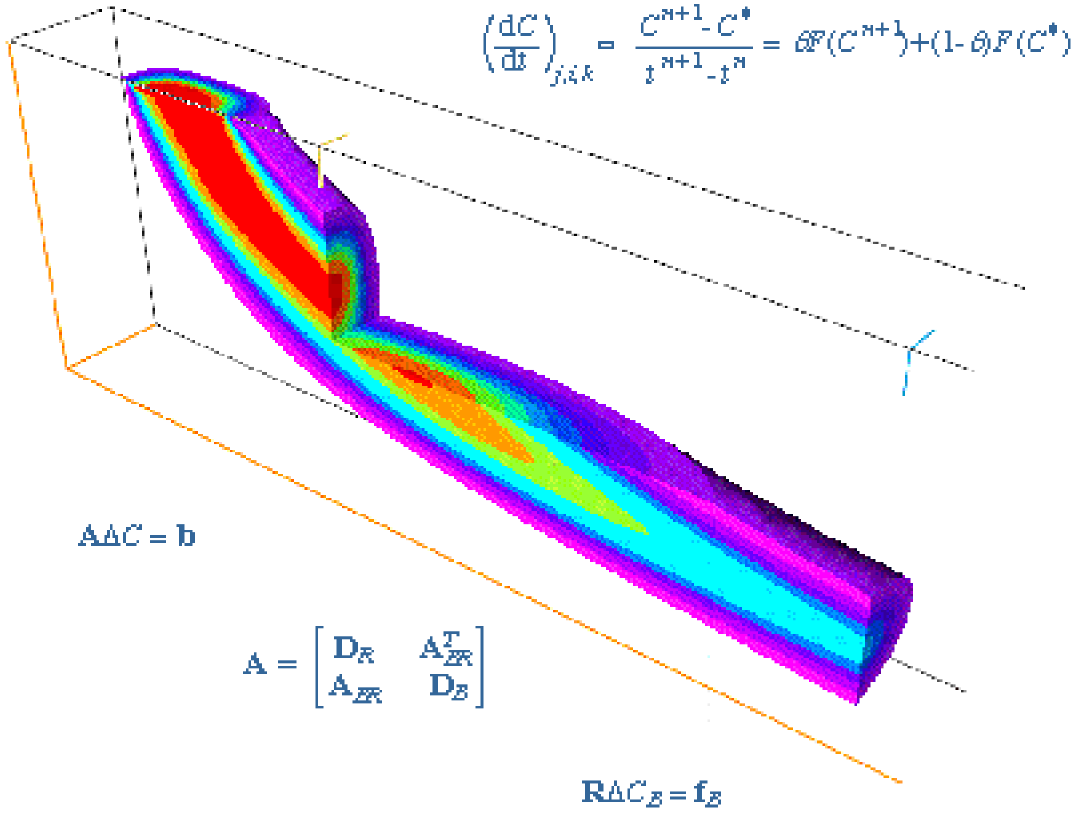
An Implicit Dispersive Transport Algorithm for the U.S. Geological Survey *MOC3D* Solute-Transport Model


$\left(\frac{dC}{dt}\right)_{j,i,k} = \frac{C^{n+1} - C^{*}}{t^{n+1} - t^{n}} = \theta F(C^{n+1}) + (1-\theta) F(C^{*})$
$\mathbf{A}\Delta C = \mathbf{b}$
$\mathbf{A} = \begin{bmatrix} \mathbf{D}_R & \mathbf{A}^T_{BR} \\ \mathbf{A}_{BR} & \mathbf{D}_B \end{bmatrix}$
$\mathbf{R}\Delta C_B = \mathbf{f}_B$

U.S. GEOLOGICAL SURVEY

Water-Resources Investigations Report 98-4234

Cover: Visualization of part of a three-dimensional plume calculated using the implicit solver in *MOC3D* for a problem (described by Burnett and Frind, 1987) of a constant source of solute in a nonuniform flow field (see figure 13 of this report and related discussion). The calculated concentrations were visualized using EarthVision software (by Dynamic Graphics, Inc.) on a Silicon Graphics workstation, and the image includes a "chair" cut to expose the internal structure of the plume. The colorization scale was selected so that each color band represents an interval of one-tenth of the range of relative concentrations. Also depicted are four of the equations relevant to the implicit formulation (see equations 10, 14, 15, and 16 of this report).

An Implicit Dispersive Transport Algorithm for the U.S. Geological Survey *MOC3D* Solute-Transport Model

By K.L. Kipp, Jr., L.F. Konikow, and G.Z. Hornberger

U. S. GEOLOGICAL SURVEY

Water-Resources Investigations Report 98-4234

Reston, Virginia
1998

USGS
science for a changing world

U.S. DEPARTMENT OF THE INTERIOR
BRUCE BABBITT, Secretary

U.S. GEOLOGICAL SURVEY
Charles G. Groat, Director

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

For additional information write to:

Solute-Transport Project
U.S. Geological Survey
Water Resources Division
431 National Center
Reston, VA 20192

Copies of this report can be purchased from:

U.S. Geological Survey
Information Services
Box 25286
Federal Center
Denver, CO 80225

PREFACE

The *MOC3D* computer code simulates the transport of a single solute in ground water that flows through porous media. The model is a package for the U.S. Geological Survey (USGS) *MODFLOW* ground-water model. The new algorithm documented in this report incorporates an implicit difference approximation in time of the dispersive transport equation and is an alternative to the explicit difference approximation. The code for this algorithm is integrated with the code for *MOC3D*, forming Version 2. This extension offers significantly improved efficiency of *MOC3D* for transport problems that are dominated by dispersion or diffusion relative to advection.

Version 2 of the *MOC3D* code, which includes the new extension, is available for downloading over the Internet from a USGS software repository. The repository is accessible on the World Wide Web (WWW) from the USGS Water-Resources Information Web page at URL **http://water.usgs.gov/**. The URL for the public repository is: **http://water.usgs.gov/software/**. The public anonymous FTP site is on the Water-Resources Information server (water.usgs.gov or 130.11.50.175) in the /pub/software directory. When this code is revised or updated in the future, new versions or releases will be made available at these same sites.

Although extensive testing of *MOC3D* (Version 2) with the implicit dispersive transport algorithm indicates that this model will yield reliable calculations for a wide variety of field problems, the user is cautioned that the accuracy and efficiency of the model can be affected significantly for certain combinations of parameter values and boundary conditions.

CONTENTS

FIGURES

Page

1–3. Diagrams showing:

4–7. Graphs showing:

8-14 Plots showing:

TABLES

ABSTRACT

This report documents an extension to the U.S. Geological Survey *MOC3D* transport model that incorporates an implicit-in-time difference approximation for the dispersive transport equation, including source/sink terms. The original *MOC3D* transport model (Version 1) uses the method of characteristics to solve the transport equation on the basis of the velocity field. The original *MOC3D* solution algorithm incorporates particle tracking to represent advective processes and an explicit finite-difference formulation to calculate dispersive fluxes. The new implicit procedure eliminates several stability criteria required for the previous explicit formulation. This allows much larger transport time increments to be used in dispersion-dominated problems. The decoupling of advective and dispersive transport in *MOC3D*, however, is unchanged. With the implicit extension, the *MOC3D* model is upgraded to Version 2.

A description of the numerical method of the implicit dispersion calculation, the data-input requirements and output options, and the results of simulator testing and evaluation are presented. Version 2 of *MOC3D* was evaluated for the same set of problems used for verification of Version 1. These test results indicate that the implicit calculation of Version 2 matches the accuracy of Version 1, yet is more efficient than the explicit calculation for transport problems that are characterized by a grid Peclet number less than about 1.0.

INTRODUCTION

This report documents an extension to the U.S. Geological Survey (USGS) *MOC3D* transport model (Konikow and others, 1996) that incorporates an implicit-in-time formulation for the dispersion equation. *MOC3D* simulates three-dimensional solute transport in flowing ground water for a single dissolved chemical constituent and represents the processes of advective transport, hydrodynamic dispersion (including both mechanical dispersion and diffusion), mixing (or dilution) from fluid sources, and simple chemical reactions (including linear sorption and decay). The *MOC3D* transport model uses the method of characteristics to solve the transport equation on the basis of the velocity field, which is calculated from the head distribution determined by MODFLOW-96 (Harbaugh and McDonald, 1996a).

In its original implementation (Konikow and Bredehoeft, 1978; Konikow and others, 1996), the method of characteristics uses particle tracking to represent advective transport and explicit finite-difference methods to calculate concentration changes over time that result from dispersive fluxes and mixing with solute from fluid sources. Explicit methods, however, have stability criteria associated with them. In some cases, these criteria cause the model to take extremely small time increments and thus become costly in computational time, especially for three-dimensional systems. These cases are characterized by relatively large dispersivity values.

The new algorithm, documented in this report, employs an implicit finite-difference equation to calculate concentration changes over time that results from dispersive fluxes and mixing with fluid sources. An iterative equation solver is used to solve the simultaneous difference equations, requiring five new parameters to be specified as input data in addition to the parameters previously needed for Version 1 of *MOC3D*. The implicit solution is unconditionally stable, which allows larger time steps to be taken than with the explicit solution. Thus, it can be much less costly in computation time for dispersion-dominated problems than the original explicit method. The implicit algorithm code, however, requires more computer memory than the explicit method for a given size problem.

With the implicit extension, the *MOC3D* model is upgraded to Version 2. *MOC3D* is integrated with *MODFLOW-96*, the U.S. Geological Survey's (USGS) modular, three-dimensional, finite-difference, ground-water flow model (McDonald and Harbaugh, 1988; Harbaugh and McDonald, 1996a and 1996b). *MODFLOW* solves the ground-water flow equation and the reader is referred to the documentation for that model and its subsequent packages and modules for complete details. In this report it is assumed that the reader is familiar with the *MODFLOW* family of codes, including *MOC3D* (Version 1).

MOC3D (Version 2) is offered as a general simulator that is applicable to a wide range of field problems that involve solute transport. The user, however, should first become aware of the assumptions and limitations inherent in the simulator, as described in this report and by Konikow and others (1996). There are some situations in which the model results could be inaccurate or model operation costly. This report includes guidelines for recognizing these situations and avoiding such problems.

MOC3D is limited to fluid properties, such as density and viscosity, that are uniform and constant, and thus independent of concentration values. Within the finite-difference grid used to solve the flow equation in *MODFLOW*, the user is able to specify a window or subgrid over which *MOC3D* will solve the solute-transport equation. *MOC3D*, however, requires that the horizontal (row and column) grid spacing be constant in each direction within the subgrid. The types of reactions that are incorporated into *MOC3D* are restricted to those that can be represented by a first-order rate reaction, such as radioactive decay, or by a retardation factor, such as equilibrium, reversible, sorption-desorption reactions that are governed by a linear isotherm and constant distribution coefficient.

The computer program for the implicit dispersive transport extension is written in FORTRAN-77 and has been developed in a modular style**.** This documentation includes a description of the implicit finite-difference algorithm used to solve the dispersive flux and fluid source terms of the solute-transport equation in *MOC3D* (Version 2). A complete description of the data requirements, input format specifications, program options, and output formats is included. This report must be used in conjunction with the original *MOC3D* documentation (Konikow and others, 1996), which provides information on all previously existing features of *MOC3D*, including details on the method of characteristics, code structure, and model use.

Acknowledgments. The authors appreciate the helpful model evaluation and review comments provided by USGS colleagues D.J. Goode and C.E. Heywood.

THEORETICAL BACKGROUND AND GOVERNING EQUATIONS

The ground-water flow and interstitial velocity equations used in *MOC3D* are given by Konikow and others, 1996, and will not be repeated here. Solution to the flow equation provides the interstitial velocity field, which couples the solute-transport equation to the ground-water flow equation.

Governing Equation for Solute Transport

The solute-transport equation is that presented in Konikow and others (1996, equation 6):

$$\frac{\partial(\varepsilon C)}{\partial t} + \frac{\partial(\rho_b \overline{C})}{\partial t} + \frac{\partial}{\partial x_i}\left(\varepsilon C V_i\right) - \frac{\partial}{\partial x_i}\left(\varepsilon D_{ij}\frac{\partial C}{\partial x_j}\right) - \sum C'W + \lambda\left(\varepsilon C + \rho_b \overline{C}\right) = 0 \ , \qquad (1)$$

where C is volumetric concentration (mass of solute per unit volume of fluid, ML^{-3}), ρ_b is the bulk density of the aquifer material (mass of solids per unit volume of aquifer, ML^{-3}), $\overline{C}$ is the mass concentration of solute sorbed on or contained within the solid aquifer material (mass of solute per unit mass of aquifer material, MM^{-1}), ε is the effective porosity (dimensionless), V is a vector of interstitial fluid velocity components (LT^{-1}), D is a second-rank tensor of dispersion coefficients (L^2T^{-1}), W is a volumetric fluid sink (W<0) or fluid source (W>0) rate per unit volume of aquifer (T^{-1}), C' is the volumetric concentration in the sink/source fluid (ML^{-3}), λ is the decay rate (T^{-1}), t is time (T), and x_i are the Cartesian coordinates (L).

For the case of reversible, linear equilibrium sorption, the form of the solute-transport equation that is solved in Version 2 of *MOC3D* is the same as that solved in Version 1:

$$\frac{\partial C}{\partial t} + \frac{V_i}{R_f}\frac{\partial C}{\partial x_i} - \frac{1}{\varepsilon R_f}\frac{\partial}{\partial x_i}\left(\varepsilon D_{ij}\frac{\partial C}{\partial x_j}\right) - \frac{\Sigma\left[W\left(C' - C\right)\right]}{\varepsilon R_f} + \lambda C = 0. \qquad (2)$$

Review of Assumptions

As described by Konikow and others (1996), a number of assumptions have been made in the development of the governing equations. Following is a list of the main assumptions for review:

1. Darcy's law is valid and hydraulic-head gradients are the only significant driving mechanism for fluid flow.
2. The hydraulic conductivity of the aquifer system is constant with time. Also, if the system is anisotropic, it is assumed that the principal axes of the hydraulic-conductivity tensor are aligned with the coordinate system of the grid, so that the cross-product terms of the hydraulic-conductivity tensor are eliminated.
3. Gradients of fluid density, viscosity, and temperature do not affect the velocity distribution.
4. No chemical reactions occur that affect the fluid or aquifer properties.
5. The dispersivity coefficients are constant with time, and the aquifer is isotropic with respect to longitudinal dispersivity.

As noted by Konikow and Bredehoeft (1978), the nature of a specific field problem may be such that not all of these underlying assumptions are valid. The degree to which field conditions deviate from these assumptions will affect the applicability and reliability of the model for that problem. If the deviation from a particular assumption is significant, the governing equations and the numerical simulator may have to be modified to account for the appropriate processes or factors.

NUMERICAL METHODS

The notation and conventions used in this report and in the computer code to describe the grid and to number the nodes are illustrated in figures 1 and 2. The indexing notation used here is consistent with that used in the

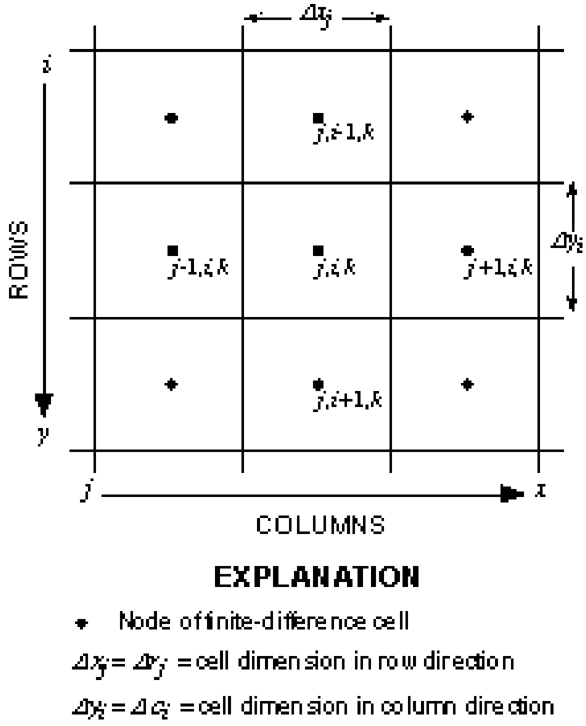

Δx_j
i
ROWS
y
j,i-1,k
j-1,i,k
j,i,k
j+1,i,k
j,i+1,k
Δy_i
j
x
COLUMNS
EXPLANATION
Node of finite-difference cell
Δx_j = Δr_j = cell dimension in row direction
Δy_i = Δc_i = cell dimension in column direction

Figure 1. Notation used to label rows, columns, and nodes within one layer (*k*) of a three-dimensional, block-centered, finite-difference grid for *MOC3D.*

computer code for *MODFLOW* by McDonald and Harbaugh (1988), although not the notation used in some sections of their report. Our indexing notation maintains conformity between the text of this report and the FORTRAN code in *MOC3D*, and the index order corresponds to an x,y,z sequence. However, our notation differs from that used in some other ground-water models in that the x-direction is indexed by *"j"* and increases from left to right along a row to indicate the column number. Our use of Δx and Δy is synonymous with the use of Δr and Δc, respectively, by McDonald and Harbaugh (1988). The y-direction is indexed by *"i"* and increases from the top of the grid to the bottom within a column to indicate the row number. Thus, in a map view of any one horizontal layer, as illustrated in figure 1, the node representing a cell in the first row and first column of the grid would lie in the upper left corner of the grid.

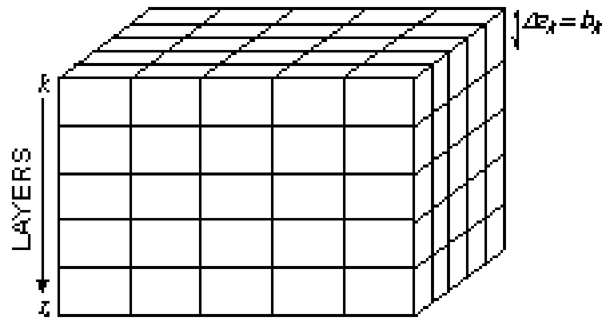


Figure 2. Representative three-dimensional grid for *MOC3D* illustrating notation for layers.

The z-direction represents layers and is indexed by *"k."* As indicated in figure 2, the first layer (k = 1) in a multilayer grid would be the top (or highest elevation) layer. The saturated thickness of a cell ($b_{j,i,k}$) is equivalent to Δz.

Ground-Water Flow Equation

A numerical solution of the three-dimensional ground-water flow equation is obtained by the *MODFLOW* code using implicit (backward-in-time) finite-difference methods. Successful use of *MOC3D* requires a thorough familiarity with the use of *MODFLOW.* Comprehensive documentation of *MODFLOW* is presented by McDonald and Harbaugh (1988), Harbaugh and McDonald (1996a and 1996b), and the various reports for additional implemented packages and modules.

Average Interstitial Velocity

The solution of the transport equation requires knowledge of the velocity (or specific discharge) field. Therefore, after the head distribution has been calculated for a given time step or steady-state flow condition, the specific discharge across every face of each finite-difference cell within the transport subgrid is calculated using a finite-difference approximation (see Konikow and others, 1996).

The particle-tracking algorithm requires that the seepage velocity at any point within a cell be defined to compute advective transport. It is calculated at points within a finite-difference cell based on interpolated estimates of specific discharge at those points divided by the effective porosity of the cell.

Solute-Transport Equation

The mathematical properties of the transport equation vary depending upon which terms in the equation are dominant in a particular system. Where solute transport is dominated by advection, as is common in many field problems, the transport equation resembles a hyperbolic type of equation (similar to equations that describe the propagation of a wave or of a shock front). In contrast, where a system is dominated by dispersive and diffusive fluxes, such as might occur where fluid velocities are relatively low and aquifer dispersivities are relatively high, the transport equation becomes more parabolic in nature (similar to the transient ground-water flow equation). Because system properties and fluid velocity may vary significantly, the dominant process (and the mathematical properties of the governing equation) may vary from point to point and over time within the domain of simulation.

