DRAFT

User’s Guide for the PLUMECALC Application

Version 2.2

Bruce A. Robinson
Zora V. Dash

REVISION HISTORY

Revision	Date	Purpose of the Revision
0	2/2/07	Initial Implementation

CONTENTS

A Particle Tracking Transport Method for the Simulation of Resident and Flux-Averaged Concentration of Solute Plumes in Groundwater Models

A new numerical technique called the Convolution-Based Particle Tracking (CBPT) method is developed to simulate resident or flux-averaged solute concentrations in groundwater models. The method is valid for steady state flow and linear transport processes such as sorption with a linear sorption isotherm, diffusion into matrix rock, and first order decay. Under these constraints, we can take advantage of the principle of superposition of multiple solute sources and numerical convolution to handle time-varying sources. A pulse of particles is introduced at each source location, and the technique accounts for the time variation of each input source function during the course of the calculation. In principle, the mathematical derivation employing the convolution and superposition assumptions could have been applied to any numerical technique for solving the Advective-Dispersion equation. The CBPT method uses particle tracking to take advantage of the ability of particle-based approaches to maintain sharp fronts for advection-dominated transport problems common in groundwater modeling, and because of the flexibility of the random walk method to simulate a wide range of possible forms of the dispersion tensor. Furthermore, the algorithm for carrying out the convolution and superposition calculation from particle tracking results is very efficient. We show that from a single particle tracking run, source term variability, sorption, diffusion, and decay can all be simulated rapidly without rerunning the underlying transport model unless the flow field or dispersion parameters are changed. A series of verification and demonstration simulations are presented to demonstrate the use of PLUMECALC, the code which implements this method.

1. INTRODUCTION

1.1 PURPOSE

The objective of the User Information Document is to provide information to the end-user on the theory of the convolution-based particle tracking method and how to effectively install and use the software

1.2 SOFTWARE IDENTIFICATION

PLUMECALC Version 2.2

1.3 DEFINITIONS

CBPT	**-** Convolution-based particle tracking method.
FEHM	- Finite-element heat- and mass-transfer code.
OMR	**-** Octree mesh refinement.

2. THEORY

2.1 INTRODUCTION

Particle tracking methods have become popular for the study of flow and solute transport in groundwater modeling. The most common applications of particle tracking models are for the delineation of pathlines in a flow model. Particles undergoing only advection are introduced at various locations in the flow model, and pathlines and travel times are simulated and visualized. Reverse particle tracking is another common technique used to predict the source location of fluid present at certain locations in the model, such as at pumping wells or points of discharge. Methods for interpolating the velocity within numerical grid cells are well known for structured grids with rectangular-shaped control volume cells (Pollock, 1988, Schafer-Perini and Wilson, 1991), and similar interpolation techniques for unstructured grids are also becoming available (Cordes and Kinzelbach, 1992, Prevost et al., 2001).

The simulation of the Advective-Dispersion equation (ADE) requires the introduction of the dispersion tensor in the particle tracking method. The random walk particle tracking method is fairly commonly used to investigate advection and dispersion in groundwater systems. Tompson et al. (1987) and Tompson and Gelhar (1990) outlined the general theory of particle tracking and the computation of random walk terms needed to reproduce the ADE using particles. These simulations can be used to model plume concentrations for conservative or simple sorbing solutes (Kinzelbach, 1988, Tompson, 1993). Traditionally, the solute source term has been quite simple in numerical modeling studies using particle tracking, such as a pulse input or input of limited duration. This type of source term is useful for investigating the basic advective and dispersive properties of groundwater systems (e.g. Tompson and Gelhar, 1990), as it is straightforward to compute the spatial and temporal moments. Reverse particle tracking with dispersion has been proposed to solve the adjoint problem for transport (Uffink, 1989) in order to determine, for example, the probability that water produced at a pumping well came from a given source region (Neupauer and Wilson, 2001).

The particle tracking method has several attractive features that make it a popular choice for groundwater modeling studies. Perhaps most important is the ability to maintain sharp fronts for low-dispersion systems. This advantage is often stated in terms of the ability to model low longitudinal dispersion, but in large-scale groundwater flow problems, the most severe numerical constraints are posed by low transverse dispersion. Gelhar (1997) suggests that in groundwater systems, transverse dispersivity in the horizontal direction is on the order of 1 meter, and vertical transverse dispersivities can be as low as about 1 cm or less. Clearly, solutions free of numerical dispersion are very difficult to obtain in such systems. When the goal is the computation of concentration of a plume in a groundwater model, numerical dispersion becomes the paramount issue. A variety of numerical techniques have been developed to address this issue, including method of characteristics (Chiang et al., 1989), dynamic mesh refinement (Wolfsberg and Freyberg, 1994), and Total Variation Diminishing (TVD) integration schemes (e.g. Cox and Nishikawa, 1991). Particle tracking methods also offer the promise of eliminating artificial solute spreading in groundwater transport simulations. Another desirable feature of particle tracking methods is the ease with which various forms of the dispersion tensor can be implemented in a manner that minimizes grid effects for flow not orthogonal to the grid (Lichtner et al., 2002). In contrast, finite difference solute transport solutions for anisotropic dispersion tensors require complex extensions to conventional models to minimize grid effects, such as the expansion of the stencil used for the integration at each cell. Finally, the use of particle tracking allows advanced methods to be implemented in which the time variable is also a stochastic variable to allow diffusion into the rock matrix to be modeled (Robinson et al., 2003).

There are also a few drawbacks of the particle tracking technique. Labolle et al. (1996) showed that care must be taken to minimize local mass conservation errors in systems with contrasting transport velocities. In short, random-walk displacements can result in excessive trapping of particles in low-velocity zones unless specialized methods are introduced to compute precisely the correction terms that have been derived to address this issue. Even if it is assumed that this problem is circumvented using techniques such as those of Labolle et al. (1996), the use of particles to simulate aqueous concentration of a solute requires that enough particles be used to minimize jaggedness in the computed concentration field. This issue is especially relevant for non-point sources and for source terms that vary over time. In principle, time-varying sources can be implemented by scaling the rate of introduction of particles to the input solute mass flux, or by varying the mass associated with each particle. However, if that input varies over a wide range of mass flux, or if the duration of solute input is long, the spatial density of particles during some parts of the simulation will likely be very low, or else an extremely large number of particles will need to be used. To date, this problem has perhaps limited the usefulness of particle tracking approaches to theoretical studies in which the source term is fairly simple. For practical problems of simulating transport in complex groundwater models, calculation of concentration of a solute plume is still most often done through finite difference, finite element, and the other methods mentioned above for solving the ADE.

In this document, a new numerical technique called the Convolution-Based Particle Tracking (CBPT) method is developed that allows for an accurate solution of plume concentrations using the particle tracking method. The method, which is valid for steady state flow fields and linear transport processes, employs the principle of superposition in space and time to take advantage of efficiencies that can be gained under the assumption of linearity. Using this method, the system response to an instantaneous pulse (or pulses for multiple solute sources) of particles is used to compute plume concentration for an arbitrary solute mass input function. In the derivation we distinguish between the resident concentration, the average concentration of fluid in a control volume, and the flux-averaged concentration, the average concentration in fluid weighted by the relative fluid flux associated with different parcels of fluid. We show that particle tracking results can be recorded in a manner that facilitates the efficient calculation of plume concentrations (resident concentration) or mixed average concentrations at fluid outlets or within the system (flux-averaged concentration).

The remainder of the section is organized as follows: First, the theory of convolution and superposition is applied to the problem of computing solute concentrations. The derivation presented is general, without reference to any particular numerical method for solving the transport equation. Next, after briefly describing the random walk particle tracking model employed in this software, we outline the CBPT method, which uses particle tracking results in the convolution process. We also discuss implementation details related to the computation of resident and flux-averaged concentrations, and demonstrate how the method can be extended to account for sorption and diffusion into the rock matrix.

