. User must specify the appropriate values for the GKS installation

Locator Devices

MODPATH-PLOT supports up to 2 locator devices that can be used to move a graphics cursor and perform graphic input operations. The most common locator devices supported by most GKS systems are the keyboard arrow keys and a mouse.

```
LOCATOR DEVICES :  Locators   LocatorNumbers(Locators)
```

Locators-- the number of locators supported by the MODPATH-PLOT implementation (0, 1, or 2)

LocatorNumbers-- an array containing the GKS locator numbers (omit if Locators=0).

Defaults: Locators = 0

The "LOCATOR DEVICES" command must be followed by the following lines of data that supply the installation-specific information necessary to implement each locator device:

For each locator device, include the following sequence of lines, as appropriate:

• **Line 1**:

```
DataRecord   Nintegers   Nreals   Nstrings
```

DataRecord-- text string "yes" or "no" indicating if a GKS data record must be created for the device

Nintegers-- number of integer values specified in the data record
Nreals-- number of real values specified in the data record
Nstrings-- number of text strings specified in the data record

• **Line 2**: If DataRecord = yes and Nintegers > 0, include the line:

```
Ia(Nintegers)
```

Ia-- array containing integer values for the data record

• **Line 3**: If DataRecord = yes and Nreals > 0, include the line:

```
Ra(Nreals)
```

Ra-- array containing real values for the data record

• **Line 4**: If DataRecord = yes and Nstrings > 0, include the line:

```
LStr(Nstrings)
```

LStr-- array containing the lengths of all the text strings to be recorded in the data record

• **Nstrings lines**: If DataRecord = yes and Nstrings > 0, include Nstrings lines of the form:

```
Str(i)
```

Str(i)-- one string in the array containing text strings to be recorded in the data record

Device

This command specifies data needed to setup a graphics workstation device. A maximum of 50 devices can be specified in the GKS Setup file.

```
DEVICE : Type   Ncolors  ID   [FileName]  [UnitNumber]
```

Type-- GKS workstation type

Ncolors-- number of foreground colors supported by the device.

ID-- GKS connection identifier

FileName-- name of a graphics output file that will be opened by MODPATH-PLOT. Omit this parameter if graphics output goes directly to a device.

UnitNumber-- the file unit number upon which a graphics output file will be opened if the GKS system requires MODPATH-PLOT to open a file directly. UnitNumber is an optional parameter that should be omitted in certain situations. If FileName is specified and UnitNumber is omitted, MODPATH-PLOT opens the file specified by FileName on unit 90. If FileName is omitted, UnitNumber also must be omitted.

Device File

The device file provides information about the types of graphical output devices displayed on the menu during interactive input. The device file must be named <device.dat>. A maximum of 10 devices can be specified in <device.dat>. The file <device.dat> contains one line of data for each device. Data is free format. Each line of data contains four data items in the following order:

```
Workstation  Kind  Text  File
```

Workstation-- is the GKS workstation type. The GKS workstation type is an integer that depends on the GKS installation.

Kind-- indicates the display capabilities of the output device.

Kind = -1; device is display monitor. Interactive mode is on using locator device number 1 (see section, **GKS File**).

Kind = -2; device is display monitor. Interactive mode is on using locator device number 2 (see section, **GKS File**).

Kind = 0; device is display monitor and interactive graphics mode is off

Kind = 1; device is a hard copy device such as a printer or plotter. Devices that send output to an intermediate plot file, such as a postscript or HPGL file, are also designated by Kind=1.

Kind = 2; device is a CGM or GKS metafile.

Text-- character string containing the text of the menu entry for the device.

File-- name of a settings file that will be used whenever the device is selected. A pathname or a local file name can be specified. Omitting the settings file name means that no settings file will be used for the device.

An example of a typical <device.dat> file would be:

```
1  0  monitor          monitor.set
6  1  printer
7  1  plotter          plotter.set
4  2  'GKS metafile'   gksm.set
```

The GKS workstation numbers in the first column would be obtained from the GKS manual for the specific installation. In this example, three settings files, <monitor.set> and <plotter.set>, and <gksm.set> are used to automatically redefine some of the default parameters whenever a monitor, plotter, or metafile device is selected. Note that in this example no settings file was specified for the monochrome printer. Settings files can be any legal fortran name. The device setup file must have the name <device.dat>.

Appendix E

Data Files for Example Problems

Problem 1 MODFLOW Data Files

Basic Package [file: demo-s.bas]

```
MODPATH SAMPLE PROBLEM

         5         27         27          1          4
 11 12 00 14 00 00 00 18 19 00 00 22
         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: demo-s.bcf]

```
         1         50
 1 0 0 0 0
         0        1.0
        11        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.
        11        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.
```

River Package [file: demo-s.riv]

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

Recharge Package [file: demo-s.rch]

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

Well Package [file: demo-s.wel]

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

Output Control [file: demo-s.oc]

```
     4           4          60           0
     0           1           1           1
     1           0           1           0
```

SIP Package [file: demo.sip]

```
   400           5
    1.       .0001           0      0.0005           1
```

Problem 1 MODPATH Data Files

Main Data File [file: demo-s.mdf]

```
27 27 5   1  1  0  0   999.9  1.0E+30  0
compact
1 0 0 0 0
1 0 0 0 0
internal  1.0  (free) 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.
internal  1.0  (free) 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.
internal  1.0  (free) 0
 100.  50.  50.  50.  50.
internal  1.0  (free) 0
  20.   0.   0.   0.   0.
220.
open/close  ibound.1  1  (free)  0
constant  1
constant  1
open/close  ibound.4  1  (free)  0
constant  1
constant  0.3
constant  0.3
constant  0.3
constant  0.3
constant  0.3
constant  0.3
```

Starting Locations File for Run 3 [file: demo-s3.loc]

```
1 14 1   383 0.5 1.0   2 0 0   0.0
1 14 1   766 0.5 1.0   2 0 0   0.0
1 14 1  1149 0.5 1.0   2 0 0   0.0
1 14 1  1532 0.5 1.0   2 0 0   0.0
1 14 1  1915 0.5 1.0   2 0 0   0.0
1 14 1  2298 0.5 1.0   2 0 0   0.0
1 14 1  2681 0.5 1.0   2 0 0   0.0
1 14 1  3064 0.5 1.0   2 0 0   0.0
1 14 1  3447 0.5 1.0   2 0 0   0.0
1 14 1  3830 0.5 1.0   2 0 0   0.0
1 14 1  4213 0.5 1.0   2 0 0   0.0
1 14 1  4596 0.5 1.0   2 0 0   0.0
1 14 1  4979 0.5 1.0   2 0 0   0.0
1 14 1  5362 0.5 1.0   2 0 0   0.0
1 14 1  5745 0.5 1.0   2 0 0   0.0
1 14 1  6128 0.5 1.0   2 0 0   0.0
1 14 1  6511 0.5 1.0   2 0 0   0.0
1 14 1  6894 0.5 1.0   2 0 0   0.0
1 14 1  7277 0.5 1.0   2 0 0   0.0
1 14 1  7660 0.5 1.0   2 0 0   0.0
```

Name Files

Run 1 [file: demo-s1.nam]

```
main           10  demo-s.mdf
wel            11  demo-s.wel
riv            12  demo-s.riv
rch            13  demo-s.rch
budget         50  demo-s.bud
head(binary)   60  demo-s.hed
endpoint       75  endpoint.s1
```

Run 2 [file: demo-s2.nam]

```
main           10  demo-s.mdf
wel            11  demo-s.wel
riv            12  demo-s.riv
rch            13  demo-s.rch
budget         50  demo-s.bud
head(binary)   60  demo-s.hed
endpoint       75  endpoint.s2
```

Run 3 [file: demo-s3.nam]