Method of Characteristics

The approach of the method of characteristics is not to solve equation 2 itself, but rather to solve an equivalent system of three ordinary differential equations and one partial differential equation:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{V_x}{R_f}, \tag{3}$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \frac{V_y}{R_f}, \tag{4}$$

$$\frac{\mathrm{dz}}{\mathrm{d}t} = \frac{V_z}{R_f}, \tag{5}$$

$$\frac{\mathrm{d}C}{\mathrm{d}t} = \frac{1}{\varepsilon R_f}\frac{\partial}{\partial x_i}\left(\varepsilon D_{ij}\frac{\partial C}{\partial x_j}\right) + \frac{\Sigma\left[W\left(C' - C\right)\right]}{\varepsilon R_f} - \lambda C. \tag{6}$$

Although the concentration in equation 6 is now that of a parcel of fluid moving in space with the retarded velocity (V/R_f), we retain the same symbol, *C*, as a matter of notational convenience.

Solutions to equations 3-5 yield the characteristic curves [$x = x(t)$, $y = y(t)$, and $z = z(t)$]. This is accomplished by introducing a set of moving points (or reference particles) that can be traced within the stationary coordinates of a finite-difference grid. Each particle corresponds to one characteristic curve, and values of x, y, and z are obtained as functions of t for each characteristic (Garder and others, 1964). Each particle has an associated concentration and moves through the flow field by the flow velocity acting along its trajectory. For the algorithm documented in this report, equation 6 is solved along the characteristic curves for $C(t)$ by using an implicit finite-difference formulation in time. Previously, equation 6 was solved by using an explicit finite-difference formulation.

Along the characteristic curves, the processes of advection, dispersion, mixing, and reactions are occurring continuously and simultaneously (Konikow and Bredehoeft, 1978). Therefore, equations 3-6 should be solved simultaneously, but for practical reasons, they are solved sequentially. Sequential solution follows the concept of the method of fractional steps or operator splitting presented by Yanenko (1971). Sensitivity to the sequence of solving the characteristic and dispersive transport equations is minimized by solving equation 6 using concentration gradients that are based on the average of the concentrations at each node before and after advection. This effectively gives equal weight to the concentration gradients before and after advection when computing the solute flux due to dispersion. These averaged concentrations, designated as $C_{j,i,k}$, are calculated as:

$$C^{*}_{j,i,k} = \frac{C^{n}_{j,i,k} + C^{n+adv}_{j,i,k}}{2}, \qquad (7)$$

where $C^{n+adv}_{j,i,k}$ is the concentration at the new time level after advection alone.

Particle Tracking

Advection in flowing ground water is simulated by particle tracking. The other solute transport terms—dispersion, sources, and decay—are simulated by computing changes in the concentration associated with each particle. The concentration changes caused by dispersion and fluid sources are computed on the finite-difference grid, fixed in space, whereas concentration changes caused by decay are calculated directly on the moving particles. After the flow equation is solved for a new time step, specific discharges are recomputed on the basis of the new head distribution, and the movement of particles during this flow time step is based only on these specific discharges.

Decay

Decay is simulated by reducing the particle concentrations after advection (Konikow and others, 1996). A major advantage of calculating the effect of decay directly on the particles, rather than on the nodal concentrations, is that this procedure eliminates any numerical dispersion caused by the interpolation between concentrations on the moving particles and on the fixed grid (that is, averaging from particle concentrations to nodal concentrations and back to particle concentrations).

This decay algorithm has no numerical stability restrictions associated with it (Goode and Konikow, 1989). If the half-life is on the order of or smaller than the transport time step, however, some accuracy will be lost because of the explicit decoupling of decay and other transport processes.

When a solute subject to decay enters the aquifer through a fluid source, it is assumed that the fluid source contains the solute in the concentration specified by C'. The *MOC3D* simulator allows decay to occur only within the ground-water system, and not within the source reservoir. In other words, for a given stress period, C' remains constant in time. Because decay is assumed to be continuous over a time increment, however, the effective value of C' for a solute subject to decay is adjusted by the factor $e^{-\lambda \Delta t/2}$ to account for the fact that solute injected into the aquifer at the beginning of the time increment will have already decayed by the end of the time increment. If the problem being simulated requires that the solute in the source fluid itself undergo decay, then the code will have to be modified.

Node Concentrations

After all particles have been moved, the concentration at each node is temporarily assigned the average concentration of all particles then located within the volume of that cell; this average concentration is denoted as $C^{n+adv}_{j,i,k}$.

$$C^{adv}_{j,i,k} = \frac{\sum_{p=1}^{N} C^{d}_{p}\,\delta\left(j^{t+1}_{p} = j, i^{t+1}_{p} = i, k^{t+1}_{p} = k\right)}{\sum_{p=1}^{N} \delta\left(j^{t+1}_{p} = j, i^{t+1}_{p} = i, k^{t+1}_{p} = k\right)}, \qquad (8)$$

where the δ function is 1 if the particle is within the cell j,i,k and is zero otherwise, and $C^{\smile}_{p}$ is the decayed particle concentration at the end of a transport time increment. The time index is labeled “*n+adv*” because this temporarily assigned average concentration represents conditions at the new time level only with respect to advective transport and decay. The effect of advective transport is to move particles having different concentrations into and out of each cell.

Finite-Difference Approximations

The dispersive transport equation along a characteristic curve, equation 6 without the decay term, is discretized in space using finite-difference approximations. The rate of change in concentration due to dispersion and sources can be written:

$$
\begin{aligned}
\left(\frac{dC}{dt}\right)_{j,i,k} = \frac{1}{\left(R_f\right)_k \left(\varepsilon b\right)^{t+1}_{j,i,k}} \Bigg\{ & \Delta x^{-1}\left[\left(\varepsilon b D_{1m}\frac{\partial C}{\partial x_m}\right)^{*}_{j+1/2,i,k} - \left(\varepsilon b D_{1m}\frac{\partial C}{\partial x_m}\right)^{*}_{j-1/2,i,k}\right] \\
& + \Delta y^{-1}\left[\left(\varepsilon b D_{2m}\frac{\partial C}{\partial x_m}\right)^{*}_{j,i+1/2,k} - \left(\varepsilon b D_{2m}\frac{\partial C}{\partial x_m}\right)^{*}_{j,i-1/2,k}\right] \\
& + \left[\left(\varepsilon D_{3m}\frac{\partial C}{\partial x_m}\right)^{*}_{j,i,k+1/2} - \left(\varepsilon D_{3m}\frac{\partial C}{\partial x_m}\right)^{*}_{j,i,k-1/2}\right] + \sum_{W>0}\left[W_{j,i,k}\left(C'_{j,i,k} - C^{*}_{j,i,k}\right)\right]\Bigg\},
\end{aligned} \tag{9}
$$

where subscript *m* is a summation index for the dispersion term. The *j,i,k* subscripts in equation 9 denote the spatial finite-difference grid indexing, as discussed previously in the section “Numerical Methods.” Equation 9 is the dispersive transport equation applied to the fixed mesh that advances the numerical solution to the final concentrations, C^{n+1}, at the end of the time increment. The components of the dispersive flux in each direction across cell faces are calculated using finite-difference approximations that are centered-in-space.

To complete the formulation of the finite-difference equation represented by equation 9, we must approximate the time derivative of concentration. We use the concept of the method of fractional steps (Yanenko, 1971) to implement a sequential solution to equations 3-6. Summarizing the approach, as described in more detail by Konikow and others (1996), advective transport is represented by particle tracking. Concentration changes caused by dispersive fluxes and fluid sources are calculated using concentration gradients in equation 9 that at any point in space are the average of (1) the concentration gradients at the start of the time increment ($C^{n}_{j,i,k}$) and (2) the concentration gradients computed after advection of particles ($C^{n+adv}_{j,i,k}$), as denoted in equation 7. Therefore, the time derivative on the left side of equation 9 is approximated by the general finite difference for a fractional step (Yanenko, 1971, p.23):

$$
\begin{aligned}
\left(\frac{dC}{dt}\right)_{j,i,k} &\approx \frac{C^{n+1} - C^{adv}}{t^{n+1} - t^{n}} \\
&= \theta F(C^{n+1}) + (1-\theta)F(C^{*}),
\end{aligned} \tag{10}
$$

where t^n is the time at level *n* (that is, the beginning of a time increment, Δt), θ is the time difference weighting factor, and *F* is the right-hand side of equation 9. The superscript “*” indicates that the terms depend on the average of the concentration at the old time level and the concentration at the new time level after advection and decay (see equation 7). Equation 10 is not a classical difference equation because it depends on intermediate concentrations, not just values at the beginning and end of the time step. Setting the time weighting factor, θ, to 1 gives a fully implicit or backwards-in-time (BT) difference equation. Setting the time weighting factor to 0.5 gives a difference equation similar to Crank-Nicolson or centered-in-time (CT).

The time truncation error associated with centered-in-time differencing is on the order of $(\Delta t)^2$; the error associated with backwards-in-time differencing is on the order of Δt. For typically small values of Δt, the truncation error for CT will be smaller than that for BT. However, CT differencing has the potential for introducing oscillations into the numerical solution. See Kipp (1987, p. 112-114) for additional details about the numerical properties of alternative finite-difference formulations.

The next step is to express the difference equation in residual form by writing:

$$C^{n+1} = C^{adv} + \Delta C, \qquad (11)$$

where ΔC is the change in C over a time increment (Δt) due to dispersive transport and sources. Equation 11 is inserted into equation 10, yielding the finite-difference approximation to equation 9 for an interior node, j,i,k:

$$\begin{aligned}
\Delta C = \Delta t\big[&\theta T_{Sxx(j+1/2,i,k)}\left(\Delta C_{j+1,i,k} - \Delta C_{j,i,k}\right) - \theta T_{Sxx(j-1/2,i,k)}\left(\Delta C_{j,i,k} - \Delta C_{j-1,i,k}\right)\\
&+\theta T_{Syy(j,i+1/2,k)}\left(\Delta C_{j,i+1,k} - \Delta C_{j,i,k}\right) - \theta T_{Syy(j,i-1/2,k)}\left(\Delta C_{j,i,k} - \Delta C_{j,i-1,k}\right)\\
&+\theta T_{Szz(j,i,k+1/2)}\left(\Delta C_{j,i,k+1} - \Delta C_{j,i,k}\right) - \theta T_{Szz(j,i,k-1/2)}\left(\Delta C_{j,i,k} - \Delta C_{j,i,k-1}\right)\\
&+\theta T_{Sxx(j+1/2,i,k)}\left(C^*_{j+1,i,k} - C^*_{j,i,k}\right) - \theta T_{Sxx(j-1/2,i,k)}\left(C^*_{j,i,k} - C^*_{j-1,i,k}\right)\\
&+\theta T_{Syy(j,i+1/2,k)}\left(C^*_{j,i+1,k} - C^*_{j,i,k}\right) - \theta T_{Syy(j,i-1/2,k)}\left(C^*_{j,i,k} - C^*_{j,i-1,k}\right)\\
&+\theta T_{Szz(j,i,k+1/2)}\left(C^*_{j,i,k+1} - C^*_{j,i,k}\right) - \theta T_{Szz(j,i,k-1/2)}\left(C^*_{j,i,k} - C^*_{j,i,k-1}\right)\\
&+(1-\theta)T_{Sxx(j+1/2,i,k)}\left(C^*_{j+1,i,k} - C^*_{j,i,k}\right) - (1-\theta)T_{Sxx(j-1/2,i,k)}\left(C^*_{j,i,k} - C^*_{j-1,i,k}\right)\\
&+(1-\theta)T_{Syy(j,i+1/2,k)}\left(C^*_{j,i+1,k} - C^*_{j,i,k}\right) - (1-\theta)T_{Syy(j,i-1/2,k)}\left(C^*_{j,i,k} - C^*_{j,i-1,k}\right)\\
&+(1-\theta)T_{Szz(j,i,k+1/2)}\left(C^*_{j,i,k+1} - C^*_{j,i,k}\right) - (1-\theta)T_{Szz(j,i,k-1/2)}\left(C^*_{j,i,k} - C^*_{j,i,k-1}\right)\\
&+T_{Sxy(j+1/2,i,k)}\left(C^*_{j+1,i+1,k} + C^*_{j,i+1,k} - C^*_{j+1,i-1,k} - C^*_{j,i-1,k}\right)\\
&+\tilde{T}_{Sxz(j+1/2,i,k)}\left(C^*_{j+1,i,k+1} - C^*_{j+1,i,k-1}\right) + \hat{T}_{Sxz(j+1/2,i,k)}\left(C^*_{j,i,k+1} - C^*_{j,i,k-1}\right)\\
&-T_{Sxy(j-1/2,i,k)}\left(C^*_{j,i+1,k} + C^*_{j-1,i+1,k} - C^*_{j,i-1,k} - C^*_{j-1,i-1,k}\right)\\
&-\tilde{T}_{Sxz(j-1/2,i,k)}\left(C^*_{j,i,k+1} - C^*_{j,i,k-1}\right) - \hat{T}_{Sxz(j-1/2,i,k)}\left(C^*_{j-1,i,k+1} - C^*_{j-1,i,k-1}\right)\\
&+T_{Syx(j,i+1/2,k)}\left(C^*_{j+1,i+1,k} + C_{j,i+1,k} - C_{j+1,i-1,k} - C_{j,i-1,k}\right)\\
&+\tilde{T}_{Syz(j,i+1/2,k)}\left(C^*_{j,i+1,k+1} - C^*_{j,i+1,k-1}\right) + \hat{T}_{Syz(j,i+1/2,k)}\left(C^*_{j,i,k+1} - C^*_{j,i,k-1}\right)\\
&-T_{Syx(j,i-1/2,k)}\left(C^*_{j,i+1,k} + C^*_{j-1,i+1,k} - C_{j+1,i-1,k} - C^*_{j,i-1,k}\right)
\end{aligned}$$

$$
\begin{aligned}
&-\tilde{T}_{Syz(j,i-1/2,k)}\left(C^{*}_{j,i,k+1}-C^{*}_{j,i,k-1}\right)-\hat{T}_{Syz(j,i-1/2,k)}\left(C^{*}_{j-1,i,k+1}-C^{*}_{j-1,i,k-1}\right)\\
&+T_{Szx(j,i,k+1/2)}\left(C^{*}_{j+1,i+1,k}+C_{j,i+1,k}-C^{*}_{j+1,i-1,k}-C_{j,i-1,k}\right)\\
&+T_{Szy(j,i,k+1/2)}\left(C^{*}_{j,i+1,k+1}-C^{*}_{j,i+1,k-1}+C^{*}_{j,i,k+1}-C^{*}_{j,i,k-1}\right)\\
&-T_{Szx(j,i,k-1/2)}\left(C^{*}_{j,i+1,k}+C^{*}_{j-1,i+1,k}-C_{j+1,i-1,k}-C^{*}_{j,i-1,k}\right)\\
&-T_{Szy(j,i,k-1/2)}\left(C^{*}_{j,i,k+1}-C^{*}_{j,i,k-1}+C^{*}_{j-1,i,k+1}-C^{*}_{j-1,i,k-1}\right)\\
&-\theta\sum_{W>0}W_{j,i,k}\left(\Delta C_{j,i,k}\right)+\sum_{W>0}W_{j,i,k}\left(C'\right)-\sum_{W>0}W_{j,i,k}\left(C\;\right)\Big],
\end{aligned}
\tag{12}
$$

where

$$T_{Sxx(j+1/2,i,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{xx}\right)_{j+1/2,i,k}}{\Delta x^{2}}\right\},$$

$$T_{Sxy(j+1/2,i,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{xy}\right)_{j+1/2,i,k}}{2\Delta x\Delta y}\right\},$$

$$\tilde{T}_{Sxz(j+1/2,i,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{xz}\right)_{j+1/2,i,k}}{2\Delta x2B_{j+1,i,k}}\right\},$$

$$\hat{T}_{Sxz(j+1/2,i,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{xz}\right)_{j+1/2,i,k}}{2\Delta x2B_{j,i,k}}\right\},$$

$$T_{Syy(j,i+1/2,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{yy}\right)_{j,i+1/2,k}}{\Delta y^{2}}\right\},$$

$$T_{Syx(j,i+1/2,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{yx}\right)_{j,i+1/2,k}}{2\Delta y\Delta x}\right\},$$

$$\tilde{T}_{Syz(j,i+1/2,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{yz}\right)_{j,i+1/2,k}}{2\Delta y2B_{j,i+1,k}}\right\},$$

$$\hat{T}_{Syz(j,i+1/2,k)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon bD_{yz}\right)_{j,i+1/2,k}}{2\Delta y2B_{j,i,k}}\right\},$$

$$T_{Szz(j,i,k+1/2)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon D_{zz}\right)_{j,i,k+1/2}}{b_{j,i,k+1/2}}\right\},$$

$$T_{Szx(j,i,k+1/2)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon D_{zx}\right)_{j,i,k+1/2}}{2\Delta x}\right\},$$

$$T_{Szy(j,i,k+1/2)}=\frac{1}{R_{f,k}\left(\varepsilon b\right)_{j,i,k}}\left\{\frac{\left(\varepsilon D_{zy}\right)_{j,i,k+1/2}}{2\Delta y}\right\},\tag{13}$$

and where the T_S terms are the conductances for solute transport (1/T), $2B_{j,i,k} \equiv b_{j,i,k} + 1/2(b_{j,i,k-1} + b_{j,i,k+1})$ is the vertical distance between nodes $(j,i,k+1)$ and $(j,i,k-1)$, and εbD_{mn} are the dispersion terms (L^3/T) given by equations A5-A22 in Konikow and others (1996). Note that $\tilde{T}$ and $\hat{T}$ are denoted differently to help clarify that these conductances are in the vertical direction, and depend on different values of B. Also note that C' is constant over any time increment, hence there is no time level index associated with it in equation 12.

The coefficients T_S are evaluated at time n, that is, at the beginning of the time step. The time weighting factor, θ, can actually be set to any value from 0.5 to 1.0 for an implicit difference equation in time.

The source concentration, C', is a specified function of time and source location. When the source flow rate is negative, so that the node represents a fluid sink, the source concentration, C', becomes that of the cell. Note that the source-sink terms must be summed individually because multiple terms can exist at the same node and each fluid flux specification can have a different concentration associated with it.

Excluded cells in the simulation region are handled by replacing equation 12 with a trivial equation for the change in concentration at these locations. That is, the difference equation for that cell becomes $\Delta C = 0$.

Note that the cross-dispersive flux terms have been evaluated explicitly in time using the intermediate concentrations (C^*) in order to retain the 7-point stencil for the finite-difference equation. The 7-point stencil is formed by the connection of a given node to its six nearest neighbors (ahead, behind, in front, in back, above, and below) in each of the three coordinate directions. This enables a renumbering scheme, described below, to be used to form a reduced matrix for the linear equation solver. Because the cross-dispersive terms of equation 12 are explicit (not a function of C^{n+1}), at least two iterations are necessary for solving the dispersive transport equation at each time step. The number of iterations is specified by the user with the value of NCXIT in the input data set (see Appendix A).

The difference equations represented by equation 12 form a set of simultaneous linear equations of the form

$$\mathbf{A}\Delta C = \mathbf{b} \qquad (14)$$

where $\mathbf{A}$ is a symmetric, sparse matrix of the coefficients of ΔC in equation 12, and $\mathbf{b}$ is the known right-hand-side vector formed by the terms containing C^* in equation 12. A symmetric matrix means that the elements above the diagonal are a mirror image reflection of those below the diagonal. A sparse matrix is one where most (more than 75 percent) of the elements are zero. Before solving equation 14, it is advantageous to transform to a reduced set of equations.