2.2 CONVOLUTION AND SUPERPOSITION

2.2.1 Advective-Dispersion Equation

We begin the development by considering transport of a solute in a steady state flow system. The solute undergoes transport with advection and dispersion, assuming the conventional form of the Advective-Dispersion equation (ADE):

$$\frac{\partial(\theta C)}{\partial t} + \nabla \cdot (\theta C \tilde{v}) - \nabla \cdot (\theta \tilde{D} \cdot \nabla C) = 0 \qquad (1)$$

where C is the concentration in the fluid (moles/liter fluid), $\tilde{v}$ is the Darcy velocity vector, θ is the volumetric water content (porosity for a saturated medium), and $\tilde{D}$ is the dispersion tensor. For many practical applications, the terms in Eq. 1 are taken to be linear, such that the terms in the equation are not themselves functions of concentration. The assumption of linearity and steady state flow forms the basis for the calculation approach to determining the concentration distribution within a complex model domain. First, we focus on the resident concentration, defined as the mass of solute within a control volume $V(\tilde{x})$ divided by the fluid volume in the control volume. The resident concentration is denoted by the variable C in this document. Later, we consider an alternate concentration definition that is appropriate under some restricted conditions, called the flux-averaged concentration. The meaning of these concentrations, their importance in modeling Eq. 1, and the usefulness of each when characterizing solute transport systems, are discussed in detail in Kreft and Zuber (1978) and Parker and van Genuchten (1984).

2.2.2 Calculation of Resident Concentration

First, consider a solute source location ξ in the system, as a result of source fluid with a given concentration, the dissolution of a solid waste, or some other contaminant source. Ultimately, the goal of this approach is to compute the concentration within the system or at fluid exit points such as wellbores or points of discharge in groundwater system. The first step towards this goal is to define the mass density function within the system $\hat{c}(\xi, \tilde{x}, t)$ as the probability of locating mass from source location ξ at a control volume $V(\tilde{x})$ centered at location $\tilde{x}$ at time t. The function $\hat{c}(\xi, \tilde{x}, t)$ can be determined by simulating the solution to Eq. 1 in response to a Dirac delta function input of solute mass at the starting location, and normalizing the concentration by dividing by the input solute mass. The units of $\hat{c}(\xi, \tilde{x}, t)$ are thus 1/m^3. Conservation of solute mass over the computational domain Ω requires that

$$\int_{\Omega} \hat{c}(\xi, \tilde{x}, t) dV = 1 - \zeta(\xi, t) \tag{2}$$

where $\zeta(\xi, t)$ is the cumulative mass from source ξ that has left the system via a fluid sink up to time t.

Next, we treat a time-varying input mass flux $\dot{m}(\xi, t)$ (unit of moles/s) at ξ. Recognizing that under the assumptions stated above, the principle of superposition in time applies, the concentration $c(\xi, \tilde{x}, \tau)$ at time τ can be computed using the following numerical convolution equation:

$$\theta c(\xi, \tilde{x}, \tau) = \int_{0}^{\tau} \dot{m}(\xi, \tau - t) \hat{c}(\xi, \tilde{x}, t) dt \tag{3}$$

To understand Eq. 3 intuitively, we note that the solute mass arriving at the control volume at $\tilde{x}$ for a transient solute input contains contributions from mass that entered at all times before τ. The requirement that the mass contribute to the calculation of $c(\xi, \tilde{x}, \tau)$ is that it arrives at this location at time τ; this can happen due to recently injected mass that quickly reaches $\tilde{x}$, or mass that was injected initially and took the full time τ to reach $\tilde{x}$. Equation 3 mathematically sums up all of the contributions of mass traveling at different rates that arrive at this location at time τ.

The final step in the calculation is to incorporate spatial variability in the solute source terms. If a different mass flux $\dot{m}(\xi_i, t)$ is assumed to enter the system at N_s source locations, then the total concentration $C(\tilde{x}, \tau)$ (the capital C denoting total concentration in response to multiple sources) is determined by superposition of the individual concentrations $c(\xi_i, \tilde{x}, \tau)$ obtained from each source term:

$$C(\tilde{x}, \tau) = \sum_{i=1}^{N_s} c(\xi_i, \tilde{x}, \tau) \tag{4}$$

In a practical model application, the discretization used for the source regions to determine $C(\tilde{x},\tau)$ using Eq. 4 will depend on the number of individual source zones that are required to accurately depict the spatial and temporal variability in the source.

Other simple reactions and transport processes such as first-order decay reactions, equilibrium sorption with a linear isotherm, or diffusion into dead-end pore space can all be incorporated into this method provided simple, linear processes are assumed. For first-order decay reactions such as radioactive decay, we recognize that the concentration associated with a parcel of solute mass is a simple function of the time since the mass entered the system. Therefore, Eq. 3 is corrected for decay as follows:

$$\theta c(\xi,\tilde{x},\tau)=\int_0^\tau \dot{m}(\xi,\tau-t)e^{-kt}\hat{c}(\xi,\tilde{x},\tau)dt \qquad (5)$$

where the decay constant $k=\ln(2)/t_{1/2}$ and $t_{1/2}$ is the half-life. Adaptations to introduce sorption and diffusion into dead end pore space are deferred to Section 2.3 because they are intimately related to the particle tracking method.

2.2.3 Calculation of Flux-Averaged Concentration

Although the in situ or resident concentration C is the most intuitive and perhaps the most meaningful type of concentration that can be computed from Eq. 1, there are instances in which alternative definitions are more useful. In groundwater models, the most notable case is the situation in which fluid from a variety of locations converges at a well bore or a point of discharge from the model. This case is typically handled with a boundary condition in which the fluids mix and leave the system with a single value of concentration. A physical analogy for this boundary condition is the collection of water in a bucket for a short period of time, and the measurement of the concentration when the bucket is filled. It is clear that this concentration is an average of the concentrations of individual streams of fluid, weighted by the fluid flux of each stream. Thus the concentration so determined has been called the flux-averaged concentration. It is possible to compute such a concentration within the flow system as well at a computational grid cell or cells. In this instance, for a control volume $V(\tilde{x})$ we compute the flux-averaged concentration as the mass flux of solute leaving $V(\tilde{x})$ divided by the fluid volumetric flow rate leaving $V(\tilde{x})$.

In the past 50 years, a vast literature on the topic of mixing and residence time distributions in continuous flow systems has been developed in the field of chemical engineering (e.g. the textbook of Nauman and Buffham, 1983, summarizes the theory and applications), and this theory can be used with minor modifications for the simulation of solute concentrations leaving a given location in a steady flow system. We first consider the flux-averaged concentration leaving the system at a fluid exit point. From the same pulse input of solute mass used to compute $\hat{c}(\xi,\tilde{x},t)$ above, we can compute the exit age distribution (also referred to as the residence time distribution, Danckwerts, 1953) $f(\xi,\tilde{x},t)$ as follows: $f(\xi,\tilde{x},t)dt$ is the fraction of the mass injected at source region ξ leaving the system at $\tilde{x}$ between time t and $t+dt$.

For an exit fluid volumetric flow rate $\dot{q}_e(\tilde{x})$, the flux-averaged concentration $\hat{c}(\xi,\tilde{x},\tau)$ exiting at $\tilde{x}$ due to injection at source region ξ is given by the following numerical convolution equation:

$$\hat{c}(\xi,\tilde{x},\tau) = \int_0^{\tau} \dot{m}(\xi,\tau - t)e^{-kt} f(\xi,\tilde{x},t)dt \tag{6}$$

Similar to the calculation of resident concentrations, first order decay reactions are simply incorporated into the calculation of flux-averaged concentrations, as shown in Eq. 6. Furthermore, we note again that Eq. 6, relying on the same set of assumptions as Eq. 5, is restricted to steady state flow and linear transport processes. Finally, multiple sources are handled in a manner analogous to Eq. 4.

