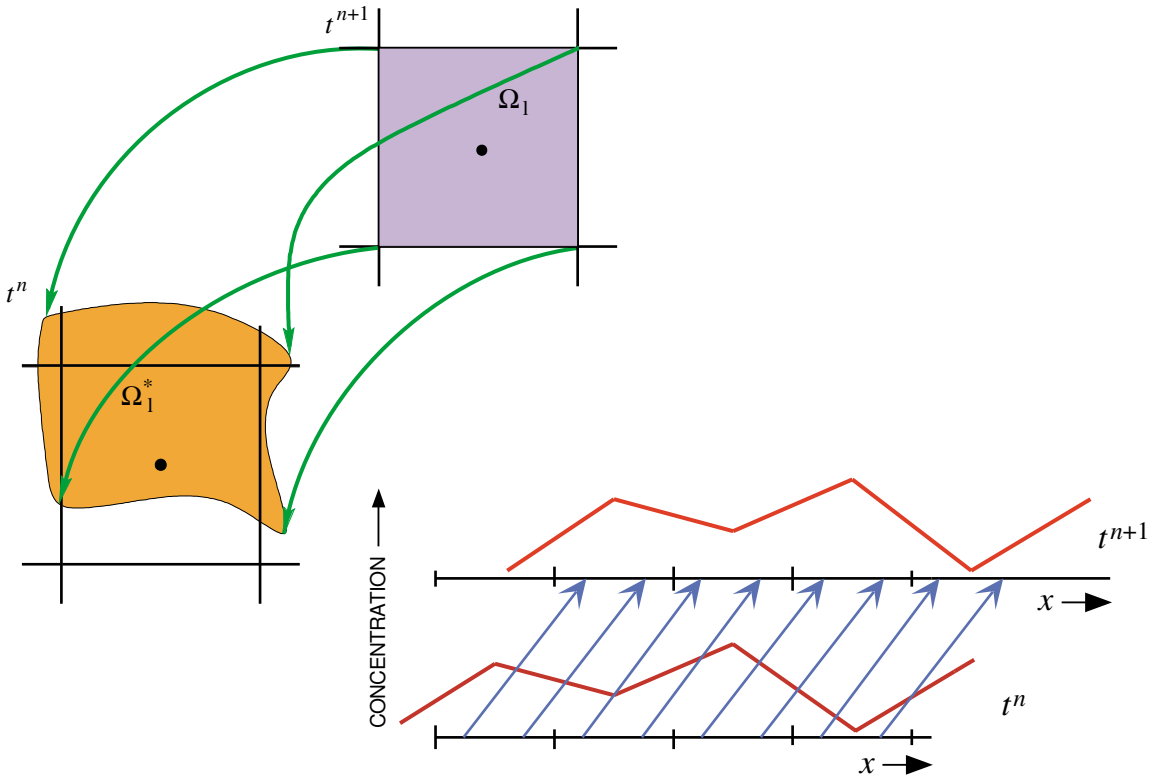
USGS
science for a changing world

A Three-Dimensional Finite-Volume Eulerian-Lagrangian Localized Adjoint Method (ELLAM) for Solute-Transport Modeling



U.S. GEOLOGICAL SURVEY
Water-Resources Investigations Report 00-4087

Cover: Illustrations of backtracking and forward tracking approaches for calculating changes in concentration of a solute caused by advective transport (see figures 5 and 6 of this report).

A Three-Dimensional Finite-Volume Eulerian-Lagrangian Localized Adjoint Method (ELLAM) for Solute-Transport Modeling

By C.I. Heberton[1], T.F. Russell[1], L.F. Konikow[2], and G.Z. Hornberger[2]

[1] Dept. of Mathematics, University of Colorado at Denver
[2] U.S. Geological Survey

U.S. GEOLOGICAL SURVEY

Water-Resources Investigations Report 00-4087

Reston, Virginia
2000

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
Branch of 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 (*ELLAM*) documented in this report solves an integral form of the solute-transport equation, which is an expression of local conservation of mass. The code incorporates an implicit difference approximation in time for the dispersive term. Thus *ELLAM* offers an alternative to previously documented particle-tracking methods of *MOC3D*. It yields a precise global mass balance and can provide qualitatively good results for advection-dominated systems having very high Courant numbers at low computational cost.

The new *MOC3D* code 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 **http://water.usgs.gov/**. The site 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. The code and documentation will also be available through an alternative web page for USGS ground-water models at **http://water.usgs.gov/nrp/gwsoftware/**. When this code is revised or updated in the future, new releases will be made available at these same sites. The code that incorporates the *ELLAM* algorithm, as documented in this report, is designated as Version 3.5.

Although extensive testing of the *ELLAM* 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

Page

ABSTRACT 1
INTRODUCTION 1
THEORETICAL BACKGROUND AND GOVERNING EQUATIONS 3
Governing Equation for Solute Transport 3
Review of Assumptions 5
NUMERICAL METHODS 5
Ground-Water Flow Equation 6
Average Interstitial Velocity 6
Solute-Transport Equation 7
ELLAM 7
Cell Integral Equations 7
Outflow Boundary Equations 8
Mass Tracking 9
Decay 10
Numerical Integration 11
Dispersion 11
Mass Storage at New Time Level 11
Mass Storage at Old Time Level 13
Approximate Test Functions 13
Source Integral 16
Sink Integral 16
Inflow Boundary Integral 17
Outflow Integrals 17
Accuracy Criteria 18
Mass Balance 19
Special Problems 20
Review of *ELLAM* Assumptions 20
COMPUTER PROGRAM 21
Program Segments 22
Guidance on Input Parameter Values 24
MODEL TESTING AND EVALUATION 25
One-Dimensional Flow 26
Uniform Flow, Three-Dimensional Transport 30
Two-Dimensional Radial Flow 32
Point Initial Condition in Uniform Flow 33
Constant Source in Nonuniform Flow 37
Relative Computational and Storage Efficiency 40
CONCLUSIONS 41
REFERENCES 42
APPENDIX A: DATA INPUT INSTRUCTIONS FOR *MOC3D (Version 3.5)* 43
MODFLOW Name File 43
MODFLOW Source and Sink Packages 43

Page

MOC3D Input Data Files 43
MOC3D Transport Name File (CONC) 44
Main *MOC3D* Package Input (*MOC*, *MOCIMP*, or *ELLAM*) 45
Source Concentration in Recharge File (CRCH) 53
Observation Well File (OBS) 53
APPENDIX B: ANNOTATED INPUT DATA SET FOR EXAMPLE PROBLEM 54
APPENDIX C: SELECTED OUTPUT FOR EXAMPLE PROBLEM 58

FIGURES

1-9. Diagrams showing:
1. Notation used to label rows, columns, and nodes within one layer (k) of a three-dimensional, block-centered, finite-difference grid for *MOC3D* (from Konikow and others, 1996) 6
2. Representative three-dimensional grid for *MOC3D* illustrating notation for layers (from Konikow and others, 1996) 6
3. Schematic representation (for a simple case having constant velocity) showing how mass is advected into cell Ω_l at time level t^{n+1} from the inflow boundary, from a fluid source in cell 1, and from storage at time level t^n in cells 1 and 2 8
4. Schematic representation showing how mass is advected to an outflow boundary at time level t^{n+1} from the inflow boundary, from a fluid source in cell 3, and from storage at time level t^n in all five cells 9
5. Two-dimensional example illustrating that pre-image of a cell may be irregularly shaped and not easily defined by backtracking from t^{n+1} to t^n 10
6. Solute concentration versus distance to illustrate one-dimensional advection of known mass distribution from old time level to new time level (Courant number = 1) 10
7. Examples of spatial distribution of approximate test functions (w_l) for selected one-dimensional cases 14
8. Simplified flow chart for the transport loop of the *ELLAM* calculation process 24
9. Schematic representation of tracking mass from an inflow boundary 25

10–13. Graphs showing:
10. Numerical and analytical solutions at three different locations for solute transport in a one-dimensional, steady flow field 27
11. Numerical and analytical solutions for the case of increased dispersivity 28

Page

12. Numerical and analytical solutions for three different times for same one-dimensional, steady flow, solute-transport problem shown in figure 10..... 29

13. Numerical and analytical solutions for three different times for case in which solute is subject to decay at rate of $\lambda = 0.01\ s^{-1}$ 29

14-22. Diagrams showing:

14. Concentration contours for (a) analytical and (b-d) *ELLAM* numerical solutions in the horizontal plane containing the solute source (layer 1) for three-dimensional solute transport in a uniform steady flow field at $t = 400$ days..... 31

15. Contours of relative concentrations calculated using (a) analytical and (b-e) numerical *ELLAM* models for solute transport in a steady radial flow field 33

16. Concentration contours for (a) analytical and (b) numerical solutions for transport of a point initial condition in uniform flow in the x-direction at $t = 90$ days..... 35

17. Concentration contours for (a) analytical and (b) *ELLAM* numerical solutions for transport of a dispersed-point initial condition in uniform flow in the x-direction at $t = 130$ days..... 35

18. Concentration contours for (a) analytical and (b) *ELLAM* numerical solutions for transport of a dispersed-point initial condition in uniform flow at 45 degrees to the x-direction at $t = 130$ days 36

19. Concentration contours for *ELLAM* numerical solution showing areas of calculated negative concentrations for problem represented in figure 18b..... 37

20. 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) *ELLAM* solution using CELDIS = 30..... 38

21. Three-dimensional simulation results for nonuniform-flow test case in which $\alpha_{TH} = 0.1$ m and $\alpha_{TV} = 0.01$ m: (a) finite-element model (modified from Burnett and Frind, 1987, figure 8c), and (b) numerical *ELLAM* solution using CELDIS = 30 39

22. 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) numerical *ELLAM* solution using CELDIS = 21..... 39

TABLES

Page

1. *ELLAM* transport loop 22
2. Calling tree for *ELLAM* code, showing hierarchy of secondary subroutines for transport calculations 23
3. Base-case parameters used in *ELLAM* simulation of solute transport in a one-dimensional, steady-state flow system 26
4. Base-case parameters used in *ELLAM* simulation of transport from a continuous point source in a three-dimensional, uniform, steady-state flow system 30
5. Parameters used in *ELLAM* simulation of two-dimensional, steady-state, radial flow case, showing range of values tested for selected numerical parameters 32
6. Parameters used in *ELLAM* 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 34
7. Parameters used for *ELLAM* 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) 38
8. Execution times and storage requirements for *MOC3D* and *ELLAM* for selected test cases..... 40
9. Formats associated with *MOC3D* print flags 50

A Three-Dimensional Finite-Volume Eulerian-Lagrangian Localized Adjoint Method (ELLAM) for Solute-Transport Modeling

by C.I. Heberton
T.F. Russell
L.F. Konikow
G.Z. Hornberger

ABSTRACT

This report documents the U.S. Geological Survey *Eulerian-Lagrangian Localized Adjoint Method* (*ELLAM*) algorithm that solves an integral form of the solute-transport equation, incorporating an implicit-in-time difference approximation for the dispersive and sink terms. Like the algorithm in the original version of the U.S. Geological Survey *MOC3D* transport model, *ELLAM* uses a method of characteristics approach to solve the transport equation on the basis of the velocity field. The *ELLAM* algorithm, however, is based on an integral formulation of conservation of mass and uses appropriate numerical techniques to obtain global conservation of mass. The implicit procedure eliminates several stability criteria required for an explicit formulation. Consequently, *ELLAM* allows large transport time increments to be used. *ELLAM* can produce qualitatively good results using a small number of transport time steps.

A description of the *ELLAM* numerical method, the data-input requirements and output options, and the results of simulator testing and evaluation are presented. The *ELLAM* algorithm was evaluated for the same set of problems used to test and evaluate Version 1 and Version 2 of *MOC3D*. These test results indicate that *ELLAM* offers a viable alternative to the explicit and implicit solvers in *MOC3D*. Its use is desirable when mass balance is imperative or a fast, qualitative model result is needed. Although accurate solutions can be generated using *ELLAM*, its efficiency relative to the two previously documented solution algorithms is problem dependent.

INTRODUCTION

This report documents the Finite-Volume *Eulerian-Lagrangian Localized Adjoint Method* (*ELLAM*) (see Healy and Russell, 1993), an alternative algorithm to the U.S. Geological Survey (USGS) *MOC3D* transport model that 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) (Konikow and others, 1996; Kipp and others, 1998). *ELLAM* solves a mass-conservative integral form of the solute-transport equation (Celia and others, 1990). The code uses an implicit method for dispersion calculations, which allows for large time steps without stability constraints. *ELLAM* uses an Eulerian-Lagrangian approach, first solving for advective transport by tracking mass through time and then solving a dispersion equation on a grid fixed in space. This is particularly advantageous for advection-dominated problems, as are typical of many field problems involving ground-water contamination, because the Eulerian-Lagrangian approach will generate less numerical dispersion than standard Eulerian approaches using finite-difference and finite-element methods. *ELLAM* solves integral equations and thus tracks mass associated with fluid volumes, so that it conserves mass locally and globally. The original *MOC3D* approach of inserting and tracking fluid particles carrying concentration, which could yield only an approximate mass balance, is therefore not used in *ELLAM*.

ELLAM is integrated into the *MOC3D* model, which is itself integrated with *MODFLOW-96*, the 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. This report also assumes reader familiarity with documentation for *MOC3D* (Version 1), to which this report is an extension.

MOC3D and *ELLAM* are offered as general simulators that are 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 prior reports by Konikow and others (1996) and Kipp and others (1998). In some situations, the model results could be inaccurate or model operation costly. This report includes guidelines for recognizing these situations and avoiding such problems.

This implementation of *MOC3D* is limited to fluid properties, such as density and viscosity, that are uniform and constant, and thus independent of concentration values. 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 *MOC3D* computer program is written in FORTRAN-77 and has been developed in a modular style. This documentation contains a description of the *ELLAM* algorithm used to solve the solute-transport equation in *MOC3D*. A description of the data requirements, input format specifications, program options, and output formats is included in the appendixes. This report should be used in conjunction with the original *MOC3D* documentation (Konikow and others, 1996), which provides information on all features of *ELLAM* previously existing in *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 R.W. Healy and K.L. Kipp.

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, eq. 6):

$$\frac{\partial(\varepsilon C)}{\partial t}+\frac{\partial\left(\rho_b \overline{C}\right)}{\partial t}+\frac{\partial}{\partial x_i}\left(\varepsilon C V_i\right)-\frac{\partial}{\partial x_i}\left(\varepsilon D_{ij}\frac{\partial C}{\partial x_j}\right)$$

$$-\sum C'W+\lambda\left(\varepsilon C+\rho_b \overline{C}\right)=0 \qquad (1)$$

(summation over repeated indices is understood), 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), $\mathbf{V}$ is a vector of interstitial fluid velocity components (LT^{-1}), $\mathbf{D}$ 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).

The terms controlling sorption are combined into a single parameter--the retardation factor (R_f)--assumed to be constant in time because on a linear isotherm, $C/\overline{C}$ is constant. The retardation factor is defined as:

$$R_f=1+\frac{\rho_b \overline{C}}{\varepsilon C}. \qquad (2)$$

An integral form of the solute-transport equation, which is a statement of conservation of mass over the domain of integration, is the governing equation for this finite-volume *ELLAM* approach. Integration against a "test function" is used to provide the formulation of conservation of mass, including treatment of cell or subdomain boundary conditions and solute decay.

The test function effectively specifies the domain of integration for the transport equation by the portion of the space-time domain where its value is nonzero. On a subdomain of integration, the test function can be seen as an integration weight at each point. Varying the weight along streamlines of the flow is a convenient mechanism to provide solute growth or decay.

If we divide eq. 1 by R_f, multiply by a test function u, integrate over time and space, and assume R_f is constant in time, we have:

$$\int_\Omega\int_0^T\left(u\frac{\partial(\varepsilon C)}{\partial t}+\frac{u}{R_f}\nabla\cdot\left(\varepsilon C\mathbf{V}-\varepsilon\mathbf{D}\nabla C\right)\right.$$

$$\left.-\frac{u}{R_f}\sum C'W+u\lambda\varepsilon C\right)\mathrm{d}t\mathrm{d}x=0 \qquad (3)$$

where Ω is the entire spatial transport subdomain, and T is the end of the simulation time period starting at time zero.

Equation 3 is integrated by parts using

$$u\frac{\partial(\varepsilon C)}{\partial t}=\frac{\partial(u\varepsilon C)}{\partial t}-\frac{\partial u}{\partial t}\varepsilon C \qquad (4)$$

and

$$\frac{u}{R_f}\nabla\cdot\left(\varepsilon C\mathbf{V}-\varepsilon\mathbf{D}\nabla C\right)=$$

$$\frac{1}{R_f}\nabla\cdot\left(u\varepsilon C\mathbf{V}-u\varepsilon\mathbf{D}\nabla C\right)$$

$$-\frac{1}{R_f}\nabla u\cdot\left(\varepsilon C\mathbf{V}-\varepsilon\mathbf{D}\nabla C\right) \qquad (5)$$

to yield the global equation,

$$\int_\Omega \int_0^T \left(\frac{\partial(u\varepsilon C)}{\partial t} + \frac{1}{R_f} \nabla \cdot (u\varepsilon C\mathbf{V} - u\varepsilon \mathbf{D} \nabla C) \right.$$
$$+ \frac{1}{R_f} \nabla u \cdot \varepsilon \mathbf{D} \nabla C - \frac{u}{R_f} \sum C'W$$
$$\left. - \varepsilon C \left(\frac{\partial u}{\partial t} + \frac{\mathbf{V}}{R_f} \cdot \nabla u - \lambda u \right) \right) dt\, d\mathbf{x} = \mathbf{0} \quad (6)$$

The Eulerian-Lagrangian aspects of the method derive from the requirement that the test function satisfy the adjoint equation, $\frac{\partial u}{\partial t} + \frac{\mathbf{V}}{R_f} \cdot \nabla u - \lambda u = 0$. Thus, for the time step from t^n to t^{n+1}, use u of the form $u(\mathbf{x},t) = f(\mathbf{x},t) e^{-\lambda\left(t^{n+1} - t\right)}$, where $\frac{\partial f}{\partial t} + \frac{\mathbf{V}}{R_f} \cdot \nabla f = 0$, so that f is constant along characteristics of the retarded interstitial velocity field. Note that with $u = e^{-\lambda(t^{n+1} - t)}$ (that is, $f = 1$) in the following for $t^n \le t \le t^{n+1}$, and $u = 0$ otherwise, we arrive at a statement of global conservation of mass for a time step:

$$\int_\Omega (u\varepsilon C)^{n+1} d\mathbf{x} - \int_\Omega (u\varepsilon C)^n d\mathbf{x}$$
$$+ \int_\Omega \int_{t^n}^{t^{n+1}} \left(\frac{1}{R_f} \nabla \cdot (u\varepsilon C\mathbf{V} - u\varepsilon \mathbf{D} \nabla C) \right.$$
$$\left. + \frac{1}{R_f} \nabla u \cdot \varepsilon \mathbf{D} \nabla C - \frac{u}{R_f} \sum C'W \right) dt\, d\mathbf{x} = 0 \quad (7)$$

For a local conservation equation on each finite-difference cell Ω_l in the transport subdomain, let

$$u_l(\mathbf{x},t) = f_l(\mathbf{x},t) e^{-\lambda\left(t^{n+1} - t\right)} \quad (8)$$

where $f_l(\mathrm{x}, t^{n+1}) = 1$ on Ω_l and $f_l(\mathrm{x}, t^{n+1}) = 0$ elsewhere.