Reduced Matrix Equation

Equation 14 is transformed to a reduced matrix equation, also known as the Schur complement (Axelsson, 1994), obtained by reordering the nodes using a red-black renumbering scheme (Price and Coats, 1974; Aziz and Settari, 1979). For red-black renumbering, the nodes are renumbered in sweeps along a primary coordinate direction skipping every other node. The secondary direction is incremented at the end of each primary sweep and the tertiary direction is incremented at the end of a set of secondary sweeps. After the first sweep cycle is complete, a second cycle is done to renumber the remaining nodes in the same fashion. In two dimensions this renumbering is analogous to numbering all the red squares of a checkerboard followed by numbering all the black squares. The selection of the primary, secondary, and tertiary sweep directions is up to the user. The convergence rate of the conjugate-gradient solver, however, can be markedly different under different reorderings. There can be as much as a factor of two between the minimum and maximum number of iterations of the solver needed to converge to a solution at a given time step depending on the sequence of directions used for the renumbering.

The red-black renumbering scheme leads to a partitioned coefficient matrix, $\mathbf{A}$, of the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{D}_R & \mathbf{A}_{BR}^T \\ \mathbf{A}_{BR} & \mathbf{D}_B \end{bmatrix}, \qquad (15)$$

where $\mathbf{D}_R$ and $\mathbf{D}_B$ are diagonal matrices. Elimination in equation 14 yields the reduced matrix equation

$$\mathbf{R}\Delta C_B = \mathbf{f}_B, \qquad (16)$$

where ΔC_B is the vector of changes in concentration for the black nodes and $\mathbf{f}_B$ is the right-hand-side vector for the black nodes. The reduced matrix, **R**, is given by

$$\mathbf{R} = \mathbf{D}_B - \mathbf{A}_{BR}\mathbf{D}_R^{-1}\mathbf{A}_{BR}^T, \qquad (17)$$

and $\mathbf{f}_B$ is given by

$$\mathbf{f}_B = \mathbf{b}_B - \mathbf{A}_{BR}\mathbf{D}_R^{-1}\mathbf{b}_R. \qquad (18)$$

The advantages to forming the reduced matrix equation are (1) it contains only half the number of unknown elements for ΔC, and (2) it increases the convergence rate of the iterative solver. The disadvantage is that the **R** matrix requires ten elements of storage per equation rather than the four elements per equation required by the **A** matrix. After equation 16 is solved for ΔC_B, then ΔC_R is easily obtained from

$$\Delta C_R = \mathbf{D}_R^{-1}[\mathbf{f}_R - \mathbf{A}_{BR}^T \Delta C_B], \qquad (19)$$

where ΔC_R is the vector of changes in concentration for the red nodes and $\mathbf{f}_R$ is the right-hand-side vector for the red nodes.

Matrix Equation Solver

An iterative solution algorithm for the set of linear, symmetric, sparse-matrix equations (represented by equation 16) has been developed. It is a conjugate-gradient method with preconditioning based on incomplete Cholesky (IC) factorization, as described by Stoer and Bulirsch (1991) and Axelsson (1994). Incomplete Cholesky factorization is a powerful preconditioning technique for use with iterative methods applied to sparse linear systems of equations (Meijerink and Van der Vorst, 1977). The preconditioned system is

$$\mathbf{M}\mathbf{R}\Delta C_B = \mathbf{M}\mathbf{b}_B, \qquad (20)$$

where **M** is an approximate inverse of **R**. For incomplete Cholesky preconditioning,

$$\mathbf{M} \approx \left[\mathbf{L}\mathbf{D}\mathbf{L}^T\right]^{-1}, \qquad (21)$$

where **L** is the lower triangular factor of matrix **R**; that is,

$$\mathbf{L}\mathbf{D}\mathbf{L}^T = \mathbf{R}. \qquad (22)$$

Different orders of IC factorization can be employed (Meijerink and Van der Vorst, 1981). Order one means no fill-in is allowed of any zero (null) element locations in the original **R** matrix during the IC factorization. Order one IC factorization, used here, corresponds to method ICCG(1,1) of Meijerink and Van der Vorst (1981). Dupont and others (1968) proposed a modified incomplete factorization (MIC) with the requirement that *rowsum*(**M**) = *rowsum*(**R**), where *rowsum* (**M**) means the sum of the elements along a row of matrix **M**. This requirement is met by adding any discarded elements from the IC factorization to the diagonal element of the corresponding row. MIC preconditioning is the method used in the present version of the solver.

Choosing the Parameters for the Iterative Equation Solver

As with any iterative solver, a convergence criterion and maximum number of iterations need to be specified. In addition, the red-black renumbering scheme has six possible permutations of the renumbering sequence for forming the reduced matrix. Short trial simulations of one or two time steps are sufficient to find the optimum choice for fastest convergence. Actually, the six choices usually group into three pairs of nearly equal iteration counts.

The tolerance for convergence of the iterative solver sets the maximum acceptable value for the Euclidean norm of the residual vector. The residual vector is defined as

$$\mathbf{r} = \mathbf{b}_B - \mathbf{R}\Delta C_B, \qquad (23)$$

and the Euclidean norm is defined as

$$\|\mathbf{r}\|_2 = \left(\sum_{i=1}^{k} \mathbf{r}_i^2\right)^{1/2}. \qquad (24)$$

Actually, the norm of the residual is scaled to the Euclidean norm of the initial residual vector. Thus,

$$\left(\frac{\mathbf{r}}{\mathbf{r}_0}\right) = \left(\frac{\mathbf{r}}{\mathbf{b}}\right) \le \varepsilon \ , \qquad (25)$$

where ε is the convergence tolerance.

Experience with test problems has shown that a convergence tolerance of 10^{-5} to 10^{-7} is necessary to obtain three or four digit agreement with results of a direct solver. No algorithm is presently available to set this tolerance based on the problem specifications. For large problems, use of a direct solver is impractical, therefore the user must experiment with several values of the convergence tolerance to determine the largest value that provides agreement to the desired number of digits with a more accurate solution obtained using a smaller tolerance. Finally, the iteration limit prevents run away conditions when the convergence rate becomes very slow. Under some conditions, the user may need to double or triple the suggested limit of 100. More than a few hundred iterations, however, indicates that adjustments probably need to be made in the spatial or temporal discretization.

Accuracy Criteria

One advantage of solving the dispersion equation implicitly is that this formulation is unconditionally stable. In contrast, the explicit formulation requires the observance of stability criteria, which limit the allowable time step (Konikow and others, 1996). Thus, the implicit algorithm allows for significantly larger time steps during the simulation. It should be noted that stability does not imply accuracy. Solution accuracy decreases as the time step increases. A stable solution may be based on such a large time step that its accuracy is very poor.

When the implicit differencing is implemented using a time weighting near 0.5, a potential exists for stable oscillations to be produced in the concentration solution. Also, for all amounts of implicit temporal differencing, the use of a symmetric spatial differencing for the cross-product terms of the dispersion tensor gives a potential for overshoot and undershoot in the calculated concentration solution, particularly when the velocity field is oblique to the axes of the grid. Remedies for excessive overshoot and undershoot are: (1) suppress the calculation of the cross-dispersive flux terms, or (2) refine the finite-difference mesh. Option 1 requires patching the source code and option 2 may lead to excessively long simulation times.

An accuracy criterion incorporated in both Version 1 and Version 2 of *MOC3D* constrains the movement of particles during each time step. For reasons described by Konikow and others (1996), an accurate computation of concentration changes caused by advective transport requires the maintenance of a relatively uniformly spaced field of marker particles that are moving along relatively smooth and continuous pathlines. Thus, a restriction must be placed on the size of the time step to ensure that the distance a particle moves in the *x*-, *y*-, or *z*-directions does not exceed some critical distance, related to the grid spacing. The simulator allows the user to specify the value of this critical distance (named *CELDIS* in the code and input instructions). This translates into a limitation on the time-step length. If the time step used to solve the flow equation exceeds the time limit, the time step will be subdivided into an appropriate number of equal-sized smaller time increments.

The original explicit algorithm in *MOC3D* also includes a stability criterion (Konikow and others, 1996, p. 24, equation 61) that constrains Δt for the explicit finite-difference solution of the term describing concentration changes due to fluid sources. This check assures that not more than one pore volume of fluid is displaced by fluid injection (recharge) during any single time increment. In the implicit solution, one term of the finite-difference approximation is at the n+1 time level. This relaxes the stability criterion by a factor of two. Therefore, in the code implementing the implicit solution algorithms for dispersive flux, we have incorporated an automatic check of the magnitude of fluid sources, and constrain the transport time increment, if necessary, to assure that not more than one pore volume is displaced in 0.5 Δt.

Mass Balance

As described by Konikow and others (1996), mass-balance calculations are performed to help check the numerical accuracy and precision of the solution. One modification of the previously described mass-balance calculations has been implemented to assure consistency with the new implicit algorithm. In calculating the cumulative mass flux out of the system, the explicit procedure assumes that the concentration associated with a fluid sink is $C^{n}_{j,i,k}$, the node concentration at the beginning of the time increment (see Konikow and others, 1996, equation 66). If the implicit solver is selected, however, the code will assume that the concentration associated with a fluid sink is the average concentration during the time increment, $\left(C^{n}_{j,i,k} + C^{n+1}_{j,i,k}\right)\Big/2$.

Review of *MOC3D* Assumptions

The assumptions that have been incorporated into Version 2 of the *MOC3D* simulator are the same as for Version 1. These are relevant to both grid design and model application. Efficient and accurate application of *MOC3D* requires the user to be aware of these assumptions. Therefore, the user should review the description of these items as presented by Konikow and others (1996).

COMPUTER PROGRAM

MOC3D Version 2 is implemented as a package for *MODFLOW*. *MOC3D* uses the flow components calculated by *MODFLOW* to compute velocities across each cell face in the transport domain. The computed velocities are used in an interpolation scheme to move each particle a distance and direction with time to represent advection. The transport mechanisms of fluid sources, dispersion, and decay are subsequently applied to the concentrations associated with the particles.

A separate executable version of *MODFLOW*, which is adapted to link with and use the *MOC3D* package, must first be created to run *MOC3D* Version 2 simulations. The *MOC3D* code is written in standard FORTRAN-77, and it has been successfully compiled and executed on multiple platforms, including 486- and Pentium-based personal computers, Macintosh personal computers, and Data General and Silicon Graphics Unix workstations. FORTRAN compilers for each of these platforms vary in their characteristics and may require the use of certain options to compile *MOC3D* successfully. For instance, the compiler should initialize all variables to zero. Depending on the size of the X-array (defined by LENX in the *MODFLOW* source code), options to enable the compiler to handle large-array addressing may be needed. Most real variables in *MOC3D* are defined as single precision variables in the FORTRAN code. In our experience, use of double-precision definitions for most variables has not been necessary.

Implementing *MOC3D* requires the use of a separate file that contains file names similar to the one used in *MODFLOW*. The principal *MOC3D* input data (such as subgrid dimensions, hydraulic properties, and particle information) are read from the main *MOC3D* data file. Other files are used for observation wells, concentrations in recharge, and several input and output options. Detailed input-data requirements and instructions are presented in Appendix A.

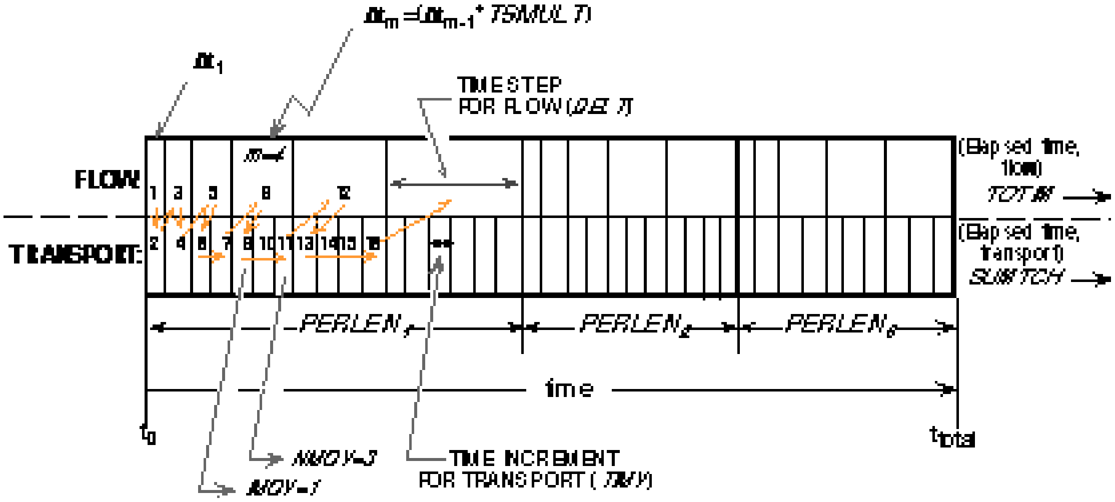
Version 2 also incorporates a new option in *MOC3D* for the format of concentration data written to a separate output file. This new option, which is highlighted in Appendix A, allows data to be saved as a table of values in which each line (or record) contains the location of a node and the value of concentration at that node. This will facilitate three-dimensional visualization of the output because this new format is compatible with many commercially available software packages that enable rendering and viewing of three-dimensional data sets.

The input data set used for the first test problem (involving one-dimensional steady flow) is included in Appendix B to provide the reader with an illustrative example.

MOC3D output is routed to a main file, separate from the *MODFLOW* main output file, and optionally to additional output files. Appendix C contains output from the example input data set contained in Appendix B.

General Program Features

Because the model is based on the assumption that the fluid properties (such as density and viscosity) are constant and uniform and independent of changes in concentration, the head distribution and flow field are independent of the solution to the solute-transport equation. Therefore, the flow and transport equations can be solved sequentially, rather than simultaneously. Because transport depends on fluid velocity, which is calculated from the solution to the flow equation, the flow equation must be solved first. The sequence is illustrated in figure 3 for a hypothetical problem involving transient flow and three stress periods. The numbered sequence from 1



$\Delta t_m = (\Delta t_{m-1} * TSMULT)$
Δt_1
TIME STEP FOR FLOW (DELT)
m=4
FLOW
1
3
5
8
12
TRANSPORT:
2
4
6
7
9
10
11
13
14
15
16
(Elapsed time, flow) TOTIM
(Elapsed time, transport) SUMTCH
PERLEN1
PERLEN2
PERLEN3
time
t_0
t_{total}
NMOV=3
IMOV=1
TIME INCREMENT FOR TRANSPORT (TIMV)

Figure 3. Double time-line illustrating the sequence of progression in the *MOC3D* model for solving the flow and transport equations. This example is for transient flow and three stress periods (*NPER* = 3) of durations *PERLEN*$_1$, *PERLEN*$_2$, and *PERLEN*$_3$. Each time step for solving the flow equation (of duration *DELT*) is divided into one or more time increments (of duration *TIMV*) for solving the transport equation; all particles are moved once during each transport time increment. For illustration purposes, the sequence of solving the two equations is labeled for the first five time steps of the first stress period, and the indices for counting time steps for flow and time increments for transport are labeled for the fourth time step (from Konikow and others, 1996).

through 16, which starts at the left edge of the double time line, illustrates the order of equation solution for the first five time steps of the first stress period. This figure illustrates the nomenclature used for time parameters in *MODFLOW* and *MOC3D*, as well as the relation between them.

The implicit solution to the flow equation in *MODFLOW* generally allows the use of time steps of increasing length during a given stress period. The length of the first time step for solving the flow equation is calculated by *MODFLOW* on the basis of user-defined values for the number of time steps (NSTP), a time-step multiplier (TSMULT), and the length of the stress period (PERLEN). After the flow equation is solved for the first time step (Δt_1), the *MOC3D* simulator compares the length of this time step with the limitations imposed by the accuracy criteria for solving the transport equation. If the limitations are exceeded, *MOC3D* will subdivide the flow time step into the minimum number of equal-sized time increments that meet the criteria. In the example shown in figure 3, the first two time steps for flow are sufficiently small so that the transport equation can use a time increment of the same duration as the flow time step (that is, TIMV = DELT). As this time-marching sequence progresses and time-step lengths are increased, the accuracy time-step constraints are eventually exceeded. Figure 3 shows that after the second time step, the transport equation had to be solved over shorter time increments than the flow equation.

As mentioned previously, transport may be simulated within a subgrid, which is a “window” within the primary *MODFLOW* grid used to simulate flow (see Konikow and others, 1996, figure 9). The grid dimensions are limited only by the size of the “X” array (see “Space Allocation” in the *MODFLOW* documentation).

Program Segments

MOC3D Version *2* input and output utilizes the standard *MODFLOW* array reading and writing utilities as much as possible. Konikow and others (1996) describe briefly each of the subroutines in *MOC3D* that are used for ten different categories of functions. Following is a brief description of the additional subroutines that are added to form Version 2 of the *MOC3D* code. In addition, several existing *MOC3D* Version 1 subroutines were modified to create Version 2.

New subroutines related to dispersion calculations are listed in table 1. Dispersion coefficients are determined on cell faces. To improve efficiency, however, the dispersion coefficients are lumped with the porosity, thickness, and an appropriate grid dimension factor of the cell into combined parameters called “dispersion equation coefficients.” For example, the dispersion equation coefficient for the $j+1/2,i,k$ face in the column direction is

$$\frac{\left(\varepsilon b D_{xx}\right)_{j+1/2,\,i,k}}{\Delta x}.$$

These combined coefficients are the ones that are written to the output files.

MODFLOW source and sink packages contain an option called CBCALLOCATE. When used, the package will save the cell-by-cell flow terms across all faces of every source or sink cell. *MOC3D* uses these fluid fluxes to calculate solute flux to or from the source/sink nodes. Because these individual solute fluxes are required to compute the solute mass balance, the CBCALLOCATE option must always be selected when using *MOC3D.* Implicit calculations of concentration changes at nodes caused by mixing with fluid sources are controlled by the “SRC” subroutine in table 2.

Subroutines related to initialization of the iterative solver for the implicit dispersion

Table 1. *MOC3D* subroutines controlling implicit dispersion calculations

Subroutine	Description	Called from
LOADT	Build solute conductance factors and load into one-dimensional arrays	MOC5LDAS
ASEMBL	Assemble matrix coefficients and right hand side vector for the dispersion equation	MOC5LDAS
RHSN	Calculate explicit terms for right hand side of matrix equation using intermediate concentrations (*C**)	MOC5MVOT

Table 2. *MOC3D* subroutine controlling implicit calculations relating to sources and sinks

Subroutine	Description	Called from
SRC5IAP	Calculate changes in concentration due to implicit flux terms at fluid sources	MOC5MVOT

Table 3. *MOC3D* subroutines controlling initialization of iterative solver for implicit dispersion calculation

Subroutine	Description	Called from
INIT	Initialize static data for solver	MOC5RPCK
REORDR	Perform red-black reordering of nodes based on the selected permutation of the coordinate directions	INIT
LDCI	Load array for connection indices based on renumbered mesh	REORDR
LDCIR	Load array for connection indices for reduced matrix	REORDR
LDIND	Load array with natural numbering for selected permutation of the coordinate directions	REORDR
RBORD	Map natural node number into red-black node number	REORDR

equation on a reduced matrix are listed in table 3. Table 4 lists subroutines related to operation of the iterative solver itself.

Also, the main *MODFLOW* routine in *MOC3D* Version 2 has been shortened. Many of the calls to *MOC3D*-specific subroutines were combined into new modules designed to control the linkages between the flow and transport routines. This simply minimizes the number of calls from the main *MODFLOW* routine; there is no change in the functionality of the overall code. These changes are beneficial to programmers, but are invisible to model users. The new transport-controlling subroutines are listed in table 5.

MODEL TESTING AND EVALUATION

The *MOC3D* Version 2 simulator was tested and evaluated by running the same suite of test cases as was applied to *MOC3D* Version 1 by Konikow and others (1996). This suite included base results generated by analytical solutions and by other numerical models. It spanned a range of conditions and problem types so that the user will gain an appreciation for both the strengths and weaknesses of this particular code. It should be noted that all test cases involve steady flow conditions.