```
main           10  demo-s.mdf
wel            11  demo-s.wel
riv            12  demo-s.riv
rch            13  demo-s.rch
budget         50  demo-s.bud
head(binary)   60  demo-s.hed
endpoint       75  endpoint.s3
pathline       76  pathline.s3
```

Run 4 [file: demo-s4.nam]

```
main           10  demo-s.mdf
wel            11  demo-s.wel
riv            12  demo-s.riv
rch            13  demo-s.rch
budget         50  demo-s.bud
head(binary)   60  demo-s.hed
endpoint       75  endpoint.s4
time-series    77  timesers.s4
```

IBOUND Array for Layer 1 [file: ibound.1]

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	3	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

IBOUND Array for Layer 4 [file: ibound.4]

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Problem 1 Response Files

MODPATH Problem 1, Run 1

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------
@  File: path-s1.rsp
@
@  Response file for steady state flow sample problem: run 1
@-------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-s1.nam
* DO YOU WANT TO STOP COMPUTING PATHS AFTER A SPECIFIED LENGTH OF TIME ?
@RESPONSE: HELP LABEL = 2.1.41
n
* SELECT THE OUTPUT MODE:
*      1 = ENDPOINTS
*      2 = PATHLINE
*      3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
1
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*      1 = FROM AN EXISTING DATA FILE
*      2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
2
* DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
@RESPONSE: HELP LABEL = 2.1.44
n
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*      1 = FORWARD IN THE DIRECTION OF FLOW
*      2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
1
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*      1 = PASS THROUGH WEAK SINK CELLS
*      2 = STOP AT WEAK SINK CELLS
*      3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
*   STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.
*
* SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
@RESPONSE: HELP LABEL = 2.1.46
n
```

```
* SELECT AN OPTION FOR RELEASING PARTICLES:
*   1 = SINGLE, INSTANTANEOUS RELEASE
*   2 = MULTIPLE RELEASE TIMES
@RESPONSE: HELP LABEL = 2.1.14A
1
*  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
@RESPONSE: HELP LABEL = 2.1.14
1 27 1 27 1 1
* WHERE SHOULD THE PARTICLES BE LOCATED ?
*   1 = WITHIN A CELL
*   2 = ON ONE OR MORE OF THE CELL FACES
@RESPONSE: HELP LABEL = 2.1.15
2
* WHAT CELL FACES DO YOU WANT TO PLACE PARTICLES ON ?
*   (ENTER ALL THE FACE NUMBERS ON A SINGLE LINE)
@RESPONSE: HELP LABEL = 2.1.47
6
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 6:  NX  NY
@RESPONSE: HELP LABEL = 2.1.22
2 2
*    2916 PARTICLES USED OUT OF 500000 MAXIMUM
* DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
@RESPONSE: HELP LABEL = 2.1.53
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
y
* SPECIFY AN ERROR TOLERANCE (IN PERCENT):
@RESPONSE: HELP LABEL = 3.1.1
0.1
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
y
* ENTER J I K COORDINATES OF CELL:  J   I   K
@RESPONSE: HELP LABEL = 5.7.2
14 14 4
*  DO YOU WANT TO CHECK DATA FOR ANOTHER CELL ?
@RESPONSE: HELP LABEL = 5.7.3
n
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
n
```

MODPATH-PLOT Problem 1, Run 1

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------------
@  File: plot-s1.rsp
@
@  Response file for steady state flow sample problem: run 1
@------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-s1.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Capture area of well as defined by forward tracking from water table.
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
1
* ENTER THE TYPE OF PLOT:
*     1 = PATHLINE PLOT
*     2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*     5 = TIME SERIES
*     6 = GRID ONLY
*     7 = CONTOUR PLOT ONLY
@RESPONSE: HELP LABEL = 2.1.3
2
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*   1 = DRAW ACTIVE GRID BOUNDARY ONLY
*   2 = DRAW FULL GRID BOUNDARY ONLY
*   3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
```

```
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*   1 = DO NOT SHADE
*   2 = SOLID SHADING
*   3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* WHAT LAYER SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
1
* WHAT DATA SHOULD BE PLOTTED ?
*   1 = PLOT ALL DATA BY PROJECTION
*   2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* SELECT AN OPTION FOR SCREENING ENDPOINTS:
*   1 = PLOT ONLY THOSE PARTICLES THAT ARE STILL ACTIVE
*   2 = PLOT ONLY THOSE PARTICLES THAT TERMINATED NORMALLY
*   3 = PLOT BOTH ACTIVE AND NORMALLY-TERMINATED PARTICLES
@RESPONSE: HELP LABEL = 2.1.15A
2
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*      MIN COLUMN, MAX COLUMN, MIN ROW, MAX ROW
*              ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.18
    1   27    1   27
* DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
@RESPONSE: HELP LABEL = 2.1.50
n
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*       (ENTER "0" TO CYCLE THROUGH COLORS)
*       (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
0
* SELECT A CONTOURING OPTION:
*   0 = NO CONTOURS
*   1 = HEAD IN A LAYER
*   2 = DRAWDOWN IN A LAYER
*   3 = OTHER GRIDDED DATA
*   4 = HEAD DIFFERENCE BETWEEN TWO LAYERS
@RESPONSE: HELP LABEL = 6.1.1
0
* SPECIFY THE NAME OF A DRAWING COMMANDS FILE:
* ( <CR> = NO DRAWING COMMANDS FILE )
@RESPONSE: HELP LABEL = 2.1.44

* SELECT AN OPTION FOR SIZING THE PLOT:
*   1 = DRAW PLOT TO AN EXACT MAP SCALE
*   2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*   (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