We thus arrive at a system of local *ELLAM* equations,

$$\int_{\Omega_l} (\varepsilon C)^{n+1} d\mathbf{x} - e^{-\lambda \Delta t} \int_{\Omega_l^*} (\varepsilon C)^n d\mathbf{x}$$
$$+ \iint\limits_{\text{supp } u_l \cap \Gamma^{n+1}} \frac{e^{-\lambda\left(t^{n+1} - t\right)}}{R_f} (\varepsilon C\mathbf{V} - \varepsilon \mathbf{D} \nabla C) \cdot \mathbf{n}\, dt\, d\mathbf{s}$$
$$- \int_{t^n}^{t^{n+1}} \int_{\partial\, \text{supp } u_l} \frac{e^{-\lambda\left(t^{n+1} - t\right)}}{R_f} (\varepsilon \mathbf{D} \nabla C) \cdot \mathbf{n}\, dt\, d\mathbf{s}$$
$$- \iint\limits_{\text{supp } u_l \cap \text{supp } W} \frac{e^{-\lambda\left(t^{n+1} - t\right)}}{R_f} \sum C'W\, dt\, d\mathbf{x} = 0. \quad (9)$$

The notations introduced in eq. 9 are explained as follows. First, *supp f* denotes the support of a function *f*, that is, the part of its domain on which it is nonzero; for example, *supp W* is the union of all finite-difference cells containing a source or sink having a nonzero flow rate. If *S* is a subset of the spatial transport subdomain Ω, then ∂S denotes its boundary, and *S** represents its pre-image at time t^n under advection along characteristics; that is, $\mathbf{x} \in S^*$ whenever the characteristic that starts at *x* at time t^n and tracks forward in time lands in *S* at time t^{n+1}. Next, $\Gamma^{n+1} \equiv \partial\Omega \times \left(t^n, t^{n+1}\right)$ denotes the space-time boundary over the current time step. If *B* is a subset of Γ^{n+1}, the pre-image *B** can again be defined as a subset of Ω; $\mathbf{x} \in B^*$ whenever the forward characteristic from $(\mathbf{x}, t^n)$ meets the outflow boundary somewhere in *B*. Finally, **n** is the outward unit normal vector on the specified boundary, and $d\mathbf{x}$ and $d\mathbf{s}$ signify differential volume and boundary area, respectively.

Note that eq. 9 has the form of space-time integrals of dispersion equations. *ELLAM* can be viewed as a method of characteristics, tracking mass along streamlines of the flow to accumulate data to the right-hand side of the system of equations.

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-derivative terms of the hydraulic-conductivity tensor are eliminated.
3. Gradients of fluid density, viscosity, and temperature do not affect the velocity distribution.
4. Chemical reactions do not affect the fluid or aquifer properties.
5. The dispersivity coefficients are constant over a flow time step, 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 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. Our notation, however, 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.

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

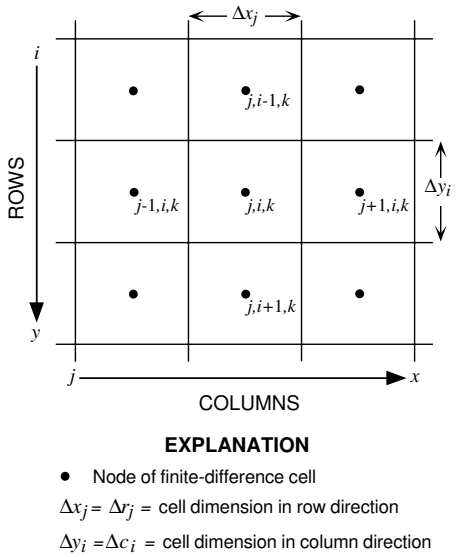


Figure 1. Notation used to label rows, columns, and nodes within one layer (*k*) of a three-dimensional, block-centered, finite-difference grid for *MOC3D* (from Konikow and others, 1996).

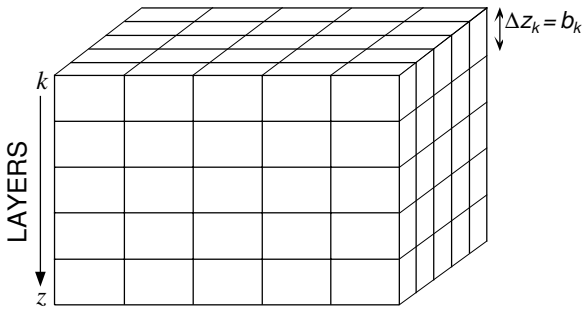


Figure 2. Representative three-dimensional grid for *MOC3D* illustrating notation for layers (from Konikow and others, 1996)

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 mass-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 linearly interpolated estimates of specific discharge at those points divided by the effective porosity of the cell (see Konikow and others, 1996).

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. The challenge for a numerical solution method is to represent solute fronts without introducing either erroneous oscillations or nonphysical dispersion, and to do so without requiring a cost-prohibitive use of computer resources.

ELLAM

The *ELLAM* method advances in time by considering the space-time domain $\Omega \times \left(t^n, t^{n+1}\right)$. Part of this domain flows out of the transport subdomain during the time step, and separate outflow boundary equations below address this part. The remaining part is modeled using cell integral equations that do not contain outflow terms. The next two subsections present these equations in turn. Overall, they can be coupled in such a way that the outflow concentrations contribute to the right-hand side of the system of cell integral equations.

Like the algorithms in the previous versions of *MOC3D, ELLAM* approximates total solute flux across the transport subdomain boundary by the advective flux; that is, the dispersive flux is neglected. This is not required by *ELLAM* methods in general, but rather is a feature of this particular implementation. As a result, boundary-face concentrations are not coupled to cell-center concentrations through a dispersive concentration gradient, simplifying the separation of outflow boundary equations from cell integral equations. All mass moving across the transport subdomain boundary can be tracked by the advective algorithm, and at outflow boundaries, this mass provides data for the outflow boundary equations. Solutions of these outflow equations, together with user-input inflow concentrations, appear on the right-hand side of the cell integral equations, which represent local mass conservation on finite-difference cells of the transport subdomain.

ELLAM approaches the hyperbolic-parabolic nature of the solute-transport equation by combining a method of characteristics technique for advection with a backward Euler in time and centered differences in space solution to a diffusion equation. The following sections describe key elements of the numerical methods in more detail.

Cell Integral Equations

Taking the conservation of mass equation for each cell (eq. 9), we approximate the total boundary flux with advective flux (that is, the dispersive flux is taken to be zero) and the dispersion time integral with a backward Euler formulation, and then

rearrange terms. The system of equations to be solved is then:

$$\int_{\Omega_l} (\varepsilon C)^{n+1} d\mathbf{x} - \Delta t \int_{\partial \Omega_l} \frac{1}{R_f} (\varepsilon \mathbf{D} \nabla C)^{n+1} \cdot \mathbf{n}\, d\mathbf{s} =$$

$$e^{-\lambda \Delta t} \int_{\Omega_l^*} (\varepsilon C)^n d\mathbf{x}$$

$$- \iint\limits_{\text{supp } u_l \cap \Gamma^{n+1}} e^{-\lambda\left(t^{n+1}-t\right)} \varepsilon\, C_{\text{inflow}} \frac{\mathbf{V}}{R_f} \cdot \mathbf{n} dt\, d\mathbf{s}$$

$$+ \iint\limits_{\text{supp } u_l \cap \text{supp } W} e^{-\lambda\left(t^{n+1}-t\right)} \sum C' \frac{W}{R_f} dt\, d\mathbf{x} \qquad (10)$$

where Ω_l^* means the pre-image in the spatial domain at t^n of Ω_l at t^{n+1}; $\Gamma^{n+1} \equiv \partial\Omega \times \left(t^n, t^{n+1}\right)$ is the space-time boundary at time step n+1; and $supp\, f \equiv \{x / f(x) \neq 0\}$. These integral equations are solved for C^{n+1}, the concentration at the new (n+1) time level at each cell center. Note that the right-hand side of eq. 10 consists of advective mass contributions from storage (that is, advection of mass in the domain at the start of the time increment), inflow boundaries, and sources. This is illustrated schematically for a simple case having constant velocity in figure 3, which shows how mass is advected into cell Ω_l at time level t^{n+1} from an inflow boundary, from a fluid source in a nearby cell, and from the mass present at time level t in nearby cells.

Outflow Boundary Equations

The integral in eq. 7 expressing mass crossing the transport subdomain boundary during a time step is:

$$\int_{\Omega} \int_{t^n}^{t^{n+1}} \frac{1}{R_f} \nabla \cdot \left(u\varepsilon C \mathbf{V} - u\varepsilon \mathbf{D} \nabla C\right) dt\, d\mathbf{x} =$$

$$\int_{t^n}^{t^{n+1}} \int_{\partial \Omega} \frac{1}{R_f} \left(u\varepsilon C \mathbf{V} - u\varepsilon \mathbf{D} \nabla C\right) \cdot \mathbf{n}\, d\mathbf{s} dt \qquad (11)$$

Considering just the outflow portion of the boundary, this becomes

$$\int_{t^n}^{t^{n+1}} \int_{(\partial \Omega)_{\text{outflow}}} \frac{1}{R_f} \left(u\varepsilon C \mathbf{V} - u\varepsilon \mathbf{D} \nabla C\right) \cdot \mathbf{n}\, d\mathbf{s} dt$$

$$\approx \int_{(\partial \Omega)_{\text{outflow}}} \int_{t^n}^{t^{n+1}} \left(u\varepsilon\, C_{\text{outflow}} \frac{\mathbf{V}}{R_f} \right) \cdot \mathbf{n}\, dt\, d\mathbf{s} \qquad (12)$$

where total flux across the boundary is now approximated by advective flux.

We index the outflow boundary faces with *ll* and define the following test functions:

$$u_{ll} = \begin{cases} e^{-\lambda\left(t^{n+1}-t\right)} & \text{on characteristics from } \Omega \text{ at time level } n \text{ into boundary area } (\partial\Omega)_{ll} \text{ at any time during time step} \\ 0 & \text{otherwise.} \end{cases} \qquad (13)$$

The mass across outflow boundary face *ll* is the mass stored at the previous time level that flows across the face, together with any inflow and source mass that both enters the transport subdomain and leaves through face *ll* during the time step.

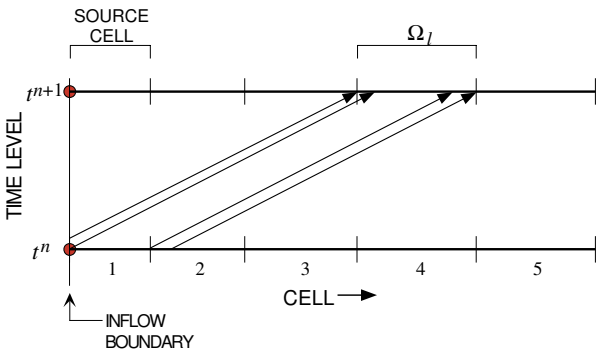


Figure 3. Schematic representation (for a simple case having constant velocity) showing how mass is advected into cell Ω_l at time level t^{n+1} from the inflow boundary, from a fluid source in cell 1, and from storage at time level t^n in cells 1 and 2.

Taking $u = u_{ll}$ on the right-hand side of eq. 12, and including those terms from eq. 7 that are appropriate in the context of the outflow boundary, we can write three terms representing mass contributions from storage, inflow, and sources, as follows:

$$\int_{(\partial\Omega)_{ll}} \int_{t^n}^{t^{n+1}} \left(e^{-\lambda\left(t^{n+1}-t\right)} \varepsilon C_{\text{outflow}} \frac{\mathbf{V}}{R_f} \right) \cdot \mathbf{n}\, dt\, ds = e^{-\lambda\Delta t} \int_{(\partial\Omega)_{ll}^*} (\varepsilon C)^n d\mathbf{x}$$

$$- \iint_{\text{supp } u_{ll} \cap \partial\Omega_{\text{inflow}} \times \left(t^n, t^{n+1}\right)} e^{-\lambda\left(t^{n+1}-t\right)} \varepsilon C_{\text{inflow}} \frac{\mathbf{V}}{R_f} \cdot \mathbf{n}\, dt\, d\mathbf{s}$$

$$+ \iint_{\text{supp } u_{ll} \cap \text{supp W} \times \left(t^n, t^{n+1}\right)} e^{-\lambda\left(t^{n+1}-t\right)} \sum C' \frac{W}{R_f} dt\, d\mathbf{x} \qquad (14)$$

where $(\partial\Omega)_{ll}$ is a discretized portion of the boundary face, and $(\partial\Omega)_{ll}^*$ is its pre-image, in the manner described following eq. 9. The advection of mass to an outflow boundary is illustrated schematically for a simple case having constant velocity in figure 4, where mass is advected to an outflow boundary at time level t^{n+1} from an inflow boundary, from a fluid source in a nearby cell, and from the mass present at time level t in nearby cells.

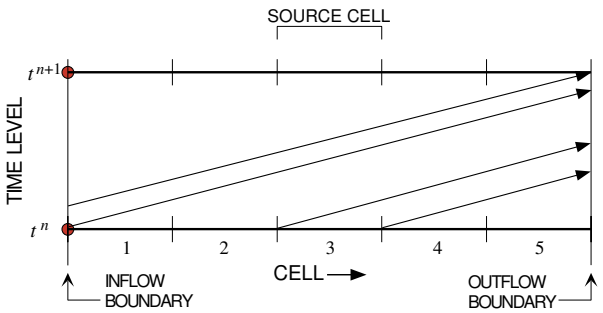


Figure 4. Schematic representation showing how mass is advected to an outflow boundary at time level t^{n+1} from the inflow boundary, from a fluid source in cell 3, and from storage at time level t^n in all five cells.

ELLAM equations are a formulation of mass conservation on each cell. Therefore, approximations to concentrations result that conserve mass locally (on each finite-difference cell) and globally (on the entire transport subdomain).

Mass Tracking

For each cell in the fixed finite-difference grid, the integrals on the right-hand side of eq. 10 represent solute mass advected into the cell during the time step from storage (that is, advection of mass in the domain at the start of the time increment), the transport subdomain boundary, or a fluid source, respectively.

Advection in flowing ground water is simulated by mass tracking along the characteristic curves determined by the seepage velocity. Calculation of advective movement during a flow time step is based on the specific discharges computed at the end of the step.

As in *MOC3D*, tracking is performed using linear interpolation of velocity in the direction of the component of interest and piecewise-constant interpolation in the other two directions. The approach is to solve a system of three ordinary differential equations to find the characteristic curves [$x = x(t)$, $y = y(t)$, and $z = z(t)$] along which fluid is advected:

$$\frac{dx}{dt} = \frac{V_x}{R_f} \qquad (15)$$

$$\frac{dy}{dt} = \frac{V_y}{R_f} \tag{16}$$

$$\frac{dz}{dt} = \frac{V_z}{R_f} \tag{17}$$

This is accomplished by introducing a set of moving points that can be traced within the stationary coordinates of a finite-difference grid. Each point 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 point moves through the flow field by the flow velocity acting along its trajectory.

The *ELLAM* equations, eqs. 10 and 14, suggest that mass is tracked backwards along characteristics to the pre-image of each cell or boundary face. It is not possible, however, to exactly locate all of the mass at the previous time level by backtracking a finite number of points (see figure 5). In order to achieve mass balance, this implementation of the *ELLAM* algorithm tracks the known mass distribution forward from the old time level to the new time level (see figure 6). The accuracy of point tracking can be related to the Courant number, which is the ratio of (1) the distance a point will move in one time increment (velocity times Δt) to (2) the grid spacing (Δx).

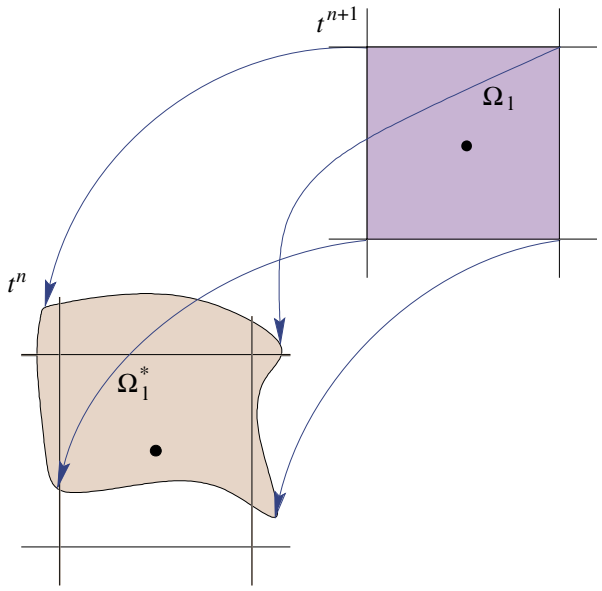


Figure 5. Two-dimensional example illustrating that pre-image of a cell may be irregularly shaped and not easily defined by backtracking from t^{n+1} to t^n.