Table 4. *MOC3D* subroutines and functions for iterative solver on reduced matrix

Subroutine	Description	Called from
CGRIES	Solve the matrix equation using the conjugate-gradient method	MOC5MVOT
ABMULT	Multiply black node matrix times a vector	CGRIES
ARMULT	Multiply red node matrix times a vector	CGRIES
DBMULT	Multiply black node diagonal matrix times a vector	CGRIES
FORMR	Form reduced matrix	CGRIES
LSOLV	Solve lower triangular black-node matrix equation	CGRIES
RFACTM	Factor reduced matrix using modified incomplete decomposition	CGRIES
USOLV	Solve upper triangular black-node matrix equation	CGRIES
VPSV	Calculate a vector plus a scalar times a vector	CGRIES
MTOIJK	Return the index (i,j,k) of the point with natural index M	several
XIP	Calculate a vector inner product (function)	several

Table 5. *MOC3D* subroutines controlling overall solute-transport calculations

Subroutine	Description	Called from
MOC5RPCK	Control calls for reading and preparing data and for performing consistency checks	main routine
MOC5INIT	Control calls for initializing transport terms	main routine
MOC5VELO	Control calls for calculating fluxes and printing velocities	main routine
MOC5DISP	Control calls for calculating and printing dispersion coefficients	main routine
MOC5LDAS	Load solute conductance factors and assemble matrix for implicit solution of the dispersion equation	main routine
MOC5MVOT	Control calls for advective transport (particle tracking), total concentration change, solute mass balance, and solute output	main routine

One-Dimensional Flow

The first test case evaluates *MOC3D* for a relatively simple system involving one-dimensional solute transport in a finite-length aquifer having a third-type source boundary condition, as described by Konikow and others (1996). The numerical results are compared to an analytical solution by Wexler (1992, p. 17). The length of the system is 12 cm; other parameters are summarized in table 6. The solute-transport equation was solved using *MOC3D* Version 2 on a 120-cell subgrid to assure a constant velocity within the transport domain and to allow an accurate match to the boundary conditions of the analytical solution.

Two different values of dispersion coefficients were evaluated in the first set of tests. The values were $D_{xx} = 0.1$ and 0.01 cm^2/s, which are equivalent to $\alpha_L = 1.0$ and 0.1 cm, respectively. Breakthrough curves showing concentration changes over time at three different distances from the boundary for the lower dispersion case, as calculated with both analytical and numerical solutions, are compared in figure 4. Numerical results using the implicit dispersion solver were generated using both the Crank-Nicolson and backward-

Table 6. Parameters used in implicit *MOC3D* simulation of solute transport in a one-dimensional, steady-state flow system

Parameter	Value
$T_{xx} = T_{yy}$	0.01 cm^2/s
ε	0.1
α_L	0.1 cm
$\alpha_{TH} = \alpha_{TV}$	0.1 cm
PERLEN (length of stress period)	120 s
V_x	0.1 cm/s
$V_y = V_z$	0.0 cm/s
Initial concentration (C_0)	0.0
Source concentration (C')	1.0
Number of rows	1
Number of columns	122
Number of layers	1
DELR (Δx)	0.1 cm
DELC (Δy)	0.1 cm
Thickness (b)	1.0 cm
NPTPND (Initial number of particles per cell)	3
CELDIS	0.5
INTRPL (Interpolation scheme)	2
FDTMTH	0.5
NXCIT	2
IDIR	1
EPSSLV	1 x 10^{-5}
MAXIT	100

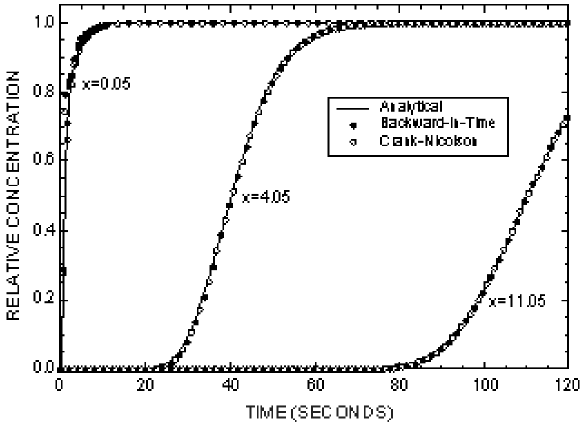


Figure 4. Numerical (implicit) and analytical solutions at three different locations for solute transport in a one-dimensional, steady flow field. Parameter values for this base case are listed in table 6.

in-time approximations (specified by the value of *FDTMTH* in the input data). To improve clarity, this plot only shows every fourth data point for the numerical model results, except for the curve for x = 0.05 cm, where every data point is shown for times less than 10 seconds and every tenth data point is shown for times greater than 10 seconds. Note that this distance (x = 0.05) represents the first node downgradient from the source location. There is a very close match between the numerical and analytical solutions. At early times and short distances the numerical solution exhibits some oscillation about the mean, which is related to the discrete nature of the particles used to represent the advection process. This small loss of precision, however, is not a cumulative error, as it vanishes after moderate travel times or distances. For this case, the results obtained using the explicit solution for dispersive fluxes yield an almost identical numerical solution (see Konikow and others, 1996, figure 18). All numerical solutions required 241 time increments to solve the transport equation because CELDIS was always the limiting criteria.

The results for the higher dispersion case are presented in figure 5. For clarity, only every tenth data point is shown for the numerical solutions at x = 4.05 cm and x = 11.05 cm and for the solution at x = 0.05 cm for times greater than 50 seconds (every data point is shown for x = 0.05 cm for times less than 50 seconds). In general, the implicit numerical results show a very close agreement with the analytical solution. The only notable exception is for x = 0.05 cm (representing the first node of the grid downstream from the solute and fluid source) at early times, where the backward-in-time solution shows concentrations that are too high. This

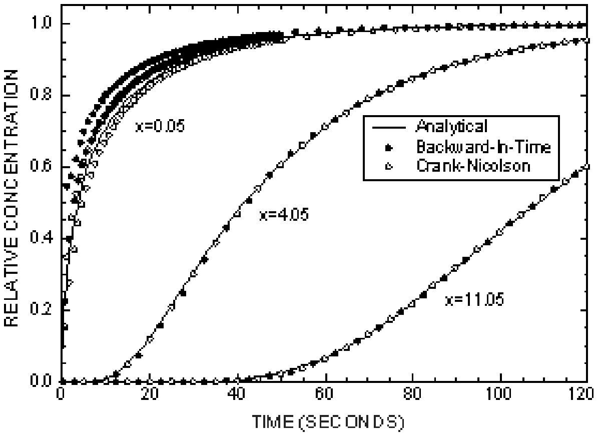


Figure 5. Numerical (implicit) and analytical solutions at three different locations for solute transport in a one-dimensional, steady flow field for case of increased dispersivity (α_L = 1.0 cm, D_{xx} = 0.1 cm^2/s, and other parameters as defined in table 6).

discrepancy, however, disappears for later times and at longer distances. The oscillations and loss of precision at nodes very close to the source are related to the discrete nature of the particles used to represent advection, as discussed by Konikow and others (1996, p. 43-44). The magnitude of the oscillations diminishes over time as dispersion reduces the local concentration gradients.

This high-dispersion case illustrates the relative computational efficiency of the implicit formulation relative to the explicit formulation using *MOC3D.* In the latter case, the dispersion coefficient imposed the limiting stability criteria, and 2401 time increments (or particle moves) were required to solve the transport equation. The implicit solver of *MOC3D* (Version 2), however, required only 241 moves. The explicit results are virtually identical to the implicit Crank-Nicolson concentrations.

Konikow and others (1996) also present the results of these tests in the form of concentration profiles in space at various times and for various retardation factors (see their figures 22 and 23). Replication of these tests using the implicit formulation yields almost as good of a match to the analytical solution as, for example, seen in figure 6 for the nonreactive case. In figure 6, only every fourth data point is shown, except for t = 6 s, where every data point is shown for distances less than 1.5 cm. For brevity, the comparisons for reactive cases are not presented here.

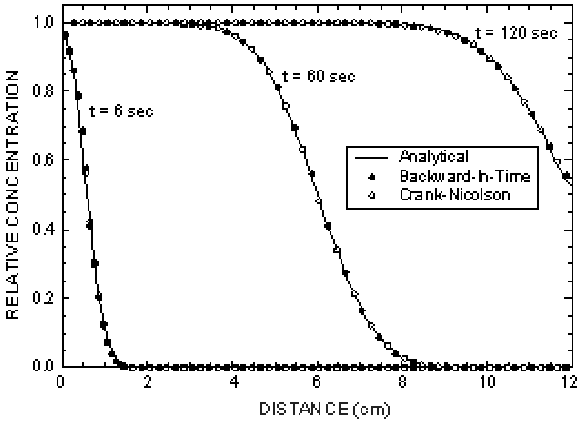


Figure 6. Numerical (implicit) and analytical solutions for three different times for same one-dimensional, steady flow, solute-transport problem shown in figure 4.

The effect of decay is evaluated by specifying the decay rate as λ = 0.01 s^{-1} for the same low-dispersion, no sorption, problem as defined for figure 4. These results are presented in figure 7, which shows excellent agreement between the analytical and implicit numerical solutions. Only every fourth data point is plotted in figure 7 for the numerical results. There is no discernible difference between the Crank-Nicolson and backward-in-time solutions, and both match the analytical solution very closely. Also, the implicit solutions closely match the explicit solutions, and both required the same number of time increments (241) to solve the transport equation.

Uniform, Three-Dimensional Flow

To evaluate and test *MOC3D* Version 2 with implicit dispersive transport calculations for three-dimensional cases, we compared numerical results with those of the analytical

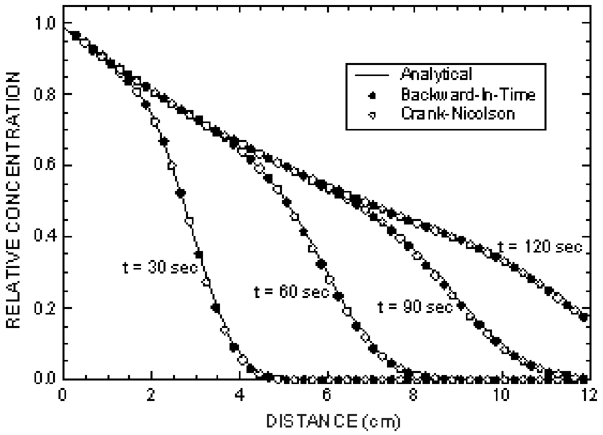


Figure 7. Numerical (implicit) and analytical solutions for four different times for solute transport in a one-dimensional, steady flow field for case with decay at rate of $\lambda = 0.01\ s^{-1}$.

solution developed by Wexler (1992) for the case of three-dimensional solute transport from a continuous point source in a steady, uniform flow field in a homogeneous aquifer of infinite extent. Konikow and others (1996) note that this evaluation primarily is a test of the accuracy of the calculated dispersive flux in three directions because the flow field is aligned with the grid. The problem and analytical solution are described in detail by Konikow and others (1996, p. 45-48); the parameters and boundary conditions for this test case are summarized in table 7.

Table 7. Parameters used in implicit *MOC3D* simulation of transport from a continuous point source in a three-dimensional, uniform, steady-state flow system

Parameter	Value
$T_{xx} = T_{yy}$	0.0125 m^2/day
ε	0.25
α_L	0.6 m
α_{TH}	0.03 m
α_{TV}	0.006 m
PERLEN (length of stress period)	400 days
V_x	0.1 m/day
$V_y = V_z$	0.0 m/day
Initial concentration (C_0)	0.0
Source concentration (C')	2.5×10^6 g/m^3
Q (at well)	1.0×10^{-6} m^3/d
Source location	row 8, column 1, layer 1
Number of rows	30
Number of columns	12
Number of layers	40
DELR (Δx)	3 m
DELC (Δy)	0.05 m
Thickness (*b*)	1.0 cm
NPTPND (Initial number of particles per cell)	3
CELDIS	0.1
INTRPL (Interpolation scheme)	1
FDTMTH	0.5
NXCIT	2
IDIR	1
EPSSLV	1×10^{-5}
MAXIT	100

The results of *MOC3D* are compared graphically with those of the analytical solution for the *x*-*y* plane passing through the point source in figure 8. Figure 8a shows the concentrations in this plane as calculated using the analytical solution and figure 8b shows the same for the Crank-Nicolson implicit *MOC3D* results (the backward-in-time results were indistinguishable from those in figure 8b, so are not reproduced here). The results agree very closely, although a slightly greater distance of transport or spreading is evident in the *MOC3D* results, both upstream as well as downstream of the source. Part of this difference, however, can be explained simply by the fact that the source is applied over a larger area in the horizontal plane of the *MOC3D* model, in which the length of the source cell is 3 m in the direction parallel to flow, whereas the source is represented as a true point in the analytical solution.

The results obtained using the explicit *MOC3D* formulation are presented by Konikow and others (1996, figure 25b). The

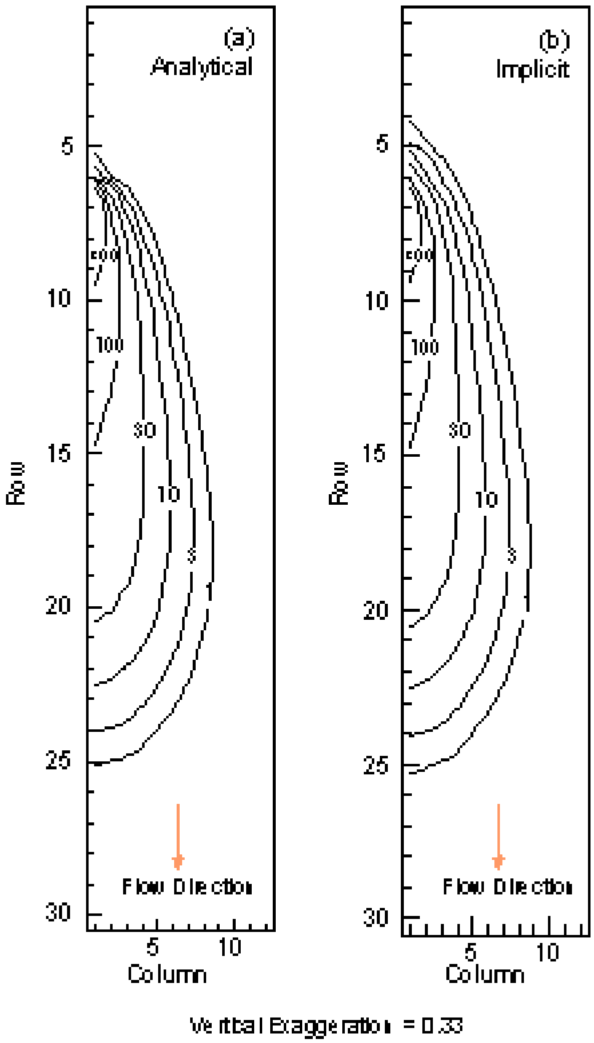


Figure 8. Concentration contours for (a) analytical and (b) numerical solutions in the horizontal plane containing the solute source (layer 1) for three-dimensional solute transport in a uniform steady flow field. Parameters are defined in table 7.

explicit and implicit results are nearly identical. The explicit solution method used 207 time increments and the implicit solution methods used only 134, so the implicit method is somewhat more efficient for this case.

Konikow and others (1996) also present comparisons for this case for vertical planes parallel and perpendicular to the flow direction. The comparisons with the implicit results are as close as in figure 8, and so for brevity are not reproduced here.

Two-Dimensional Radial Flow

A radial flow and dispersion problem was also used to compare the implicit dispersive transport *MOC3D* solution to the analytical solution given by Hsieh (1986) for a finite-radius injection well in an infinite aquifer of two dimensions. The problem involves flow from a single injection well; the velocities vary in space and are inversely related to the distance from the injection well.

The parameters for the problem are summarized in table 8 and the analytical solution and other details about this test case are presented by Konikow and others (1996, p. 49-50). The problem was again modeled using

Table 8. Parameters used in implicit *MOC3D* simulation of two-dimensional, steady-state, radial flow case

Parameter	Value
$T_{xx} = T_{yy}$	3.6 m^2/hour
ε	0.2
α_L	10.0 m
$\alpha_{TH} = \alpha_{TV}$	10.0 m
PERLEN (length of stress period)	1000 hours
Q (at well)	56.25 m^3/hour
Source concentration (C')	1.0
Number of rows	30
Number of columns	30
Number of layers	1
DELR (Δx) = *DELC* (Δy)	10.0 m
Thickness (b)	10.0 m
NPTPND (Initial number of particles per cell)	46
CELDIS	0.5
INTRPL (Interpolation scheme)	2
FDTMTH	1.0
NXCIT	2
IDIR	1
EPSSLV	1 x 10^{-5}
MAXIT	100

a grid having 30 cells in the x-direction and 30 cells in the y-direction, representing one quadrant of the radial flow field (90 of 360 degrees). Initial particle positions were defined using the custom particle placement option as described by Konikow and others (1996). The implicit solutions using Crank-Nicolson and backward-in-time differencing were essentially identical, and both matched the analytical solution almost exactly (see Konikow and others, 1996, figure 29). The implicit solutions of *MOC3D* also agree very closely with the explicit dispersive transport solution of *MOC3D* presented by Konikow and others (1996). The explicit solution, however, required 596 time increments, whereas the implicit solutions required only 282 time increments.

Point Initial Condition in Uniform Flow

A problem including three-dimensional solute transport from an instantaneous point source, or Dirac initial condition, in a uniform flow field was used as another test problem. An analytical solution for an instantaneous point source in a homogeneous infinite aquifer is given by Wexler (1992, p. 42), who presents the *POINT3* code for a related case of a continuous point source. The *POINT3* code was modified to solve for the desired case of an instantaneous point source. Test problems were designed to evaluate the numerical solution for two cases—one in which flow is parallel to the grid (in the x-direction) and one in which flow occurs at 45 degrees to the x- and y-axes. This allows us to evaluate the accuracy and sensitivity of the numerical solution to the orientation of the flow relative to the grid. The assumptions and parameters for this test case are summarized in table 9 and are described in more detail by Konikow and others (1996).