2.3 SOLUTION FOR CONCENTRATION FROM PARTICLE TRACKING MODEL RESULTS

2.3.1 Random Walk Particle Tracking Technique

From the previous derivation, we showed that an evolving plume within a steady state groundwater flow domain can be computed for an arbitrarily complex source term by determining $\hat{c}(\xi,\tilde{x},t)$ for all sources and using Eq. 4 and 5, which employ the principles of convolution and superposition. The functions $\hat{c}(\xi,\tilde{x},t)$ or $f(\xi,\tilde{x},t)$ are obtained as the solution to Eq. 1 in response to a pulse of solute input at time 0. Up until this point, the theory is independent of the particular method used to solve the ADE. For reasons stated in the introduction, and because we will show that the method is very efficient, in this software we apply a particle tracking technique to solve for $\hat{c}(\xi,\tilde{x},t)$ and $f(\xi,\tilde{x},t)$. The particle tracking model used here, described in Lichtner et al. (2002), computes the locations of particles using well-known methods obtained from a solution of the Fokker Planck equation. The method combines a deterministic component to treat advection and a random-walk component to simulate dispersion. The derivation of the relations used to compute the random-walk particle trajectories that simulate Eq. 1 has been described in detail elsewhere (e.g. Tompson and Gelhar, 1990, Labolle et al., 1996, Lichtner et al., 2002) and will not be repeated here. The well-known final result of the derivation for particle displacements is

$$X_p(t+\Delta t) = X_p(t) + A[X_p(t)]\Delta t + B[X_p(t)] \cdot Z\sqrt{\Delta t} \tag{7}$$

where X_p is the particle location, Z is a vector of three independent random numbers (mean of 0, variance of 1), and A and B are related to the flow and transport properties of the medium as follows:

$$A = \tilde{v} + \nabla \cdot \tilde{D} + \frac{1}{\theta}\tilde{D} \cdot \nabla\theta \tag{8}$$

$$B \cdot B^T = 2\tilde{D} \tag{9}$$

The characteristics of dispersion in porous media are the subject of a great deal of past and current research. In this work, the form of the dispersion tensor $\tilde{D}$ proposed by Lichtner et al. (2002) for axisymmetric media is used, and the procedure for deriving the random-walk displacement scaling matrix, B, is the one presented in that study. Note that the selection of a different mathematical model for dispersion, including a non-Gaussian dispersion approach, poses no restriction on the plume concentration method of the PLUMECALC software, as long as the assumption of steady state flow is valid and $\hat{c}(\xi, \tilde{x}, t)$ or $f(\xi, \tilde{x}, t)$ can be obtained using particle tracking.

2.3.2 Considerations of Concentration Averaging

Before outlining the numerical approach to determine concentration, a brief discussion of the interpretation of concentration and its calculation from particle tracking results is appropriate. In a groundwater flow system, the locations of solute mass within the flow domain define the concentration field. With a numerical method such as particle tracking, the resident concentration is proportional to the spatial density of particles. This particle density is, in principle, independent of the numerical grid, the geometric representation of the control volumes in a finite volume numerical scheme, or any other arbitrary discretization method to subdivide the domain. However, most numerical models used to compute concentration fields define a system of connected control volumes and compute the average concentration within each volume. This type of averaging homogenizes the concentration within each cell, whereas in reality, a distribution of concentrations may exist at scales smaller than the dimensions of the cell. Bagtzoglou et al. (1992) outlined a variety of interpolation schemes involving what they called projection functions to control the smoothness of the concentration fields determined from particle tracking simulations.

The degree to which this averaging influences the results of a particular application depends on the purpose of the simulation. For example, in groundwater contaminant transport applications, the quantity of interest might be the concentration that would be encountered if a specified quantity of water is extracted by a well. If this fluid volume is similar to or larger than the volume of water residing in a typical cell, then homogenization of concentration does not pose a problem. On the other hand, if accurate local concentrations are important to simulate regardless of the quantity of fluid involved, then the selection of a small enough control volume to capture the concentrations accurately is essential. The issue of small-scale dilution and the estimation of local concentration and its relation to the macroscopic dimensions of a solute plume have been have treated by several authors, including Kapoor and Gelhar (1994) and Kitanidis (1994). In the development of this method, we assume that for the purposes of computing concentrations, the average concentration within the control volumes defined by the finite volume grid used for the flow simulations is an appropriate average. Furthermore, the simplest projection function of Bagtzoglou et al. (1992), the nearest grid point method, is used, which simply counts all the particles within each control volume to determine $\hat{c}(\xi, \tilde{x}, t)$, which is then defined at the center of the control volume. For conservative solutes, this particle density function is proportional to solute mass per volume of the medium. It should be noted that schemes that subdivide the grid, or even those that overlay a completely independent grid, are also possible. Such techniques were not implemented in the current version of the software, but will be added to a future version.

2.3.3 Convolution-Based Particle Tracking Method: Resident Concentration

For the calculation of resident concentration, solution of Eq. 5 using information from particle tracking results can be accomplished in a variety of ways. The simplest, though not necessarily the most accurate or efficient, would be to record the function $\hat{c}(\xi, \tilde{x}, t)$ at a large number of times and perform the convolution integration in a straightforward manner using a simple numerical integration scheme. This method is potentially very memory intensive, requiring storage of $\hat{c}(\xi, \tilde{x}, t)$ at many times over an entire computational domain. A more efficient approach is to store information regarding the time history of the location of each particle and perform the contribution to the convolution integral on a particle-by-particle basis. To derive the equation for this approach, we define the following terms:

t_{in} = the time at which a particle enters cell $\tilde{x}$.

t_{out} = the time at which a particle exits cell $\tilde{x}$.

Then, the contribution to the convolution integral for time τ associated with the residence of this particle within the cell is given by

$$\int_{\tau_{out}}^{\tau_{in}} \dot{m}(\xi, t) e^{-k(\tau - t)} dt \qquad (10)$$

where $\tau_{in} = \max(0, \tau - t_{in})$ and $\tau_{out} = \max(0, \tau - t_{out})$. The limits of integration define the time interval during which the input mass flux values must be determined in order to translate input mass into the cell at time τ, and the max functions ensure that only particles that have spent time in the cell at or before time τ are included. Then, the aqueous concentration calculation sums the contributions for all particles:

$$c(\xi, \tilde{x}, \tau) = \frac{\sum\limits_{p \in N_p(\xi)} \int_{\tau_{out}}^{\tau_{in}} \dot{m}(\xi, t) e^{-k(\tau - t)} dt}{N_p(\xi) \theta(\tilde{x}) V(\tilde{x})} \qquad (11)$$

where $N_p(\xi)$ is the total number of particles associated with source location ξ, $\theta(\tilde{x})$ is the volumetric water content at this location, $V(\tilde{x})$ is the volume of the cell, and the summation in the numerator occurs over all particles spending time in the cell. Finally, Eq. 4 is applied by summing over all source zones by repeating Eq.11 for each source location and time varying mass flux $\dot{m}(\xi, t)$ to obtain $C(\tilde{x}, \tau)$.

2.3.4 Convolution-Based Particle Tracking Method: Flux-Averaged Concentration

The calculation of the flux-averaged concentration $\widehat{c}(\xi,\widetilde{x},t)$ using a particle tracking model of $f(\xi,\widetilde{x},t)$ requires that a temporal averaging be performed, analogous to the spatial averaging carried out for the estimation of the resident concentration $\hat{c}(\xi,\widetilde{x},t)$. Contributions to $f(\xi,\widetilde{x},t)$ come in the form of incremental jumps in the output as individual particles leave a computational grid cell or cells at $\widetilde{x}$, rather than as a smooth function that can be integrated readily in Eq. 6. Therefore we require a method that produces the solution in a manner that conserves solute mass globally, but does not introduce artificial smoothing of the results. The potential for artificial smoothing is most relevant for low-dispersivity systems and abrupt changes in $\dot{m}(\xi,t)$ with time, the combination of which might result in a smeared arrival of solute rather than an abrupt rise in outlet concentration.