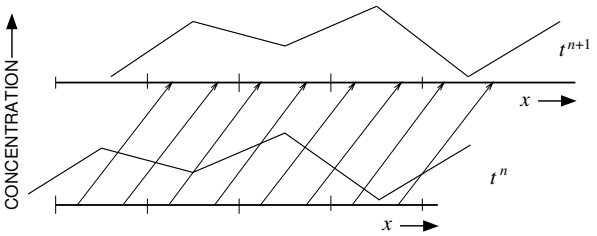


Figure 6. Plots of solute concentration versus distance that illustrate one-dimensional advection of known mass distribution from old time level to new time level (Courant number = 1).

ELLAM tracks points that are the centers of volumes of fluid. Thus, mass in a fluid volume is tracked under advection during a time step, distributed among destination cells, and accumulated to the right hand side storage, inflow, or source integral for each cell.

Decay

When simulating linear decay, all mass in the system at the beginning of each transport time step is decayed over the entire time step by a factor $e^{-\lambda\Delta t}$, where λ is the decay rate. Inflow and source mass are decayed in the same way, where the time interval is now not the entire time step, but the part of it during which new mass is in the transport subdomain.

This decay algorithm has no numerical stability restrictions associated with it. If the half-life is on the order of, or smaller than, the transport time step, however, some accuracy will be lost.

When a solute subject to decay enters the aquifer through a fluid source, it is assumed that the fluid source contains the solute at the concentration specified by C'. *MOC3D* 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. If the problem being simulated requires that the solute in the source fluid itself undergo decay, then the source code will have to be modified.

Numerical Integration

The numerical treatment of each term in eqs. 10 and 14 will next be discussed. The *j,i,k* subscripts for a cell Ω_l will denote the spatial finite-difference grid indexing, as discussed previously in the section "Numerical Methods."

The equations are first divided through by porosity, which is represented by piecewise constants in space and time. This is valid because there are no spatial derivatives of porosity in the local *ELLAM* equations.

Dispersion

Time integration is accomplished using a one point in time backward Euler rule. Spatially, a one point integration rule with a seven point stencil is used:

$$\Delta t \int_{\partial\Omega_l} \frac{1}{\varepsilon R_f}(\varepsilon \mathbf{D}\nabla C)^{n+1}\cdot \mathbf{n}\,ds \approx \frac{\Delta t}{\left(R_f\right)_k(\varepsilon b)^{n+1}_{j,i,k}}\left\{\left[\left(\varepsilon b D_{1m}\frac{\partial C}{\partial x_m}\right)^{n+1}_{j+\frac{1}{2},i,k}-\left(\varepsilon b D_{1m}\frac{\partial C}{\partial x_m}\right)^{n+1}_{j-\frac{1}{2},i,k}\right]\Delta y_i\, b^{n+1}_{j,i,k}\right.$$
$$+\left[\left(\varepsilon b D_{2m}\frac{\partial C}{\partial x_m}\right)^{n+1}_{j,i+\frac{1}{2},k}-\left(\varepsilon b D_{2m}\frac{\partial C}{\partial x_m}\right)^{n+1}_{j,i-\frac{1}{2},k}\right]\Delta x_j\, b^{n+1}_{j,i,k}$$
$$\left.+\left[\left(\varepsilon b D_{3m}\frac{\partial C}{\partial x_m}\right)^{n+1}_{j,i,k+\frac{1}{2}}-\left(\varepsilon b D_{3m}\frac{\partial C}{\partial x_m}\right)^{n+1}_{j,i,k-\frac{1}{2}}\right]\Delta x_j\,\Delta y_i\right\} \qquad (18)$$

where m = 1,2,3 is a summation index for the dispersion term. Finite-difference approximations to the space derivatives in the dispersion integral are calculated using centered differences as in *MOC3D*, but with *ELLAM* they are modified for varying grid dimensions. (See Konikow and others (1996), p. 64 for $\varepsilon b D_m \frac{\partial C}{\partial x_m}$ expansion.)

Mass Storage at New Time Level

The quantity mass/porosity in a cell at the new time level t^{n+1} is expressed using the trapezoidal rule for integration, formulated over each cell octant. Concentrations at octant corners are weighted averages of neighboring node concentrations, determined by trilinear interpolation.

For each octant,

$$\frac{\text{mass}}{\text{porosity}} = \frac{1}{64}\Delta x_j \Delta y_i b^{n+1}_{j,i,k}\sum\nolimits_{corner=1}^{8} C_{corner}$$
$$= \frac{1}{64}\Delta x_j \Delta y_i b^{n+1}_{j,i,k}$$
$$\sum\nolimits_{corner=1}^{8}\sum\nolimits_{nbr=1}^{8}(weight)_{nbr} C_{nbr} \qquad (19)$$

where for an interior octant, *nbr* = *neighbor* and *nbr* is one of the eight grid nodes between which concentration varies trilinearly. In the case of a boundary octant, a boundary face value is needed for calculation, and is taken to be the following:

- Inflow: user input;
- No flow: same as associated interior node; and
- Outflow: calculated using cell parameters, boundary flow rate, and mass tracked across boundary during transport time step.

One important implication of the above procedure, which is designed to be mass conservative, is that the concentration calculated and reported (in the output) for the location of the block-centered node, $C_{j,i,k}$, represents the estimated concentration at that point and not the average concentration in the cell. Thus, unlike many other numerical methods, $C_{j,i,k}$ multiplied by the volume of water in the cell would not necessarily equal the solute mass in the cell.

Coefficients calculated by ((1/8) x *octant volume* x *nodal weight*) for all nodes neighboring a cell comprise the storage matrix entries for the equation for each cell. Boundary terms are put on the right-hand side of the equation because all boundary face concentrations are determined before the solution of the interior equations.

It should be noted that linear interpolation in the vertical dimension is approximate in the case where adjacent cells in the same layer of the transport subdomain have varying thicknesses, as is allowed by *MODFLOW*. Extreme variations could affect accuracy of the solution.

For an interior cell with all neighbors active and using b at time n+1:

$$\int_{\Omega_l} C^{n+1} d\mathbf{x} = \left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)-4\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)$$
$$\left(\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)+\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)-4\right)C_{j,i,k}$$
$$-\left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j-1,i-1,k-1}+\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j-1,i-1,k+1}\right.$$
$$+\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j-1,i+1,k+1}+\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j-1,i+1,k-1}$$
$$+\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j+1,i-1,k-1}+\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j+1,i-1,k+1}$$
$$\left.+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j+1,i+1,k-1}+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j+1,i+1,k+1}\right)$$
$$+\left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)-4\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j,i-1,k-1}+\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j,i-1,k+1}\right.$$
$$\left.+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j,i+1,k-1}+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j,i+1,k+1}\right)$$
$$+\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)\left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j-1,i,k+1}+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j+1,i,k-1}\right.$$
$$\left.+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)C_{j+1,i,k+1}+\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)C_{j-1,i,k-1}\right)$$

$$+\left(\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)+\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)-4\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)C_{j+1,i-1,k}+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)C_{j+1,i+1,k}\right.$$
$$\left.+\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)C_{j-1,i-1,k}+\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)C_{j-1,i+1,k}\right)$$
$$-\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)\left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)+\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)-4\right)\right.$$
$$\left.+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)-4\left(\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)\right)C_{j,i,k-1}$$
$$-\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)\left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)+\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)\left(\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)-4\right)\right.$$
$$\left.+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)-4\left(\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)\right)C_{j,i,k+1}$$
$$-\left(\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k-1}}\right)+\left(\frac{b_{j,i,k}}{b_{j,i,k}+b_{j,i,k+1}}\right)-4\right)\left(\left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)-4\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)C_{j,i-1,k}\right.\right.$$
$$\left.\left.+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)C_{j,i+1,k}\right)\left(\left(\frac{\Delta y_i}{\Delta y_{i-1}+\Delta y_i}\right)+\left(\frac{\Delta y_i}{\Delta y_{i+1}+\Delta y_i}\right)-4\right)\left(\left(\frac{\Delta x_j}{\Delta x_{j-1}+\Delta x_j}\right)C_{j-1,i-1,k}+\left(\frac{\Delta x_j}{\Delta x_{j+1}+\Delta x_j}\right)C_{j+1,i,k}\right)\right) \quad (20)$$

Mass Storage at Old Time Level

The total mass advected into each cell during a transport time step that was already stored within the system at the old time level is needed for the right-hand side of the *ELLAM* equation. Numerically, this is accomplished by tracking mass forward from the old time level, *n*, along characteristics. Each cell is divided into subcells determined by parameters NSC, NSR, NSL, specifying the number of subcells in the column, row, and layer direction, respectively. The center of each subcell is tracked through the time step under advection. Depending on the exact location of this point in the destination cell at the new time, all of the mass in the subcell may or may not also be found in that destination cell. In order to mitigate the effects of unwarranted mass lumping, subcell mass is distributed among cells neighboring the destination cell using the "approximate test functions," w_l, described below. The value of w_l at the subcell center destination point is the fraction of subcell mass to be distributed to cell Ω_l.

This yields the formulation,

$$e^{-\lambda\Delta t}\int_{\Omega_l^*} C^n d\mathbf{x} \approx$$
$$e^{-\lambda\Delta t}\sum_{j,i,k}\sum_{\substack{\text{p=subcell}\\ \text{center}}}\left[\frac{\Delta x_j \Delta y_i b_{j,i,k}}{(NSC)(NSR)(NSL)}\left(w_l\left(p^f\right)C(p)\right)\right] \quad (21)$$

where summation runs through all subcells of each cell in the transport subdomain, and p^f is the image of p under forward tracking to the new time level.

Approximate Test Functions

An approximate test function is defined for each active cell for the purpose of distributing advected mass among neighboring cells. The designation "approximate test

function" is given because the graph of this function looks like a characteristic (indicator) function with slanted sides extending into adjacent cells, whereas the test functions described in the derivation of the governing equation are exactly characteristic functions in space at time t^{n+1}. Examples of approximate test functions are illustrated in figure 7 for one direction. An approximate test function is determined by NSC, NSR, and NSL, the proximity of the transport subdomain boundary, and the active status of neighboring cells. Mass is not split across the transport boundary or into inactive cells.

We define local reference coordinates $\hat{x},\hat{y},\hat{z}$ centered around cell Ω_l with node indices *j,i,k* by

$$\hat{x} = \frac{x - x_j}{\Delta x_j} \tag{22}$$

and similarly for $\hat{y}$ and $\hat{z}$. For $\hat{x},\hat{y},\hat{z} \in \left(-\frac{1}{2},\frac{1}{2}\right)$, the corresponding point (*x, y, z*) is in cell Ω_l. For an interior cell on a uniform grid with all surrounding cells active, one approximate test function is

$$w_{jik}(\hat{x},\hat{y},\hat{z}) = f(\hat{x})\,g(\hat{y})\,h(\hat{z}) \tag{23}$$

where

$$f(\hat{x}) = \begin{cases} 0 & \hat{x} \le -\frac{1}{2} - \frac{1}{2NSC} \\ NSC\,\hat{x} + \frac{1}{2}(NSC+1) & -\frac{1}{2} - \frac{1}{2NSC} < \hat{x} < -\frac{1}{2} + \frac{1}{2NSC} \\ 1 & -\frac{1}{2} + \frac{1}{2NSC} \le \hat{x} \le \frac{1}{2} - \frac{1}{2NSC} \\ -NSC\,\hat{x} + \frac{1}{2}(NSC+1) & \frac{1}{2} - \frac{1}{2NSC} < \hat{x} < \frac{1}{2} + \frac{1}{2NSC} \\ 0 & \frac{1}{2} + \frac{1}{2NSC} \le \hat{x} \end{cases} \tag{24}$$

and similarly for *g* and *h*. This function, in one direction on a uniform grid, is shown graphically in figures 7a and 7b. Approximate test functions in each direction are multiplied together (see eq. 23) to get the test functions used to distribute the advected mass.

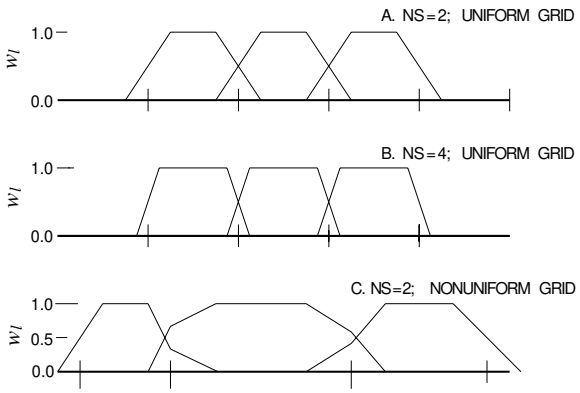


Figure 7. Examples of spatial distribution of approximate test functions (w_l) for selected one-dimensional cases. Vertical ticks represent cell boundaries.

In the general case of a possibly nonuniform grid, the single variable functions *f*, *g*, and *h* are given by:

$$f(\hat{x}) = \begin{cases} 0 & \hat{x} \le -\frac{1}{2} - \frac{1}{2NSC} \\ 1 + 2NSC\left(\frac{\Delta x_j}{\Delta x_{j-1} + \Delta x_j}\right)(1+\hat{x}) - \left(1 - \frac{\Delta x_j}{\Delta x_{j-1} + \Delta x_j}\right) - NSC\left(\frac{\Delta x_j}{\Delta x_{j-1} + \Delta x_j}\right) & -\frac{1}{2} - \frac{1}{2NSC} < \hat{x} < -\frac{1}{2} \\ 2NSC\left(1 - \left(\frac{\Delta x_j}{\Delta x_{j-1} + \Delta x_j}\right)\right)\hat{x} + \left(\frac{\Delta x_j}{\Delta x_{j-1} + \Delta x_j}\right) + NSC\left(1 - \left(\frac{\Delta x_j}{\Delta x_{j-1} + \Delta x_j}\right)\right) & -\frac{1}{2} \le \hat{x} < -\frac{1}{2} + \frac{1}{2NSC} \\ 1 & -\frac{1}{2} + \frac{1}{2NSC} \le \hat{x} \le \frac{1}{2} - \frac{1}{2NSC} \\ 2NSC\left(\left(\frac{\Delta x_j}{\Delta x_{j+1} + \Delta x_j}\right) - 1\right)\hat{x} + \left(\frac{\Delta x_j}{\Delta x_{j+1} + \Delta x_j}\right) + NSC\left(1 - \left(\frac{\Delta x_j}{\Delta x_{j+1} + \Delta x_j}\right)\right) & \frac{1}{2} - \frac{1}{2NSC} < \hat{x} < \frac{1}{2} \\ 1 + 2NSC\left(\frac{\Delta x_j}{\Delta x_{j+1} + \Delta x_j}\right)(1-\hat{x}) - \left(1 - \frac{\Delta x_j}{\Delta x_{j+1} + \Delta x_j}\right) - NSC\left(\frac{\Delta x_j}{\Delta x_{j+1} + \Delta x_j}\right) & \frac{1}{2} < \hat{x} < \frac{1}{2} + \frac{1}{2NSC} \\ 0 & \frac{1}{2} + \frac{1}{2NSC} \le \hat{x} \end{cases} \quad (25)$$

with g and h defined analogously, where for points outside of Ω_l, $\hat{x},\hat{y},\hat{z}$ must be defined according to the scale of the appropriate cell; for example, in the neighboring cell having coordinates j-1, we have $\hat{x} \in \left(-\frac{3}{2}, -\frac{1}{2}\right)$, $\hat{x} = -1 + \frac{x - x_{j-1}}{\Delta x_{j-1}}$. This function is shown graphically in figure 7c.

In practice, to evaluate all test functions at a given point in any Ω_l, use cell coordinates j,i,k and reference coordinates $\hat{x},\hat{y},\hat{z} \in \left(-\frac{1}{2}, \frac{1}{2}\right)$ for that point within that cell. Then eqs. 25 are used to evaluate f, g, h and eq. 23 to find $w_{j,i,k}$. If $\hat{x},\hat{y},\hat{z} > 0$, there are potentially seven more nonzero test functions, with values given by:

$$w_{j+1,i,k} = \left(1 - f(\hat{x})\right) g(\hat{y})\, h(\hat{z}) \quad (26a)$$

$$w_{j,i+1,k} = f(\hat{x})\left(1 - g(\hat{y})\right) h(\hat{z}) \quad (26b)$$

$$w_{j,i,k+1} = f(\hat{x})\, g(\hat{y})\left(1 - h(\hat{z})\right) \quad (26c)$$

$$w_{j+1,i+1,k} = \left(1 - f(\hat{x})\right)\left(1 - g(\hat{y})\right) h(\hat{z}) \quad (26d)$$

$$w_{j,i+1,k+1} = f(\hat{x})\left(1 - g(\hat{y})\right)\left(1 - h(\hat{z})\right) \quad (26e)$$

$$w_{j+1,i,k+1} = \left(1 - f(\hat{x})\right) g(\hat{y})\left(1 - h(\hat{z})\right) \quad (26f)$$

$$w_{j+1,i+1,k+1} = \left(1 - f(\hat{x})\right)\left(1 - g(\hat{y})\right)\left(1 - h(\hat{z})\right) \quad (26g)$$

Analogous expressions hold for reference coordinates with different signs. Test functions for neighboring cells are evaluated as in eq. 26, and $w_{l'} \equiv 0$ for all l' indexing cells not adjacent to Ω_l.

The sum of the values of the test functions at any point in the transport domain is one, thus conserving mass for the integral equation. This may be expressed as $\sum_l w_l(x) = 1$.

The test function component in the direction of the boundary extends from the center of a boundary cell to the boundary face with the value of one. Thus, there is no splitting of mass across the boundary.

There is no test function associated with an inactive cell. In a cell adjacent to an inactive cell, the value of the test function that would normally be assigned to the inactive cell is distributed proportionally to other test functions that are nonzero at that point. All test functions are zero in inactive cells.

Extreme variation in cell thickness among neighboring cells in a layer may adversely affect model results. In this case, linear interpolation of concentration in the

vertical direction, or approximate distribution of advected mass, may be inaccurate.