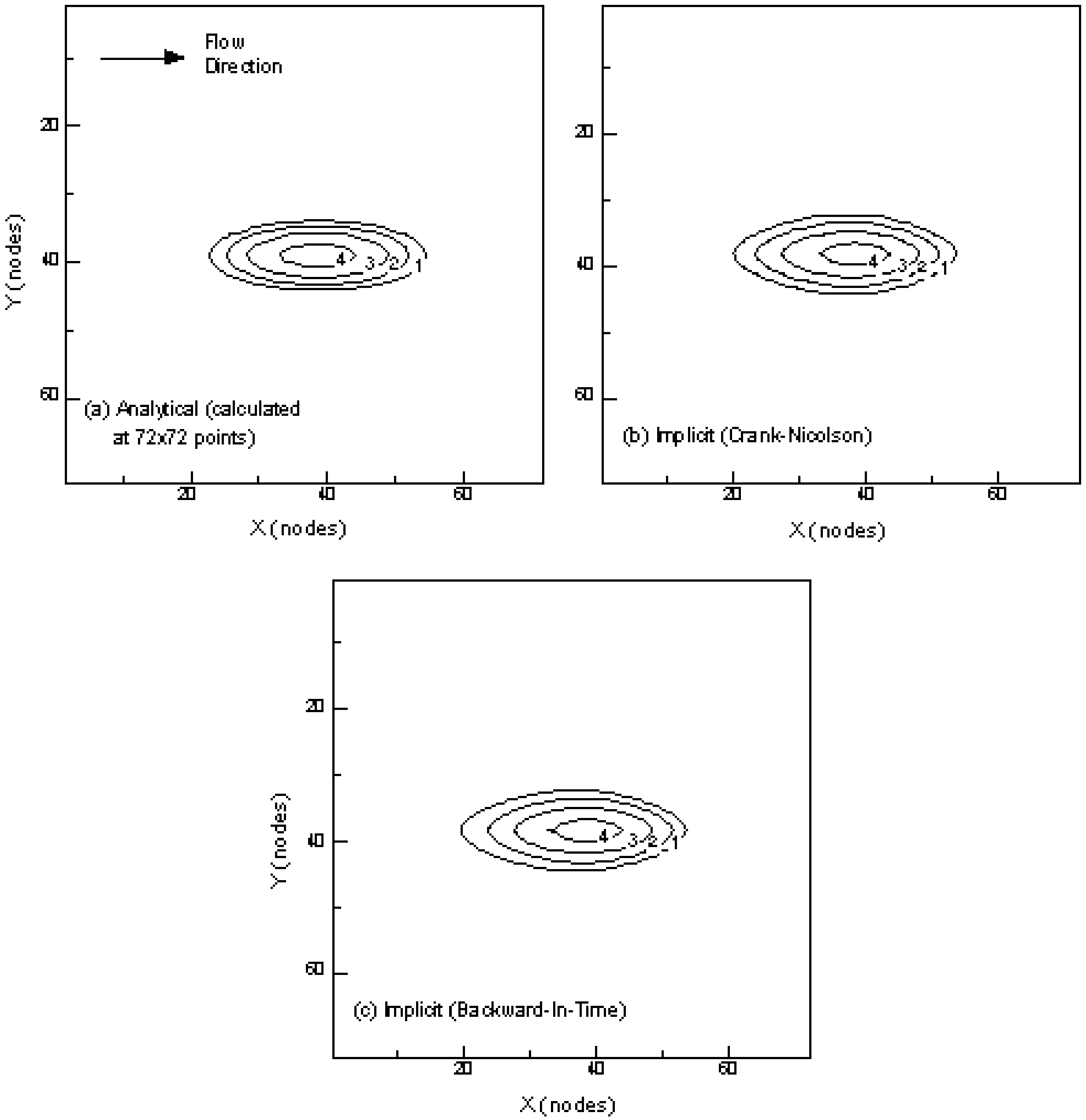
Table 9. Parameters used in implicit *MOC3D* simulation of three-dimensional transport from a point source with flow in the x-direction and flow at 45 degrees to x- and y-axes

Parameter	Value
$T_{xx} = T_{yy}$	10.0 m^2/day
ε	0.1
α_L	1.0 m
$\alpha_{TH} = \alpha_{TV}$	0.1 m
PERLEN (length of stress period)	90 days
V_x	1.0 m/day
$V_y = V_z$	0.0 m/day*
Initial concentration at source	1×10^6
Source location	x = 30 m, y = 120 m, z = 40 m**
Number of rows	24 and 72
Number of columns	24 and 72
Number of layers	24
DELR (Δx)	10.0 and 3.33 m
DELC (Δy)	10.0 and 3.33 m
Thickness (b)	10.0 m
NPTPND (Initial number of particles per cell)	8
CELDIS	0.5
INTRPL (Interpolation scheme)	2
FDTMTH	0.5
NXCIT	2
IDIR	1
EPSSLV	1×10^{-7}
MAXIT	100

* For flow at 45 degrees to x- and y-axes, V_y = 1.0 m/day

** For flow at 45 degrees to x- and y-axes, the source location is x = 30 m, y = 30 m, and z = 120 m.

As described by Konikow and others (1996), we specified for the test case of flow in the x-direction that V_x = 1.0 m/d, and $V_y = V_z$ = 0.0 m/d. For flow at 45 degrees to x and y, we specified $V_x = V_y$ = 1.0 m/d, and V_z = 0.0 m/d. For both cases, the distance the center of mass of the plume travels in the x-direction is the same for equal simulation times. Note, however, that the magnitude of velocity is


(a) Analytical (calculated at 72x72 points)

(b) Implicit (Crank-Nicolson)

(c) Implicit (Backward-In-Time)

Figure 9. Concentration contours for (a) analytical and (b, c) numerical solutions for transport of a point initial condition in uniform flow in the *x*-direction. The *z*-component of flow is zero, but there is dispersion in all three directions. Contour values are the log of the concentrations.

higher in the latter case; therefore, there will be more dispersion in that problem during an equivalent time interval.

The results for both the analytical and numerical solutions for the case in which flow occurs only in the *x*-direction are shown in figure 9. The implicit dispersive transport algorithms in *MOC3D* gave results for a 72 by 72 grid that are very close to those of the analytical solution (figure 9a). The numerical results, however, do show some slight spreading (or numerical dispersion) relative to the analytical solution in both the transverse and longitudinal directions. The Crank-Nicolson solution (figure 9b) agrees very closely with the backward-in-time solution (figure 9c), except for a slightly longer tail on the highest contour in the backward-in-time solution. The implicit *MOC3D* solutions also closely match the explicit *MOC3D* results presented by Konikow and others (1996, figure 30d). All numerical solutions took 56 time increments to solve the transport equation and CELDIS was the limiting criteria in all cases.

The results of the test problem for flow at 45 degrees to the grid are shown in figure 10. The analytical solution (figure 10a), which provides the basis for the evaluation, was solved on a 72× 72 grid, and the implicit *MOC3D* solutions are shown for a 24× 24× 24 grid (figure 10b) and a 72× 72× 24 grid

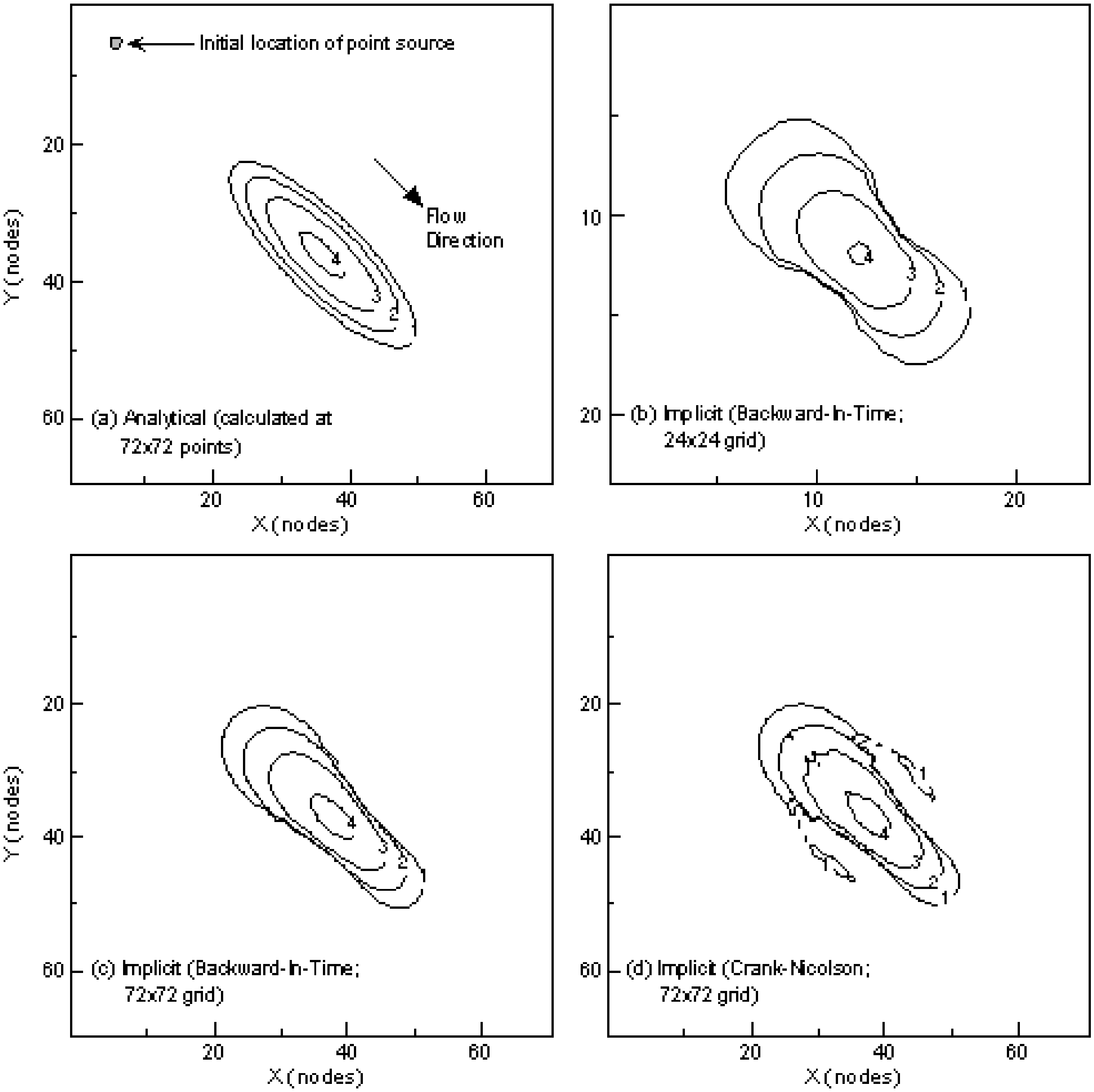


Figure 10. Concentration contours for (a) analytical and (b, c, d) numerical solutions for transport of a point initial condition in uniform flow at 45 degrees to *x* and *y*. Contour values are the log of the concentrations.

(figures 10c and 10d). Unlike the previous case (where flow is aligned with the grid), the numerical results in figure 10 show a noticeable difference in the shape of the plume relative to the analytical solution. The numerically calculated “hourglass” shape is characteristic of a grid-orientation effect and is caused primarily by the off-diagonal (cross-product) terms of the dispersion tensor. When flow is oriented parallel to the grid, or when longitudinal and transverse dispersivities are equal, the cross-product terms of the dispersion equation are zero. Because flow is at 45 degrees to the grid, the cross-product dispersive flux terms are of maximum size and negative concentrations are most likely to occur. The calculated concentration field is less accurate in this case largely because the standard differencing scheme for the cross-product dispersive flux terms can cause overshoot and undershoot of concentrations. If the base (or background) is zero concentration, then undershoot will cause negative concentrations. The magnitude of this overshoot and undershoot effect is reduced by using a finer grid. For this test problem, the coarsest grid (figure 10b) exhibits too much spreading relative to the analytical solution, and the finer grid results in less numerical dispersion (although the grid-orientation effect is not eliminated). The backward-in-time solution (figure 10c) matches the analytical solution more closely than does the Crank-Nicolson solution (figure 10d), which characteristically yields more oscillatory behavior. The implicit and explicit *MOC3D* results all show a slight asymmetry in the shape of the plume in the direction of flow (that is, there is slightly less forward spreading compared to backward spreading), although this effect is less pronounced in the explicit solution (see Konikow and others, 1996, figure 31d). This asymmetry is inconsistent with symmetrical spreading found in the analytical solution, and is caused by the sequence in which the dispersive and advective equations of the transport are solved. Both explicit and implicit solutions using a 72×72 grid required 56 moves, so the simpler explicit solution was more efficient for this particular problem. The solution on the 24×24 grid required only 19 moves.

Some small areas of negative concentrations were calculated, but they do not appear in figure 10 because they are removed by using logarithmic-scale contouring. To show the extent of the areas of negative concentration, we have replotted the results illustrated in figures 10b-d in figure 11, using two types of shading for all areas where the relative concentration is less than -0.05 and less than -10.0. The coarse grid solution shows the largest area in which negative concentrations occur. The use of a finer grid reduces the magnitude of the problem, but does not eliminate it. We also tested the sensitivity of the extent of negative concentrations to the size of the transport time increment by reducing the value of CELDIS to 0.1 (the base case used a value of CELDIS = 0.5). The area over which negative concentrations occurred was only slightly smaller. The increase in execution time, however, was considerable, so the very small improvement does not appear to justify the extra computational costs.

Constant Source in Nonuniform Flow

Burnett and Frind (1987) used a numerical model to simulate a hypothetical problem having a constant source of solute over a finite area at the surface of an aquifer having homogeneous properties, but nonuniform boundary conditions, which result in nonuniform flow. Because an analytical solution is not available for such a complex system, we use their results for this test case as a benchmark for comparison with the results of applying the implicit algorithm in *MOC3D*, as was also done by Konikow and others (1996). Burnett and Frind (1987) used an alternating-direction Galerkin finite-element technique to

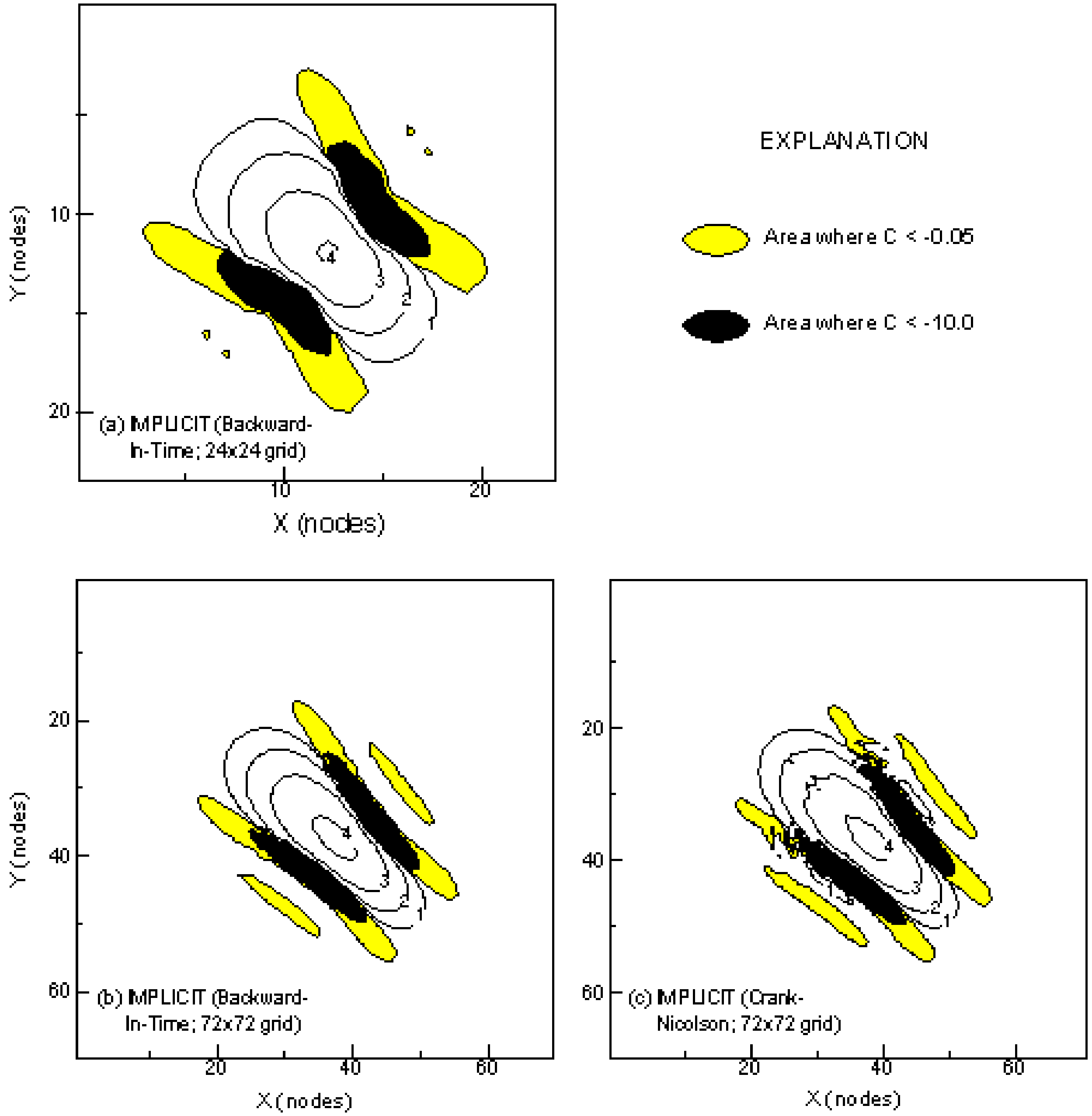


Figure 11. Concentration contours showing calculated areas of negative concentrations for same problem as represented in figure 10.

solve the flow and solute-transport equations in both two and three dimensions. Their model also includes the capability to vary α_T as a function of coordinate direction, thereby allowing this feature of *MOC3D* to be evaluated. A detailed description of the problem geometry and of the parameters for the numerical simulation using *MOC3D* are presented by Konikow and others (1996, p. 55-60).

Cases of both two- and three-dimensional transport were examined for this problem. The grids used in the implicit *MOC3D* simulations were designed to match as closely as possible the finite-element mesh used by Burnett and Frind (1987). However, some differences in discretization could not be avoided because the finite-element method uses a point-centered grid whereas *MOC3D* uses a block-centered (or cell-centered) grid. The former allows specifications of values at nodes, which can be placed directly on boundaries of the model domain. Nodes in *MOC3D* are located at the centers of cells, and block-centered nodes are always one-half of the grid spacing away from the edge of the model domain. Among the small differences arising from the alternative discretization schemes are that, in the *MOC3D* grid, (1) the modeled location of the 14.25 m long source area is offset by 0.225 m towards the right, and (2) the total length of the domain is 199.5 m.

The first analysis of this test case was a two-dimensional model. The input data values for this analysis are listed in table 10. The top flow layer consisted of constant-head nodes and the solute source. We were able to increase the efficiency of the simulation by using a custom initial particle placement of only three particles in each cell and still achieve reasonably accurate results (see Konikow and others, 1996, p. 56-57).

Results for the two-dimensional case from the implicit calculation of the *MOC3D* simulator closely match those of Burnett and Frind (1987) (see figure 12). The shape of the plume is almost exactly the same for both models. In the implicit *MOC3D* results, however, the contours extend slightly further downgradient than those of Burnett and Frind (1987). This may be attributable in part to small differences in the numerical treatment of the source between the two models. The results using the backward-in-time approximation and CELDIS = 0.5 are presented in figure 12b. This simulation required 381 time increments, and the contours exhibit some very small "wiggles." When this was run using CELDIS = 1.0, the simulation required only 191 time increments (effectively doubling the average length of the time increment), but the oscillations were noticeably worse (although these results are not shown). When this was run again using CELDIS = 0.1, the simulation required 1901 time increments; these results are presented in figure 12c. The oscillations are somewhat reduced relative to those in figure 12b, and the furthest downstream extent of the 0.9 and 0.3 contours has shifted away from the center of mass. The

Table 10. Parameters used for implicit *MOC3D* simulation of transport in a vertical plane from a continuous point source in a nonuniform, steady-state, two-dimensional flow system (described by Burnett and Frind, 1987)

Parameter	Value
K	1.0 m/day
ε	0.35
α_L	3.0 m
α_{TH}	0.10 m
α_{TV}	0.01 m
PERLEN (length of stress period)	12,000 days
Q (at well)	56.25 m^3/hour
Source concentration (C')	1.0
Number of rows	1
Number of columns[1]	141
Number of layers[1]	91
DELR (Δx)	1.425 m
DELC (Δy)	1.0 m
Thickness (b)	0.2222-0.2333 m
NPTPND (Initial number of particles per cell)	3
CELDIS	0.5
INTRPL (Interpolation scheme)	1
FDTMTH	1.0
NXCIT	2
IDIR	1
EPSSLV	1 x 10^{-7}
MAXIT	100

[1] One row and layer were allocated to defining boundary conditions, so concentrations calculated in only 140 columns and 90 layers were used for comparison.