The most straightforward approach to dealing with these issues is to select a time interval Δt_m over which the flux-averaged concentration is computed, and marching forward in time to compute the breakthrough curve for each time. Each particle exiting a a cell or cells in a given time interval contributes mass to the outlet concentration of the fluid that exits during the time interval, and Eq. 6 defines these contributions. We now derive the relation for approximating Eq. 6 to determine the flux-averaged concentration at a given time. First, we define the following quantities:

t_1 = previous time in the calculation of the breakthrough curve

$t_2 = t_1 + \Delta t_m$ = upcoming time for which the concentration is being computed in the breakthrough curve

The contribution to the integral for the solute mass in the exit fluid from a single particle is given by

$$\int_{\tau_1}^{\tau_2} \dot{m}(\xi,t)e^{-kt_{out}}\,dt \tag{12}$$

where $\tau_1 = \max(0, t_1 - t_{out})$, $\tau_2 = \max(0, t_2 - t_{out})$, and, t_{out} is the residence time of the particle when it leaves the cell at which the flux-averaged concentration is being computed. Then, the flux-averaged concentration is determined as follows:

$$\widehat{c}(\xi,\widetilde{x},\tau) = \frac{\sum\limits_{p\in N_p(\xi)} e^{-kt_{out}} \int_{\tau_1}^{\tau_2} \dot{m}(\xi,t)dt}{N_p(\xi)\dot{q}_e(\widetilde{x})\Delta t_m} \tag{13}$$

where the summation occurs over all particles that exit at $\widetilde{x}$. Conceptually, each particle leaving at $\widetilde{x}$ represents a stream of solute mass that has taken time t_{out} to reach the exit; the integration sums the mass that entered the system during an interval such that after traveling for time t_{out}, it reaches the exit within the time interval from t_1 to t_2. The total mass leaving ($\sum_{p \in N_p(\xi)} e^{-kt_{out}} \int_{\tau_1}^{\tau_2} \dot{m}(\xi,t)dt / N_p(\xi)$) divided by the volume of water leaving ($\dot{q}_e(\tilde{x})\Delta t_m$) equals the flux-averaged concentration.

2.3.5 Extension for Sorption

To include sorption to the medium in the CBPT method assuming a linear, equilibrium sorption model, the basic approach to particle tracking is the same as for conservative solutes, except that the vector A is corrected by dividing by the retardation factor $R_f = 1 + \rho_b K_d / \theta$, where ρ_b is the bulk rock density and K_d is the sorption distribution coefficient (Tompson, 1993). It is generally suggested that particle tracking methods be revised to include sorption within the particle tracking run. However, equivalent results are obtained efficiently by simulating the conservative particle tracking behavior in a manner outlined above, and applying a spatially dependent R_f to adjust the transit time for each stage of the particle trajectory within the algorithm developed herein. There is no restriction on the complexity of the spatial R_f field that can be applied using the method, as long as R_f can be mapped onto the same grid used to specify the particle trajectory paths during the transport simulation. Thus, a single simulation of particle trajectories can therefore be used to simulate both conservative and sorbing simulations, thereby reducing the number of particle tracking runs required in sensitivity studies involving sorption. When determining the aqueous concentration using the convolution and superposition methods outlined here, it must also be recognized the particle tracking solution for $\hat{c}(\xi,\tilde{x},t)$ records the total mass at a cell, both in the fluid and sorbed to the medium. This means that aqueous concentrations for cells with sorption are determined by multiplying the denominator of Eq. 11 by R_f (Kinzelbach, 1988).

For flux-averaged concentration $\hat{c}(\xi,\tilde{x},t)$, particle paths are delayed to account for sorption in the same manner, but division of the denominator of Eq. 13 is not performed because the solute is not sorbing to the medium once it leaves the cell or cells at which the flux-averaged concentration is being computed.

2.3.6 Extension for Matrix Diffusion in Dual Porosity Transport

To incorporate transport in a dual porosity system, Robinson and Bussod (2000) and Arnold et al. (2002) outlined a technique called the residence time transfer function (RTTF) technique for particle tracking models to simulate the retardation associated with diffusion into dead end pore space. The primary porosity, such as the fractures in a fractured porous medium, conducts the fluid, and solute mass can undergo diffusion into the stagnant fluid in the secondary porosity. Models for linear, equilibrium sorption in the primary and secondary porosity are also included. The RTTF technique introduces a probabilistic particle delay that, in the limit of a large number of particles, reproduces the solution for the breakthrough curve for a dual porosity system. Because this model includes only linear transport processes, the convolution and superposition principles introduced in the CBPT method can be applied. As was the case for sorption in a continuum, this method of introducing delays to the particle travel time can either be applied at the time the particle tracking runs are performed, or external to the random walk simulation using particle tracking results for a conservative solute. In PLUMECALC, we assume that the user wishes to conduct the sorption or diffusion calculations in PLUMECALC, to increase the computational efficiency of the modeling process.

In contrast to conservative species or simple sorption in a porous continuum model, there is a complex distribution of solute mass in the primary porosity fluid, the secondary porosity fluid, and sorbed to both media. When dual porosity transport is implemented using probabilistic particle delays, there is no simple way to back out resident concentrations from the simulations. The total solute mass per unit volume of porous media and water is well-defined, but the partitioning is complex because conceptually, there are concentration gradients within the secondary medium. Therefore, to use the CBPT method, the most appropriate concentration to compute is the flux-averaged concentration. These concentrations are well defined for fluid exiting the system or at an internal location, regardless of whether the particle tracking simulation is a single continuum or a dual continuum. At internal cells for which no fluid sink exists, the flux-averaged concentration can be computed by assigning the total fluid volumetric flow rate passing through the cell to the term $\dot{q}_e(\tilde{x})$ in Eq. 13.

2.3.7 Numerical Implementation Details

Particle tracking model requirements: Efficient implementation of the methods outlined above follows directly from Eq. 11 (resident concentration) and Eq. 13 (flux-averaged concentration). Simulation of particle tracking results in a time history of spatial locations and residence time distributions. In addition, for a control-volume finite-difference code, the particle tracking model can easily be made to keep track of the number of the control volume cell in which the particle resides. At times determined by the progress of the simulation, particles shift cells either by advection or during the random-walk dispersion jump. During the particle tracking simulation, information is recorded at each time and cell number when a particle shifts from one cell to the next. For the model developed and implemented in PLUMECALC, the particle tracking model in the flow and transport code FEHM (Zyvoloski et al., 1997) was enhanced to write the information on particle times and cell numbers for each particle.

Simulation of Plume Concentrations: The PLUMECALC code was developed to implement the CBPT method to compute resident and flux-averaged concentrations for an arbitrary number of time-dependent mass flux functions $\dot{m}(\xi,t)$. The code performs the simulation through a numerical implementation of Eq. 11 or Eq. 13 for the convolution part of the model. Sorption is implemented by first correcting the travel times of the individual segments of the conservative particle tracking paths to include the delay due to sorption. For multiple sources, Eq. 4 is implemented. To do this, the code requires particle tracking results for all of the sources, and an indexing array that ties each particle to an input source function $\dot{m}(\xi,t)$. Then, the numerical convolution process consists of looping though each particle, and, using the proper function $\dot{m}(\xi,t)$ for that particle, computing the contribution to the total concentration at a given location and time using Eq. 10 or Eq. 12. The indexing ensures that Eq. 4 is automatically taken care of in the course of the integration. The functions $\dot{m}(\xi,t)$ are implemented as discrete points of time and mass flux in PLUMECALC (although other methods are possible) and the integration is computed by assuming linear interpolation between the points. In addition to the mass flux and particle tracking information, the code requires basic grid geometric information such as cell volumes, and the rock and transport properties of the original flow model on a cell-by-cell basis.