Source Integral

Source and sink integrals correspond to the last term in eq. 10, divided through by porosity as noted above. In general, *MOC3D* assumes that a source or sink is distributed uniformly over the finite-difference cell containing it.

For a source, a time step $\left(t^n, t^{n+1}\right)$ is discretized into *NT* uniform sub-time steps, $t_0 = t^n,\ t_{NT} = t^{n+1},\ t_m = t^n + \Delta t\, m/NT$ for *m* = 1,2,...,*NT*-1. Inflow of mass is integrated using the compound trapezoid rule. In effect, this means that the amount of mass associated with a sub-time step $\Delta t/NT$ is introduced at each of the times $t_1, t_2, ..., t_{NT-1}$, and half this amount is introduced at each of t_0 and t_{NT}. Starting from its designated time, each packet of mass is tracked forward to time t^{n+1} and accumulated in the same manner as non-source mass that began the time step already in the transport subdomain.

This accumulation for the last term in eq. 10 is done with the following integration for cell Ω_l. To account for all source mass that flows into Ω_l during the time step, all sources that intersect the support of the space-time test function associated with Ω_l are included. Note that the integration determines the source mass flowing into Ω_l, not the source mass originating in Ω_l. Multiple sources within the same cell are summed. This yields:

$$\iint\limits_{\text{supp } u_l \cap \text{supp } W} e^{-\lambda\left(t^{n+1}-t\right)} \sum C' \frac{W}{R_f} dt\, d\mathbf{x} \approx$$

$$\frac{1}{\varepsilon_{jik}} \sum_{\substack{ALL \\ SOURCE \\ CELLS}} \sum_{\substack{source \\ p=subcell \\ center}} \sum_{m=0}^{NT} e^{-\lambda\left(t^{n+1}-t_m\right)} \frac{\Delta T_m}{(NSC)\ (NSR)(NSL)} w_l\left(p^f\right) \sum_s C'_s \frac{Q_s}{R_f} \qquad (27)$$

where summation runs through all subcells of each source cell in the transport subgrid; p^f is the image of *p* under forward tracking to the new time level; t_m represents the time during the time step at which discretized source mass enters the system ($t_0 = t^n$, $t_{NT} = t^{n+1}$, $t_m = t^n + \Delta t\, m/NT$); and $\Delta T_m = \Delta t/NT$ or $\Delta T_m = \Delta t/2NT$ if *m* equals 0 or *NT*, *s* indexes all source terms within the source cell of interest, and Q_s is the volumetric flow rate associated with the fluid source having a concentration C'_s. Note that Q_s equals W_s multiplied by the volume of the source cell.

Sink Integral

Analytically, the domain of integration is the support of the space-time test function for a cell Ω_l intersected with any sink cells. To approximate, this term is only formulated if cell Ω_l contains a sink, and the sink concentration is assumed to be the average nodal concentration for the transport time step, with the exception of a sink related to evapotranspiration, where sink concentration is taken to be zero. Integration rules are a one point in space and a one point backward Euler in time. Multiple sinks within a sink cell are summed. The averaging of concentration results in this integral approximation contributing to both the left and right hand sides of the equation for sink cell Ω_l with coordinates *j,i,k*:

$$\iint\limits_{\text{supp } u_l \cap \text{ supp } W} e^{-\lambda\left(t^{n+1}-t\right)} \sum C' \frac{W}{\varepsilon R_f} dt\, d\mathbf{x}$$

$$\approx \Delta t \int_{\Omega_{\text{sink}}} \frac{1}{\varepsilon}\left(C_{average} \sum \frac{W}{R_f} + C_{\text{ET}} \frac{W_{\text{ET}}}{R_f} \right) d\mathbf{x}$$

$$= \frac{\Delta t}{\varepsilon_{jik} R_f}\left[\frac{\left(C^{n+1} + C^n\right)}{2} \sum Q_{\text{sink}} + C_{\text{ET}} Q_{\text{ET}} \right] \qquad (28)$$

where the subscript ET refers to evapotranspirative flux.

Inflow Boundary Integral

Inflow boundary integrals correspond to the next-to-last term in eq. 10, divided through by porosity. For an inflow boundary, as for a source, a single time step is discretized into a number of sub-time steps determined by parameter *NT*. The composite trapezoidal rule is applied in time. At each sub-time step, inflow mass is spatially discretized, tracked, and accumulated, just like mass already in the system at the start of the transport time step, but for the shorter interval. The only difference in the treatment of the inflow boundary from the treatment of the source is that only the two-dimensional boundary face is discretized, whereas for a source, the entire cell is discretized. For a cell Ω_l, the integration is performed over the intersection of the space-time test function for that cell and the transport subdomain boundary; that is, all mass entering through the boundary and advected to Ω_l during the time step is accumulated to the right-hand side of local equation *l*, yielding:

$$\iint_{\text{supp } u_l \cap \Gamma^{n+1}} e^{-\lambda\left(t^{n+1}-t\right)} C_{\text{inflow}} \frac{\mathbf{V}}{R_f} \cdot \mathbf{n}\, dt\, d\mathbf{s}$$

$$\approx \sum_{\substack{all \\ inflow \\ faces}} \sum_{\substack{face \\ p=subarea \\ center}} \sum_{m=0}^{NT} e^{-\lambda\left(t^{n+1}-t_m\right)} \frac{\Delta T_m}{ref\ area} w_l\left(p^f\right) C_{\text{inflow}} \frac{Q_{\text{inflow}}}{R_f} \qquad (29)$$

where *ref area* = NSC×NSR, NSC×NSL, or NSR×NSL, depending on plane of face; p^f is the image of *p* under forward tracking to the new time level; Q_{inflow} is the volumetric rate of inflow across the face; and t_m and ΔT_m are defined in eq. 27. Summation runs over each *p* on the transport subgrid boundary, with the approximate test function w_l used to select mass advected to Ω_l.

Outflow Integrals

Concentration is calculated at each outflow boundary face using cell parameters, velocity information from *MODFLOW*, and the amount of mass tracked across the cell boundary determined by *MOC3D*.

On the left-hand side of the system of boundary equations (eq. 14) is an integral approximated using a one point in space, one point backward Euler in time formulation. This time approximation eliminates the exponential factor, as follows:

$$\int_{(\partial\Omega)_{ll}} \int_{t^n}^{t^{n+1}} e^{-\lambda\left(t^{n+1}-t\right)} C_{\text{outflow}} \frac{\mathbf{V}}{R_f} \cdot \mathbf{n}\, dt\, d\mathbf{s}$$

$$\approx \Delta t \int_{(\partial\Omega)_{ll}} C_{\text{outflow}} \frac{\mathbf{V}}{R_f} \cdot \mathbf{n}\, d\mathbf{s}$$

$$= \Delta t \frac{Q_{\text{outflow}}}{R_f} C_{\text{outflow}} \qquad (30)$$

where *ll* is the index for boundary faces; and Q_{outflow} is determined using the outflow velocity calculated from *MODFLOW* output and cell parameters. The concentration on face *ll* is the unknown in the boundary equation.

The right-hand side boundary integrals are constructed from the mass contributions tracked across the boundary from interior cells, sources, and inflow boundaries during the transport time step. All mass associated with a tracked point that reaches the outflow boundary at any time during the time step is

considered to leave the transport subdomain. Test functions are evaluated to distribute mass among neighboring boundary outflow faces.

The three terms on the right-hand side of eq. 14 are approximated in a manner analogous to eqs. 21, 29, and 27, respectively, where now outflow boundary integrals, instead of cell integrals, are being considered. We define test functions W_{ll} associated with outflow boundary faces analogous to the test functions W_l for cells. Thus, $W_{ll}(p^f)$ will be nonzero if, tracking p^f forward from sub-time step t_m, p^f reaches the given outflow boundary face during time interval $[t_m, t^{n+1}]$. Spatially along the boundary, W_{ll} will have a profile analogous to those in figure 7, thus distributing mass to neighboring boundary faces. Substituting the right-hand side of eq. 30 for the left-hand side of eq. 14, approximating the right-hand side of eq. 14 as described above, and using a trapezoidal rule in time and a cell-midpoint rule in space, we obtain:

$$\Delta t \frac{Q_{\text{outflow}}}{R_f} C_{\text{outflow}} \approx e^{-\lambda \Delta t} \sum_{j,i,k} \sum_{\substack{\text{p=subcell} \\ \text{center}}} \frac{\Delta x_j \, \Delta y_i \, b_{j,i,k}}{(NSC)(NSR)(NSL)} w_l\left(p^f\right) C(p)$$

$$+ \sum_{\substack{all \\ inflow \\ faces}} \sum_{\substack{face \\ p=subarea \\ center}} \sum_{m=0}^{NT} e^{-\lambda\left(t^{n+1}-t_m\right)} \frac{\Delta T_m}{ref\ area} w_l\left(p^f\right) C_{\text{inflow}} \frac{Q_{\text{inflow}}}{R_f}$$

$$+ \frac{1}{\varepsilon_{jik}} \sum_{\substack{ALL \\ SOURCE \\ CELLS}} \sum_{\substack{source \\ p=subcell \\ center}} \sum_{m=0}^{NT} e^{-\lambda\left(t^{n+1}-t_m\right)} \frac{\Delta T_m}{(NSC)\ (NSR)(NSL)} w_l\left(p^f\right) \sum_s C_s' \frac{Q_s}{R_f} \qquad (31)$$

We thus have a system of equations represented by a diagonal matrix, to be solved for C_{outflow}.

Accuracy Criteria

An accuracy criterion incorporated in *MOC3D* constrains the distance that solute mass is advected during each transport time step. A restriction can be placed on the size of the time step to ensure that the number of grid cells a point moves in the *x*-, *y*-, or *z*-directions does not exceed some maximum. The simulator allows the user to specify this maximum (named *CELDIS* in the code and input instructions). This translates into a limitation on the transport time-step length. If the time step used to solve the flow equation exceeds the time limit, the flow time step will be subdivided into an appropriate number of equal-sized smaller time increments for solving transport.

For advective transport, a mesh density sufficient so that at least four grid nodes are represented across a solute front (or zone of relatively steep concentration gradient) is needed for good accuracy. Similarly, for advecting a peak concentration, the area of the peak should be represented across at least eight nodes of the grid for good accuracy. In such cases, testing suggests that a peak concentration value can be advected with a very small dissipation of the maximum concentration per time step for a variety of

Courant numbers. With insufficient mesh density, a peak will dissipate rapidly for an initial period of time during which it spreads out and oscillates; thereafter, the numerical decay slows and the oscillations do not worsen. A fine discretization of tracked mass (large NSC, NSR, NSL) reduces the rate of peak decay when modeling with many transport time steps (see sections "Special Problems" and "Input Parameter Values"). Regardless of the solution accuracy, global mass is conserved.

The accuracy of the dispersion calculation is governed in part by the accuracy of the central-difference approximations to the space derivatives, meaning a finer mesh will result in better accuracy. The implicit formulation for the solution of the dispersion equation is unconditionally stable. This allows for large time steps during the simulation. Because *ELLAM* solves for advection along characteristics, thus avoiding large values of the second time derivative of the solution at passage of a steep front, error in calculation of the time derivative may be expected to be small compared to a standard finite-difference solution to an advection-dispersion equation. Some dependence of the accuracy of the dispersion calculation on the size of the time step is retained, however. Note that stability does not imply accuracy; accuracy of the solution to the dispersion equation decreases as the time step size increases. On the other hand, modeling with many time steps in order to resolve dispersion to the desired accuracy could result in a loss of peak to numerical dispersion inherent in the treatment of advection, an effect that can be reduced by increasing NSC, NSR, and NSL.

One additional difficulty encountered with implicit temporal differencing results from the use of a symmetric spatial differencing for the cross-derivative terms of the dispersion tensor. This creates a potential for overshoot and undershoot in the calculated concentration solution, particularly when the velocity field is oblique to the axes of the grid. A remedy for excessive overshoot and undershoot is to refine the finite-difference mesh. This may, however, increase simulation times.

ELLAM can produce qualitatively good results in a small number of time steps, provided the NT value is sufficient to yield smooth distribution of mass along the inflow path. (See sections "Special Problems" and "Input Parameter Values".)

Mass Balance

As described by Konikow and others (1996), global mass-balance calculations are ordinarily performed to help check the numerical accuracy and precision of the solution. Modifications to the previously described mass-balance calculations have been implemented to assure consistency with the 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, eq. 66). The *ELLAM* code assumes that the concentration associated with a flux in or out of the system is the average nodal concentration during the time increment, $\left(C^n_{j,i,k} + C^{n+1}_{j,i,k}\right)/2$.

ELLAM conserves mass globally, regardless of the accuracy of the solution. Mass balance errors of less than 10^{-4} percent can generally be expected (this depends on the value of approximate zero, which is FORTRAN variable AZERO in the code, and solver tolerance, both of which are currently predefined in the code and cannot be specified in the input data).

Special Problems

Fronts too sharp for the given mesh density (grid spacing) may produce negative concentration values and/or numerical dispersion. For the mass in one cell to be positive, the mass in the adjacent cell to be zero, and concentration to vary linearly, the cell with zero mass may show a negative concentration. For non-integer Courant numbers, numerical dispersion results from the solution algorithm being insensitive to the exact location of advected mass in a destination cell. On a well-discretized front, these effects are minimal due to error cancellation. Thus, solving on a fine grid with few time steps may mitigate these difficulties.

Numerical dispersion may also result from tracking subdivisions of mass that are too coarse. The level of discretization of mass tracked and accumulated to the right-hand side vector is determined by parameters, NSC, NSR, and NSL. These parameters define the number of subcells in the column, row, and layer direction, respectively. To increase the resolution of mass tracking under advection, it may be desirable to increase the values of these parameters.

Parameter NT defines the number of sub-time steps per transport time increment. NT should be large enough so that all cells in the path of flow from the inflow boundary or source to the location of the front at the end of the time step receive incoming mass. This is to avoid artificial mass lumping. See section "Input Parameter Values."

To avoid non-physical accumulation of mass at an outflow boundary, the spatial NS parameter in the direction normal to the boundary must be such that 1/(2NS)<Courant. This is to ensure that at least some mass is calculated by the algorithm as reaching the boundary during a time step.

Extreme variation in cell thickness among neighboring cells in a layer may adversely affect model results. This is caused by the inherent geometric inconsistency in the vertical direction between adjacent cells that have different thicknesses. (Also see McDonald and Harbaugh, 1988, figure 9 and related discussion.)

Review of *ELLAM* Assumptions

The assumptions that have been incorporated into the *ELLAM* simulator are very similar to those for *MOC3D* Version 1 and Version 2. They are relevant to both grid design and model application. Efficient and accurate application of *ELLAM* 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).

Transport subgrid boundaries are assumed to be far enough from the plume that any errors in the treatment of the boundaries will not have a significant effect on the solution. The boundary condition is that the normal component of the concentration gradient on the boundary is zero, meaning there is no dispersive flux across the transport subdomain boundary.

Unlike the previous *MOC3D* explicit and implicit difference approximations, *ELLAM* does not require a uniform grid spacing within the transport subdomain. Likewise, there is no longer a formal restriction on variations in the product of porosity and thickness within the subdomain.

ELLAM does assume:

- Concentration at an outflow boundary face at the new time level is well approximated by the mass crossing the face during the

time step divided by the fluid volume across the face.

- Mass in or out of the transport subdomain during a time step via a source or sink cell is well approximated by the average nodal concentration during the time step times the fluid volume through the source or sink. Mass loss in flow through upstream sinks is negligible.
- Cell thicknesses are smoothly varying within a horizontal layer.

COMPUTER PROGRAM

ELLAM is implemented as a package for *MODFLOW*. *ELLAM* 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 mass-bearing volume a distance and direction with time to represent advection. An integral formulation of conservation of mass is applied, yielding a dispersion equation including boundary fluxes, fluid sources, and decay.

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.

A separate executable version of *MODFLOW*, which is adapted to link with and use the *ELLAM* package, must first be created to run the simulations. *MOC3D* is written in standard FORTRAN-77, and it has been successfully compiled and executed on multiple platforms, including Pentium-based personal computers, Macintosh personal computers, and Data General, Sun, 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 these variables has not been necessary.

Implementing *ELLAM* requires the use of a separate "name" file that contains file names, similar to the one used in *MODFLOW*. The principal *ELLAM* input data (such as subgrid dimensions, hydraulic properties, and particle information) are read from the main *ELLAM* 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. *ELLAM* input requirements differ from those of *MOC3D* in that NSCEXP, NSREXP, NSLEXP, and NTEXP values must be provided, whereas parameter values related to fluid particles are omitted.

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.

Program Segments

MOC3D 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. Discussion related to table 4 in that report (p. 37) is to some extent irrelevant, inasmuch as it pertains to particle manipulations. Several existing *MOC3D* subroutines were modified. *ELLAM* routines have been added to the main *MOC3D* transport time loop. Table 1 provides a list of the primary subroutines and their descriptions. Table 2 contains secondary subroutines and the calling tree. The ELMOVE routine in *ELLAM* uses a linear velocity interpolation. Output routines from earlier versions of the code are retained. A flow chart of the program segments controlling the transport calculations for *ELLAM* is shown in figure 8 (also see figures 12 and 13 of Konikow and others, 1996).