MODPATH Problem 1, Run 2

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------
@  File: path-s2.rsp
@
@  Response file for steady state flow sample problem: run 2
@------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-s2.nam
* DO YOU WANT TO STOP COMPUTING PATHS AFTER A SPECIFIED LENGTH OF TIME ?
@RESPONSE: HELP LABEL = 2.1.41
n
* SELECT THE OUTPUT MODE:
*      1 = ENDPOINTS
*      2 = PATHLINE
*      3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
1
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*      1 = FROM AN EXISTING DATA FILE
*      2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
2
* DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
@RESPONSE: HELP LABEL = 2.1.44
n
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*      1 = FORWARD IN THE DIRECTION OF FLOW
*      2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
2
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*      1 = PASS THROUGH WEAK SINK CELLS
*      2 = STOP AT WEAK SINK CELLS
*      3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
*   STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.
*
* SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
@RESPONSE: HELP LABEL = 2.1.46
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
@RESPONSE: HELP LABEL = 2.1.14
14 14 14 14 4 4
* WHERE SHOULD THE PARTICLES BE LOCATED ?
*   1 = WITHIN A CELL
*   2 = ON ONE OR MORE OF THE CELL FACES
@RESPONSE: HELP LABEL = 2.1.15
2
* WHAT CELL FACES DO YOU WANT TO PLACE PARTICLES ON ?
*   (ENTER ALL THE FACE NUMBERS ON A SINGLE LINE)
@RESPONSE: HELP LABEL = 2.1.47
1 2 3 4 6
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 1:  NY  NZ
@RESPONSE: HELP LABEL = 2.1.17
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 2:  NY  NZ
@RESPONSE: HELP LABEL = 2.1.18
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 3:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.19
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 4:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.20
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 6:  NX  NY
@RESPONSE: HELP LABEL = 2.1.22
2 2
*     200 PARTICLES USED OUT OF 500000 MAXIMUM
* DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
@RESPONSE: HELP LABEL = 2.1.53
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
y
* SPECIFY AN ERROR TOLERANCE (IN PERCENT):
@RESPONSE: HELP LABEL = 3.1.1
0.1
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
n
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
n
```

MODPATH-PLOT Problem 1, Run 2

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------------
@  File: plot-s2.rsp
@
@  Response file for steady state flow sample problem: run 2
@------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-s2.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Capture area of well as defined by backtracking from the well cell.
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
1
* ENTER THE TYPE OF PLOT:
*     1 = PATHLINE PLOT
*     2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*     5 = TIME SERIES
*     6 = GRID ONLY
*     7 = CONTOUR PLOT ONLY
@RESPONSE: HELP LABEL = 2.1.3
4
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*   1 = DRAW ACTIVE GRID BOUNDARY ONLY
*   2 = DRAW FULL GRID BOUNDARY ONLY
*   3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
```

```
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*    1 = DO NOT SHADE
*    2 = SOLID SHADING
*    3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* WHAT LAYER SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
1
* WHAT DATA SHOULD BE PLOTTED ?
*    1 = PLOT ALL DATA BY PROJECTION
*    2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* SELECT AN OPTION FOR SCREENING ENDPOINTS:
*    1 = PLOT ONLY THOSE PARTICLES THAT ARE STILL ACTIVE
*    2 = PLOT ONLY THOSE PARTICLES THAT TERMINATED NORMALLY
*    3 = PLOT BOTH ACTIVE AND NORMALLY-TERMINATED PARTICLES
@RESPONSE: HELP LABEL = 2.1.15A
2
* PLOT ONLY THOSE PARTICLES WHOSE FINAL LOCATIONS ARE IN A SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.49
n
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*       MIN COLUMN, MAX COLUMN, MIN ROW, MAX ROW
*                ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.18
    1   27    1   27
* DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
@RESPONSE: HELP LABEL = 2.1.50
n
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*        (ENTER "0" TO CYCLE THROUGH COLORS)
*        (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
0
* SELECT A CONTOURING OPTION:
*    0 = NO CONTOURS
*    1 = HEAD IN A LAYER
*    2 = DRAWDOWN IN A LAYER
*    3 = OTHER GRIDDED DATA
*    4 = HEAD DIFFERENCE BETWEEN TWO LAYERS
@RESPONSE: HELP LABEL = 6.1.1
0
* SPECIFY THE NAME OF A DRAWING COMMANDS FILE:
* ( <CR> = NO DRAWING COMMANDS FILE )
@RESPONSE: HELP LABEL = 2.1.44

* SELECT AN OPTION FOR SIZING THE PLOT:
*    1 = DRAW PLOT TO AN EXACT MAP SCALE
*    2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
```

```
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*    (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

MODPATH Problem 1, Run 3

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------------
@  File: path-s3.rsp
@
@  Response file for steady state flow sample problem: run 3
@-------------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-s3.nam
* DO YOU WANT TO STOP COMPUTING PATHS AFTER A SPECIFIED LENGTH OF TIME ?
@RESPONSE: HELP LABEL = 2.1.41
n
* SELECT THE OUTPUT MODE:
*      1 = ENDPOINTS
*      2 = PATHLINE
*      3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
2
* DO YOU WANT TO COMPUTE LOCATIONS AT SPECIFIC POINTS IN TIME ?
@RESPONSE: HELP LABEL = 2.1.43
n
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*      1 = FROM AN EXISTING DATA FILE
*      2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
1
* ENTER NAME OF DATA FILE CONTAINING STARTING LOCATIONS:
@RESPONSE: HELP LABEL = 2.1.39
demo-s3.loc
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*      1 = FORWARD IN THE DIRECTION OF FLOW
*      2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
1
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*      1 = PASS THROUGH WEAK SINK CELLS
*      2 = STOP AT WEAK SINK CELLS
*      3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
y
* SPECIFY AN ERROR TOLERANCE (IN PERCENT):
@RESPONSE: HELP LABEL = 3.1.1
0.1
```

```
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
n
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
y
```

MODPATH-PLOT Problem 1, Run 3

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------------
@  File: plot-s3.rsp
@
@  Response file for steady state flow sample problem: run 3
@------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-s3.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Pathlines along a cross section through the well.
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
3
* SELECT AN ORIENTATION FOR THE CROSS SECTION:
*   1 = REGULAR
*   2 = REVERSED
@RESPONSE: HELP LABEL = 2.1.6A
1
* ENTER THE TYPE OF PLOT:
*     1 = PATHLINE PLOT
*     2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*     5 = TIME SERIES
*     6 = GRID ONLY
@RESPONSE: HELP LABEL = 2.1.3
1
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*   1 = DRAW ACTIVE GRID BOUNDARY ONLY
*   2 = DRAW FULL GRID BOUNDARY ONLY
*   3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
```

```
* SELECT AN OPTION FOR SIZING THE PLOT:
*    1 = DRAW PLOT TO AN EXACT MAP SCALE
*    2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*    (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

```
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*   1 = DO NOT SHADE
*   2 = SOLID SHADING
*   3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* SELECT OPTION FOR SHADING QUASI-3D CONFINING BEDS:
*   1 = DO NOT SHADE
*   2 = SOLID SHADING
*   3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45C
1
* WHAT ROW SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
14
* WHAT DATA SHOULD BE PLOTTED ?
*   1 = PLOT ALL DATA BY PROJECTION
*   2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* WERE THE FLOWLINES GENERATED BY FORWARD OR BACKWARD TRACKING ?
*    1 = FORWARD
*    2 = BACKWARD
@RESPONSE: HELP LABEL = 2.1.14
1
* DO YOU WANT TO SKIP OVER PATH LINES THAT DISCHARGE IN ZONE 1 ?
@RESPONSE: HELP LABEL = 2.1.46
n
* DO YOU WANT TO PLOT POINTS AT SPECIFIED TIME INTERVALS ?
@RESPONSE: HELP LABEL = 2.1.47
n
* DO YOU WANT TO STOP DRAWING PATH LINES AT A SPECIFIED TIME?
@RESPONSE: HELP LABEL = 2.1.48
n
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*      MIN COLUMN, MAX COLUMN, MIN LAYER, MAX LAYER
*              ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.22
    1   27    1    5
* WHAT IS THE VERTICAL EXAGGERATION ?
@RESPONSE: HELP LABEL = 2.1.39
10
* DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
@RESPONSE: HELP LABEL = 2.1.50
n
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*       (ENTER "0" TO CYCLE THROUGH COLORS)
*       (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
0
```

MODPATH Problem 1, Run 4

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------------
@  File: path-s4.rsp
@
@  Response file for steady state flow sample problem: run 4
@-------------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-s4.nam
* DO YOU WANT TO STOP COMPUTING PATHS AFTER A SPECIFIED LENGTH OF TIME ?
@RESPONSE: HELP LABEL = 2.1.41
y
* ENTER: MAXIMUM TRACKING TIME & TIME UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.30
50 365
* SELECT THE OUTPUT MODE:
*      1 = ENDPOINTS
*      2 = PATHLINE
*      3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
3
* HOW SHOULD POINTS IN TIME BE SPECIFIED ?
*      1 = WITH A CONSTANT TIME INTERVAL
*      2 = VALUES OF TIME POINTS ARE READ FROM A FILE
@RESPONSE: HELP LABEL = 2.1.7
1
* ENTER: TIME INTERVAL & TIME UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.34
5 365
* ENTER THE MAXIMUM NUMBER OF TIME POINTS ALLOWED
@RESPONSE: HELP LABEL = 2.1.8
10
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*      1 = FROM AN EXISTING DATA FILE
*      2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
2
* DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
@RESPONSE: HELP LABEL = 2.1.44
n
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*      1 = FORWARD IN THE DIRECTION OF FLOW
*      2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
1
```