The final issue related to the integration relates to the accurate computation of the integrals associated with each segment of a particle path (Eq. 10 or 12). We require an accurate representation of these integrals for the general case in which first order decay occurs and for situations in which the mass flux term is not constant over the time interval of interest. For a piecewise linear mass flux versus time curve, Eq. 10 can be solved by first defining the mass flux for a linear segment between τ_{out} and τ_{in}:

$$\dot{m}(\xi,t) = \dot{m}(\xi,\tau_{out}) + \beta(t-\tau_{out}) \tag{14}$$

where

$$\beta = \frac{\dot{m}(\xi,\tau_{in}) - \dot{m}(\xi,\tau_{out})}{\tau_{in} - \tau_{out}} \tag{15}$$

Substituting Eq. 14 into Eq. 10 and performing the integration, we obtain the following expression for the integral:

$$\int_{\tau_{out}}^{\tau_{in}} \dot{m}(\xi,t)e^{-k(\tau-t)}dt = \frac{\dot{m}(\xi,\tau_{in}) - \beta/k}{k}e^{-kt_{in}} - \frac{\dot{m}(\xi,\tau_{out}) - \beta/k}{k}e^{-kt_{out}} \tag{16}$$

This expression is valid unless $k = 0$, for which a simpler equation can be derived:

$$\int_{\tau_{out}}^{\tau_{in}} \dot{m}(\xi,t)dt = \frac{(\tau_{in} - \tau_{out})\left[\dot{m}(\xi,\tau_{out}) + \dot{m}(\xi,\tau_{in})\right]}{2} \tag{17}$$

When integrating from τ_{out} to τ_{in}, it is possible that the discretization in the mass flux input curve requires that several segments of the curve be considered. If this is the case, then Eq. (16) or (17) can be carried out for each segment. For the computation of the integral for the flux-averaged concentration (Eq. 14), an expression analogous to Eq. 17 is used. Note that in this case, the decay correction is a constant ($e^{-kt_{out}}$) that can be pulled outside the integral, reflecting the fact that the solute mass represented by a given particle spends a residence time t_{out} in the system.

Transfer Functions for Matrix Diffusion in Dual Porosity Transport: The process for incorporating matrix diffusion in a PLUMECALC simulation is the same as that described previously for the FEHM code (Arnold et al., 2003), so only a brief summary is provided here. To include matrix diffusion, a sub-grid-block model consisting of flow in parallel fractures, with diffusion into stagnant water in the rock matrix pores, is assumed.

Arnold et al. (2003) describe the equations and the analytical solution used for this submodel. In PLUMECALC, the incorporation of matrix diffusion is accomplished by allowing the user to define the relevant diffusion parameters on a cell-by-cell basis, and the code imparts a probabilistic travel time delay for each segment of the particle transport. The analytical solution involves two dimensionless parameters that define the entire range of behavior, from fracture-dominated transport to equivalent continuum behavior when the diffusion times are short compared to advection. The code implements the submodel with transfer functions, which are a series of analytical solutions for different values of the dimensionless parameters. For each segment of the particle path in which diffusion is simulated, the code performs an interpolation to find the appropriate transfer function, and randomly selects the travel time of the particle based on that transfer function. In the limit of a large number of particles, the matrix diffusion system as defined in the submodel is reproduced. Currently, the analytical solution developed in Arnold et al. (2003) is used, but the PLUMECALC code is designed to accommodate different conceptual models for sub-grid-block transport by providing a different set of transfer functions, perhaps accompanied by small changes to the code (depending on the nature of the submodel).

Introduction of Particles in a Flow Model: The underlying particle tracking simulation performed as a precursor to the plume concentration calculation must utilize an appropriate spatial distribution of particles to simulate the solute source accurately. To simulate an input boundary condition of a fluid source of time-varying concentration, we input a uniform spatial distribution of particles within the source region ξ, and determine the equivalent solute input mass flux $\dot{m}(\xi,t)$ using

$$\dot{m}(\xi,t) = \dot{q}_i(\xi)c_{in}(\xi,t) \tag{18}$$

where $\dot{q}_i(\xi)$ is the fluid volumetric flow rate entering the system at source region ξ, and $c_{in}(\xi,t)$ is the time-varying concentration of solute in the source fluid. Alternatively, the input mass flux $\dot{m}(\xi,t)$ can in some applications be defined, and the method simply requires that the particles be distributed uniformly in the region. This region can either be defined as an area or a volume, depending on the application. For example, if a fluid source such as groundwater recharge enters the flow domain of a finite difference model, the most realistic conceptualization is that the fluid injection is occurring on the faces of the grid blocks on the outside of the model domain. For time-dependent release of a solute internal to the model, such as occurs when a contaminant enters the water through dissolution of a solid phase, then the particles can be introduced in a volume defined by the size of the source. Neither case requires that the particles coincide with the numerical grid faces or control volumes, but the concentrations computed once the simulation begins will be affected by the grid, especially close to the source when source regions are small in extent (smaller than a grid cell). These considerations must be examined for the specific application to ensure proper interpretation of the results.

3. USER INFORMATION

3.1 INTRODUCTION

The PLUMECALC application determines resident or flux-averaged concentrations in groundwater flow models using the results from a random-walk particle tracking model simulation. The model assumes that the particle tracking simulation accurately characterizes the transport solution to the Advection-Dispersion equation (ADE) for one or more source locations. These particle tracking results, combined with solute input information such as mass flux input functions, sorption, diffusion, and decay parameters, are used to resolve the concentration within the model system or at fluid exit points.

The code is currently developed to be compatible with the FEHM fluid flow and random-walk particle tracking model. Therefore, the code requires auxiliary input information related to the flow and transport model, such as the grid geometric information. The code implements the Convolution-Based Particle Tracking (CBPT) method: the theory associated with this numerical technique is described in Section 2.

3.2 HOW TO USE THE SOFTWARE

To run plumecalc, the program executable file name and optional command line arguments are entered at the system prompt:

<PROMPT> plumecalc_V2.2 [*ctrl_file err_file*]

Where the first command line argument (*ctrl_file*) is the name of the file that contains the I/O file information (see Section 3.3.2) and the second argument (*err_file*) is the name of the file where run time information/error messages will be output. The code will look for a file named “plumecalc.files” in the current working directory if no command line arguments are input. If the file does not exist, the user will be prompted to enter the name of the *ctrl_file*. If a name for the information/error output file is not entered on the command line, the default name “plumecalc.err” will be used.

3.3 INPUT SPECIFICATION

3.3.1 General information

Throughout this manual, when a keyword is provided as an optional input, it is not designated as a separate group of input in the tables describing the input, but must be placed on a separate line for the code to function properly.

The FEHM file associated with the streamline particle tracking output used by PLUMECALC is the *.sptr2 file, generated using the option to select a reduced set of output for the particle paths. This option in the current version of the FEHM particle tracking module requires that the input for the sptr parameter iprto be assigned a value of –1, -2, or -3. The choice of the value of iprto specifies the output format option:

-1: Formatted output (ASCII format)
-2: Unformatted output
-3: Binary output

Binary output yields the smallest files, which is an important issue with this method, given the large file size needed to represent a simulation with a large number of particles. ASCII output allows the file to be read on the screen, of course. Although binary output of the FEHM particle tracking results reduces the file size significantly, the results may be machine dependent, i.e. binary files written on one system may not be readable on another. The output in the condensed version of the *.sptr2 file is:

The number of particles used in the simulation, followed by

The particle number, time that the particle is leaving a cell (days), and the cell that the particle is leaving, for each travel segment of each particle.