Dispersion coefficients are calculated at cell faces. To improve efficiency, 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." Here, the *ELLAM* version of subroutine DSP6FM has been modified to calculate the distance between cell nodes in the column or row direction using the newly implemented variable grid dimensions, and to use an appropriate grid dimension factor for the *ELLAM* integral formulation of the problem. For example, the dispersion equation coefficient for the j+1/2,i,k face in the column direction is

$$\frac{2(\varepsilon b D_{xx})_{j+1/2,i,k}}{\Delta x_j + \Delta x_{j+1}} \qquad (32)$$

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

Table 1. *ELLAM* transport loop

Subroutine	Description
ELLBDY	Create boundary arrays; Track inflow mass, and accumulate to RHS storage or outflow integral
BDYINT	Track boundary layer mass, and accumulate to RHS storage or outflow integral; Calculate LHS storage coefficient, and save value if inflow or outflow face (1st transport time step)
ELLLHI	Track interior mass, and accumulate to RHS storage or outflow integral; Calculate LHS storage coefficient (1st transport time step)
ELLDIS	Build LHS dispersion matrix (1st transport time step)
ELLSRC	Sink: accumulate to dispersion matrix (1st transport time step) and RHS; Source: track source mass; accumulate to RHS storage or outflow integral
ELIUPD	Move inflow boundary concentrations to solution vector
ELLOUT	Outflow and inflow processing: solve outflow boundary equations; accumulate inflow and outflow boundary mass to RHS; accumulate boundary mass for mass balance
CONVERT	Convert storage plus dispersion matrix into SLAP column format (1st transport time step)
SMOC5I	Convert storage matrix into SLAP column format (1st transport time step); Calculate initial mass in system
ELLSLV	Solve interior equations
ELNUPD	Move no-flow boundary concentrations to solution vector
SMOC6BE	Mass-balance calculations
SMOC6C and SOBS5O	Print results

Table 2. Calling tree for *ELLAM* code, showing hierarchy of secondary subroutines for transport calculations. Subroutine to the left calls subroutine(s) to the right. Subroutines in column 1 are called from the main transport loop (subroutine MOC6MVOT).

ELLBDY	BDYENT	BDYRHS	ELMOVE EVALTF BNDYTF	EMOVTIM EVALTF
BDYINT	BDYCON	BUILDNEI BDYCR BDYCL	 BINTERP ELMOVE EVALTF BNDYTF DISTRIB EDGWGHT	 EDGWGHT EMOVTIM EVALTF
ELLLHI	CLAYER BLAYER ELLRHI	CROW HFACED VFACED EDGCOR ELLCR	 CINTERP ELMOVE EVALTF BNDYTF	 EMOVTIM EVALTF
ELLDIS				
ELLSRC	BDYRHS	ELMOVE EVALTF BNDYTF	EMOVTIM EVALTF	
ELIUPD				
ELLOUT				
CONVERT	GET_IDISP			
SMOC5I	GET_IDISP SMOC5A			
ELLSLV	SSLUGM	solver routines		
ELNUPD				
SMOC6BE	SMOC5A			
SMOC6C SOBS5O	print routines			

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 ELLSRC subroutine.

The *ELLAM* code includes a preconditioned generalized minimum residual method (GMRES) sparse iterative solver with an incomplete lower-upper (ILU) decomposition preconditioner to solve the non-symmetric system of equations. The solver routines are code from the SLATEC Common Mathematical Library, available through the NETLIB public domain repository; see http://www.netlib.org/slatec/index.html. Users interested in seeing more details about the numerical solvers should also examine the FORTRAN source code listing for *MOC3D* and its *ELLAM* subroutines, as they are very clearly documented with explanatory "comment" lines throughout the code.

Some mass-balance calculations have been reordered compared to the original *MOC3D* sequence. The initial mass calculation is called the during the first transport time step from within the transport loop, after the matrix of storage coefficients has been created.

Mass associated with inflow and outflow boundary faces is accumulated for mass-balance reporting each time step. This mass is calculated using integration weights associated with boundary nodes and boundary cell porosities, along with current boundary concentration values. Total amounts are added to interior and no-flow boundary mass calculated from the storage coefficient matrix and current concentrations.

Similarly, decayed mass from inflow faces and sources is accumulated for mass-balance reporting. This total, and decay of mass stored at the beginning of the time step, are both reported as contributions to mass flux out of the transport domain.

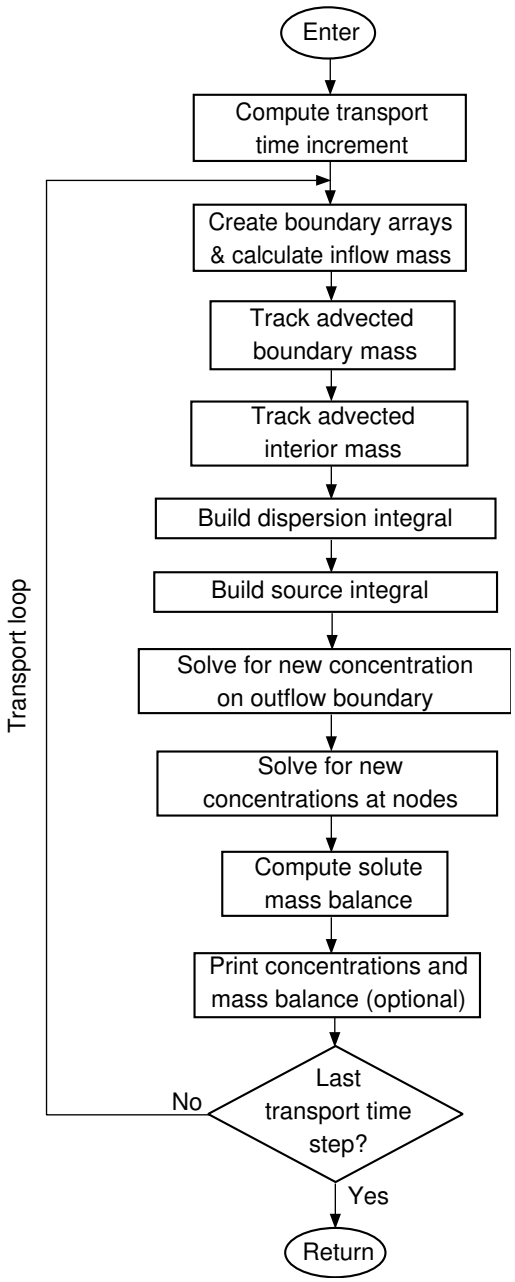


Figure 8. Simplified flow chart for the transport loop of the *ELLAM* calculation process.

Guidance on Input Parameter Values

Discretization parameters NSC, NSR, NSL, and NT must all be powers of two. In each case, the input parameter specified by the user is the exponent: NSCEXP, NSREXP, NSLEXP, and NTEXP, respectively.

In general, use NSC = NSR = NSL = 4 (and therefore NSCEXP = NSREXP = NSLEXP = 2) except if modeling a one- or

two-dimensional problem. Here, a value of 4 is only needed in the dimension(s) of the problem, with NS values of 2 adequate in the missing direction(s). NS values greater than four may be useful when modeling with a complicated velocity field, or with numerous transport time steps, or to improve accuracy near a boundary. Computational efficiency is strongly related to the values of the NS parameters, but the impact of changing NS values is highly problem dependent.

The number of discrete mass-bearing volumes entering the transport subdomain during a time step is NT+1. This number must be large enough so that each grid cell, from the one at the first boundary to the one where mass entering at the beginning of that time step is advected, can receive a portion of the discretized inflow mass. If the solution shows mass becoming distributed down the flow path in clumps (this is illustrated schematically in figure 9 for a case in which two sub-time steps are used), increase NTEXP (thus increasing NT). Increasing NT will smooth the front to the point of ensuring an even distribution of mass among destination cells. NT only comes into effect for problems with concentrations entering via fluid sources or inflow across boundaries of the transport subgrid.

For example, consider the case of one-dimensional flow discussed below as the first test problem. Here, $V = 0.1$ cm/s and each cell has a width of 0.1 cm. NTEXP = 1 (thus NT = 2) will result in a discretization of inflow mass such that one fourth enters the system at the beginning of the time step, one half in the middle, and one fourth at the end. Thus, the initial quarter of the mass will be advected the farthest and will be distributed among neighboring cells as mandated by the test functions determined by NSC. The mass entering at the end of the time step remains in the the first grid cell. The point representing half of the mass will be advected a number of cells equal to the magnitude of (TIMV/2). For a time step (TIMV) of 5 s or longer, TIMV/2 > 2.5, so the point will move past the middle of the third cell. No advected mass will end up in the second grid cell, for any value of NSC. Increasing NTEXP will eliminate this problem.

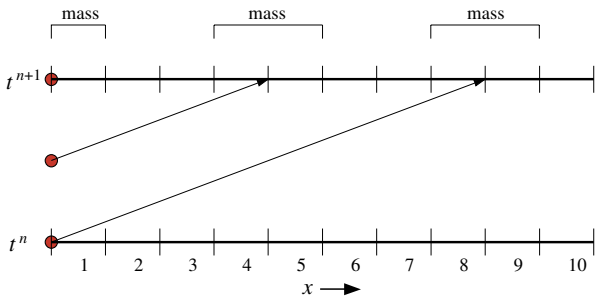


Figure 9. Schematic representation of tracking mass from an inflow boundary. For a Courant number of 8 and NT = 2, not every cell along the inflow path receives mass.

MODEL TESTING AND EVALUATION

The *ELLAM* 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) and *MOC3D* Version 2 by Kipp and others (1998). This suite includes results generated by analytical solutions and by other numerical models. It spans 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.

One-Dimensional Flow

The first test case evaluates *ELLAM* 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). Note that when modeling using *MOC3D*, total flux boundary conditions are approximated using only advective flux.

The length of the transport system is 12 cm; other parameters are summarized in table 3. The solute-transport equation was solved using *ELLAM* on a 120-cell subgrid to assure a constant velocity within the transport domain. This use of the subgrid eliminates the local influence of a zero-velocity condition at the upstream no-flow boundary on the calculated velocities within the grid and allows an accurate match to the boundary conditions and velocity assumptions for the analytical solution.

In *MOC3D* Version 1, the explicit formulation of the dispersion coefficient imposed a limiting time step to meet the stability criteria, and 2401 time increments were required to solve the transport equation. The implicit solver of *MOC3D* Version 2, however, required only 241 time increments. *ELLAM* results for CELDIS = 0.5 (241 time steps), NSC = 4, NSR = NSL = 2, NT = 128 are essentially identical to the analytical results (these are the base-case parameters listed in table 3). Therefore, these solutions are not plotted. Instead, results are plotted for substantially fewer time increments (larger values of CELDIS). (Note that table 3 lists the "EXP" input parameters that yield the values for the four numerical *ELLAM* parameters used in the simulation: NSC, NSR, NSL, and NT.) As discussed below, we also evaluated several variants of this problem, in which we varied dispersivity, decay, and retardation parameters; in all cases the base-case numerical parameters yielded results that were indistinguishable from the analytical results.

Table 3. Base-case parameters used in *ELLAM* 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
CELDIS	0.5
NSCEXP	2
NSREXP	1
NSLEXP	1
NTEXP	7

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 α_L = 1.0 and 0.1 cm, respectively. Breakthrough curves showing concentration changes over time at different distances from the boundary, as calculated using both analytical and numerical solutions, are shown.

Figure 10 shows the analytical solution and two *ELLAM* solutions for the low-dispersivity case (α_L = 0.1 cm). For CELDIS = 1, which required 121 transport time increments, there is a very close match between the numerical and analytical solutions, using NSC = 32, NSR = NSL = 2, and NT = 128. To improve clarity in showing the results for this case in figure 10, only every fourth data point is shown, except for times less than 10 seconds at x = 0.05, where every point is plotted. The efficiency of the numerical solution can be improved by about a factor of four by setting NSC = 4; the results are very similar, so are not plotted, although the concentration is just slightly low at the first grid cell. The simulation took 205 seconds when NSC = 32, but only 119 seconds when NSC = 4 (simulations were executed on a Data General Unix workstation). For CELDIS = 10.1 (12 time increments), using NSC = 4, NSR = NSL = 2, and NT = 128, concentrations at early times and short distances are somewhat low, but elsewhere the results look excellent. Thus, there is an overall good agreement with the analytical results, as well as with the previously published *MOC3D* results that used 20 times as many time increments. For comparison, the simulation using CELDIS = 10.1 took only 9 seconds.

Concentrations for the high dispersion case are plotted in figure 11. For CELDIS = 1 (121 transport time increments), the results are a very close match to those obtained using the analytical solution, and ELLAM yields a smooth solution without the oscillations produced by *MOC3D* at short distances. For CELDIS = 10.1 (12 time increments and execution time about one tenth of that for CELDIS = 1), small discrepancies are apparent near both boundaries, in that calculated concentrations are too low near the inflow boundary and too high near the outflow boundary.

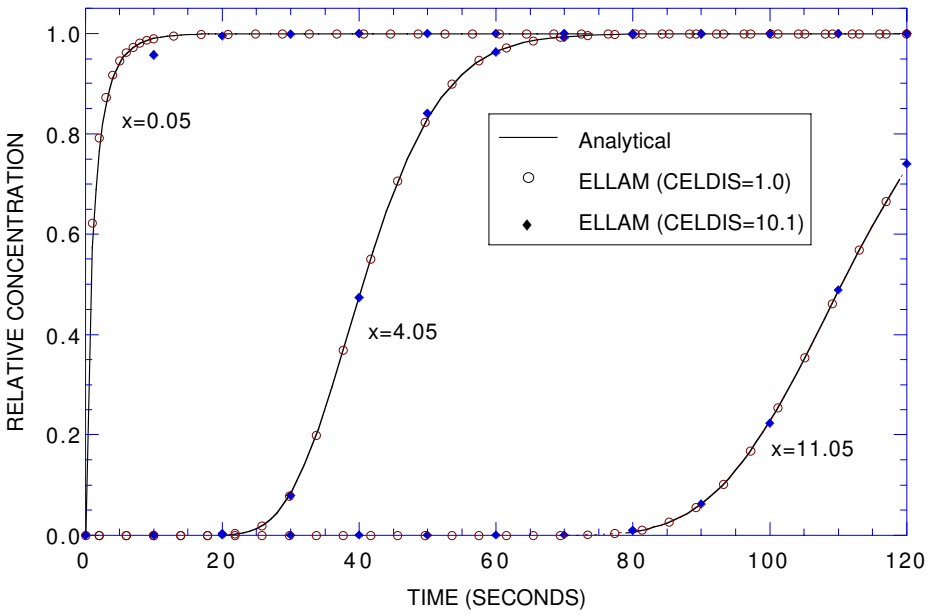


Figure 10. Numerical and analytical solutions at three different locations for solute transport in a one-dimensional, steady flow field. Parameter values for the base case are listed in table 3.

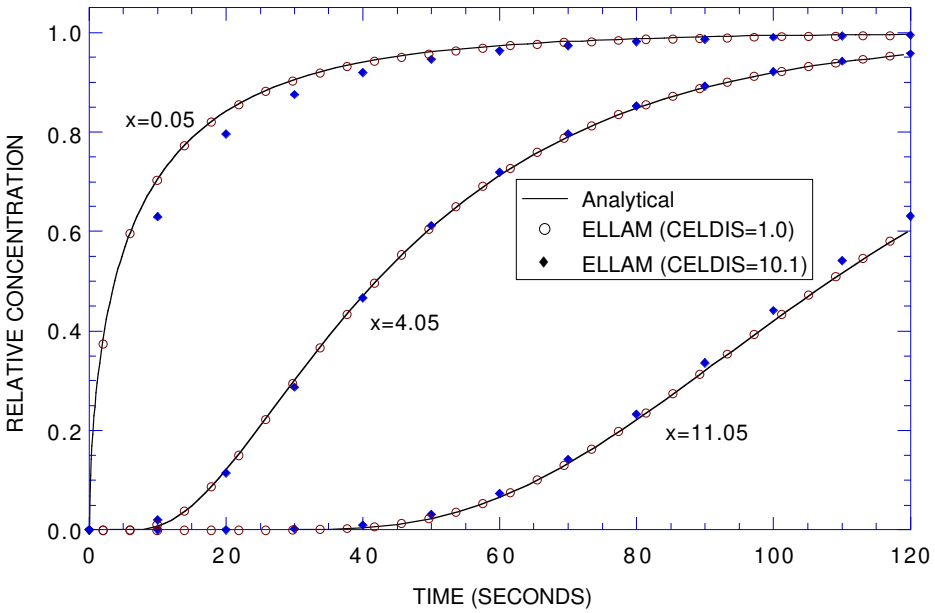


Figure 11. Numerical and analytical solutions for the case of increased dispersivity (α_L = 1.0 cm, D_{xx} = 0.1 cm^2/s, and other parameters as defined in table 3).

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 *ELLAM* formulation yields results comparable to those just described, as seen in figure 12 for CELDIS = 10.1 and CELDIS = 1 in the nonreactive case. To test the limits of the *ELLAM* method, we also solved this problem using CELDIS = 61 (2 time increments), NSC = 4, NSR = NSL = 2, and NT = 128. Although these numerical parameters yield too few time increments to even expect an accurate or precise match to the breakthrough curves, figure 12 indicates that even in this extreme case, a qualitatively good match for most of the breakthrough was calculated, except notably near the outflow face. Although such large values of CELDIS are not recommended, the results for CELDIS = 61 demonstrate the apparent robustness of the method.

The accuracy of the numerical method for problems in which decay is occurring was evaluated by specifying the decay rate as λ = 0.01 s^{-1} for the same low-dispersion, no sorption, problem as defined for figure 10. The results for CELDIS = 1, NSC = 32, NSR = NSL = 2, and NT = 128 are presented in figure 13, which shows excellent agreement between the analytical and numerical solutions. For clarity, only every fourth data point of the numerical solution is shown. As in the case of no decay, NSC = 4 (not plotted) produces a slightly low concentration at short distance.

In all cases described above, the mass-balance error was less than 0.001 percent. In contrast, the mass-balance errors for these problems using the explicit and implicit versions of the method-of-characteristics code yielded mass-balance errors of up to a few percent in some cases. *ELLAM* is mass conservative whereas *MOC* and *MOCIMP* are not. Also, as illustrated by the results for CELDIS = 61 in figure 12, an accurate mass balance does not prove that you have an accurate solution.