```
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*     1 = PASS THROUGH WEAK SINK CELLS
*     2 = STOP AT WEAK SINK CELLS
*     3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
*  STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.
*
* SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
@RESPONSE: HELP LABEL = 2.1.46
n
* SELECT AN OPTION FOR RELEASING PARTICLES:
*   1 = SINGLE, INSTANTANEOUS RELEASE
*   2 = MULTIPLE RELEASE TIMES
@RESPONSE: HELP LABEL = 2.1.14A
2
*  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
@RESPONSE: HELP LABEL = 2.1.14
8 9 14 14 1 1
* WHERE SHOULD THE PARTICLES BE LOCATED ?
*   1 = WITHIN A CELL
*   2 = ON ONE OR MORE OF THE CELL FACES
@RESPONSE: HELP LABEL = 2.1.15
2
* WHAT CELL FACES DO YOU WANT TO PLACE PARTICLES ON ?
*   (ENTER ALL THE FACE NUMBERS ON A SINGLE LINE)
@RESPONSE: HELP LABEL = 2.1.47
6
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 6:  NX  NY
@RESPONSE: HELP LABEL = 2.1.22
16 1
* SPECIFY THE TIME PERIOD FOR RELEASING PARTICLES:
*   INITIAL TIME, FINAL TIME, RELEASE INTERVAL, & TIME CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.53A
0 10 0.5 365
*     672 PARTICLES USED OUT OF 500000 MAXIMUM
* DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
@RESPONSE: HELP LABEL = 2.1.53
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
y
* SPECIFY AN ERROR TOLERANCE (IN PERCENT):
@RESPONSE: HELP LABEL = 3.1.1
0.1
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
n
```

```
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
n
```

MODPATH-PLOT Problem 1, Run 4

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------------
@  File: plot-s4.rsp
@
@  Response file for steady state flow sample problem: run 4
@
@  This response file shows the responses necessary to plot a single time step
@  (time step number 2, which displays the plume after 10 years of releasing
@  particles). You can plot other time steps by substituting the appropriate
@  time step number.
@
@  Another way to plot additional time steps quickly is to execute MODPATH-PLOT
@  and specifiy the name of this file when prompted for the name of a response
@  file. The text string "(?)" has been added to the prompt termination lines
@  for the prompts that ask for the time step number and the title. Whenever
@  MODPATH or MODPATH-PLOT finds the character string, "(?)", in a prompt
@  termination line, that prompt is issued interactively. Otherwise, responses
@  are read directly from the file. In this case, you will be prompted to enter
@  a title and the number of the time step that you want to plot.
@------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-s4.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Plume after 10 years.
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
3
* SELECT AN ORIENTATION FOR THE CROSS SECTION:
*   1 = REGULAR
*   2 = REVERSED
@RESPONSE: HELP LABEL = 2.1.6A
1
```

```
* ENTER THE TYPE OF PLOT:
*      1 = PATHLINE PLOT
*      2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*      3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*      4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*      5 = TIME SERIES
*      6 = GRID ONLY
@RESPONSE: HELP LABEL = 2.1.3
5
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*    1 = DRAW ACTIVE GRID BOUNDARY ONLY
*    2 = DRAW FULL GRID BOUNDARY ONLY
*    3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*    1 = DO NOT SHADE
*    2 = SOLID SHADING
*    3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* SELECT OPTION FOR SHADING QUASI-3D CONFINING BEDS:
*    1 = DO NOT SHADE
*    2 = SOLID SHADING
*    3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45C
1
* WHAT ROW SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
14
* WHAT DATA SHOULD BE PLOTTED ?
*    1 = PLOT ALL DATA BY PROJECTION
*    2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* 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)
@RESPONSE: HELP LABEL = 2.1.16
1
* ENTER THE TIME STEP NUMBERS THAT YOU WANT TO PLOT:
@RESPONSE: HELP LABEL = 2.1.17
2
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*       MIN COLUMN, MAX COLUMN, MIN LAYER, MAX LAYER
*                ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.22
    1   27    1    5
* WHAT IS THE VERTICAL EXAGGERATION ?
@RESPONSE: HELP LABEL = 2.1.39
5
```

```
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*       (ENTER "0" TO CYCLE THROUGH COLORS)
*       (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
2
* SELECT AN OPTION FOR SIZING THE PLOT:
*   1 = DRAW PLOT TO AN EXACT MAP SCALE
*   2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*   (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

Problem 2 MODFLOW Data Files

Basic Package [file: demo-t.bas]

```
MODPATH SAMPLE PROBLEM:
   TRANSIENT FLOW EXAMPLE
         5         27         27          3          4
 11 12 00 14 00 00 00 18 19 00 00 22
         0          0
         0          1
         0          1
         0          1
         0          1
         0          1
     999.9
       -65         1.
       -65         1.
       -65         1.
       -65         1.
       -65         1.
  182500.0          1        1.0   period 1: 500 years, 1 time step
   10950.0          8        2.0   period 2:  30 years, 8 time steps
  456250.0          1        1.0   period 3: 500 years, 1 time step
```

BCF Package [file: demo-t.bcf]

```
         0        50
 1 0 0 0 0
         0       1.0
        11       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.
        11       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       0.2
         0       50.
         0      220.
         0    0.0005
         0    0.0001
         0     1250.
         0      0.01
         0    0.0001
         0     1250.
         0      0.01
         0    0.0001
         0     1250.
         0      0.01
         0    0.0001
         0     1250.
```

River Package [file: demo-t.riv]

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

Recharge Package [file: demo-t.rch]

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

Well Package [file: demo-t.wel]

```
 2          0
 1                                          period 1
 4         14         14    -80000.          0
 2                                          period 2
 4         14         14    -80000.          0
 1         14         14    -80000.          0
-1                                          period 3
```

Output Control [file: demo-t.oc]

```
4          4          60          0
0          1           1          1  p1 s1
1          0           1          0
0          1           1          1  p2 s1
1          0           1          0
0          1           1          1  p2 s2
1          0           1          0
0          1           1          1  p2 s3
1          0           1          0
0          1           1          1  p2 s4
1          0           1          0
0          1           1          1  p2 s5
1          0           1          0
0          1           1          1  p2 s6
1          0           1          0
0          1           1          1  p2 s7
1          0           1          0
0          1           1          1  p2 s8
1          0           1          0
0          1           1          1  p3 s1
1          0           1          0
```

Problem 2 MODPATH Data Files

Main Data File [file: demo-t.mdf]

```
27 27 5   1  1  3  0   999.9  1.0E+30  0
compact
1 0 0 0 0
1 0 0 0 0
internal  1.0  (free) 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.
internal  1.0  (free) 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.
internal  1.0  (free) 0
 100.  50.  50.  50.  50.
internal  1.0  (free) 0
  20.   0.   0.   0.   0.
220.
open/close  ibound.1  1  (free) 0
constant  1
constant  1
open/close  ibound.4  1  (free) 0
constant  1
constant  0.3
constant  0.3
constant  0.3
constant  0.3
constant  0.3
constant  0.3
0.0
  182500.0          1       1.0   period 1: 500 years, 1 time step
   10950.0          8       2.0   period 2:  10 years, 8 time steps
  456250.0          1       1.0   period 3: 500 years, 1 time step
1 1  3 1
```

Problem 2 Response Files

MODPATH Problem 2, Run 1

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------------
@  File: plot-t1.rsp
@
@  Response file for transient flow sample problem: run 1
@
@  This file shows responses for a backward pathline analysis using two
@  particles that originate at the well cell in layer 1 at 1 year after the
@  start of pumping in layer 1. Additional runs can be made by selecting a
@  different reference time for releasing the particles.
@-------------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-t1.nam
* DEFINE A REFERENCE TIME FOR RELEASING PARTICLES ...
*   SELECT AN OPTION:
*      1 = SPECIFY BY ENTERING A STRESS PERIOD AND TIME STEP
*      2 = SPECIFY BY ENTERING A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
2
*   ENTER: REFERENCE TIME  &  TIME UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.31
501 365
* STOP COMPUTING PATHS AT A SPECIFIED VALUE OF TRACKING TIME ?
@RESPONSE: HELP LABEL = 2.1.42
n
* SPECIFY AN OPTION FOR READING HEAD AND FLOW RATE DATA:
*   1 = READ STANDARD MODFLOW UNFORMATTED FILES & GENERATE A
*       COMPOSITE BUDGET FILE
*   2 = READ FROM AN EXISTING COMPOSITE BUDGET FILE
@RESPONSE: HELP LABEL = 2.1.3
1
* SELECT THE OUTPUT MODE:
*     1 = ENDPOINTS
*     2 = PATHLINE
*     3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
2
* DO YOU WANT TO COMPUTE LOCATIONS AT SPECIFIC POINTS IN TIME ?
@RESPONSE: HELP LABEL = 2.1.43
n
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*     1 = FROM AN EXISTING DATA FILE
*     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
2
```