3.3.2 I/O input file: (default name plumecalc.files)

The I/O input file contains the input and output file information. The name of this file is provided to the program on the command line, or if not entered, defaults to plumecalc.files, and the file must be located in the current working directory.

The following is a summary of the I/O file input:

Group number	Input Variable	Type	Definition
1	grid_file	character*100	Name of the FEHM grid file for the particle tracking simulation.
(optional)	file_format	character*12	Keyword “ascii”, “formatted”, or “unformatted” denoting the format used for the FEHM storage coefficient file. If the keyword is omitted, ASCII formatting is assumed.
2	stor_file	character*100	Name of the FEHM storage coefficient file for the particle tracking simulation.
(optional)	keyword	character*4	Keyword “flux” to denote flux values will be read from a FEHM restart file.
(optional)	file_format	character*12	Keyword “ascii”, “formatted”, or “binary” denoting the format used for writing the FEHM restart file. If the keyword is omitted, ASCII formatting is assumed.
(optional)	flux_file	character*100	Name of the input file containing cell fluxes. This input is read from a FEHM restart file that contains steady state fluxes.

Group number	Input Variable	Type	Definition
(optional)	sptr_num	integer	Number of particle tracking output files to be used for the calculations. If not entered the default is 1. Note: sptr_num file formats (optional) and filenames (Group 3) need to be entered.
(optional)	file_format	character*12	Keyword “ascii”, “formatted”, “binary”, or “unformatted” denoting the format used for writing the FEHM particle tracking output. If the keyword is omitted, ASCII formatting is assumed.
3	sptr_file	character*100	Name of the FEHM particle tracking output file to be used in the calculation of plume concentrations
4	rock_file	character*100	Name of the input file containing the rock property information. This input is in the style of the FEHM ‘rock’ macro.
5	sim_file	character*100	Name of the simulation control input file for the plumecalc simulation.
6	output_file	character*100	Name of the output file for the plumecalc simulation.
7	tcurve_file	character*100	Name of the transfer function curve data file.

3.3.3 Input files from the FEHM Simulation

The five files that can be input / used directly from the FEHM flow and particle tracking transport solution are the grid file, the storage coefficient file, the restart file (containing flux data), the transfer function curve data file, and the streamline particle tracking (sptr2) output file. A detailed description of these files can be found in the FEHM Users Manual (Dash, 2003).

Previous versions of PLUMECALC required the use of a structured numerical grid. With the advance of particle tracking simulations on unstructured (OMR) grids this restriction no longer applies. A new option has been added to FEHM to allow the generation of a modified storage coefficient file for OMR grids for use with PLUMECALC. This became necessary to address the modified control volumes used for interpolation of velocities in OMR regions. The file generated by this option contains only the storage file header information and volumetric coefficients needed by PLUMECALC.

The PLUMECALC simulation implicitly adopts all of the input parameters associated with that model run. In addition, there are restrictions in the simulation of the particle tracking model that must be observed in order for the PLUMECALC code to yield meaningful results:

- Steady state flow
- Particle tracking simulations for a conservative solute (sorption and decay are handled within PLUMECALC)
- Particles introduced to the model in a manner that is consistent with the plume calculation being performed (see Section 2 for a discussion on the method for introducing particles into the flow model domain)

3.3.4 Simulation control input file

Input Variable	**Type**	**Definition**
Group 1: n_sources, kdecay		
n_sources	integer	Number of solute sources
kdecay	real*8	First order decay constant for the medium (day^{-1}).
Optional keyword “do”		
dummy2	character*2	Keyword: if ‘do’ is input, then the input for Group 2 is of the “do loop” form. Otherwise, starting points are input (see the remaining input for this group below)
If “do loop” form is chosen, n_sources lines of input of the following parameters are used to assign which particles belong to each mass flux source input. Group 2: start_no(i), end_no(i), step_no(i)		
start_no	integer array, size n_sources	Beginning particle number associated with the current source.
end_no	integer array, size n_sources	Ending particle number associated with the current source
step_no	integer array, size n_sources	Do loop step for assigning particle numbers to the current source
Otherwise, if starting and ending particles are used to assign particles to sources, all “n_sources” values of this parameter are input on a single line. This option assumes contiguous particle numbering such that particles for the first source go from 1 to np1, the second source from np1 + 1 to np2, and so on. Group 2: start_no(i), for i = 1 to n_sources		
start_no	integer array, size n_sources	Beginning particle number associated with each source.
Group 3: column_number(i), for i = 1 to n_sources		
column_number	integer array, size n_sources	In the input files containing the solute mass flux input versus time information, the column_number in which the source is contained. The first column is assumed to be time in days, and is not counted as one of the columns when setting column_number

Input Variable	Type	Definition
Group 4: current_file		
current_file	character*100	File name for each solute mass flux source (n_source lines). If all mass flux information is contained in a single file, repeat this file n_sources times, and use column_number to provide the indexing to the correct column.
Optional keyword “favg”		
conc_string	character*4	Keyword denoting the type of concentration to compute: if ‘favg’ is input, flux-averaged concentration is determined. Omitting this keyword means that resident concentration is computed.
The following are input in the order shown below only if conc_string = ‘favg’ and flux values are not read from a FEHM restart file (note that each cell is treated as a separate zone when fluxes are read from a restart file): nfavgzones water_flux(i), for i = 1, nfavgzones inodes_favg index_favg(i), for i = 1 to inodes_favg inodes_favg and index_favg are input in nfavgzones sets, once for each zone for which flux-averaged concentration is being computed		
nfavgzones	integer	Number of zones of nodes at which the flux-averaged concentration is to be calculated
water_flux	real*8 array, size nfavgzones	Water volumetric flow rate exiting each of the zones for which the flux-averaged concentration is being computed (liters/day)
inodes_favg	integer	Number of nodes contained in the list of nodes associated with this flux_averaged concentration zone
index_favg	integer array, size inodes_favg	List of nodes associated with this flux_averaged concentration zone
Group 5: total_time, n_out_times		
total_time	real*8	Total time of the plume concentration simulation (days)

Input Variable	Type	Definition
n_out_times	real*8	Time step parameter: If >0: abs(n_out_times) is the total number of equally spaced times for which the calculation is performed If < 0: the code uses this value as the time step, with appropriate rounding to ensure equally spaced time steps If = 0: Times at which results are computed are input individually (see optional input below). This option requires input of an integration time interval, delta_time, for flux-averaged concentration.
The following are input after Group 5 only if n_out_times = 0 ntimes, delta_time out_times(i), for i = 1, ntimes		
ntimes	integer	Number of input times at which calculations are to be performed
delta_time	real*8	Time interval (days) for integration when computing flux averaged concentrations (input only if conc_string = 'favg')
out_times	integer array, size ntimes	Array of times (days) at which calculations are to be performed
Group 6: out_string		
out_string	character*4	Keyword denoting the type of nodal output when the code is computing resident concentration: If 'pckd': output is a "packed" output containing concentrations of cells that have particles passing through them (the others are always 0). If 'node': concentrations are output at nodes specified by the following input. If 'tecp' or 'tecn': output uses tecplot style headers and formatting for packed or node output.
The following are input only if out_string = 'node' or 'tecn' noutnodes ioutnode(i), for i = 1 to noutnodes		
noutnodes	integer	Number of nodes at which output concentrations are specified
ioutnode	Integer array, size noutnodes	Array of nodes at which resident concentrations are to be output

3.3.5 Rock properties input file

The rock properties input file contains rock property and diffusion model data. The rock file uses macro input formats similar to those used by the FEHM zone (list and nnum options) and rock macros. See the FEHM user's manual for the format of the zone and rock macros. In the case of PLUMECALC, the rock macro must reside in a separate file that contains an optional zone macro, a rock macro, and an optional diff macro. Comment lines (denoted with the '#" sign) may be entered before or after the macros, but not within the macro data. Also, if zones are specified, they must be specified with the 'list' or 'nnum' techniques, which designate zones based on lists of node coordinates or node numbers for each zone.