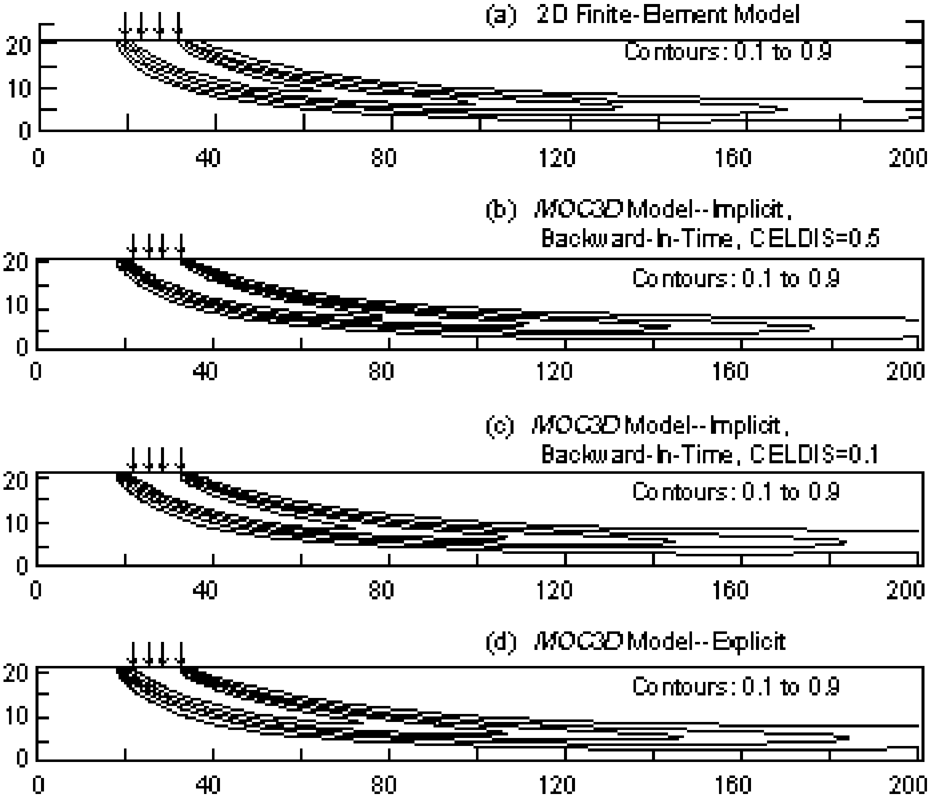


Figure 12. Two-dimensional simulation results for nonuniform-flow test case showing plume positions as contours of relative concentration: (a) finite-element model (modified from Burnett and Frind, 1987, figure 8a), and (b-d) *MOC3D*. Contour interval is 0.2 relative concentration.

simulation was also run using an even smaller value of CELDIS (0.05), which caused the model to take 3,801 time increments. But this further reduction of the size of the time increment had no discernible effect on the results. Similarly, when the simulations were made using the Crank-Nicolson difference, results were essentially identical to those obtained using backward-in-time. The results obtained using the explicit *MOC3D* solution are presented in figure 12d for comparison, and they appear to be almost identical to those in figure 12c. The explicit simulation required 4,218 time increments (dispersion was the limiting criteria). For this problem, it is clear that the implicit *MOC3D* algorithm is significantly more efficient. The implicit results show some sensitivity to the length of the time increment. In general, the user should evaluate this sensitivity for their particular problem by making several runs in which the value of CELDIS is varied.

As was done for the explicit *MOC3D* tests (Konikow and others, 1996), the implicit *MOC3D* grid was expanded to 15 rows having Δy of 1.0 m for the three-dimensional analyses of this case. Figure 13 shows the transport results in a vertical plane at the middle of the plume for both models for the case in which α_{T_V} = 0.01 m and α_{T_H} = 0.1 m. The implicit *MOC3D* results for the vertical plane in the first row are contoured in figure 13b (because of symmetry, we only simulate half of the plume, as explained by Konikow and others, 1996). The *MOC3D* plume closely matches that calculated by the finite-element model, although the former shows slightly further downstream migration of solute. Konikow and others

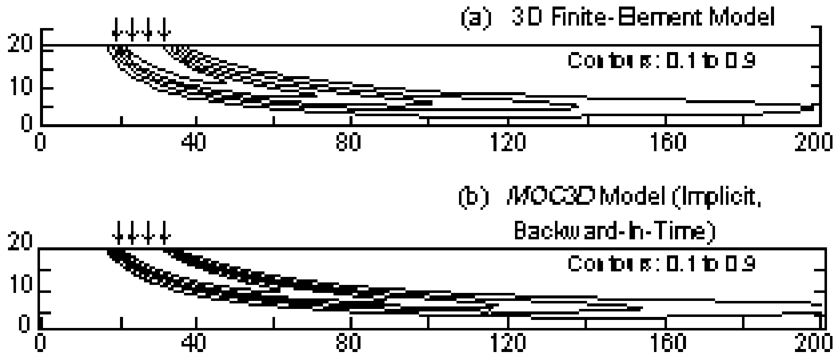


Figure 13. Three-dimensional simulation results for nonuniform-flow test case in which α_{TH} = 0.1 m and α_{TV} = 0.01 m: (a) finite-element model (modified from Burnett and Frind, 1987, figure 8c), and (b) implicit *MOC3D* solution using backward-in-time approximation and CELDIS = 0.5. Plume positions are represented by contours of relative concentration in vertical plane of three-dimensional grid; contour interval is 0.2 relative concentration.

(1996) note that there is a slight discrepancy in the basis of comparison because concentrations from *MOC3D* are evaluated at the center of the block (1/2 of a cell width from the plane of symmetry), whereas those from Burnett and Frind (1987) are evaluated on the cell faces (directly on the plane of symmetry). The Crank-Nicolson differencing scheme was also used to compute the concentration distribution, and those results matched those calculated using the backward-in-time scheme (figure 13b) almost exactly. These implicit solutions required the same number of time increments (381) as did the previously described two-dimensional simulation. The explicit solution (Konikow and others, 1996, figure 36b), however, required 4,218 time increments and was clearly less efficient for achieving equivalent accuracy.

Figure 14 shows the results for the case in which the vertical transverse dispersivity is increased by a factor of ten, so that $\alpha_{TH} = \alpha_{TV}$ = 0.1 m. Overall, the implicit dispersive transport *MOC3D* results (figure 14b) agree very closely with those of Burnett and Frind (1987) (figures 14a). As before, the results shown in figure 14b for the backward-in-time approximation agree almost exactly with those obtained using the Crank-Nicolson implicit solution and the explicit *MOC3D* solution.

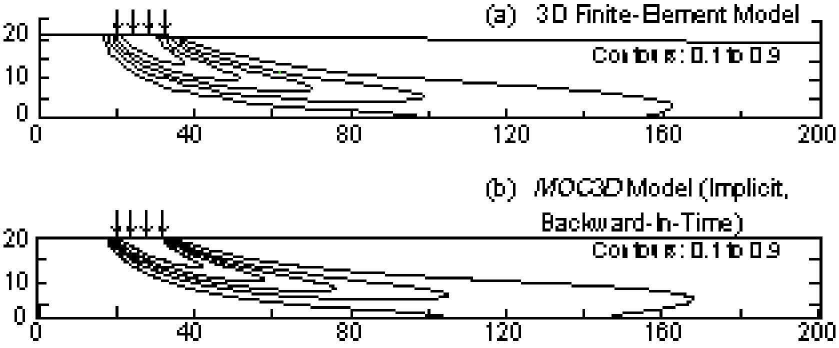


Figure 14. Three-dimensional simulation results for nonuniform-flow test case in which $\alpha_{TH} = \alpha_{TV}$ = 0.1 m: (a) finite-element model (modified from Burnett and Frind, 1987, figure 9b), and (b) implicit *MOC3D* solution using backward-in-time approximation and CELDIS = 0.5. Plume positions are represented by contours of relative concentration in vertical plane of three-dimensional grid; contour interval is 0.2 relative concentration.

Relative Computational and Storage Efficiency

Computer-memory storage requirements for the implicit dispersive transport algorithm of *MOC3D* are greater than those for the explicit dispersive transport algorithm. The additional arrays required for using the implicit algorithm routines can increase the memory size requirement by as much as a factor of two (see table 11). Computational effort required for the implicit algorithm, as opposed to the explicit one, depends on the dominant transport mechanism in the model--advection or dispersion. The first test problem, which was a one-dimensional, steady-state flow problem, was run using various values of longitudinal dispersivity with both the explicit and implicit algorithms. As presented in table 12, the implicit algorithm is more efficient than the explicit algorithm when the dispersivity is greater than 0.1 (the grid spacing for this problem is 0.1 cm). This corresponds to a grid Peclet number less than 1, where the grid Peclet number is defined as,

$$Pe_g = \frac{\Delta x}{\alpha} \qquad (26)$$

The computational effort required by the implicit dispersive transport algorithm in the *MOC3D* simulator is strongly dependent on the size of the problem being solved, as determined by the total number of nodes, total number of particles, and total number of time increments, but is independent of the value of the time weighting factor (FDTMTH). Analyses indicate that the greatest computational effort, as measured by *CPU* time, is typically expended in the particle tracking routines. For a given problem, computational effort may vary significantly as a function of the characteristics of the particular computer on which the simulation is performed, and on which FORTRAN compiler and options were used to generate the executable code.

Table 11. Execution times and storage requirements for explicit and implicit solutions using *MOC3D* (Version 2) for selected test cases.

	Run Time in CPU-seconds		"X" Array Elements Used[1]	
Problem Description	Explicit	Implicit	Explicit	Implicit
One-Dimensional Steady Flow[2]	7	10	11,457	17,400
Three-Dimensional Steady Flow[2]	404	175	897,331	1,602,994
Two-Dimensional Radial Flow and Dispersion[2]	930	445	455,737	499,900
Point Initial Condition in Uniform Flow (flow at 45 degrees to grid) [2]	210	310	1,728,673	2,406,112
Constant Source in Nonuniform Flow (Two-Dimensional)[3]	13,360	2,450	868,951	1,457,602
Constant Source in Nonuniform Flow (Three-Dimensional) [3]	38,117	13,026	12,823,151	21,652,034

[1] All data arrays and lists for *MODFLOW*, explicit and implicit *MOC3D* are allocated space in one array, the *MODFLOW* "X" array.

[2] Data General server with a Motorola 88110 chip running DG Unix 5.4R3.10 with 256MB RAM and a 45 MHz processor was used for this problem. Green Hills Software FORTRAN-88000 was used to compile *MOC3D*.

[3] Silicon Graphics server with an R8000 chip running Irix 6.0.1 with 576MB RAM and a 90 MHz processor was used for this problem. MIPSpro F77 was used to compile *MOC3D*.

Table 12. Relative simulation times for explicit and implicit solutions using *MOC3D* (Version 2) for varying values of longitudinal dispersivity for the one-dimensional, steady-state flow problem.

Longitudinal Dispersivity (cm)	Peclet Number	*MOC3D*	
		Explicit	Implicit
0.001	100.0	1.0*	2.1
0.01	10.0	1.0	2.1
0.02	5.0	1.0	2.1
0.1	1.0	1.0	2.1
0.2	0.50	1.6	1.8
0.5	0.20	3.7	1.7
1.0	0.10	7.1	1.7
10.0	0.01	67.2	1.6

* A value of 1.0 is equivalent to the base run, which took 6.9 cpu seconds to execute on a Data General server with a Motorola 88110 chip running DG Unix 5.4R3.10 with 256 MB RAM and a 45MHz processor.

To provide a qualitative indication of these relations, we have run all of the sample problems described in this report on a variety of computers. The costs, given as actual run times for each problem on selected computer systems, are presented in table 13. The run times are measured as *CPU* time in seconds. As shown in table 13, the running times for a given problem may vary by more than a factor of ten, depending on which computer was used. However, for the given test problems, the cost was much more sensitive to the overall size of the problem. The *CPU* time on a given computer varied by about four orders of magnitude between the simplest one-dimensional problem and the most complex three-dimensional problem. Thus, the model user should be aware that simulation cost may be a serious constraint on the size of problem that is feasible to simulate.

CONCLUSIONS

The implicit dispersive transport algorithm presented as an extension to the *MOC3D* simulator can simulate the transient, three-dimensional, transport of a solute subject to decay and retardation. The numerical methods used to solve the governing equations have broad general capability and flexibility for application to a wide range of hydrogeological problems.

The accuracy and precision of the numerical results of the implicit dispersive transport algorithm were tested and evaluated by comparison to analytical and numerical solutions for the same set of test problems as reported for the explicit algorithm in *MOC3D* (Version 1). These evaluation tests indicate that both the explicit and implicit solution algorithms in the *MOC3D* (Version 2) model can successfully and accurately simulate three-dimensional transport and dispersion of a solute in flowing ground water. An advantage of the implicit algorithm is that it has less computational cost for a given accuracy for dispersion-dominated problems. As such, it complements the explicit dispersive transport algorithm, which is more economical for advection-dominated problems when time step limitations on the particle transport calculation become limiting.

Table 13. Comparison of *MOC3D* (Version 2) simulation times using implicit solver for selected test cases on various computer platforms

				Run Time in CPU-Seconds			
Problem Description	Number of Nodes	Number of Moves	Maximum Number of Particles	Silicon Graphics[1]	Data General Server[2]	PC (133 MHz, Pentium)[3]	Macintosh PowerPC (7300/200)[4]
One-Dimensional Steady Flow	120	241	360	1.5	10	3	4.5
Three-Dimensional Steady Flow	14,400	27	43,200	16	175	30	83
Two-Dimensional Radial Flow and Dispersion	900	282	41,400	33	445	77	158
Point Initial Condition in Uniform Flow (flow at 45 degrees to grid)	13,824	19	110,729	20	310	40	90
Constant Source in Nonuniform Flow (Two-Dimensional)[5]	12,831	381	57,066	210	2450	420	783
Constant Source in Nonuniform Flow (Three-Dimensional)[5]	192,465	381	855,330	2425	NT	NT	NT

NT = Not Tested

[1] Silicon Graphics server with an R8000 chip running Irix 6.0.1 with 576MB RAM and a 90 MHz processor. MIPSpro F77 was used to compile *MOC3D*.

[2] Data General server with a Motorola 88110 chip running DG Unix 5.4R3.10 with 256MB RAM and a 45 MHz processor. Green Hills Software FORTRAN-88000 was used to compile *MOC3D*.

[3] IBM-compatible Pentium PC running WindowsNT with 32MB RAM and a 133 MHz processor. Lahey LF90 version 4.50e was used to compile *MOC3D*.

[4] Macintosh 7300 with 200 MHz PowerPC 604e processor and 32MB of RAM. *MOC3D* was compiles using Fortner Research LS FORTRAN Version 1.1 for Power Mac.

[5] CELDIS = 0.5

REFERENCES

Axelsson, Owe, 1994, Iterative Solution Methods: New York, Cambridge University Press, 654 p.

Aziz, Khalid, and Settari, Antonin, 1979, Petroleum reservoir simulation: London, England, Applied Science, 476 p.

Burnett, R.D., and Frind, E.O., 1987, Simulation of contaminant transport in three dimensions, 2. Dimensionality effects: Water Resources Research, v. 23, no. 4, p. 695-705.

Dupont, R.P., Kendall, Todd, and Rachford, Henry, 1968, An approximate factorization procedure for solving self-adjoint elliptic difference equations: SIAM Journal of Numerical Analysis, v. 53, p. 559-573

Garder, A.O., Peaceman, D.W., Pozzi, A.L., 1964, Numerical calculation of multidimensional miscible displacement by the method of characteristics: Soc. Petroleum Eng. Jour., v. 4, no. 1, p. 26-36.

Goode, D.J., and Konikow, L.F., 1989, Modification of a method-of-characteristics solute-transport model to incorporate decay and equilibrium-controlled sorption or ion exchange: U.S. Geological Survey Water-Resources Investigations Report 89-4030, 65 p.

Harbaugh, A.W., and McDonald, M.G., 1996a, User's documentation for MODFLOW-96, an update to the U.S. Geological Survey modular finite-difference ground-water flow model: U.S. Geological Survey Open-File Report 96-485, 56 p.

Harbaugh, A.W., and McDonald, M.G., 1996b, Programmer's documentation for MODFLOW-96, an update to the U.S. Geological Survey modular finite-difference ground-water flow model: U.S. Geological Survey Open-File Report 96-486, 220 p.

Hsieh, P.A., 1986, A new formula for the analytical solution of the radial dispersion problem: Water Resources Research, v. 22, no. 11, p. 1597-1605.

Kipp, K.L., 1987, HST3D: A computer code for simulation of heat and solute transport in three-dimensional ground-water flow: U.S. Geological Survey Water-Resources Investigations Report 86-4095, 517 p.

Konikow, L.F., and Bredehoeft, J.D., 1978, Computer model of two-dimensional solute transport and dispersion in ground water: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 7, Chapter C2, 90 p.

Konikow, L.F., Goode, D.J., and Hornberger, G.Z., 1996, A three-dimensional method-of-characteristics solute-transport model (MOC3D): U.S. Geological Survey Water-Resources Investigations Report 96-4267, 87 p.

McDonald, M.G., and Harbaugh, A.W., 1988, A modular three-dimensional finite-difference ground-water flow model: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 6, Chapter A1, 586 p.

Meijerink, J.A., and Van der Vorst, H.A., 1977, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix: Mathematics of Computation, v. 31, no. 137, p. 148-162.

Meijerink, J.A., and Van der Vorst, H.A., 1981, Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems: Journal of Computational Physics, v. 44, p. 134-155.

Price, H.S., and Coats, K.H., 1974, Direct methods in reservoir simulation: Transactions of the Society of Petroleum Engineers of the American Institute of Mining, Metallurgical, and Petroleum Engineers, v. 257, p. 295-308.

Stoer, J., and Bulirsch, R., 1991, Introduction to Numerical Analysis: 2nd ed., Springer-Verlag, New York, 660 p.

Wexler, E.J., 1992, Analytical solutions for one-, two-, and three-dimensional solute transport in ground-water systems with uniform flow: U.S. Geological Survey Techniques of Water-Resources Investigations, Book 3, Chapter B7, 190 p.

Yanenko, N.N., 1971, The Method of Fractional Steps: Springer-Verlag, New York, 160 p.

APPENDIX A: DATA INPUT INSTRUCTIONS FOR *MOC3D (Version 2)*

This Appendix includes a complete set of instructions for preparing a data set for the *MOC3D* model. Major changes that have been implemented since the release of Version 1 are shaded to highlight the new instructions.

MODFLOW Name File

Transport simulation is activated by including a record in the *MODFLOW* name file using the file type (Ftype) “CONC” to link to the transport name file. The transport name file specifies the files to be used when simulating solute transport in conjunction with a simulation of ground-water flow using *MODFLOW*. The transport name file works in the same way as the *MODFLOW* name file.

MODFLOW Source and Sink Packages

Except for recharge, concentrations associated with fluid sources (C') are read as auxiliary parameters in the *MODFLOW* source package. The source concentration is read from a new column appended to the end of each line of the data file describing a fluid sink/source (see documentation for revised *MODFLOW* model; Harbaugh and McDonald, 1996a and 1996b). For example, concentrations associated with well nodes should be appended to the line in the WEL Package where the well’s location and pumping rate are defined. These concentrations will be read if the auxiliary parameter “CONCENTRATION” (or “CONC”) appears on the first line of the well input data file. The concentration in recharge is defined separately, as described in following section “Source Concentration in Recharge File.”

To simulate solute transport the *MODFLOW* option enabling storage of cell-by-cell flow rates for each fluid source or sink is required in all fluid packages except recharge. The key word “CBCALLOCATE” (or “CBC”) must appear on the first line of each input data file for a fluid package (see Harbaugh and McDonald, 1996a and 1996b).

MOC3D Input Data Files

All input variables are read using free formats, except as specifically indicated. In free format, variables are separated by one or more spaces or by a comma and optionally one or more spaces. Blank spaces are not read as zeros.

MOC3D Transport Name File (CONC)

FOR EACH SIMULATION:

1. Data: FTYPE NUNIT FNAME

The name file consists of records defining the names and units numbers of the files. Each "record" consists of a separate line of data. There must be a record for the listing file and for the main *MOC3D* input file.

The listing (or output) file ("CLST") must be the first record. The other files may be in any order. Each record can be no more than 79 characters.

FTYPE The file type, which may be one of the following character strings:

CLST *MOC3D* listing file (separate from the *MODFLOW* listing file) *[required]*.