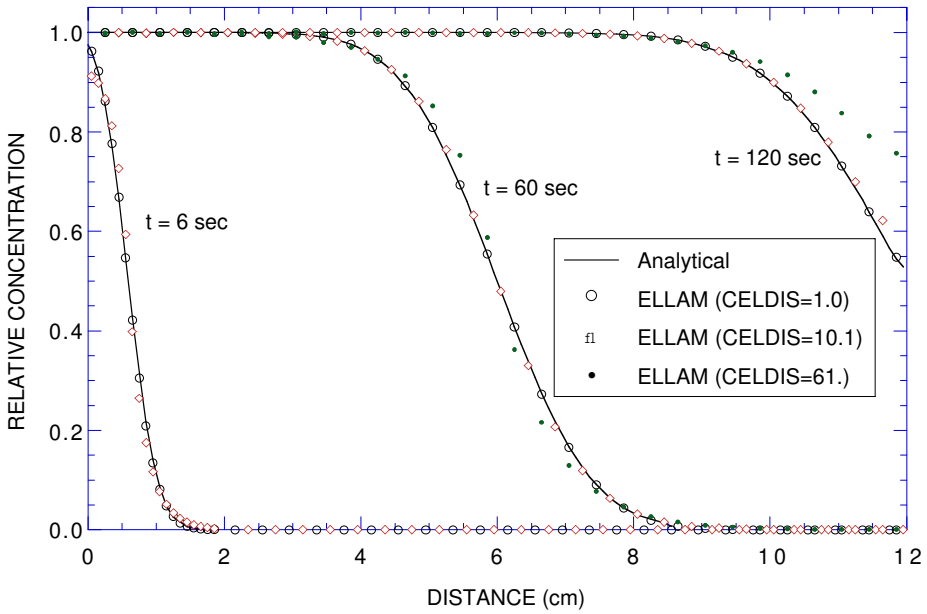


Figure 12. Numerical and analytical solutions for three different times for same one-dimensional, steady flow, solute-transport problem shown in figure 10.

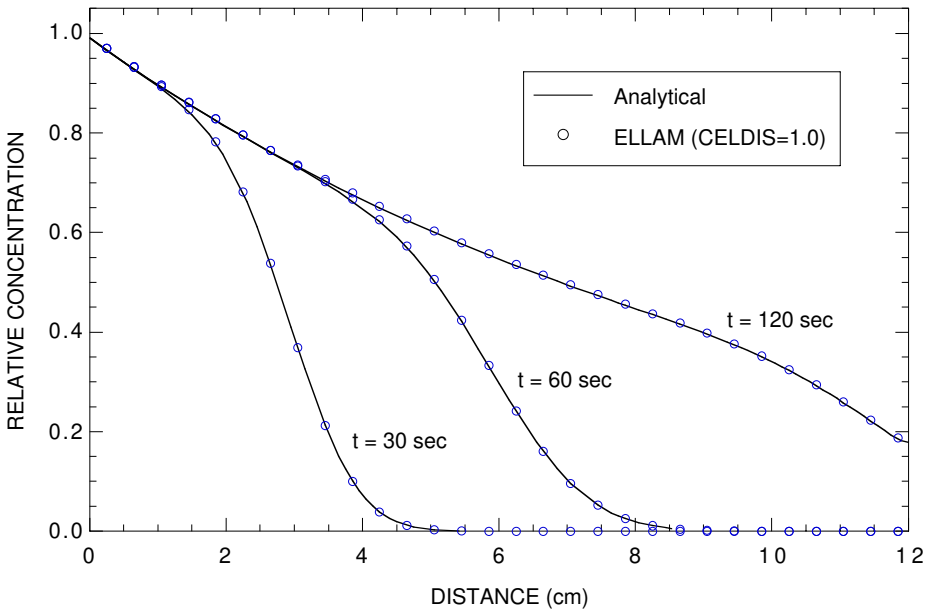


Figure 13. Numerical and analytical solutions for three different times for case in which solute is subject to decay at rate of $\lambda = 0.01\ s^{-1}$.

The sensitivity of the results to variations in the value of NT was evaluated for the case of CELDIS = 1.0, shown in figure 10. Values of NTEXP were varied from 2 to 9 (figure 10 represents NTEXP = 7). This parameter only had a small effect on the accuracy of results at the node closest to the source; there were no discernible differences

elsewhere. It also had only a minor impact on the efficiency of the solution. At the first node, the breakthrough curves were essentially identical for values of NTEXP ranging from 5 to 9, and the running time increased by less than 13 percent (from 203 s to 229 s). However, decreasing the value of NTEXP from 5 to 2 caused increasingly larger (but still minor) deviations from the analytical solution, while decreasing the running time only by 1 s (from 203 s to 202 s). For NTEXP = 2, the solution was stable but too low by about 2 percent at most times. For this case, it appears that a value of NTEXP = 5 or 6 would be optimal, but the results were relatively insensitive to variations in NTEXP.

Uniform Flow, Three-Dimensional Transport

To evaluate and test *ELLAM* for three-dimensional cases, we compared numerical results with those of the analytical 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 4. This case also represents a test of the ability of the algorithm to represent the effects of a solute source at a specified flux boundary condition.

The results of *ELLAM* are compared graphically in figure 14 with those of the analytical solution for the *x-y* plane passing through the point source. Figure 14a shows the concentrations in this plane at t = 400 days as calculated using the analytical solution. Also shown, in figures 14b-d, are the *ELLAM* solutions using CELDIS = 7 (two transport time increments), NSC = NSR = NSL = 4, and NT = 16 (figure 14b); CELDIS = 1 (14 time increments), NSC = NSR = NSL = 4, and NT = 4 (figure 14c); and CELDIS = 0.1 (134 time increments), NSC = NSL = 4, NSR = 8, and NT = 16 (figure 14d).

Table 4. Base-case parameters used in *ELLAM* 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_y	0.1 m/day
$V_x = V_z$	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	Column = 1, Row = 8, Layer = 1
Number of rows	30
Number of columns	12
Number of layers	40
DELR (Δx)	0.5 m
DELC (Δy)	3.0 m
Layer thickness (Δz)	0.05 m
CELDIS	1.0
NSCEXP	2
NSREXP	2
NSLEXP	2
NTEXP	2

As noted in previous *MOC3D* reports, a slightly greater spreading is evident in the numerical model results than in the analytical solution, both upstream as well as downstream of the source. Part of this difference, however, is explained by the fact that the numerical source is applied over a finite area in the horizontal plane of the 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 *ELLAM* results using two transport time increments (figure 14b) indicate that more time steps are needed in order to accurately simulate dispersion. The *ELLAM* results for 14 time steps (figure 14c) accurately characterize the dispersive flux without the spreading upstream from the source that is produced by *MOC3D*. The *ELLAM* results for 134 time steps (figure 14d) yield even less spreading upstream of the source, but do exhibit numerical oscillations produced because the concentration gradient is too steep relative to the grid spacing.

Konikow and others (1996) also present comparisons for this case for vertical planes parallel and perpendicular to the flow direction. These same comparisons between the analytical and *ELLAM* results are as close as between figures 14a and 14c, and are not reproduced here.

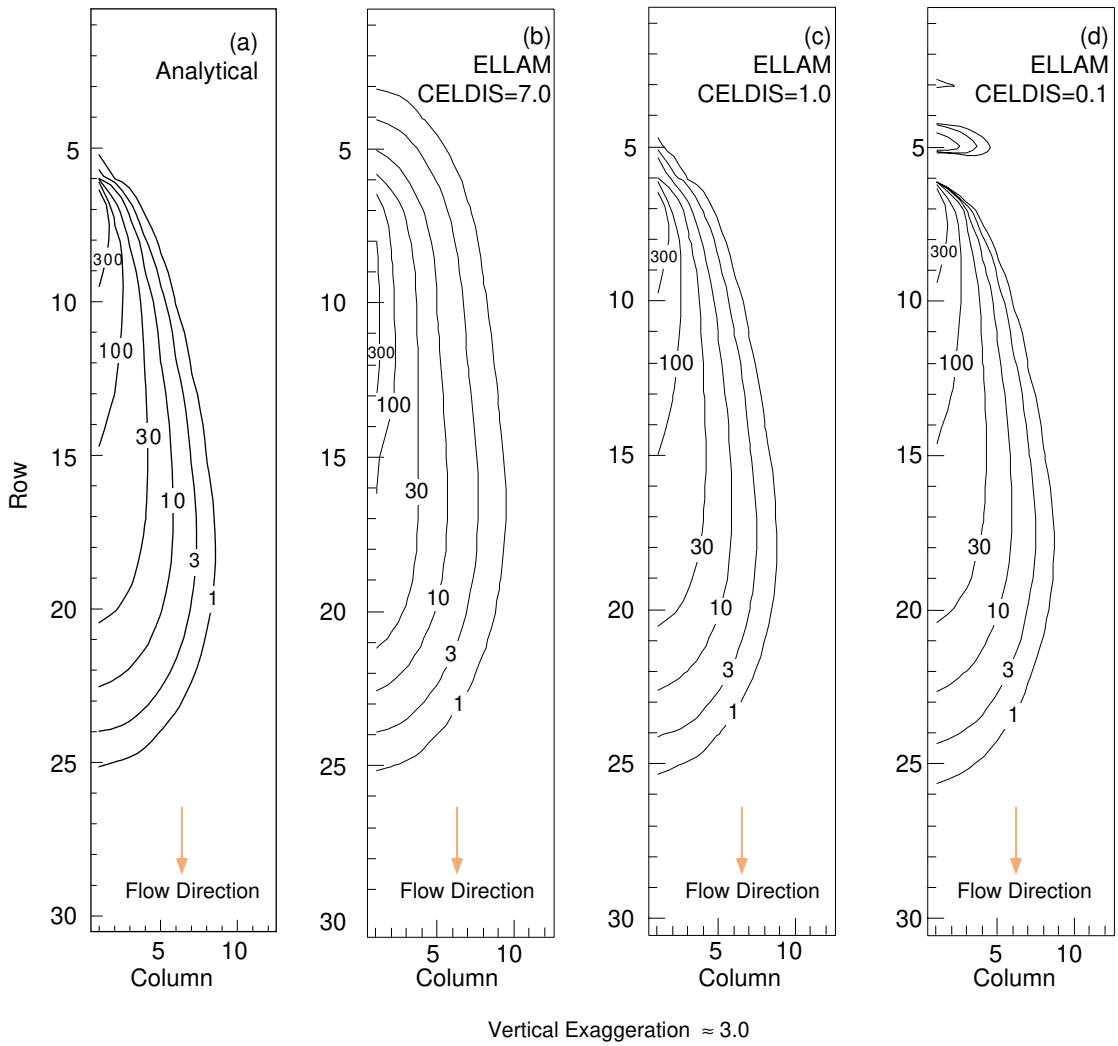


Figure 14. Concentration contours for (a) analytical and (b-d) *ELLAM* numerical solutions in the horizontal plane containing the solute source (layer 1) for three-dimensional solute transport in a uniform steady flow field at t = 400 days. Parameters are defined in table 4.

Two-Dimensional Radial Flow

A radial-flow transport problem was used to compare the *ELLAM* solution to the analytical solution given by Hsieh (1986) for a finite-radius injection well in an infinite aquifer. The problem involves flow from a single injection well; the velocities vary in space and vary inversely with the distance from the injection well.

The parameters for the problem are summarized in table 5 and the analytical solution and other details about this test case are presented by Konikow and others (1996, p. 49-50) and Kipp and others (1998, p. 21-22). The problem was modeled using 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).

Table 5. Parameters used in *ELLAM* simulation of two-dimensional, steady-state, radial flow case, showing range of values tested for selected numerical parameters

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
CELDIS	0.25-75.0
NSCEXP	2-3
NSREXP	2-3
NSLEXP	1
NTEXP	2-4

In figure 15, *ELLAM* solutions using 2, 29, and 563 transport time increments (corresponding to CELDIS values of 75, 5, and 0.25, respectively) are shown along with the analytical solution. Each run represents the qualitative features of the analytical solution (figure 15a), with the exception of the high concentration contour of the two-time-step (CELDIS = 75) run, which manifests an articulated rather than smooth shape. The high concentration contour is often the most difficult for *ELLAM* to portray accurately.

The two-time-step run shown in figure 15b uses CELDIS = 75, NSC = NSR = 4, NSL = 2, NT = 16. The high concentration contour has nonphysical oscillations. Increasing NS and NT values (not shown) smooth that one contour noticeably, but inadequately.

The 29-time-step run shown in figure 15c uses CELDIS = 5, NSC = NSR = 4, NSL = 2, NT = 4. It is a very close approximation to the analytical solution, although the high concentration contour does not have a constant radius.

Figures 15d and e show the results of two runs using 563 time steps. Parameter values for these two cases are CELDIS = 0.25, NSC = NSR = 4, NSL = 2, NT = 4, and CELDIS = 0.25, NSC = NSR = 8, NSL = 2, NT = 4. The improvement obtained using higher NS values when modeling with numerous time steps is illustrated here. The former values yield a set of contours not at constant radii, whereas the latter, using NSC = NSR = 8, produce a very close match to the analytical solution.

For comparison, the implicit solutions of *MOC3D* match the analytical solution almost exactly and agree very closely with the explicit dispersive transport solution of *MOC3D* (see Konikow and others, 1996, figure 29). The explicit solution required 596 time increments and used 750 s of cpu time, and the implicit solutions required 282 time

increments and used 445 s of cpu time. The *ELLAM* solutions represented in figures 15c-e required 138 s, 1,990 s, and 5,780 s, respectively, of cpu time (all times represent runs on a Unix workstation).

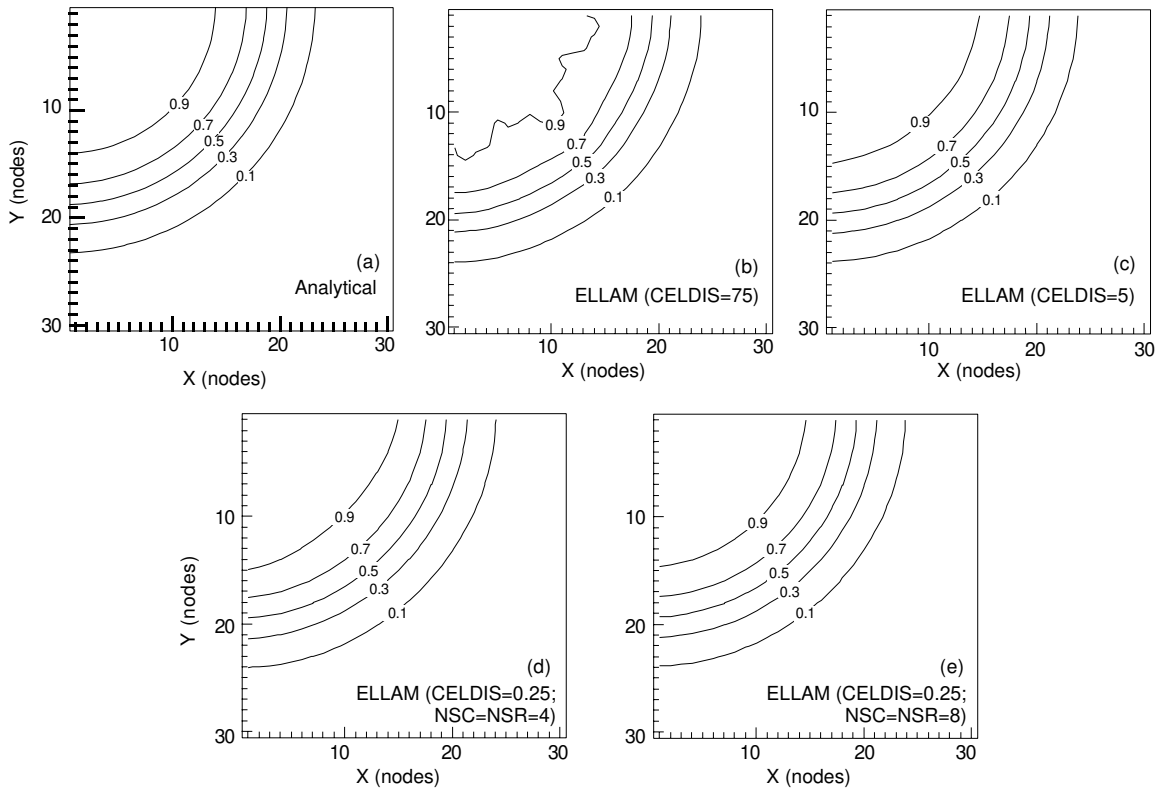


Figure 15. Contours of relative concentrations calculated using (a) analytical and (b-e) numerical *ELLAM* models for solute transport in a steady radial flow field. Source concentration is 1.0 and source is located in cell (1,1). Grid spacing is 10.0 m. Parameters are defined in table 5.

Point Initial Condition in Uniform Flow

A problem involving three-dimensional solute transport from an instantaneous point source, or Dirac initial condition, in a uniform flow field was used as another test problem for *MOC3D*. 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 grid relative to the flow. The assumptions and parameters for this test case are summarized in table 6 and are described in more detail by Konikow and others (1996).

Table 6. Parameters used in *ELLAM* 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	1.0275 m/day
$V_y = V_z$	0.0 m/day*
Initial concentration at source	1×10^6
Source location in transport grid	*Column* = 11, *Row* = 36, *Layer* = 4
Number of rows	72
Number of columns	72
Number of layers	24
DELR (Δx)	3.33 m
DELC (Δy)	3.33 m
Layer thickness ($b=\Delta z$)	10.0 m
CELDIS	5.0
NSCEXP	2
NSREXP	2
NSLEXP	2
NTEXP	1

* For flow at 45 degrees to *x*- and *y*-axes, V_y = 1.0275 m/day

We specified boundary conditions for the test case of flow in the *x*-direction such that V_x = 1.0275 m/d, and $V_y = V_z$ = 0.0 m/d. For flow at 45 degrees to *x* and *y*, we specified boundary conditions such that $V_x = V_y$ = 1.0275 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 higher in the latter case; therefore, there will be more dispersion in that problem during an equal time interval.