Input Variable	Type	Definition
Optional keyword zone		
dummy_string	character*4	Keyword "zone" designating zone information follows. See the FEHM Users Manual for a description of the zone macro input options "list" or "nnum".
The zone macro data may also be input using an optional zone macro data file where the zone keyword is followed by keyword "file" and the name of the zone macro data file.		
dummy_string	character*4	Keyword "file"
zone_file_name	character*100	Name of the zone macro data file.
For the rock macro, the parameters are input using FEHM's ja, jb, jc input format, with a blank line to terminate the macro. Group 1: rock		
dummy_string	character*4	Keyword "rock" designating rock property information follows
Group 2: ja, jb, jc, denr, kdp, ps [vcf]		
ja, jb, jc	integer	Loop indices or zone designation (see FEHM Users Manual)
denr	real*8	Bulk rock density (kg/m^3).
kdp	real*8	Sorption coefficient in the primary porosity. The retardation factor for sorption will be computed using: $rfac = 1 + \frac{\rho_b Kd}{\phi_f}$
ps	real*8	Porosity of the medium (under unsaturated conditions, this is the volumetric water content)
vcf	real*8	Velocity correction factor (optional). If omitted default value is 1. (Ratio of PLUMECALC model porosity to FEHM model porosity.)
For the diff macro, model parameters are input and then assigned to cells using FEHM's ja, jb, jc input format, with a blank line to terminate the input. Group 1: diff [matrix]		

Input Variable	Type	Definition
dummy_string	character*4	Keyword “diff” designating diffusion model information follows
dummy_string	character*5	Optional keyword “matrix” to indicate resident time concentration calculations should be computed using matrix porosity where defined. The default is to use the primary (fracture) porosity.
Group 2: rseed		
rseed	integer	Initial random number seed used by the diffusion model.
Diffusion parameters are entered for each model being defined, terminated by a blank line Group 3: kd, diffmfl, rd_frac, matrix_por, spacing_primary		
kd	real*8	Sorption coefficient in the matrix
diffmfl	real*8	Molecular diffusion coefficient in the rock matrix (m^2/s)
rd_frac	real*8	Retardation factor in the primary medium
matrix_por	real*8	Matrix porosity
spacing_primary	real*8	Length scale in the primary porosity for the diffusion model. If a negative value is input for this parameter, an error function solution is used for the diffusion otherwise transfer function curve data is used to determine diffusion.
Group 4: ja, jb, jc, itrc_diff		
ja, jb, jc	integer	Loop indices or zone designation (see FEHM Users Manual)
itrc_diff	integer	Diffusion model that applies to specified cell

3.3.6 Solute mass flux input files

The input files for the solute source mass flux contain an arbitrary number of lines of individual sets of time (days) and mass flux values (moles/day). The code reads these lines until the end of the file is reached. More than one column of mass flux values can be listed in each line: the user specifies which column is associated with a given source zone with the array column_number in control input file sim_file. The specification of a column does not include the time array, which is the first entry in each line. The times in the file must be monotonically increasing or equal to the previous time. If the latter, the code makes an abrupt change in mass flux from one value to the next at that time. Otherwise, the code performs a linear interpolation to determine the mass flux at some intermediate time.

Input Variable	Type	Definition
Group 1 is repeated for each mass flux input time. Group 1: time_mdot, mdot(i), for i = 1, number of columns of mass flux input		
time_mdot	real*8	Time (days)
mdot	real*8	Mass flux value (moles/day) at specified time

3.4 OUTPUT SPECIFICATION

The output file consists of the following output from the simulation:

Output Variable	Type	Definition
If the resident concentration is calculated and “pckd” output is requested: Group 1: ntimes, n_touched_cells		
ntimes	integer	Number of output times
n_touched_cells	integer	Total number of cells that have any particles traveling through them. These are the only cells where it is possible to have a non-zero concentration.
Group 2: touched_cells(i), for i = 1 , n_touched_cellss		
touched_cells	integer	List of the nodes that have non-zero concentrations
Groups 3 and 4 are repeated ntimes Group 3: current_time		
current_time	real*8	Current simulation time (days).
Group 4: concentration(i), for i = 1 to n_touched_cells		
concentration	real*8	Cell concentration at current time (moles/m^3).
If the resident concentration is calculated and “node” output is requested: Group 1: ntimes, n_touched_cells		
ntimes	integer	Number of output times
n_touched_cells	integer	Total number of cells that have any particles traveling through them. These are the only cells where it is possible to have a non-zero concentration.
Groups 2 and 3 are repeated ntimes, with Group 3 repeated for each output node Group 2: current_time Group 3: ioutnode(i), concentration(ioutnode(i)), for i = 1 to noutnodes		
current_time	real*8	Current simulation time (days).

Output Variable	Type	Definition
ioutnode	integer	Output node number
concentration	real*8	Cell concentration at current time for specified output node (moles/m^3).
If the resident concentration is calculated and “tecp” (pckd output using tecplot format) or “tecn” (node output using tecplot format) output is requested: Header: variables="x","y","z","node","concentration" Groups 1 and 2 are used for the first output time Group 1: zone t="time current_time "		
current_time	real*8	Current simulation time (days).
Group 2: touched_cells(i), x(i), y(i), z(i), concentration(i), for i = 1 to n_touched_cells Or Group 2: ioutnode(i), x(i), y(i), z(i), concentration(i), for i = 1 to noutnodes Groups 3 and 4 are repeated for each subsequent output time (up to ntimes): Group 3: zone t="time current_time , VARSHARELIST = ([1-4]=1)" Group 4: concentration(i), for i = 1 to n_touched_cells Or Group 4: concentration(i), for i = 1 to noutnodes		
current_time	real*8	Current simulation time (days).
touched_cells	integer	Node number
ioutnode	integer	Output node number
x	real*8	X coordinate of node (m)
y	real*8	Y coordinate of node (m)
z	real*8	Z coordinate of node (m)
concentration	real*8	Cell concentration at current time (moles/m^3).
If the flux-averaged concentration is calculated and output zones are specified: Group 1: ntimes, nfavgzones		
ntimes	integer	Number of output times
nfavgzones	integer	Number of zones for which flux averaged concentrations are computed
Group 2 is repeated ntimes Group 2: current_time, cfavg(i), i=1,nfavgzones		
current_time	real*8	Current simulation time (days).

Output Variable	Type	Definition
cfavg	real*8	Flux averaged concentration of the output zone (moles/l).
If the flux-averaged concentration is calculated and fluxes have been read from a restart file each node/cell is treated as a zone and output will use the same format as resident time concentrations using the “tecp” option. The concentration, however, will be output in moles/l.		

3.5 DATA FILES

The following data files are used by plumecalc. All files are formatted unless designated otherwise:

- Main input file for plumecalc. This file contains the input and output file information (see Section 3.3.2).
- FEHM grid file for the particle tracking simulation (see FEHM Users Manual).
- FEHM storage coefficient file for the particle tracking simulation (see FEHM Users Manual). This file may be formatted or unformatted.
- FEHM restart file containing steady-state fluxes used for the particle tracking simulation (see FEHM Users Manual). This file may be formatted or binary.
- FEHM particle tracking output file to be used in the calculation of plume concentrations (see FEHM Users Manual). This file may be formatted, unformatted, or binary.
- Simulation control input file for the plumecalc simulation (see Section 3.3.4).
- Rock property input file (see Section 3.3.5). An optional zone macro data file may be used within the rock property information file for the zone input (see FEHM Users Manual).
- Solute mass flux input files (see Section 3.3.6).
- Transfer function curve data file (see FEHM Users Manual).
- Output file for the plumecalc simulation (see Section 3.4).
- Error conditions and messages output file, plumecalc.err.