MOC** or **MOCIMP Main *MOC3D* input data file *[required]*. Specifying ***MOC*** indicates dispersion calculations will be explicit (as described by Konikow and others, 1996) and specifying ***MOCIMP*** indicates dispersion calculations will be implicit (as described in this report).

CRCH Concentrations in recharge *[optional]*.

CNCA Separate output file containing concentration data in ASCII (text-only) format. Frequency and format of printing controlled by NPNTCL and ICONFM *[optional]*. If concentrations are written to a separate output file, they will not be written to the main output file.

CNCB Separate output file containing concentration data in binary format *[optional]*.

VELA Separate output file with velocity data in ASCII format. Frequency and format of printing controlled by NPNTVL and IVELFM *[optional]*.

VELB Separate output file with velocity data in binary format *[optional]*.

PRTA Separate output file with particle locations printed in ASCII format. Frequency and format of printing controlled by NPNTPL *[optional]*.

PRTB Separate output file with particle locations printed in binary format *[optional]*.

OBS Observation wells input file *[optional]*.

DATA For formatted files such as those required by the OBS package and for array data separate from the main *MOC3D* input data file *[optional]*.

DATA(BINARY) For formatted input/output files *[optional]*.

NUNIT The FORTRAN unit number used to read from and write to files. Any legal unit number other than 97, 98, and 99 (which are reserved by *MODFLOW*) can be used provided that it is not previously specified in the *MODFLOW* name file.

FNAME The name of the file.

Main *MOC3D* Package Input (MOC or MOCIMP)

Input for the method-of-characteristics (*MOC3D*) solute-transport package is read from the unit specified in the transport name file. The input consists of up to 19 separate records or data sets, as described in detail below. These data are used to specify information about the transport subgrid, physical and chemical transport parameters, numerical solution variables, and output formats. Output file controls for the *MOC3D* package are specified in the transport name file, described previously. Compared to the original version of *MOC3D*, Version 2 includes one additional data set, which is used only if the implicit solver is selected. This data set is placed between the original data sets 7 and 8, and hence has been labeled as data set 7.1.

FOR EACH SIMULATION:

1. Data: HEDMOC — A two-line character-string title describing the simulation (80 text characters per line).

2. Data: HEDMOC (continued)

3. Data: ISLAY1 ISLAY2 ISROW1 ISROW2 ISCOL1 ISCOL2

ISLAY1 Number of first (uppermost) layer for transport.

ISLAY2 Last layer for transport.

ISROW1 First row for transport.

ISROW2 Last row for transport.

ISCOL1 First column for transport.

ISCOL2 Last column for transport.

Notes:

Transport may be simulated within a subgrid, which is a “window” within the primary *MODFLOW* grid used to simulate flow. Within the subgrid, the row and column spacing must be uniform, but thickness can vary from cell to cell and layer to layer. However, as discussed in the section reviewing *MOC3D* assumptions, the range in thickness values (or product of thickness and porosity) should be as small as possible.

4. Data: NODISP DECAY DIFFUS

NODISP Flag for no dispersion (set NODISP=1 if no dispersion in problem; this will reduce storage allocation).

DECAY First-order decay rate [1/T] (DECAY=0.0 indicates no decay occurs).

DIFFUS Effective molecular diffusion coefficient [L^2/T].

Notes:

The decay rate (λ) is related to the half life ($t_{1/2}$) of a constituent by $\lambda = (\ln 2)/t_{1/2}$.

The effective molecular diffusion coefficient (D_m) includes the effect of tortuosity.

5. `Data:   NPMAX    NPTPND`

`NPMAX` Maximum number of particles available for particle tracking of advective transport in *MOC3D*. If set to zero, the model will calculate NPMAX according to the following equation:

$$\text{NPMAX} = 2 \times \text{NPTPND} \times \text{NSROW} \times \text{NSCOL} \times \text{NSLAY}.$$

`NPTPND` Initial number of particles per cell in transport simulation (that is, at $t = 0.0$). Valid options for default geometry of particle placement include 1, 2, 3, or 4 for one-dimensional transport simulation; 1, 4, 9, or 16 for two-dimensional transport simulation; and 1, 8, or 27 for three-dimensional transport simulation. The user can also customize initial placement of particles by specifying NPTPND as a negative number, in which case the minus sign is recognized as a flag to indicate custom placement is desired. In this case, the user must input local particle coordinates as described below.

IF *NPTPND* IS NEGATIVE IN SIGN:

6. `Data:   PNEWL    PNEWR    PNEWC`

`PNEWL` Relative position in the layer (z) direction for initial placement of particle within any finite-difference cell.

`PNEWR` Relative position in the row (y) direction for initial placement of particle.

`PNEWC` Relative position in the column (x) direction for initial placement of particle.

Notes:

The three new (or initial) particle coordinates are entered sequentially for each of the NPTPND particles. Each line contains the three relative local coordinates for the new particles, in order of layer, row, and column. There must be NPTPND lines of data, one for each particle. The local coordinate system range is from -0.5 to 0.5, and represents the relative distance within the cell about the node location at the center of the cell, so that the node is located at 0.0 in each direction.

FOR EACH SIMULATION:

7. `Data:   CELDIS    FZERO    INTRPL`

`CELDIS` Maximum fraction of cell dimension that particle may move in one step (typically, $0.5 \leq \text{CELDIS} \leq 1.0$).

`FZERO` If the fraction of active cells having no particles exceeds FZERO, the program will automatically regenerate an initial particle distribution before continuing the simulation (typically, $0.01 \leq \text{FZERO} \leq 0.05$).

`INTRPL` Flag for interpolation scheme used to estimate velocity of particles. The default (INTRPL=1) will use a linear interpolation routine; if INTRPL=2, a scheme will be implemented that uses bilinear interpolation in the row and column (j and i) directions only (linear interpolation will still be applied in the k, or layer, direction). (See section “Discussion—Choosing appropriate interpolation scheme.”)

FOR EACH SIMULATION (only if *MOCIMP* is specified in Transport Name File):

7.1 Data: FDTMTH NCXIT IDIREC EPSSLV MAXIT

FDTMTH Weighting factor for temporal differencing of dispersion equation (0.0 ≤ FDTMTH ≤ 1.0). We suggest using either a value of FDTMTH = 0.5, a centered-in-time (or Crank-Nicolson) approximation, or FDTMTH = 1.0, a backward-in-time (or fully implicit) approximation. [Default value = 1]

NCXIT Number of iterations for the explicitly-lagged cross-dispersive flux terms (NCXIT ≥ 1). We suggest that the user initially specify a value of 2, but if the solution exhibits significant areas of negative concentrations, then the value of NCXIT should be increased to require more iterations, which typically will reduce the extent and magnitude of negative concentrations (at the cost of increased computational time). [Default value = 2]

IDIREC Direction index for permutation of the red-black node renumbering scheme. The order is as follows: 1: x,y,z; 2: x,z,y; 3: y,x,z; 4: y,z,x; 5: z,x,y; and 6: z,y,x. The first direction index is advanced most rapidly and the last direction index is advanced least rapidly. In some cases, there can be a significant variation in the number of iterations needed to achieve convergence, depending on the order of the directions for the red-black renumbering. We suggest that the user initially specify IDIREC = 1. If this leads to a relatively large number of iterations (more than 10), then the user should experiment with alternate choices to determine the one requiring the fewest number of iterations for their particular problem. [Default value = 1]

EPSSLV Tolerance on the relative residual for the conjugate-gradient solution of the matrix of the difference equations. We suggest that the user initially specify EPSSLV ≤ 10^{-5}. An adequately small value of EPSSLV has the property that a smaller value does not change the numerical solution within the number of significant digtits desired by the user. In the single-precision code implemented here, EPSSLV should not be less than 10^{-7}. [Default value = 10^{-5}]

MAXIT Maximum number of iterations allowed for the iterative solution to the differernce equations for dispersive transport. In most cases, MAXIT = 100 is satisfactory. [Default value = 100]

Notes:

Entering a zero or out-of-range value for any of these five variables will cause the code to use the indicated default value.

FOR EACH SIMULATION:

8\. Data: NPNTCL ICONFM NPNTVL IVELFM NPNTDL IDSPFM NPNTPL

NPNTCL Flag for frequency of printing concentration data. If NPNTCL=-2, concentration data will be printed at the end of every stress period; if NPNTCL=-1, data will be printed at the end of every flow time step; if NPNTCL=0, data will be printed at the end of the simulation; if NPNTCL=N>0, data will be printed every Nth particle moves, and at the end of the simulation. Initial concentrations are always printed. Solute budget and mass balance information are only printed every time concentration data are saved.

ICONFM Flag for output format control for printing concentration data. If concentration data are written to main output file (file type CNCA is not used), ICONFM represents a code indicating the format style (table 14, also see Harbaugh and McDonald, 1996a, p. 19). If concentration data are written to a separate output file (file type CNCA exists), specifying ICONFM ≥ 0 will indicate that concentration data are to be written as a matrix of values for each layer of the subgrid, whereas specifying ICONFM < 0 will indicate that concentration data are to be written as a table of values having one row for each node in the subgrid and four columns (*x*, *y*, *z*, and concentration), where *x*, *y*, and *z* are the actual nodal coordinates in the length units of the model simulation. Note that we follow the *MODFLOW* convention in that *y* increases from top to bottom row, and *z* increases from top layer to bottom layer. Also note that the *x* and *y* values are given with respect to the entire MODFLOW grid, but the *z* location is calculated only for vertical distances within the layers of the transport subgrid. If data are written in matrix style, one header line precedes and identifies the data for each layer. If data are written as a table of values, one header line is written each time that concentration data are saved.

NPNTVL Flag for printing velocity data. If NPNTVL=-1, velocity data will be printed at the end of every stress period; if NPNTVL=0, data will be printed at the end of the simulation; if NPNTVL=N>0, data will be printed every Nth flow time steps, and at the end of the simulation.

IVELFM Specification for format of velocity data, if being printed in main output file (see table 14).

NPNTDL Flag for printing dispersion equation coefficients that include cell dimension factors (see section “Program Segments”). If NPNTDL=-2, coefficients will be printed at the end of every stress period; if NPNTDL=-1, coefficients will be printed at the end of the simulation; if NPNTDL=0, coefficients will not be printed; if NPNTDL=N>0, coefficients will be printed every Nth flow time step.

IDSPFM Specification for format of dispersion equation coefficients (see table 14).

NPNTPL Flag for printing particle locations in a separate output file (only used if file types “PRTA” or “PRTB” appear in the *MOC3D* name file). If neither “PRTA” or “PRTB” is entered in the name file, NPNTPL will be read but ignored (so you must always have some value specified here). If either “PRTA” or “PRTB” is entered in the name file, initial particle locations will be printed to the separate file first, followed by particle data at intervals determined by the value of NPNTPL. If NPNTPL=-2, particle data will be printed at the end of every stress period; if NPNTPL=-1, data will be printed at the end of every flow time step; if NPNTPL=0, data will be printed at the end of the simulation; if NPNTPL=N>0, data will be printed every Nth particle moves, and at the end of the simulation.

Table 14. Formats associated with *MOC3D* print flags. (Positive values for wrap format; negative values for strip format. Also see Harbaugh and McDonald, 1996a, p. 19.)

Print flag	Format	Print flag	Format	Print flag	Format
0	10G11.4	7	20F5.0	14	10F6.1
1	11G10.3	8	20F5.1	15	10F6.2
2	9G13.6	9	20F5.2	16	10F6.3
3	15F7.1	10	20F5.3	17	10F6.4
4	15F7.2	11	20F5.4	18	10F6.5
5	15F7.3	12	10G11.4		
6	15F7.4	13	10F6.0		

FOR EACH SIMULATION:

9. Data: CNOFLO — Concentration associated with inactive cells of subgrid (used for output purposes only).

FOR EACH LAYER OF THE TRANSPORT SUBGRID:

10. Data: CINT(NSCOL,NSROW) — Initial concentration.
Module: U2DREL*

FOR EACH SIMULATION, ONLY IF TRANSPORT SUBGRID DIMENSIONS ARE SMALLER THAN FLOW GRID DIMENSIONS:

11. Data: CINFL(ICINFL) — C' to be associated with fluid inflow across the boundary of the subgrid.
Module: U1DREL*

Notes:

The model assumes that the concentration outside of the subgrid is the same within each layer, so only one value of CINFL is specified for each layer within and adjacent to the subgrid. That is, the size of the array (ICINFL) is determined by the position of the subgrid with respect to the entire (primary) *MODFLOW* grid. If the transport subgrid has the same dimensions as the flow grid,

* Module is a standard *MODFLOW* input/output module.

this parameter should not be included in the input data set. If the subgrid and flow grid have the same number of layers, but the subgrid has fewer rows or fewer columns, ICINFL=NSLAY. Values are also required if there is a flow layer above the subgrid and/or below the subgrid. The order of input is: C' for first (uppermost) transport layer (if required); C' for each successive (deeper) transport layer (if required); C' for layer above subgrid (if required); and C' for layer below subgrid (if required).

FOR EACH SIMULATION:

12. `Data:` `NZONES` Number of zone codes among fixed-head nodes in transport subgrid.

IF *NZONES* > 0:

`Data:` `IZONE` `ZONCON`

`IZONE` Value identifying a particular zone.

`ZONCON` Source concentration associated with nodes in the zone defined by IZONE above.

Notes:

Zones are defined within the IBOUND array in the BAS Package of *MODFLOW* by specifying unique negative values for fixed-head nodes to be associated with separate fluid source concentrations. Each zone is defined by a unique value of IZONE and a concentration associated with it (ZONCON). There must be NZONES lines of data, one for each zone. Note that values of IZONE in this list must be negative for consistency with the definitions of fixed-head nodes in the IBOUND array in the BAS Package. If a negative value of IBOUND is defined in the BAS package but is not assigned a concentration value here, *MOC3D* will assume that the source concentrations associated with those nodes equal 0.0.

FOR EACH LAYER OF THE TRANSPORT SUBGRID:

13. `Data:` `IGENPT(NSCOL,NSROW)` Flag to treat fluid sources and sinks as either “strong” or “weak.”

`Module:` `U2DINT`*

Notes:

Where fluid source is “strong,” new particles are added to replace old particles as they are advected out of that cell. Where a fluid sink is “strong,” particles are removed after they enter that cell and their effect accounted for. Where sources or sinks are weak, particles are neither added nor removed, and the source/sink effects are incorporated directly into appropriate changes in particle positions and concentrations. If IGENPT=0, the node will be considered a weak source or sink; if IGENPT=1, it will be a strong source or sink. See section on “Special Problems” and discussion by Konikow and Bredehoeft (1978).

* Module is a standard *MODFLOW* input/output module.

IF *NODISP* ≠ 1 (If dispersion is included in simulation):

14. Data: `ALONG(NSLAY)` Longitudinal dispersivity.
Module: `U1DREL`[*]

15. Data: `ATRANH(NSLAY)` Horizontal transverse dispersivity.
Module: `U1DREL`[*]

16. Data: `ATRANV(NSLAY)` Vertical transverse dispersivity.
Module: `U1DREL`[*]

FOR EACH SIMULATION:

17. Data: `RF(NSLAY)` Retardation factor (RF=1 indicates no retardation).
Module: `U1DREL`[*]

Notes:

If RF=0.0 in input, the code automatically resets it as RF=1.0 to indicate no retardation.

FOR EACH LAYER OF TRANSPORT SUBGRID:

18a. Data: `THCK(NSCOL,NSROW)` Cell thickness.
Module: `U2DREL`[*]

18b. Data: `POR(NSCOL,NSROW)` Cell porosity.
Module: `U2DREL`[*]

Notes:

The thickness and porosity are input as separate arrays for each layer of the transport subgrid. The sequence used in data set 18 is to first define the thickness of the first layer of the transport subgrid, and then define the porosity of that same layer. Next, that sequence is repeated for all succeeding layers. The product of thickness and porosity should not be allowed to vary greatly among cells in the transport subgrid.

[*] Module is a standard *MODFLOW* input/output module.

Source Concentration in Recharge File (CRCH)

Concentrations in recharge, if the recharge package is used, are read from a separate unit specified in the *MOC3D* name file. This is defined using the file type (Ftype) "CRCH."

FOR EACH STRESS PERIOD, IF RECHARGE PACKAGE USED:

`1. Data: INCRCH` Flag to reuse or read new recharge concentrations.

Notes:

Read new recharge concentrations if INCRCH ≥ 0. Reuse recharge concentrations from the last stress period if INCRCH < 0.

`2. Data: CRECH(NSCOL,NSROW)` Source concentration associated with fluid entering the aquifer in recharge.

`Module: U2DREL`*

Observation Well File (OBS)

Nodes of the transport subgrid can be designated as "observation wells." At each such node, the time, head, and concentration after each move increment will be written to a separate output file to facilitate graphical postprocessing of the calculated data. The input file for specifying observation wells is read if the file type (Ftype) "OBS" is included in the *MOC3D* name file.

FOR EACH SIMULATION, IF *OBS* PACKAGE USED:

`1. Data: NUMOBS IOBSFL`

`NUMOBS` Number of observation wells.

`IOBSFL` If IOBSFL = 0, well data are saved in NUMOBS separate files. If IOBSFL>0, all observation well data will be written to one file, and the file name and unit number used for this file will be that of the first observation well in the list.

FOR EACH OBSERVATION WELL:

`2. Data: LAYER ROW COLUMN UNIT`

`LAYER` Layer of observation well node.

`ROW` Row of observation well node.

`COLUMN` Column of observation well node.

`UNIT` Unit number for output file.

Notes:

If NUMOBS>1 and IOBSFL = 0, you must specify a unique unit number for each observation well and match those unit numbers to DATA file types and file names in the *MOC3D* name file. If IOBSFL>0, you must specify a unique unit number for the first observation well and match that unit number to a DATA file type and file name in the *MOC3D* name file.

* Module is a standard *MODFLOW* input/output module.

APPENDIX B: ANNOTATED INPUT DATA SET FOR EXAMPLE PROBLEM

This example input data set is the one used to generate the solution for the base case in the one-dimensional steady-state flow problem. Parameter values are indicated in table 6 and selected results are shown in figures 4 and 6. Several of the following data files (*finite.nam*, *finite.bas*, *finite.bcf*, and *finite.sip*) are those required for *MODFLOW-96*, and their formats are described by Harbaugh and McDonald (1996a).

In the data files shown below, the right side of some data lines includes a semi-colon followed by text that describes the parameters for which values are given. These comments (including the semi-colon) are not read by the program because in free format the code will only read the proper number of variables and ignore any subsequent information on that line. This style of commenting data files is optional, but users may find it helpful when preparing their own data files.

Information pertaining specifically to the implicit solution is highlighted by shading.

Following (enclosed in a border) are the contents of the *MODFLOW* name file for the sample problem; explanations are noted outside of border:

File name: `finite.nam`

```
list    16     finite.lst      ← Designates main output file for MODFLOW
bas     95     finite.bas      ← Basic input data for MODFLOW
bcf     11     finite.bcf      ← Block-centered flow package
sip     19     finite.sip      ← Input for numerical solution of flow equation
conc    33     fint_moc.nam    ← Transport name file (turns transport “on”)

 ↑       ↑           ↑
 1       2           3
```

1 Ftype (that is, the type of file)
2 Unit number
3 File name (name chosen to reflect contents of file)

Following (enclosed in a border) are the contents of the basic package input data file for the *MODFLOW* simulation of the sample problem; explanations are noted outside of border:

File name: `finite.bas`

```
Finite:  Compare to Wexler program and MOC3D                  BAS Input      ← 1
      NLAY      NROW      NCOL      NPER    ITMUNI                           ← 1
         1         1       122         1         1                           ← 2
FREE  CHTOCH                                                                 ← 3
         0         1      ; IAPART,ISTRT                                     ← 4
        95         1(25I3)                               3   ; IBOUND        ← 5
 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  ← 5
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  ← 5
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  ← 5
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  ← 5
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1 -2              ← 5
      0.00                                          ; HNOFLO                 ← 6
        95       1.0(122F6.0)                  1    ; HEAD                   ← 7
  12.1                                                                       ← 7
 120.0           1        1.    ; PERLEN,NSTP,TSMULT                         ← 8
```

[1] Two header lines of comments. For convenience and clarity, the second line is used to label names of parameters on subsequent line of file.