Name Files

Run 1 [file: demo-t1.nam]

```
main           10  demo-t.mdf
wel            11  demo-t.wel
riv            12  demo-t.riv
rch            13  demo-t.rch
budget         50  demo-t.bud
head(binary)   60  demo-t.hed
cbf            70  demo-t.cbf
endpoint       75  endpoint.t1
pathline       76  pathline.t1
```

Run 2 [file: demo-t2.nam]

```
main           10  demo-t.mdf
wel            11  demo-t.wel
riv            12  demo-t.riv
rch            13  demo-t.rch
budget         50  demo-t.bud
head(binary)   60  demo-t.hed
cbf            70  demo-t.cbf
endpoint       75  endpoint.t2
```

Run 3 [file: demo-t3.nam]

```
main           10  demo-t.mdf
wel            11  demo-t.wel
riv            12  demo-t.riv
rch            13  demo-t.rch
budget         50  demo-t.bud
head(binary)   60  demo-t.hed
cbf            70  demo-t.cbf
endpoint       75  endpoint.t3
```

Run 4 [file: demo-t4.nam]

```
main           10  demo-t.mdf
wel            11  demo-t.wel
riv            12  demo-t.riv
rch            13  demo-t.rch
budget         50  demo-t.bud
head(binary)   60  demo-t.hed
cbf            70  demo-t.cbf
endpoint       75  endpoint.t4
```

```
* DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
@RESPONSE: HELP LABEL = 2.1.44
n
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*      1 = FORWARD IN THE DIRECTION OF FLOW
*      2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
2
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*      1 = PASS THROUGH WEAK SINK CELLS
*      2 = STOP AT WEAK SINK CELLS
*      3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
*   STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.
*
* SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
@RESPONSE: HELP LABEL = 2.1.46
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
@RESPONSE: HELP LABEL = 2.1.14
14 14 14 14 1 1
* WHERE SHOULD THE PARTICLES BE LOCATED ?
*    1 = WITHIN A CELL
*    2 = ON ONE OR MORE OF THE CELL FACES
@RESPONSE: HELP LABEL = 2.1.15
2
* WHAT CELL FACES DO YOU WANT TO PLACE PARTICLES ON ?
*    (ENTER ALL THE FACE NUMBERS ON A SINGLE LINE)
@RESPONSE: HELP LABEL = 2.1.47
3 4
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 3:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.19
1 1
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 4:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.20
1 1
*        2 PARTICLES USED OUT OF 500000 MAXIMUM
* DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
@RESPONSE: HELP LABEL = 2.1.53
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
y
* SPECIFY AN ERROR TOLERANCE (IN PERCENT):
@RESPONSE: HELP LABEL = 3.1.1
0.1
```

```
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
n
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
y
```

MODPATH-PLOT Problem 2, Run 1

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------------
@  File: plot-t1.rsp
@
@  Response file for transient flow sample problem: run 1
@-------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-t1.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Backward tracked pathlines originating at layer 1 well after 1 year of pumping
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* A TIME STEP MUST BE SPECIFIED FOR COMPUTING THE ACTIVE GRID AND CONTOURS.
* SELECT AN OPTION FOR SPECIFYING THE TIME STEP:
*   1 = ENTER STRESS PERIOD AND TIME STEP DIRECTLY
*   2 = ENTER A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
2
* ENTER: SIMULATION TIME AND UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.2B
501 365
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
1
* ENTER THE TYPE OF PLOT:
*     1 = PATHLINE PLOT
*     2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*     5 = TIME SERIES
*     6 = GRID ONLY
*     7 = CONTOUR PLOT ONLY
@RESPONSE: HELP LABEL = 2.1.3
1
```

```
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*    1 = DRAW ACTIVE GRID BOUNDARY ONLY
*    2 = DRAW FULL GRID BOUNDARY ONLY
*    3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*    1 = DO NOT SHADE
*    2 = SOLID SHADING
*    3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* WHAT LAYER SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
1
* WHAT DATA SHOULD BE PLOTTED ?
*    1 = PLOT ALL DATA BY PROJECTION
*    2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* WERE THE FLOWLINES GENERATED BY FORWARD OR BACKWARD TRACKING ?
*     1 = FORWARD
*     2 = BACKWARD
@RESPONSE: HELP LABEL = 2.1.14
2
* DO YOU WANT TO PLOT POINTS AT SPECIFIED TIME INTERVALS ?
@RESPONSE: HELP LABEL = 2.1.47
n
* DO YOU WANT TO STOP DRAWING PATH LINES AT A SPECIFIED TIME?
@RESPONSE: HELP LABEL = 2.1.48
n
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*      MIN COLUMN, MAX COLUMN, MIN ROW, MAX ROW
*               ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.18
    1   27    1   27
* DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
@RESPONSE: HELP LABEL = 2.1.50
n
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*       (ENTER "0" TO CYCLE THROUGH COLORS)
*       (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
0
* SELECT A CONTOURING OPTION:
*    0 = NO CONTOURS
*    1 = HEAD IN A LAYER
*    2 = DRAWDOWN IN A LAYER
*    3 = OTHER GRIDDED DATA
*    4 = HEAD DIFFERENCE BETWEEN TWO LAYERS
@RESPONSE: HELP LABEL = 6.1.1
1
```

```
* WHAT LAYER DO YOU WANT TO CONTOUR ?
*     [ENTER TWO LAYER NUMBERS IF YOU ARE CONTOURING HEAD DIFFERENCES]
@RESPONSE: HELP LABEL = 6.1.2
1
* LABEL CONTOURS ?
@RESPONSE: HELP LABEL = 6.1.12
Y
* SELECT AN OPTION FOR GENERATING CONTOURS:
*   1 = GENERATE REGULARLY-SPACED CONTOURS AUTOMATICALLY
*   2 = READ CONTOUR LEVEL VALUES FROM A FILE
@RESPONSE: HELP LABEL = 6.1.8A
1
* ENTER: REFERENCE CONTOUR, CONTOUR INTERVAL, LABELING INTERVAL
@RESPONSE: HELP LABEL = 6.1.9
280 2 10
* DEFINE THE RANGE OF DATA TO CONTOUR --
*   ENTER: MINIMUM & MAXIMUM VALUES
*     (ENTER A BLANK LINE TO DRAW CONTOURS OVER THE FULL RANGE)
@RESPONSE: HELP LABEL = 6.1.6
285 300
* SPECIFY THE NAME OF A DRAWING COMMANDS FILE:
* ( <CR> = NO DRAWING COMMANDS FILE )
@RESPONSE: HELP LABEL = 2.1.44

* SELECT AN OPTION FOR SIZING THE PLOT:
*   1 = DRAW PLOT TO AN EXACT MAP SCALE
*   2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*   (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