The results for both the analytical and numerical solutions for the case in which flow occurs in the *x*-direction are shown in figure 16, where values of CELDIS = 5 (yielding six transport time increments), NSC = NSR = NSL = 4, and NT = 2 were used. These results represent the concentrations in the plane of the initial source of solute. The *ELLAM* transport algorithm gives results (figure 16b) for a 72 by 72 grid that are close to those of the analytical solution (figure 16a). The numerical results, however, do show some slight spreading (or numerical dispersion) relative to the analytical solution in both the transverse and longitudinal directions. Increasing the number of time increments does not completely eliminate the spreading and causes some loss of peak concentrations, even with increased NS values. In contrast to the previous *MOC3D* solutions, *ELLAM* results retain the symmetry of the analytical solution. Part of the discrepancy is attributable to the need in *ELLAM* to use four grid points to discretize a front. This precludes the possibility of modeling with high accuracy the migration of an instantaneous point source placed in a single grid cell. Therefore, we modified this test problem for *ELLAM* by using a dispersed solute mass as an initial condition. The initial condition for this *ELLAM* test is the analytical solution to the original point source problem at *t* = 90 days, and the *ELLAM* solution is evaluated against the Wexler analytical solution later in time at *t* = 130 days. These results are presented in figure 17, where it can be seen that the analytical solution (figure 17a) and the numerical solution (figure 17b) are very similar. The *ELLAM* solution, however, still clearly exhibits some numerical dispersion, which is most evident at the lower concentrations.

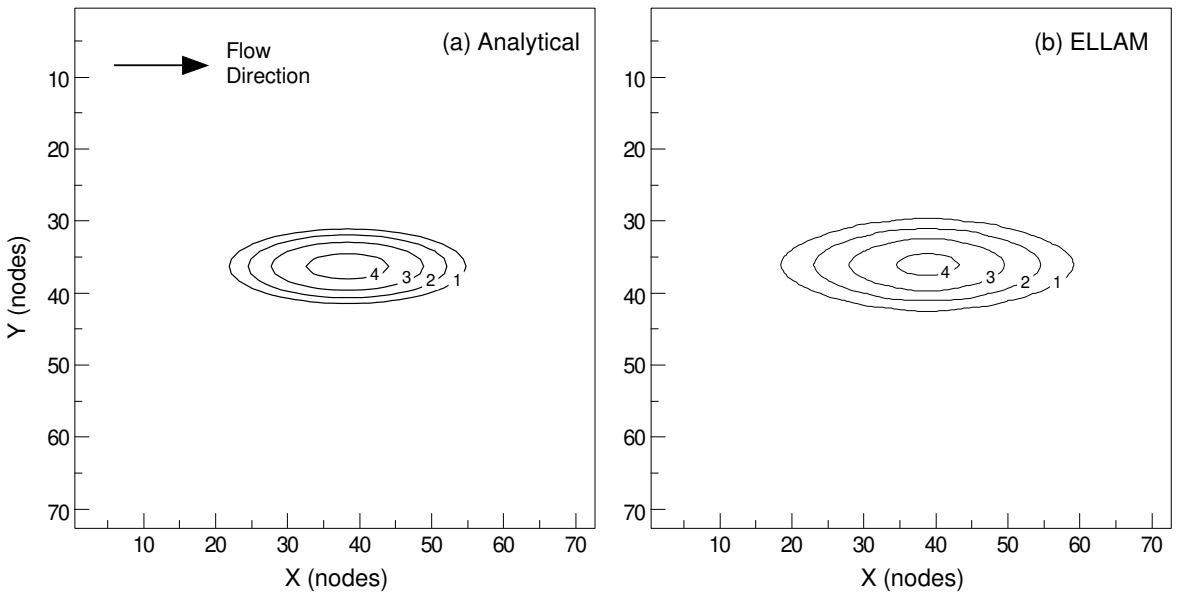


Figure 16. Concentration contours for (a) analytical and (b) numerical solutions for transport of a point initial condition in uniform flow in the *x*-direction at *t* = 90 days. The *z*-component of flow is zero, but there is dispersion in all three directions. Contour values are the log of the concentrations.

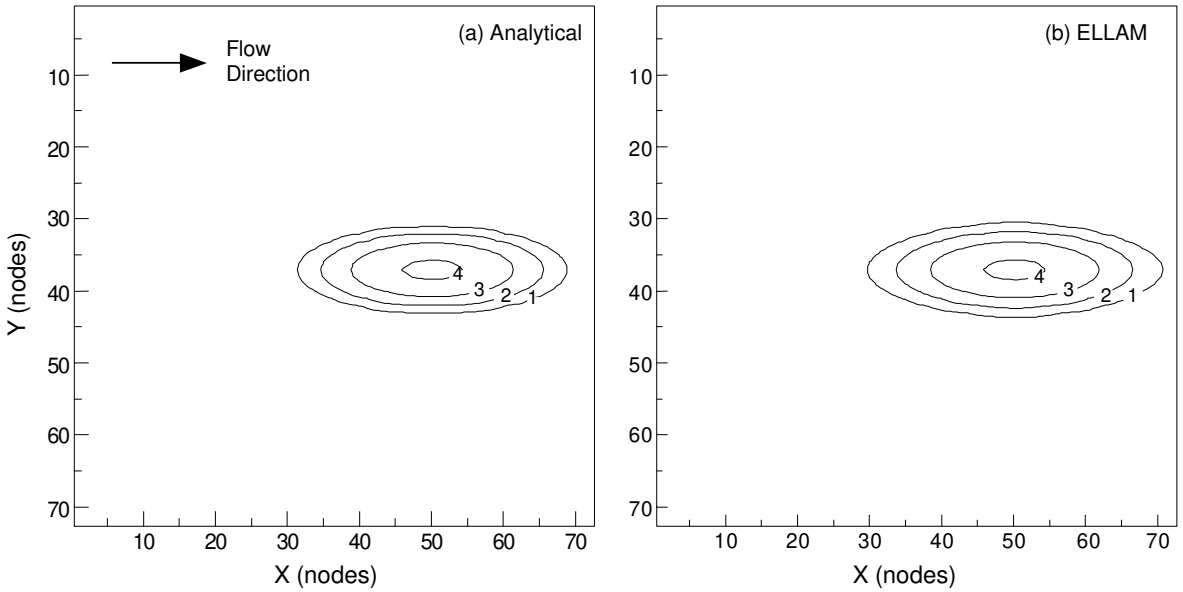


Figure 17. Concentration contours for (a) analytical and (b) *ELLAM* numerical solutions for transport of a dispersed-point initial condition in uniform flow in the *x*-direction at *t* = 130 days. The *y*- and *z*-components of flow are zero, but dispersion occurs in all three directions. Contour values are the log of the concentrations.

The results of the test problem for flow at 45 degrees to the grid are shown in figure 18, again using a 72×72×24 grid. The analytical solution for *t* = 130 days, which provides the basis for the evaluation, is shown in figure 18a. As was done for the previous analysis shown in figure 17, the *ELLAM* solution in this case also used the analytical

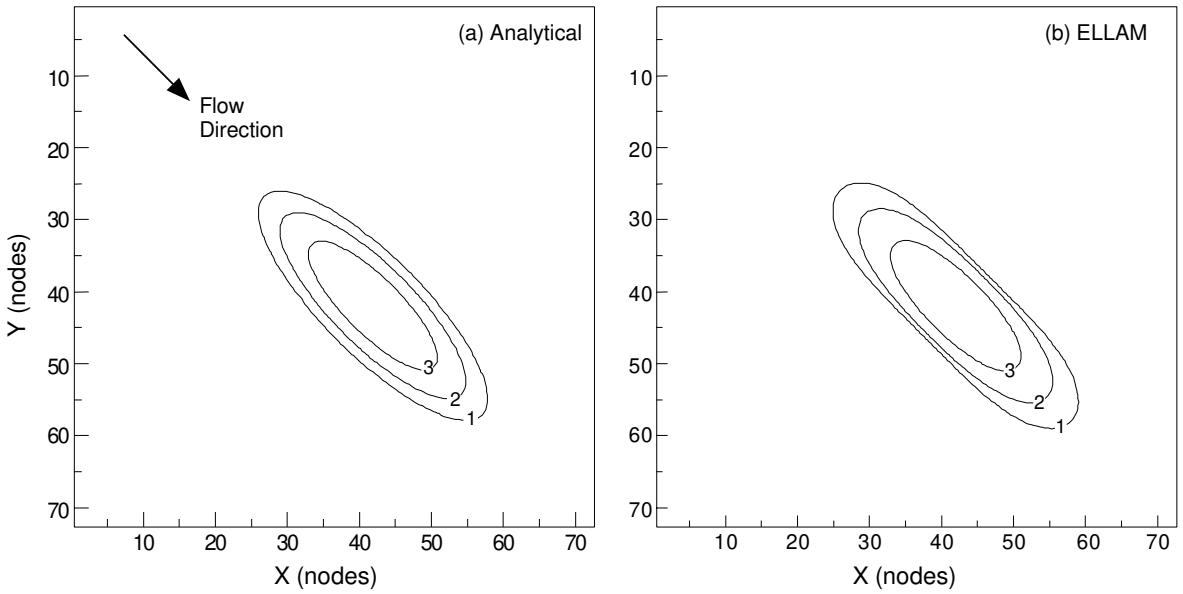


Figure 18. Concentration contours for (a) analytical and (b) *ELLAM* numerical solutions for transport of a point initial condition in uniform flow at 45 degrees to the *x*-direction at *t* = 130 days. Contour values are the log of the concentrations.

solution at *t* = 90 days as the initial conditions. The results using CELDIS = 5 (three time increments), NSC = NSR = NSL = 4, and NT = 2 are shown in figure 18b for the plane of the initial source. As in the previous case (where flow is aligned with the grid), *ELLAM* produces the symmetry characteristic of the analytical solution. There is also slight longitudinal spreading (numerical dispersion) that is not alleviated by increasing the number of time steps.

Unlike the previous case, the numerical results in figure 18b do show some distortion of the shape of the plume relative to the analytical solution. It is not as pronounced, however, as the "hourglass" shape yielded by *MOC3D* for the Dirac problem (see Kipp and others, 1998, figure 14). There is a narrowing of the plume calculated with the numerical model, which is characteristic of a grid-orientation effect and is caused primarily by the off-diagonal (cross-derivative) terms of the dispersion tensor. When flow is oriented parallel to the grid, or when longitudinal and transverse dispersivities are equal, the cross-derivative terms of the dispersion tensor are zero. Because flow is at 45 degrees to the grid in this test problem, the cross-derivative 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-derivative 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 can be reduced by using a finer grid.

Some small areas of negative concentrations were calculated, but they do not appear in figure 18b using logarithmic-scale contouring. To show the extent of the areas of negative concentration, we have replotted the results illustrated in figure 18b in figure 19, using two types of shading for areas where the relative concentration is less than -0.05 and less than -10.0. We tested the sensitivity of the extent of negative concentrations to the size of the transport time

increment by reducing the value of CELDIS to 0.25. The area over which negative concentrations occurred was only slightly smaller. The increase in execution time, however, was significant, so the very small improvement does not appear to justify the extra computational costs.

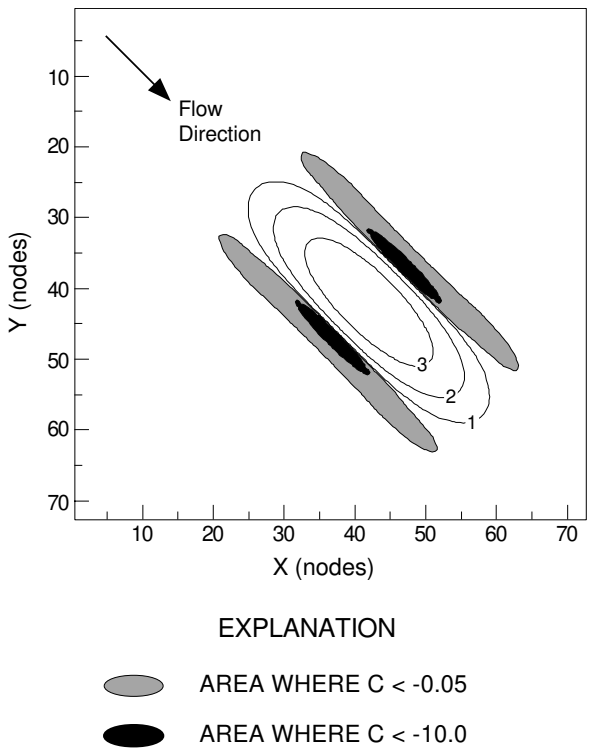


Figure 19. Concentration contours for *ELLAM* numerical solution showing areas of calculated negative concentrations for problem represented in figure 18b.

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 *ELLAM* algorithm in *MOC3D*, as was also done by Konikow and others (1996) and Kipp and others (1998). Burnett and Frind (1987) used an alternating-direction Galerkin finite-element technique to 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 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 *ELLAM* simulations were designed to match as closely as possible the finite-element mesh used by Burnett and Frind (1987). Some differences in discretization, however, could not be avoided because the finite-element method uses a point-centered grid whereas *ELLAM* 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 *ELLAM* 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 *ELLAM* 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 simulation of this test problem was for the case of a two-dimensional model. The input data values for this analysis are listed in table 7. The top discretization layer consisted of constant-head nodes and the solute source.

Results for the two-dimensional case from the *ELLAM* simulation closely match those of Burnett and Frind (1987) (see figure 20). The results using CELDIS = 30 (seven time increments), NSC = NSR = NSL = 4, and NT = 32 are shown. The shape of the plume is almost exactly the same for both models. In the *ELLAM* results, however, the highest concentration contour (0.9) does not extend as far downgradient as that of Burnett and Frind (1987), while the low concentration contour (0.3) from *ELLAM* extends slightly farther downgradient. Overall, the *ELLAM* results provide a closer match to the contours of Burnett and Frind than do the *MOC3D* contours using 381, 1901, or 4218 time increments (see Kipp and others, 1998). The *ELLAM* contours (for all NS values tested) are free of "wiggles" in the *MOC3D* solution discussed by Kipp and others (1998). Increasing the number of transport time increments produced a solution having a slightly greater downgradient extent, but still short of *MOC3D* results.

Table 7. Parameters used for *ELLAM* 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
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
Layer thickness (b=Δz)	0.2222-0.2333 m
CELDIS	30.0
NSCEXP	2
NSREXP	2
NSLEXP	2
NTEXP	5

[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.

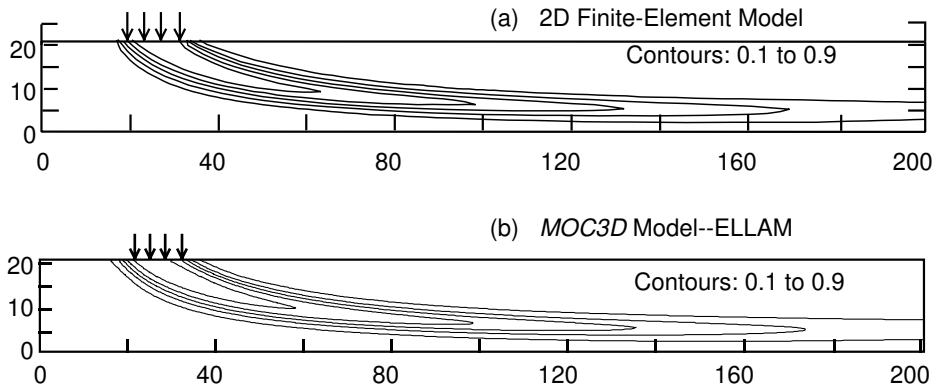


Figure 20. Two-dimensional simulation results for nonuniform-flow test case showing plume positions as contours of relative concentration; (a) finite-element model (modified from Burnett and Frind, 1987, figure 8a), and (b) *ELLAM* solution using CELDIS = 30. Contour interval is 0.2 relative concentration.

As was done for the *MOC3D* tests (Konikow and others, 1996; Kipp and others, 1998), the *ELLAM* grid was expanded laterally to 15 rows having Δy of 1.0 m for the three-dimensional version of this case. Figure 21 shows the transport results in a vertical plane at the middle of the plume for both models for the case in which $\alpha_{T_V} = 0.01$ m and $\alpha_{T_H} = 0.1$ m. The *ELLAM* results for the vertical plane in the first row are contoured in figure 21b (because of symmetry, we only simulate half of the plume, as explained by Konikow and others, 1996). The *ELLAM* plume closely matches that calculated by the finite-element model (figure 21a), although the former shows slightly farther downstream migration of low concentrations of solute. As in the two-dimensional case, the *ELLAM* solution provides a closer match to the Burnett and Frind (1987) solution than do the previous *MOC3D* results.

Figure 22 shows the results for the case in which the vertical transverse dispersivity is increased by a factor of ten, so that $\alpha_{T_H} = \alpha_{T_V} = 0.1$ m. The *ELLAM* results for CELDIS = 30 yielded concentrations that were noticeably low near the source (near the upgradient end of the plume), so the simulation was repeated using CELDIS = 21 (10 time increments). These ELLAM results are illustrated in figure 22b) and appear to agree very closely with the results of Burnett and Frind (1987) (figure 22a).

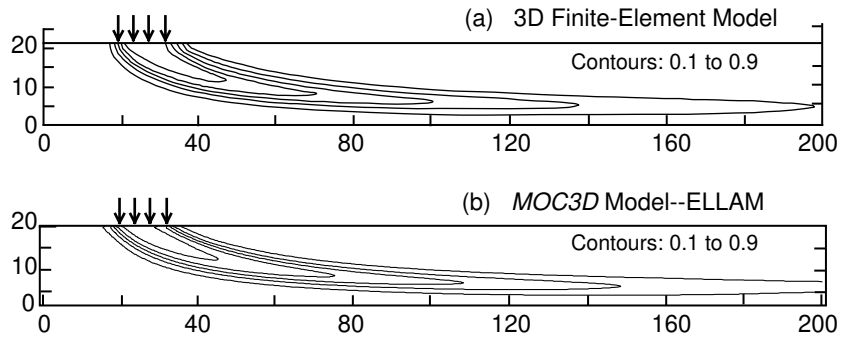


Figure 21. Three-dimensional simulation results for nonuniform-flow test case in which α_{TH} = 0.1 m and α_{TV} = 0.01 m: (a) finite-element model (modified from Burnett and Frind, 1987, figure 8c), and (b) numerical *ELLAM* solution using CELDIS = 30. Plume positions are represented by contours of relative concentration; contour interval is 0.2 relative concentration.