[2] Flow grid dimensions, number of periods, and time units.

[3] Options line (new in *MODFLOW-96*)

[4] Flags for buffer array and drawdown calculations.

[5] *IBOUND* identifiers (first line) and array

[6] Head value assigned to inactive cells

[7] Initial head information

[8] *MODFLOW* time-step information

Following (enclosed in a border) are the contents of the block-centered flow package input data file; explanations are noted outside of border:

File name: `finite.bcf`

```
         1   0 0.0 0 0.0 0 0  ; ISS,flags        BCF Input   ← 1
         0                    ; LAYCON                       ← 2
         0       1.0          ; TRPY                         ← 3
         0       0.1          ; DELR                         ← 4
         0       0.1          ; DELC                         ← 4
         0      0.01          ; TRAN
```

[1] Flag for steady-state flow, flag for cell-by-cell flow terms, five flags related to wetting

[2] Layer type

[3] Anisotropy factor

[4] Grid spacing information

Following (enclosed in a border) are the contents of the strongly implicit procedure package input data file; explanations are noted outside of border:

File name: *finite.sip*

```
500        5     ; MXITER,NPARM                SIP Input                  ← 1
1.   0.0000001     0     0.001   0 ; ACCL,HCLOSE,IPCALC,WSEED,IPRSIP      ← 2
```

1 Maximum iterations, number of iteration parameters
2 Acceleration parameter, head change criterion, flag for seed, seed, printout interval for SIP

Following (enclosed in a border) are the contents of the *MOC3D* name file for the sample problem; explanations are noted outside of border:

File name: *fint_moc.nam*

```
clst      94     finite.out       ← Designates main output file for MOC3D
mocimp    96     fint_imp.moc     ← Main input data file for MOC3D
obs       44     finite.obs       ← Input data file for observation wells
data      45     finite.oba       ← Output file for observation well data
cnca      22     finite.cna       ← Separate output file for concentration data (ASCII)

  ↑         ↑            ↑
  1         2            3
```

1 Ftype
2 Unit number
3 File name

Following (enclosed in a border) are the contents of the main input data file for the *MOC3D* simulation for the sample problem; selected explanations are noted outside of border:

File name: `finite_imp.moc`

```
One-dimensional, Steady Flow, No Decay, Low Dispersion: MOC3D (Implicit)     ← 1
    ISLAY1    ISLAY2    ISROW1    ISROW2    ISCOL1    ISCOL2                  ← 1
         1         1         1         1         2       121                  ← 2
         0      0.00       0.0 ; NODISP, DECAY, DIFFUS                        ← 3
         0         3           ; NPMAX, NPTPND                                ← 4
       0.5      0.05         2 ; CELDIS, FZERO, INTRPL                        ← 4
       0.5  2  1  1.e-05   100 ; FDTMTH,NXCIT,IDIR,EPSSLV,MAXIT               ← 5
  0 -1 0 0 0 0 0 ; NPNTCL, ICONFM, NPNTVL, IVELFM, NPNTDL, IDSPFM, NPRTPL     ← 6
       0.0                                 ; CNOFLO                           ← 7
         0       0.0 (122F3.0)             ; initial concentration
         0       1.                        ; C' inflow
         2                                 ; NZONES to follow                 ← 8
        -1       1.0                       ; IZONE, ZONCON                    ← 8
        -2       0.0                       ; IZONE, ZONCON                    ← 8
         0         0                       ; IGENPT                           ← 9
         0       0.1                       ; longitudinal disp.
         0       0.1                       ; transverse disp. horiz.
         0       0.1                       ; transverse disp. vert.
         0       1.0                       ; retardation factor
         0       1.0                       ; thickness
         0       0.1                       ; porosity
```

1 Two header lines of comments. For convenience and clarity, the second line is used to label names of parameters on subsequent line of file.

2 Indices for transport subgrid

3 Flag for no dispersion, decay rate, diffusion coefficient

4 Particle information for advective transport

5 Information to control implicit dispersion calculations

6 Print flags

7 Value of concentration associated with inactive cells

8 Concentrations associated with fixed-head nodes (fixed head nodes are defined in the IBOUND array in the *MODFLOW* BAS package)

9 Flag for "strong" sources or sinks

Following (enclosed in a border) are the contents of the observation well input data file for the sample problem; explanations are noted outside of border:

File name: `finite.obs`

```
  3  1                ;NUMOBS  IOBSFL    Observation well data    ← 1
  1  1   2  45        ;layer, row, column, unit number             ← 2
  1  1  42            ;layer, row, column                          ← 2
  1  1 112            ;layer, row, column                          ← 2
```

1 Number of observation wells, flag to print to one file or separate files

2 Node location and unit number for output file (linked to the Ftype DATA in *MOC3D* name file)

APPENDIX C: SELECTED OUTPUT FOR EXAMPLE PROBLEM

This example output was generated from the input data sets listed in Appendix B for the base case of the one-dimensional steady-state flow problem. We do not include the main *MODFLOW* listing (output) file. The line spacing and font sizes of the output files have been modified in places to enhance the clarity of reproduction in this report. Some repetitive lines of output have been deleted where indicated by an ellipsis (**. . .**). Output related specifically to the implicit solution is highlighted by shading.

Some brief annotations were added to this sample output listing to help the reader understand the purpose of various sections of output. These annotations are written in bold italics.

Following are the contents of the *MOC3D* main output file (*finite.out*) for the sample problem.

```
              U.S. GEOLOGICAL SURVEY
 METHOD-OF-CHARACTERISTICS SOLUTE-TRANSPORT MODEL
           MOC3D (Version 2.0) 11/16/98


 MOC BASIC INPUT READ FROM UNIT
 LISTING FILE: finite.out   UNIT  94

 OPENING fint_imp.moc
 FILE TYPE: MOCIMP   UNIT  96

 OPENING finite.obs
 FILE TYPE: OBS   UNIT  44

 OPENING finite.oba
 FILE TYPE: DATA   UNIT  45

 OPENING finite.cna
 FILE TYPE: CNCA   UNIT  22


 MOC BASIC INPUT READ FROM UNIT  96
```

FILE INFORMATION

2 TITLE LINES:

```
 One-dimensional, Steady Flow, No Decay, Low Dispersion: MOC3D (Implicit)
     ISLAY1    ISLAY2    ISROW1    ISROW2    ISCOL1    ISCOL2
```

PROBLEM DESCRIPTORS, INCLUDING GRID CHARACTERISTICS AND PARTICLE INFORMATION:

```
      MAPPING OF SOLUTE TRANSPORT SUBGRID IN FLOW GRID:
 FIRST LAYER FOR SOLUTE TRANSPORT =   1      LAST LAYER FOR SOLUTE TRANSPORT  =   1
 FIRST ROW FOR SOLUTE TRANSPORT   =   1      LAST ROW FOR SOLUTE TRANSPORT    =   1
 FIRST COLUMN FOR SOLUTE TRANSPORT=   2      LAST COLUMN FOR SOLUTE TRANSPORT = 121

 UNIFORM DELCOL AND DELROW IN SUBGRID FOR SOLUTE TRANSPORT

 NO. OF LAYERS =    1   NO. OF ROWS =    1   NO. OF COLUMNS =  120
 NO SOLUTE DECAY
 NO MOLECULAR DIFFUSION
 MAXIMUM NUMBER OF PARTICLES (NPMAX) =      720
    13669 ELEMENTS IN X ARRAY ARE USED BY MOC
       12 ELEMENTS IN X ARRAY ARE USED BY OBS
 NUMBER OF PARTICLES INITIALLY IN EACH ACTIVE CELL (NPTPND) =   3
 PARTICLE MAP ("o" indicates particle location; shown as
              fractions of cell distances relative to node location):

       o------o------o

     -1/3     0     1/3

 INITIAL RELATIVE PARTICLE COORDINATES
   1    0.00000   0.00000  -0.33333
   2    0.00000   0.00000   0.00000
   3    0.00000   0.00000   0.33333

 CELDIS=     0.500
 FZERO =     0.050


INTRPL= 2;  BILINEAR INTERPOLATION SCHEME

 NUMERICAL PARAMETERS FOR IMPLICIT SOLVER:
 FDTMTH =     0.50
 NCXIT  =      2
 IDIREC =      1
 EPSSLV =  1.0000E-05
 MAXIT  =    100

NPNTCL=  0:     CONCENTRATIONS WILL BE WRITTEN AT THE END OF THE SIMULATION
MODFLOW FORMAT SPECIFIER FOR CONCENTRATION DATA: ICONFM= -1

NPNTVL=  0:         VELOCITIES WILL BE WRITTEN AT THE END OF THE SIMULATION
MODFLOW FORMAT SPECIFIER FOR      VELOCITY DATA: IVELFM=  0

NPNTDL=  0: DISP. COEFFICIENTS WILL NOT BE WRITTEN

NPNTPL=  0: PARTICLE LOCATIONS WILL NOT BE WRITTEN
```

OUTPUT CONTROL

```
CONCENTRATION WILL BE SET TO 0.00000E+00 AT ALL NO-FLOW NODES (IBOUND=0).

    INITIAL CONCENTRATION =  0.0000000E+00 FOR LAYER   1


VALUES OF C' REQUIRED FOR SUBGRID BOUNDARY ARRAY =    1
ONE FOR EACH LAYER IN TRANSPORT SUBGRID

ORDER OF C' VALUES: FIRST LAYER IN SUBGRID, EACH SUBSEQUENT LAYER,
LAYER ABOVE SUBGRID, LAYER BELOW SUBGRID:

  SUBGRID BOUNDARY ARRAY  =   1.000000


 NUMBER OF ZONES FOR CONCENTRATIONS AT FIXED HEAD CELLS =    2

 ZONE FLAG =   -1     INFLOW CONCENTRATION =   1.0000E+00
 ZONE FLAG =   -2     INFLOW CONCENTRATION =   0.0000E+00
```

INITIAL AND BOUNDARY CONDITIONS FOR SOLUTE

```
         SINK-SOURCE FLAG =             0 FOR LAYER   1

 LONGITUDNL. DISPERSIVITY =  0.1000000

  HORIZ. TRANSVERSE DISP. =  0.1000000

   VERT. TRANSVERSE DISP. =  0.1000000
```

```
       RETARDATION FACTOR =   1.000000

        INITIAL THICKNESS =   1.000000     FOR LAYER   1

         INITIAL POROSITY =  0.1000000     FOR LAYER   1



COORDINATES FOR   3 OBSERVATION WELLS:

  WELL #   LAYER     ROW  COLUMN    UNIT
       1       1       1       2      45
       2       1       1      42      45
       3       1       1     112      45
ALL OBSERVATION WELL DATA WILL BE WRITTEN ON UNIT  45


CONCENTRATION DATA WILL BE SAVED ON UNIT  22 IN X,Y,Z,CONC FORMAT

 TOTAL NUMBER OF PARTICLES GENERATED =       360
 TOTAL NUMBER OF ACTIVE NODES (NACTIV) =       120
 MAX. NUMBER OF CELLS THAT CAN BE VOID OF PARTICLES (NZCRIT) =      6
     (IF NZCRIT EXCEEDED, PARTICLES ARE REGENERATED)
```

CALCULATED VELOCITIES (INCLUDING EFFECTS OF RETARDATION, IF PRESENT):

```
EFFECTIVE MEAN SOLUTE VELOCITIES IN COLUMN DIRECTION
                          AT NODES
1
VELOCITY (COL)   IN LAYER  1 AT END OF TIME STEP  1 IN STRESS PERIOD  1
  -----------------------------------------------------------------------

            1           2           3           4           5           6           7  ...
...
  ..................................................................................  ...
   1   0.1000      0.1000      0.1000      0.1000      0.1000      0.1000      0.1000  ...

...

EFFECTIVE MEAN SOLUTE VELOCITIES IN ROW DIRECTION
                          AT NODES
1
  VELOCITY (ROW)   IN LAYER  1 AT END OF TIME STEP  1 IN STRESS PERIOD  1
  -----------------------------------------------------------------------
          1          2          3          4          5          6          7          8          9      ...
...
 ....................................................................................................... ...
   1   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  ...

...

EFFECTIVE MEAN SOLUTE VELOCITIES IN LAYER DIRECTION
                          AT NODES
1
  VELOCITY (LAYER) IN LAYER  1 AT END OF TIME STEP  1 IN STRESS PERIOD  1
  -----------------------------------------------------------------------

          1          2          3          4          5          6          7          8          9      ...
...
 ....................................................................................................... ...
   1   0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  0.000E+00  ...

...
```

```
         STABILITY CRITERIA --- M.O.C.

    MAXIMUM FLUID VELOCITIES:  C-VEL =  1.00E-01     R-VEL =  1.00E-20     L-VEL =  1.00E-18
MINIMUM TIME TO TRAVEL THCK =  1.00E+18

TIMV =  5.00E-01     NTIMV  =    241

    MAX. C-VEL. IS CONSTRAINT AND OCCURS BETWEEN NODES (   2,   1,   1) AND (   1,   1,   1)
THERE ARE NO FLUID SOURCES IN THE TRANSPORT SUBGRID

NUMBER OF MOVES FOR ALL STABILITY CRITERIA:
   CELDIS   INJECTION
      241           1

CELDIS IS LIMITING

         TIME STEP    1 IN STRESS PERIOD    1

         NO. OF PARTICLE MOVES REQUIRED TO COMPLETE THIS TIME STEP  =  241
           MOVE TIME STEP (TIMV)=  4.979253113270E-01
```

(NUMERICAL SOLUTION TO TRANSPORT EQUATION STARTS AT THIS POINT)

```
  NP       =      360 AT START OF MOVE          IMOV     =            1
  No. of solver iterations =    1 Relative residual =   1.9848E-09
  NP       =      360 AT START OF MOVE          IMOV     =            2
  No. of solver iterations =    1 Relative residual =   3.6997E-08

...

  NP       =      361 AT START OF MOVE          IMOV     =          240
  No. of solver iterations =    1 Relative residual =   7.8027E-08
  NP       =      361 AT START OF MOVE          IMOV     =          241
  No. of solver iterations =    1 Relative residual =   9.7147E-08
```

TRACK PROGRESS OF MOVES, NUMBER OF ACTIVE PARTICLES, AND IMPLICIT SOLVER

```
          SOLUTE BUDGET AND MASS BALANCE FOR TRANSPORT SUBGRID

     VALUES CALCULATED AT END OF:
              STRESS PERIOD    1  OUT OF    1
             FLOW TIME STEP    1  OUT OF    1
   TRANSPORT TIME INCREMENT  241  OUT OF  241

     ELAPSED TIME =  1.2000E+02

     CHEMICAL MASS IN STORAGE:
         INITIAL:   MASS DISSOLVED =  0.0000E+00     MASS SORBED =  0.0000E+00
         PRESENT:   MASS DISSOLVED =  1.1391E-01     MASS SORBED =  0.0000E+00

              CHANGE IN MASS STORED = -1.1391E-01
```

```
CUMULATIVE SOLUTE MASS  (L**3)(M/VOL)
----------------------

     IN:
     ---
                DECAY =  0.0000E+00
        CONSTANT HEAD =  0.0000E+00
     SUBGRID BOUNDARY =  1.2001E-01
             RECHARGE =  0.0000E+00
                WELLS =  0.0000E+00
               RIVERS =  0.0000E+00
               DRAINS =  0.0000E+00
GENL. HEAD-DEP. BDYS. =  0.0000E+00
   EVAPOTRANSPIRATION =  0.0000E+00
 SPECIFIED FLOW (FHB) =  0.0000E+00

             TOTAL IN =  1.2000E-01


    OUT:
    ----
                DECAY =  0.0000E+00
        CONSTANT HEAD =  0.0000E+00
     SUBGRID BOUNDARY = -6.3127E-03
             RECHARGE =  0.0000E+00
                WELLS =  0.0000E+00
               RIVERS =  0.0000E+00
               DRAINS =  0.0000E+00
GENL. HEAD-DEP. BDYS. =  0.0000E+00
   EVAPOTRANSPIRATION =  0.0000E+00
 SPECIFIED FLOW (FHB) =  0.0000E+00

            TOTAL OUT = -6.3127E-03


    SOURCE-TERM DECAY =  0.0000E+00



             RESIDUAL = -2.2171E-04

  PERCENT DISCREPANCY = -1.8476E-01 RELATIVE TO MASS FLUX IN
```

ITEMIZED

BUDGETS FOR

SOLUTE FLUXES

Following (enclosed in a border) are the abridged contents of the observation well output file for the sample problem. This output file was generated using the option to write all observation well data to a single file (IOBSFL = 1).

File name: `finite.oba`

```
"OBSERVATION WELL DATA"
"TIME, THEN HEAD AND CONC. FOR EACH OBS. WELL AT NODE (K,I,J)"
"  TIME:      H & C AT   1,  1,  2    H & C AT   1,  1, 42    H & C AT   1,  1,112 "
0.0000E+00   0.000E+00   0.000E+00   0.000E+00   0.000E+00   0.000E+00   0.000E+00
4.9793E-01   1.200E+01   2.763E-01   8.000E+00   5.865E-32   1.000E+00   0.000E+00
9.9585E-01   1.200E+01   7.444E-01   8.000E+00   8.686E-30   1.000E+00   0.000E+00
1.4938E+00   1.200E+01   6.607E-01   8.000E+00   4.280E-28   1.000E+00   0.000E+00
1.9917E+00   1.200E+01   8.175E-01   8.000E+00   1.949E-26   1.000E+00   0.000E+00
2.4896E+00   1.200E+01   8.219E-01   8.000E+00   4.459E-25   1.000E+00   0.000E+00
...
...
1.1851E+02   1.200E+01   1.000E+00   8.000E+00   1.000E+00   1.000E+00   7.043E-01
1.1900E+02   1.200E+01   1.000E+00   8.000E+00   1.000E+00   1.000E+00   7.133E-01
1.1950E+02   1.200E+01   1.000E+00   8.000E+00   1.000E+00   1.000E+00   7.255E-01
1.2000E+02   1.200E+01   1.000E+00   8.000E+00   1.000E+00   1.000E+00   7.334E-01
```

Following (enclosed in a border) are the partial contents of the separate ASCII output file for concentration in a table format style. Initial concentrations follow the first header line; final concentrations follow the second (internal) header line.

File name: *finite.cna*

```
CONCENTRATIONS AT NODES (X,Y,Z,CONC): IMOV=    0, KSTP=    0, KPER=    0, SUMTCH=0.0000E+00
  1.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  2.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  3.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  4.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  5.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  6.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  7.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  8.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
  9.5000E-01  5.0000E-02  5.0000E-01  0.0000E+00
...

1.1650E+01  5.0000E-02  5.0000E-01  0.0000E+00
  1.1750E+01  5.0000E-02  5.0000E-01  0.0000E+00
  1.1850E+01  5.0000E-02  5.0000E-01  0.0000E+00
  1.1950E+01  5.0000E-02  5.0000E-01  0.0000E+00
  1.2050E+01  5.0000E-02  5.0000E-01  0.0000E+00
CONCENTRATIONS AT NODES (X,Y,Z,CONC): IMOV=  241, KSTP=    1, KPER=    1, SUMTCH=1.2000E+02
  1.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  2.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  3.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  4.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  5.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  6.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  7.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  8.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
  9.5000E-01  5.0000E-02  5.0000E-01  1.0000E+00
...

1.1650E+01  5.0000E-02  5.0000E-01  6.1888E-01
  1.1750E+01  5.0000E-02  5.0000E-01  5.9739E-01
  1.1850E+01  5.0000E-02  5.0000E-01  5.7546E-01
  1.1950E+01  5.0000E-02  5.0000E-01  5.5432E-01
  1.2050E+01  5.0000E-02  5.0000E-01  5.3902E-01
```