MODPATH Problem 2, Run 2

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------------
@  File: path-t2.rsp
@
@  Response file for transient flow sample problem: run 2
@-------------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-t2.nam
* DEFINE A REFERENCE TIME FOR RELEASING PARTICLES ...
*   SELECT AN OPTION:
*      1 = SPECIFY BY ENTERING A STRESS PERIOD AND TIME STEP
*      2 = SPECIFY BY ENTERING A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
2
*   ENTER: REFERENCE TIME  &  TIME UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.31
503 365
* STOP COMPUTING PATHS AT A SPECIFIED VALUE OF TRACKING TIME ?
@RESPONSE: HELP LABEL = 2.1.42
n
* SPECIFY AN OPTION FOR READING HEAD AND FLOW RATE DATA:
*   1 = READ STANDARD MODFLOW UNFORMATTED FILES & GENERATE A
*       COMPOSITE BUDGET FILE
*   2 = READ FROM AN EXISTING COMPOSITE BUDGET FILE
@RESPONSE: HELP LABEL = 2.1.3
2
* SELECT THE OUTPUT MODE:
*     1 = ENDPOINTS
*     2 = PATHLINE
*     3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
1
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*     1 = FROM AN EXISTING DATA FILE
*     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
2
* DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
@RESPONSE: HELP LABEL = 2.1.44
n
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*     1 = FORWARD IN THE DIRECTION OF FLOW
*     2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
2
```

```
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*      1 = PASS THROUGH WEAK SINK CELLS
*      2 = STOP AT WEAK SINK CELLS
*      3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
*   STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.
*
* SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
@RESPONSE: HELP LABEL = 2.1.46
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
@RESPONSE: HELP LABEL = 2.1.14
14 14 14 14 1 1
* WHERE SHOULD THE PARTICLES BE LOCATED ?
*    1 = WITHIN A CELL
*    2 = ON ONE OR MORE OF THE CELL FACES
@RESPONSE: HELP LABEL = 2.1.15
2
* WHAT CELL FACES DO YOU WANT TO PLACE PARTICLES ON ?
*    (ENTER ALL THE FACE NUMBERS ON A SINGLE LINE)
@RESPONSE: HELP LABEL = 2.1.47
1 2 3 4 5 6
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 1:  NY  NZ
@RESPONSE: HELP LABEL = 2.1.17
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 2:  NY  NZ
@RESPONSE: HELP LABEL = 2.1.18
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 3:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.19
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 4:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.20
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 5:  NX  NY
@RESPONSE: HELP LABEL = 2.1.21
2 2
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 6:  NX  NY
@RESPONSE: HELP LABEL = 2.1.22
2 2
*      204 PARTICLES USED OUT OF 500000 MAXIMUM
* DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
@RESPONSE: HELP LABEL = 2.1.53
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
n
```

```
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
n
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
n
```

MODPATH-PLOT Problem 2, Run 2

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------------
@  File: plot-t2.rsp
@
@  Response file for transient flow sample problem: run 2
@------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-t2.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Source area of water discharging to the layer 1 well after 3 years of pumping
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* A TIME STEP MUST BE SPECIFIED FOR COMPUTING THE ACTIVE GRID AND CONTOURS.
* SELECT AN OPTION FOR SPECIFYING THE TIME STEP:
*   1 = ENTER STRESS PERIOD AND TIME STEP DIRECTLY
*   2 = ENTER A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
2
* ENTER: SIMULATION TIME AND UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.2B
503 365
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
1
* ENTER THE TYPE OF PLOT:
*     1 = PATHLINE PLOT
*     2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*     5 = TIME SERIES
*     6 = GRID ONLY
*     7 = CONTOUR PLOT ONLY
@RESPONSE: HELP LABEL = 2.1.3
4
```

```
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*    1 = DRAW ACTIVE GRID BOUNDARY ONLY
*    2 = DRAW FULL GRID BOUNDARY ONLY
*    3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*    1 = DO NOT SHADE
*    2 = SOLID SHADING
*    3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* WHAT LAYER SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
1
* WHAT DATA SHOULD BE PLOTTED ?
*    1 = PLOT ALL DATA BY PROJECTION
*    2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* SELECT AN OPTION FOR SCREENING ENDPOINTS:
*    1 = PLOT ONLY THOSE PARTICLES THAT ARE STILL ACTIVE
*    2 = PLOT ONLY THOSE PARTICLES THAT TERMINATED NORMALLY
*    3 = PLOT BOTH ACTIVE AND NORMALLY-TERMINATED PARTICLES
@RESPONSE: HELP LABEL = 2.1.15A
2
* PLOT ONLY THOSE PARTICLES WHOSE FINAL LOCATIONS ARE IN A SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.49
n
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*      MIN COLUMN, MAX COLUMN, MIN ROW, MAX ROW
*               ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.18
    1   27    1   27
* DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
@RESPONSE: HELP LABEL = 2.1.50
n
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*       (ENTER "0" TO CYCLE THROUGH COLORS)
*       (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
0
* SELECT A CONTOURING OPTION:
*    0 = NO CONTOURS
*    1 = HEAD IN A LAYER
*    2 = DRAWDOWN IN A LAYER
*    3 = OTHER GRIDDED DATA
*    4 = HEAD DIFFERENCE BETWEEN TWO LAYERS
@RESPONSE: HELP LABEL = 6.1.1
0
```

```
* SPECIFY THE NAME OF A DRAWING COMMANDS FILE:
* ( <CR> = NO DRAWING COMMANDS FILE )
@RESPONSE: HELP LABEL = 2.1.44

* SELECT AN OPTION FOR SIZING THE PLOT:
*   1 = DRAW PLOT TO AN EXACT MAP SCALE
*   2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*   (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

MODPATH Problem 2, Run 3

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------------
@  File: path-t3.rsp
@
@  Response file for transient flow sample problem: run 3
@------------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-t3.nam
* DEFINE A REFERENCE TIME FOR RELEASING PARTICLES ...
*   SELECT AN OPTION:
*      1 = SPECIFY BY ENTERING A STRESS PERIOD AND TIME STEP
*      2 = SPECIFY BY ENTERING A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
1
*   ENTER: STRESS PERIOD & TIME STEP
@RESPONSE: HELP LABEL = 2.1.2
3 1
*   ENTER: RELATIVE TIME WITHIN TIME STEP
*          (VALUE FROM 0 TO 1)
@RESPONSE: HELP LABEL = 2.1.32
0
* STOP COMPUTING PATHS AT A SPECIFIED VALUE OF TRACKING TIME ?
@RESPONSE: HELP LABEL = 2.1.42
n
* SPECIFY AN OPTION FOR READING HEAD AND FLOW RATE DATA:
*   1 = READ STANDARD MODFLOW UNFORMATTED FILES & GENERATE A
*       COMPOSITE BUDGET FILE
*   2 = READ FROM AN EXISTING COMPOSITE BUDGET FILE
@RESPONSE: HELP LABEL = 2.1.3
2
* SELECT THE OUTPUT MODE:
*     1 = ENDPOINTS
*     2 = PATHLINE
*     3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
1
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*     1 = FROM AN EXISTING DATA FILE
*     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
2
* DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
@RESPONSE: HELP LABEL = 2.1.44
n
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*     1 = FORWARD IN THE DIRECTION OF FLOW
*     2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
1
```

```
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*     1 = PASS THROUGH WEAK SINK CELLS
*     2 = STOP AT WEAK SINK CELLS
*     3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
*  STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.
*
* SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
@RESPONSE: HELP LABEL = 2.1.46
n
* SELECT AN OPTION FOR RELEASING PARTICLES:
*   1 = SINGLE, INSTANTANEOUS RELEASE
*   2 = MULTIPLE RELEASE TIMES
@RESPONSE: HELP LABEL = 2.1.14A
1
*  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
@RESPONSE: HELP LABEL = 2.1.14
1 26 1 27 1 1
* WHERE SHOULD THE PARTICLES BE LOCATED ?
*   1 = WITHIN A CELL
*   2 = ON ONE OR MORE OF THE CELL FACES
@RESPONSE: HELP LABEL = 2.1.15
2
* WHAT CELL FACES DO YOU WANT TO PLACE PARTICLES ON ?
*   (ENTER ALL THE FACE NUMBERS ON A SINGLE LINE)
@RESPONSE: HELP LABEL = 2.1.47
6
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 6:  NX  NY
@RESPONSE: HELP LABEL = 2.1.22
2 2
*    2808 PARTICLES USED OUT OF 500000 MAXIMUM
* DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
@RESPONSE: HELP LABEL = 2.1.53
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
y
* SPECIFY AN ERROR TOLERANCE (IN PERCENT):
@RESPONSE: HELP LABEL = 3.1.1
0.1
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
n
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
n
```