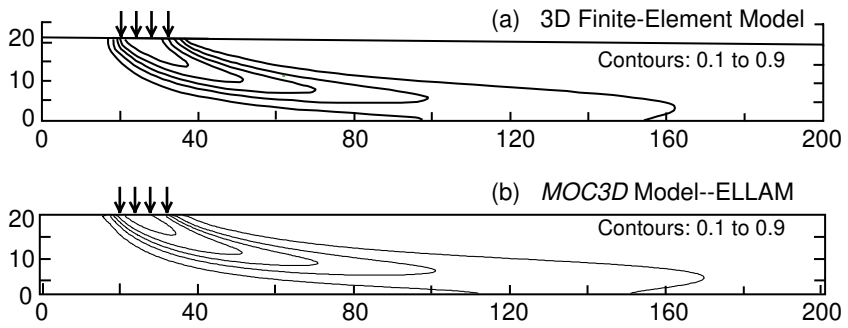


Figure 22. Three-dimensional simulation results for nonuniform-flow test case in which $\alpha_{TH} = \alpha_{TV}$ = 0.1 m: (a) finite-element model (modified from Burnett and Frind, 1987, figure 9b), and (b) numerical *ELLAM* solution using CELDIS = 21. Plume positions are represented by contours of relative concentration; contour interval is 0.2 relative concentration.

Relative Computational and Storage Efficiency

Computer-memory requirements for *ELLAM* are greater than those for the explicit or implicit *MOC3D* dispersive transport algorithm. The additional arrays required can increase the memory size requirement by as much as a factor of three (see table 8).

The computational effort required by the *ELLAM* simulator is strongly dependent on the size of the problem being solved, as determined by the total number of nodes, the NS and NT values, and the total number of time increments (controlled by CELDIS). The user is cautioned that using values for NS and NT parameters that are too small for a given problem may lead to inaccurate solutions. Sensitivity testing will help the user determine appropriate values to specify. Analyses indicate that the greatest computational effort, as measured by *CPU* time, is typically expended in the mass tracking routines. For a given problem, computational time 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.

For a given problem, the *ELLAM* algorithm can often yield an accurate solution more efficiently than the previously documented explicit or implicit MOC options. However, this will typically require the use of a CELDIS value of 5 or more; the explicit and implicit versions of MOC require that CELDIS be less than or equal to 1.0. Table 8 shows that *ELLAM* was more efficient for three of the six test problems evaluated.

Table 8. Execution times and storage requirements for *MOC3D* and *ELLAM* for selected test cases

	Run Time in CPU-seconds			Array Elements Used[1]		
Problem Description	Explicit	Implicit	ELLAM	Explicit	Implicit	ELLAM
One-Dimensional Steady Flow[2]	7	10	9 CELDIS=10.1	11,457	17,400	33,489
Three-Dimensional Steady Flow[2]	404	175	1,366 CELDIS=1	897,331	1,602,994	3,344,624
Two-Dimensional Radial Flow and Dispersion[2]	930	445	138 CELDIS=5	455,737	499,900	233,564
Point Initial Condition in Uniform Flow[2]	210	310	2,721 CELDIS=5	1,728,673	2,406,112	3,384,524
Constant Source in Nonuniform Flow (Two-Dimensional)[3]	13,360	2,450	2,245 CELDIS=30	868,951	1,457,602	2,850,056
Constant Source in Nonuniform Flow (Three-Dimensional)[3]	38,117	12,026	4,400 CELDIS=30	12,823,151	21,652,034	41,206,836

[1] Data arrays and lists for *MODFLOW* and explicit *MOC3D* are allocated space in one array, the *MODFLOW* "X" array. ELLAM also uses an "MX" array for integer arrays.

[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*.

CONCLUSIONS

The *ELLAM* advective-dispersive transport algorithm presented as an alternative solution method within the *MOC3D* simulator can model 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 *ELLAM* simulator were evaluated by comparison to analytical and numerical solutions for the same set of test problems as reported for *MOC3D* (Versions 1 and 2), with the instantaneous point source problem modified slightly. These evaluation tests indicate that the solution algorithms in the *ELLAM* model can successfully and accurately simulate three-dimensional transport and dispersion of a solute in flowing ground water. To avoid non-physical oscillations and loss of peak concentrations, care must be taken to use a grid having sufficient mesh density to adequately resolve sharp fronts. The primary advantages of the *ELLAM* code are that fewer transport time steps need be used and that mass is conserved globally. Using *ELLAM* with few time steps can provide an accurate and cost-effective way of discerning salient features of a solute-transport process under a complex set of boundary conditions. Furthermore, the *ELLAM* algorithm eliminates the previous restriction in *MOC3D* that the transport grid be uniformly spaced. The computational effort required by the *ELLAM* simulator is strongly dependent on the size of the problem being solved, as determined by the total number of nodes, the NS and NT values, and the total number of time increments (controlled by CELDIS, a model parameter that is analogous to the Courant number). For test cases in which *ELLAM* was more efficient than the explicit or implicit MOC options, use of CELDIS values equal to or greater than 5.0 were required.

REFERENCES

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.

Celia, M.A., Russell, T.F., Herrera, I., and Ewing, R.E., 1990, An Eulerian-Lagrangian localized adjoint method for the advection-diffusion equation: Advances in Water Resources, v. 13, no. 4, p. 187-206.

Garder, A.O., Peaceman, D.W., and 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., 1999, Age, Double porosity, and simple reaction modifications for the MOC3D ground-water transport model: U.S. Geological Survey Water-Resources Investigations Report 99-4041, 34 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.

Healy, R.W., and Russell, T.F., 1993, A finite-volume Eulerian-Lagrangian localized adjoint method for solution of the advection-dispersion equation: Water Resources Research, v. 29, no. 7, p. 2399-2413.

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., Konikow, L.F., and Hornberger, G.Z., 1998, An implicit dispersive transport algorithm for the U.S. Geological Survey MOC3D solute-transport model: U.S. Geological Survey Water-Resources Investigations Report 98-4234, 54 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.

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.

APPENDIX A: DATA INPUT INSTRUCTIONS FOR *MOC3D (Version 3.5)*

This Appendix includes a complete set of instructions for preparing a data set for the *MOC3D* model. For more comprehensive descriptions of input parameters, options, and underlying assumptions, however, the user should also refer to Konikow and others (1996), Kipp and others (1998), and Goode (1999). Major changes that have been implemented since the release of Version 3.0 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. Variables that are optional are enclosed in brackets, as in {option}.

MOC3D Transport Name File (CONC)

FOR EACH SIMULATION:

1. Data: FTYPE NUNIT FNAME

The name file consists of records defining the names and unit 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*** or ***ELLAM*** Main *MOC3D* input data file *[required].* Specifying ***MOC*** indicates dispersion calculations will be explicit (as described by Konikow and others, 1996); specifying ***MOCIMP*** indicates dispersion calculations will be implicit (as described by Kipp and others, 1998); and specifying ***ELLAM*** indicates that the solute-transport equation will be solved using the *ELLAM* method (as described in this report).

CRCH Concentrations in recharge *[optional].*

CNCA Separate output file containing concentration data in ASCII (text-only) format *[optional].* Frequency and format of printing controlled by NPNTCL and ICONFM. 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 *[optional].* Frequency and format of printing controlled by NPNTVL and IVELFM.

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

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

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].*

AGE Ground-water age simulation input file *[optional].* (Not compatible with *ELLAM* option.)

DP Double porosity input file *[optional].* (Not compatible with *ELLAM* option.)

DK Simple reactions(decay, zero-order growth, retardation) input file *[optional]*. (Not compatible with *ELLAM* option.)

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.

Note: AGE, DP, and DK file types are described by Goode (1999).

Main *MOC3D* Package Input (*MOC, MOCIMP,* or *ELLAM*)

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 (note that data set numbers do not necessarily correspond with line numbers in the file). 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 previous versions of *MOC3D*, use of the *ELLAM* option requires definition of several alternative input parameters and deletes two previously required data sets (6 and 13), which are used only if *MOC* or *MOCIMP* is selected.

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 if FTYPE *MOC* or *MOCIMP* are specified in the transport name file, but subgrid spacing can vary as in *MODFLOW*

if *ELLAM* is specified. The thickness can vary from cell to cell and layer to layer. However, 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.

IF Ftype *MOC* OR *MOCIMP* IS ACTIVE:

5a. 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:

NPMAX = 2× NPTPND× NSROW× NSCOL× 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 Ftype *ELLAM* IS ACTIVE:

5b. Data: NSCEXP NSREXP NSLEXP NTEXP

NSCEXP Exponent used to calculate the number of subcells in the column direction (NSC, where NSC = 2**NSCEXP).

NSREXP Exponent used to calculate the number of subcells in the row direction (NSR).

NSLEXP Exponent used to calculate the number of subcells in the layer direction (NSL).

NTEXP Exponent used to calculate the number of sub-time steps per transport time increment (NT).

Notes:

In general, numerical accuracy will be increased by increasing the value of these parameters. This will also, however, increase computational costs. For each of the four parameters above, the value represents the exponent y in the expression 2^y.

Entering a zero or negative value for any of the above variables will cause the code to use default values. Default values for NSCEXP, NSREXP, and NSLEXP are 2 in active dimensions and 1 in inactive dimensions (for example, if a simulation represented a two-dimensional areal problem in which the number of rows and columns were greater than one and the number of layers equals one, then default settings would be NSCEXP = 2, NSREXP = 2, and NSLEXP = 1, and the number of subcells in each direction would be 4, 4, and 2, respectively). The default value of NTEXP is 2.

IF *MOC* OR *MOCIMP* IS ACTIVE AND 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≤CELDIS ≤1.0). For *ELLAM*, CELDIS can be greater than 1.0, and specifying CELDIS = 0.0 will result in one transport time step being used (which is not generally recommended).

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≤FZERO ≤0.05). Only specify if *MOC* or *MOCIMP* is active.

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.") Only specify if *MOC* or *MOCIMP* is active. If *ELLAM* is specified, the code will automatically set INTRPL = 1.

IF *MOCIMP* IS ACTIVE:

7.1 Data: FDTMTH NCXIT IDIREC EPSSLV MAXIT

FDTMTH Weighting factor for temporal differencing of dispersion equation ($0.0 \leq$ 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 $\leq 10^{-5}$. An adequately small value of EPSSLV has the property that a smaller value does not change the numerical solution within the number of significant digits 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 difference 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 9, 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 9).

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 9).

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. Only specify if *MOC* or *MOCIMP* is active.

Table 9. 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)

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

MODFLOW grid. If the transport subgrid has the same dimensions as the flow grid, 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 IF *MOC* OR *MOCIMP* IS ACTIVE:

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*

Notes:

Data sets 14-16 should include one value for each layer in subgrid.

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. If the *ELLAM* option is being used, the variation in thickness (not thickness times porosity) between neighboring cells within a layer should be small.

* 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 observa-tion 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 one of the cases in the one-dimensional steady-state flow problem. Parameter values are indicated in table 4 and selected results are shown in figures 10 and 12 (this is the case of CELDIS = 1.0). 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 *ELLAM* 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  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
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.   0.0000001     0     0.001    0 ; ACCL,HCLOSE,IPCALC,WSEED,IPRSIP
```

←1
←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
ellam     96      finite.ell
obs       44      finite.obs
data      45      finite.oba
cnca      22      finite.cna
```

←Designates main output file for *MOC3D*
←Main input data file for *ELLAM*
←Input data file for observation wells
←Output file for observation well data
←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.ell*

```
One-dimensional, Steady Flow, No Decay, Low Dispersion: MOC3D (ELLAM)
    ISLAY1    ISLAY2    ISROW1    ISROW2    ISCOL1    ISCOL2
         1         1         1         1         2       121
         0      0.00       0.0 ; NODISP, DECAY, DIFFUS
   5 1 1 7                     ; NSCEXP,NSREXP,NSLEXP,NTEXP
       1.0         1           ; CELDIS, INTRPL
  0 -1 0 0 0 0 0 ; NPNTCL, ICONFM, NPNTVL, IVELFM, NPNTDL, IDSPFM, NPRTPL
       0.0                                          ; CNOFLO
         0       0.0 (122F3.0)                      ; initial concentration
         0       1.                                 ; C' inflow
         2                                          ; NZONES to follow
        -1       1.0                                ; IZONE, ZONCON
        -2       0.0                                ; IZONE, ZONCON
         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 *ELLAM* parameters (exponents for NSC, NSR, NSL, and NT)

5 Courant number and interpolation method flag

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)

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   2  45        ;layer, row, column, unit number
  1  1  42            ;layer, row, column
  1  1 112            ;layer, row, column
```

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 a 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 *ELLAM* 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 3.5) July 2000


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

OPENING finite.ell
FILE TYPE: ELLAM   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: ELLAM
    ISLAY1    ISLAY2    ISROW1    ISROW2    ISCOL1    ISCOL2
```

PROBLEM DESCRIPTORS, INCLUDING GRID CHARACTERISTICS AND ELLAM INPUT 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

NONUNIFORM 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

 ELLAM INPUT PARAMETERS:
  NSCEXP, NSREXP, NSLEXP, NTEXP
      5       1       1       7
  NSC, NSR, NSL, NT (CALCULATED)
   32    2    2  128
    18288 ELEMENTS IN X ARRAY ARE USED BY MOC
 CELDIS=     1.0

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

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

         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
```

OUTPUT CONTROL

INITIAL AND BOUNDARY CONDITIONS FOR SOLUTE

```
ALL OBSERVATION WELL DATA WILL BE WRITTEN ON UNIT  45

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

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  ...

...

 NUMBER OF MOVES FOR CELDIS CRITERIA:
       120

          TIME STEP    1 IN STRESS PERIOD    1
          NO. OF PARTICLE MOVES REQUIRED TO COMPLETE THIS TIME STEP  =  120
            MOVE TIME STEP (TIMV)=  1.000000000000E+00

           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  120  OUT OF  120

  ELAPSED TIME =  1.2000E+02

  CHEMICAL MASS IN STORAGE:
      INITIAL:   MASS DISSOLVED =  0.0000E+00     MASS SORBED =  0.0000E+00
      PRESENT:   MASS DISSOLVED =  1.0203E-01     MASS SORBED =  0.0000E+00

            CHANGE IN MASS STORED = -1.0203E-01


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

      IN:
      ---
                  DECAY =  0.0000E+00
             CONSTANT HEAD =  0.0000E+00
      SUBGRID BOUNDARY =  1.2000E-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 = -1.7972E-02
                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 = -1.7972E-03


     SOURCE-TERM DECAY =  0.0000E+00


               RESIDUAL =  2.5183E-06

   PERCENT DISCREPANCY =  2.0986E-03 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
 1.0000E+00   1.200E+01   2.587E-01   8.000E+00   8.100E-07   1.000E+00   1.887E-16
 2.0000E+00   1.200E+01   3.780E-01   8.000E+00   8.271E-06   1.000E+00   2.499E-15
 3.0000E+00   1.200E+01   4.558E-01   8.000E+00   4.364E-05   1.000E+00   2.128E-14
 4.0000E+00   1.200E+01   5.140E-01   8.000E+00   1.591E-04   1.000E+00   1.447E-13
 5.0000E+00   1.200E+01   5.606E-01   8.000E+00   4.519E-04   1.000E+00   8.378E-13
 6.0000E+00   1.200E+01   5.993E-01   8.000E+00   1.070E-03   1.000E+00   4.233E-12
 7.0000E+00   1.200E+01   6.322E-01   8.000E+00   2.202E-03   1.000E+00   1.971E-11
 8.0000E+00   1.200E+01   6.606E-01   8.000E+00   4.060E-03   1.000E+00   8.664E-11
 9.0000E+00   1.200E+01   6.856E-01   8.000E+00   6.853E-03   1.000E+00   3.646E-10
 1.0000E+01   1.200E+01   7.078E-01   8.000E+00   1.076E-02   1.000E+00   1.404E-09
...
...
 1.1100E+02   1.200E+01   9.955E-01   8.000E+00   9.428E-01   1.000E+00   5.284E-01
 1.1200E+02   1.200E+01   9.957E-01   8.000E+00   9.446E-01   1.000E+00   5.374E-01
 1.1300E+02   1.200E+01   9.958E-01   8.000E+00   9.463E-01   1.000E+00   5.464E-01
 1.1400E+02   1.200E+01   9.959E-01   8.000E+00   9.479E-01   1.000E+00   5.553E-01
 1.1500E+02   1.200E+01   9.960E-01   8.000E+00   9.495E-01   1.000E+00   5.640E-01
 1.1600E+02   1.200E+01   9.962E-01   8.000E+00   9.511E-01   1.000E+00   5.727E-01
 1.1700E+02   1.200E+01   9.963E-01   8.000E+00   9.526E-01   1.000E+00   5.813E-01
 1.1800E+02   1.200E+01   9.964E-01   8.000E+00   9.540E-01   1.000E+00   5.897E-01
 1.1900E+02   1.200E+01   9.965E-01   8.000E+00   9.554E-01   1.000E+00   5.980E-01
 1.2000E+02   1.200E+01   9.966E-01   8.000E+00   9.568E-01   1.000E+00   6.063E-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=  120, KSTP=    1, KPER=    1, SUMTCH=1.2000E+02
  1.5000E-01  5.0000E-02  5.0000E-01  9.9660E-01
  2.5000E-01  5.0000E-02  5.0000E-01  9.9667E-01
  3.5000E-01  5.0000E-02  5.0000E-01  9.9634E-01
  4.5000E-01  5.0000E-02  5.0000E-01  9.9600E-01
  5.5000E-01  5.0000E-02  5.0000E-01  9.9563E-01
  6.5000E-01  5.0000E-02  5.0000E-01  9.9524E-01
  7.5000E-01  5.0000E-02  5.0000E-01  9.9482E-01
  8.5000E-01  5.0000E-02  5.0000E-01  9.9438E-01
  9.5000E-01  5.0000E-02  5.0000E-01  9.9392E-01
...
  1.1650E+01  5.0000E-02  5.0000E-01  5.8604E-01
  1.1750E+01  5.0000E-02  5.0000E-01  5.8337E-01
  1.1850E+01  5.0000E-02  5.0000E-01  5.8129E-01
  1.1950E+01  5.0000E-02  5.0000E-01  5.7985E-01
  1.2050E+01  5.0000E-02  5.0000E-01  5.7913E-01
```