MODPATH-PLOT Problem 2, Run 3

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------------
@  File: plot-t3.rsp
@
@  Response file for transient flow sample problem: run 3
@-------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-t3.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Source areas to wells in layers 1 and 4 under new steady state conditions
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* A TIME STEP MUST BE SPECIFIED FOR COMPUTING THE ACTIVE GRID AND CONTOURS.
* SELECT AN OPTION FOR SPECIFYING THE TIME STEP:
*   1 = ENTER STRESS PERIOD AND TIME STEP DIRECTLY
*   2 = ENTER A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
1
* ENTER: STRESS PERIOD & TIME STEP USED TO COMPUTE ACTIVE GRID AND CONTOURS:
@RESPONSE: HELP LABEL = 2.1.2
3 1
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
1
* ENTER THE TYPE OF PLOT:
*     1 = PATHLINE PLOT
*     2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*     5 = TIME SERIES
*     6 = GRID ONLY
*     7 = CONTOUR PLOT ONLY
@RESPONSE: HELP LABEL = 2.1.3
2
```

```
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*   1 = DRAW ACTIVE GRID BOUNDARY ONLY
*   2 = DRAW FULL GRID BOUNDARY ONLY
*   3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*   1 = DO NOT SHADE
*   2 = SOLID SHADING
*   3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* WHAT LAYER SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
1
* WHAT DATA SHOULD BE PLOTTED ?
*   1 = PLOT ALL DATA BY PROJECTION
*   2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* SELECT AN OPTION FOR SCREENING ENDPOINTS:
*   1 = PLOT ONLY THOSE PARTICLES THAT ARE STILL ACTIVE
*   2 = PLOT ONLY THOSE PARTICLES THAT TERMINATED NORMALLY
*   3 = PLOT BOTH ACTIVE AND NORMALLY-TERMINATED PARTICLES
@RESPONSE: HELP LABEL = 2.1.15A
2
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*      MIN COLUMN, MAX COLUMN, MIN ROW, MAX ROW
*              ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.18
    1   27    1   27
* DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
@RESPONSE: HELP LABEL = 2.1.50
n
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*       (ENTER "0" TO CYCLE THROUGH COLORS)
*       (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
0
* SELECT A CONTOURING OPTION:
*   0 = NO CONTOURS
*   1 = HEAD IN A LAYER
*   2 = DRAWDOWN IN A LAYER
*   3 = OTHER GRIDDED DATA
*   4 = HEAD DIFFERENCE BETWEEN TWO LAYERS
@RESPONSE: HELP LABEL = 6.1.1
0
* SPECIFY THE NAME OF A DRAWING COMMANDS FILE:
* ( <CR> = NO DRAWING COMMANDS FILE )
@RESPONSE: HELP LABEL = 2.1.44
```

```
* SELECT AN OPTION FOR SIZING THE PLOT:
*   1 = DRAW PLOT TO AN EXACT MAP SCALE
*   2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*   (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

MODPATH Problem 2, Run 4

```
@[MODPATH Version 3.00 (V3, Release 1, 9-94) ]
@
@------------------------------------------------------------------------------
@  File: path-t4.rsp
@
@  Response file for transient flow sample problem: run 4
@------------------------------------------------------------------------------
@
* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 4.2.1
demo-t4.nam
* DEFINE A REFERENCE TIME FOR RELEASING PARTICLES ...
*   SELECT AN OPTION:
*      1 = SPECIFY BY ENTERING A STRESS PERIOD AND TIME STEP
*      2 = SPECIFY BY ENTERING A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
2
*   ENTER: REFERENCE TIME  &  TIME UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.31
510 365
* STOP COMPUTING PATHS AT A SPECIFIED VALUE OF TRACKING TIME ?
@RESPONSE: HELP LABEL = 2.1.42
n
* SPECIFY AN OPTION FOR READING HEAD AND FLOW RATE DATA:
*   1 = READ STANDARD MODFLOW UNFORMATTED FILES & GENERATE A
*       COMPOSITE BUDGET FILE
*   2 = READ FROM AN EXISTING COMPOSITE BUDGET FILE
@RESPONSE: HELP LABEL = 2.1.3
2
* SELECT THE OUTPUT MODE:
*     1 = ENDPOINTS
*     2 = PATHLINE
*     3 = TIME SERIES
@RESPONSE: HELP LABEL = 2.1.6
1
* HOW ARE STARTING LOCATIONS TO BE ENTERED?
*     1 = FROM AN EXISTING DATA FILE
*     2 = ARRAYS OF PARTICLES WILL BE GENERATED INTERNALLY
@RESPONSE: HELP LABEL = 2.1.9
2
* DO YOU WANT TO STORE INTERNALLY-GENERATED STARTING LOCATIONS ON DISK ?
@RESPONSE: HELP LABEL = 2.1.44
n
* IN WHICH DIRECTION SHOULD PARTICLES BE TRACKED?
*     1 = FORWARD IN THE DIRECTION OF FLOW
*     2 = BACKWARDS TOWARD RECHARGE LOCATIONS
@RESPONSE: HELP LABEL = 2.1.10
2
```

```
* HOW SHOULD PARTICLES BE TREATED WHEN THEY ENTER CELLS WITH INTERNAL SINKS ?
*     1 = PASS THROUGH WEAK SINK CELLS
*     2 = STOP AT WEAK SINK CELLS
*     3 = STOP AT WEAK SINK CELLS THAT EXCEED A SPECIFIED STRENGTH
@RESPONSE: HELP LABEL = 2.1.11
1
* DO YOU WANT TO STOP PARTICLES WHENEVER THEY ENTER ONE SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.45
n
*  STARTING LOCATIONS WILL BE GENERATED FOR 3-D SUBREGIONS.
*
* SHOULD PARTICLES BE PLACED IN CONSTANT HEAD CELLS ?
@RESPONSE: HELP LABEL = 2.1.46
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
@RESPONSE: HELP LABEL = 2.1.14
14 14 14 14 4 4
* WHERE SHOULD THE PARTICLES BE LOCATED ?
*   1 = WITHIN A CELL
*   2 = ON ONE OR MORE OF THE CELL FACES
@RESPONSE: HELP LABEL = 2.1.15
2
* WHAT CELL FACES DO YOU WANT TO PLACE PARTICLES ON ?
*   (ENTER ALL THE FACE NUMBERS ON A SINGLE LINE)
@RESPONSE: HELP LABEL = 2.1.47
1 2 3 4 6
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 1:  NY  NZ
@RESPONSE: HELP LABEL = 2.1.17
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 2:  NY  NZ
@RESPONSE: HELP LABEL = 2.1.18
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 3:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.19
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 4:  NX  NZ
@RESPONSE: HELP LABEL = 2.1.20
7 7
* ENTER A 2-D ARRAY OF PARTICLES FOR FACE 6:  NX  NY
@RESPONSE: HELP LABEL = 2.1.22
2 2
*     200 PARTICLES USED OUT OF 500000 MAXIMUM
* DO YOU WANT TO SPECIFY ANOTHER SUBREGION ?
@RESPONSE: HELP LABEL = 2.1.53
n
* DO YOU WANT TO COMPUTE VOLUMETRIC BUDGETS FOR ALL CELLS ?
@RESPONSE: HELP LABEL = 3.1.4
y
* SPECIFY AN ERROR TOLERANCE (IN PERCENT):
@RESPONSE: HELP LABEL = 3.1.1
0.1
```

```
*  DO YOU WANT TO CHECK DATA CELL BY CELL ?
@RESPONSE: HELP LABEL = 3.1.2
n
* SUMMARIZE FINAL STATUS OF PARTICLES IN SUMMARY.PTH FILE ?
@RESPONSE: HELP LABEL = 3.1.3
n
```

MODPATH-PLOT Problem 2, Run 4

```
@[MODPATH-PLOT Version 3.00 (V3, Release 1, 9-94) ]
@
@-------------------------------------------------------------------------------
@  File: plot-t4.rsp
@
@  Response file for transient flow sample problem: run 4
@-------------------------------------------------------------------------------
@
* TO REDEFINE SETTINGS, ENTER NAME OF FILE WITH SETTINGS DATA:
* ( <CR> = USE DEFAULT SETTINGS FOR DEVICE)
@RESPONSE: HELP LABEL = 4.22.1

* ENTER THE NAME FILE:
@RESPONSE: HELP LABEL = 5.2.1
demo-t4.nam
* ENTER TITLE (80 CHARACTERS OR LESS):
@RESPONSE: HELP LABEL = 2.1.40
Source of water to layer 4 well 10 years after start of pumping in layer 1.
* ENTER THE TYPE OF GRAPHICS OUTPUT DEVICE:
*      1 = Device 1 (usually a display monitor)
*          (availability of additional output devices depends on each
*           specific GKS installation)
@RESPONSE: HELP LABEL = 4.20.1
1
* A TIME STEP MUST BE SPECIFIED FOR COMPUTING THE ACTIVE GRID AND CONTOURS.
* SELECT AN OPTION FOR SPECIFYING THE TIME STEP:
*   1 = ENTER STRESS PERIOD AND TIME STEP DIRECTLY
*   2 = ENTER A VALUE OF SIMULATION TIME
@RESPONSE: HELP LABEL = 2.1.2A
2
* ENTER: SIMULATION TIME AND UNITS CONVERSION FACTOR
@RESPONSE: HELP LABEL = 2.1.2B
510 365
* WHAT IS THE ORIENTATION OF THE PLOT ?
*   1 = MAP VIEW
*   2 = TRUE CROSS SECTION ALONG A COLUMN
*   3 = TRUE CROSS SECTION ALONG A ROW
*   4 = NORMALIZED CROSS SECTION ALONG A COLUMN
*   5 = NORMALIZED CROSS SECTION ALONG A ROW
@RESPONSE: HELP LABEL = 2.1.6
1
* ENTER THE TYPE OF PLOT:
*     1 = PATHLINE PLOT
*     2 = STARTING LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     3 = FINAL LOCATIONS FROM FORWARD TRACKING ANALYSIS
*     4 = FINAL LOCATIONS FROM BACKWARD TRACKING ANALYSIS
*     5 = TIME SERIES
*     6 = GRID ONLY
*     7 = CONTOUR PLOT ONLY
@RESPONSE: HELP LABEL = 2.1.3
4
```

```
* SELECT A OPTION FOR DRAWING GRID BOUNDARIES:
*   1 = DRAW ACTIVE GRID BOUNDARY ONLY
*   2 = DRAW FULL GRID BOUNDARY ONLY
*   3 = DRAW ACTIVE GRID AND FULL GRID BOUNDARIES
@RESPONSE: HELP LABEL = 2.1.4
1
* DRAW INTERIOR GRID LINES ?
@RESPONSE: HELP LABEL = 2.1.45
n
* SELECT OPTION FOR SHADING INACTIVE CELLS:
*   1 = DO NOT SHADE
*   2 = SOLID SHADING
*   3 = HATCH SHADING
@RESPONSE: HELP LABEL = 2.1.45B
1
* WHAT LAYER SHOULD BE USED WHEN DRAWING THE ACTIVE GRID ?
@RESPONSE: HELP LABEL = 2.1.5
1
* WHAT DATA SHOULD BE PLOTTED ?
*   1 = PLOT ALL DATA BY PROJECTION
*   2 = ONLY PLOT DATA WITHIN A 2-D SLICE
@RESPONSE: HELP LABEL = 2.1.8
1
* SELECT AN OPTION FOR SCREENING ENDPOINTS:
*   1 = PLOT ONLY THOSE PARTICLES THAT ARE STILL ACTIVE
*   2 = PLOT ONLY THOSE PARTICLES THAT TERMINATED NORMALLY
*   3 = PLOT BOTH ACTIVE AND NORMALLY-TERMINATED PARTICLES
@RESPONSE: HELP LABEL = 2.1.15A
2
* PLOT ONLY THOSE PARTICLES WHOSE FINAL LOCATIONS ARE IN A SPECIFIC ZONE ?
@RESPONSE: HELP LABEL = 2.1.49
n
* ENTER RANGE OF GRID COORDINATES FOR THIS PLOT:
*      MIN COLUMN, MAX COLUMN, MIN ROW, MAX ROW
*              ( BLANK LINE = PLOT FULL GRID)
@RESPONSE: HELP LABEL = 2.1.18
    1   27    1   27
* DO YOU WANT TO CHANGE ANY OF THE ZONE CODES IN THE IBOUND ARRAY ?
@RESPONSE: HELP LABEL = 2.1.50
n
* SELECT A COLOR FOR PLOTTING PATHLINES AND POINTS:
*       (ENTER "0" TO CYCLE THROUGH COLORS)
*       (ENTER A BLANK LINE TO FORCE BLACK AND WHITE PLOT)
@RESPONSE: HELP LABEL = 2.1.52
0
* SELECT A CONTOURING OPTION:
*   0 = NO CONTOURS
*   1 = HEAD IN A LAYER
*   2 = DRAWDOWN IN A LAYER
*   3 = OTHER GRIDDED DATA
*   4 = HEAD DIFFERENCE BETWEEN TWO LAYERS
@RESPONSE: HELP LABEL = 6.1.1
0
```

```
* SPECIFY THE NAME OF A DRAWING COMMANDS FILE:
* ( <CR> = NO DRAWING COMMANDS FILE )
@RESPONSE: HELP LABEL = 2.1.44

* SELECT AN OPTION FOR SIZING THE PLOT:
*   1 = DRAW PLOT TO AN EXACT MAP SCALE
*   2 = SCALE PLOT AUTOMATICALLY
@RESPONSE: HELP LABEL = 2.1.53
2
* ENTER A SCALING FACTOR BETWEEN 0.05 AND 1 TO SIZE THE PLOT:
*   (1= MAXIMUM SIZE POSSIBLE FOR OUTPUT DEVICE)
@RESPONSE: HELP LABEL = 2.1.53A
1
```

Appendix F
References

American National Standards Institute, 1978, Programming Language FORTRAN, ANSI, X3.9-1978, chs. 1-18.

American National Standards Institute, 1985, Computer Graphics -- Graphical Kernel System (GKS) functional description, ANSI X3.124-1985, 106p.

Harbaugh, Arlen W., 1990, A simple contouring program for gridded data, U. S. Geological Survey Open-File Report 90-144, 37p.

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, 586p.

Pollock, David W., 1989a, 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, 188p.

Pollock, David W., 1989b, Documentation of GKS-based MODPATH-PLOT, U.S. Geological Survey Open-File Report 89-622, 49p.

Prudic, David E., 1989, Documentation of a computer program to simulate stream-aquifer relations using a modular, finite-difference, ground-water flow model, U.S. Geological Survey Open-File Report 88-729, 113p.