3.6 DEFAULTS

The default format for all files associated with the plumecalc application is ASCII (formatted) input or output. All other parameters used by the code must be read from the program input files.

3.7 ERRORS

Error conditions and messages are written to file plumecalc.err. The following errors will result in termination of the program (italicized words represent variable values that are output by the code):

Error Condition	**Error Message**
I/O file error	

File does not exist or may not be opened for reading / writing.	`ERROR opening` *`FILENAME`*
	`STOPPING execution`

File cannot be read as written.	`ERROR reading coefficient storage file`
	`STOPPING execution`

An error was encountered while trying to read flux data from the FEHM restart file.	`ERROR reading flux data`
	`STOPPING execution`

Input Error	

An illegal keyword was input for out_string in Group 6 of the simulation control input file.	`Unrecognized output option:` *`OUT_STRING`*
	`use keyword pckd, tec, or node instead`
	`STOPPING execution`

An illegal value was input for n_out_times in Group 5 of the simulation control file.	`ERROR -` *`n_out_times`* `must be > 0 or < 0 to compute flux averaged concentrations`
	`STOPPING Execution`

Illegal data has been entered in the rock property input file for macro “rock”.	`Fatal error - for array number` *`IARRAY`*
	`macro -` *`MACRO`*
	`Group number -` *`IGROUP`*
	`Something other than a real or integer has been specified`
	`Line number -` *`INUMBER`*
	`Bad input, check this line`

Error found in transfer function curve input data.	`Stopping in svdcmp_new`
	`Fatal error in transfer function`

Wrong number of parameters found in transfer function curve data file.	`Error in particle tracking interp.`
	`STOPPING execution`

Error found in transfer function curve input data.	`Decreasing data found in type curve, stop`
	`point:` *`M`* `conc=` *`CONC`*

Programming error	

Illegal call to initdata2 routine.	`Fatal error, too many real inputs to initdata2`
	`STOPPING Execution`

Illegal call to initdata2 routine.	`Fatal error, too many integer inputs to initdata2`
	`STOPPING Execution`

3.8 HARDWARE/SOFTWARE ENVIRONMENTS

No special hardware features or environments are required by the software. The code will run on Sun Workstations running Solaris 2.9 or higher, PC workstations running Windows XP, or Linux 2.4.21 or higher. Memory requirements depend on the problem being modeled (based on the number of nodes). It is suggested that the system being used have a minimum of 128 MB of memory.

3.9 EXAMPLES

The examples presented here illustrate the use of several of the features of the PLUMECALC application and demonstrate the input required to perform various types of simulations. A simple three-dimensional model with uniform properties and simple flow in the x direction, as illustrated in Figure 1, was used. The domain is discretized to allow the concentration within the domain to be computed. The discretization consists of 175,639 nodes, 101 in the x direction (20 km model length), 37 nodes in the y direction (9.6 km model width), and 47 nodes in the z direction (500 m thickness). Within the region through which the simulated plume travels, the grid spacings are $\Delta x = 200m$, $\Delta y = 100m$, and $\Delta z = 5m$.

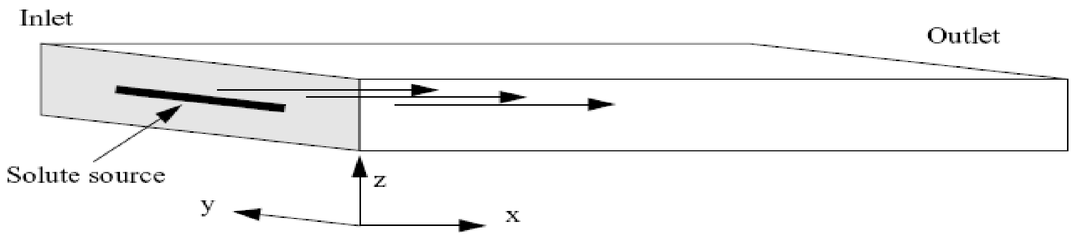

Figure 1. Schematic of the transport test problem used to demonstrate the CBPT method.

Constant head boundaries are applied on the upstream (*x = 0 m*) and downstream (*x = 20 km*) planes, and no flow conditions are assumed on the other sides, resulting in uniform, steady state flow in the x direction. The other medium property of interest is the porosity, $\theta = 0.03361$. The head difference between the two ends of the model is 0.377 MPa and the permeability is set at 10^{-12} m^2 throughout the domain, yielding a pore water velocity of 34.2 *m/y*. Dispersion is modeled with the tensor proposed by Burnett and Frind (1987), implemented in particle tracking using the method outlined by Lichtner et al. (2002). Transport parameter values are longitudinal dispersivity α_L of 100 *m*, transverse horizontal dispersivity α_T^H of 10 *m*, and transverse vertical dispersivity α_T^V of 0.01 *m*. The solute mass is input into the domain at the upstream end in a patch 1000 m wide and 15 m thick, centered in the middle of the plane.

3.9.1 Single-Source Example, Steady State Resident Concentration

In the first simulation, the resident concentration is calculated at several points within the domain at *t =1000 yr*, a time long enough for a steady state plume to be established for a constant mass flux injection. A single source is specified at the inlet with a mass flux such that, for the size of source selected (from the particle tracking run) and the inlet fluid flow rate, equates to a concentration of 1.

The FEHM particle tracking simulation that generated the particle tracks for this simulation (control file control.plume3_fine, main input file plume3_fine.dat) utilized a uniform 317x317 particle distribution (a total of 100489 particles) at the inlet in a two-dimensional patch at *x = 0*, *y = –500 to 500 m*, and *z = –242.5 to –257.5 m*. This patch spans several grid cells in both directions, but since the flux through the patch is uniform for this simple model setup, a single source zone can be used. To simulate an inlet concentration of 1 for the injected fluid in the patch, the fluid flow rate through the region defined by the patch is determined. The concentration (1 mole/liter) is multiplied by the flow rate to obtain the solute mass flux required for this region. In this case, the total water flow rate into the face of the model (147.364 kg/s) is multiplied by the ratio of the area of the patch (1000 x 15) to the total area of the inlet face (9600 x 500). The resulting calculation for this problem yields 39.788 moles/day, which is the value input in file plume3.mdot for this problem. The output is specified to be at 63 individual cells in the model. These node numbers were identified in the original FEHM simulation using the coordinate specification in the node macro to obtain three series of vertical nodes at *x*, *y* values of (4800,0), (9600,0) and (14400,0) for various *z* values ranging from –250 (the center of the plume) to –350.

The input used to perform this simulation is now explained in detail to illustrate some of the subtleties of the method. Files used for this simulation are found in plumecalc.files.fine (renamed plumecalc.files when executing the code). For a complete view of the input files, please check the actual input files supplied with the executable.

Main input file: plumecalc.files.fine:

```
geometry2
unformatted
geometry2.stor
binary
plume3_fine.sptr2
rock.macro
plume3_fine.sim
plume3_fine.output
```

Simulation control input file: plume3_fine.sim

```
1  0.
1
1
plume3.mdot
3.6525e5        1.
node
 63
   13054
   16791
   20528
   24265
   .
   .
   .
   76631
   80368
   84105
   87842
```

Rock property input file: rock.macro

```
rock
   1  0  0  2530. 1. 0.03361
```

Solute mass flux input file: plume3.mdot

```
1. 39.788
1.e7  39.788
```

The downstream concentrations at various *z* values are shown for *x = 4800 m*, *y = 0 m*, *x = 10000 m*, *y = 0 m, and x = 14400 m*, *y = 0 m* in Figure 2. The points produced from PLUMECALC agree closely with the analytical solution results of Leij et al. (1991). Dispersion horizontally and vertically have reduced the concentration to a fraction of its inlet value.

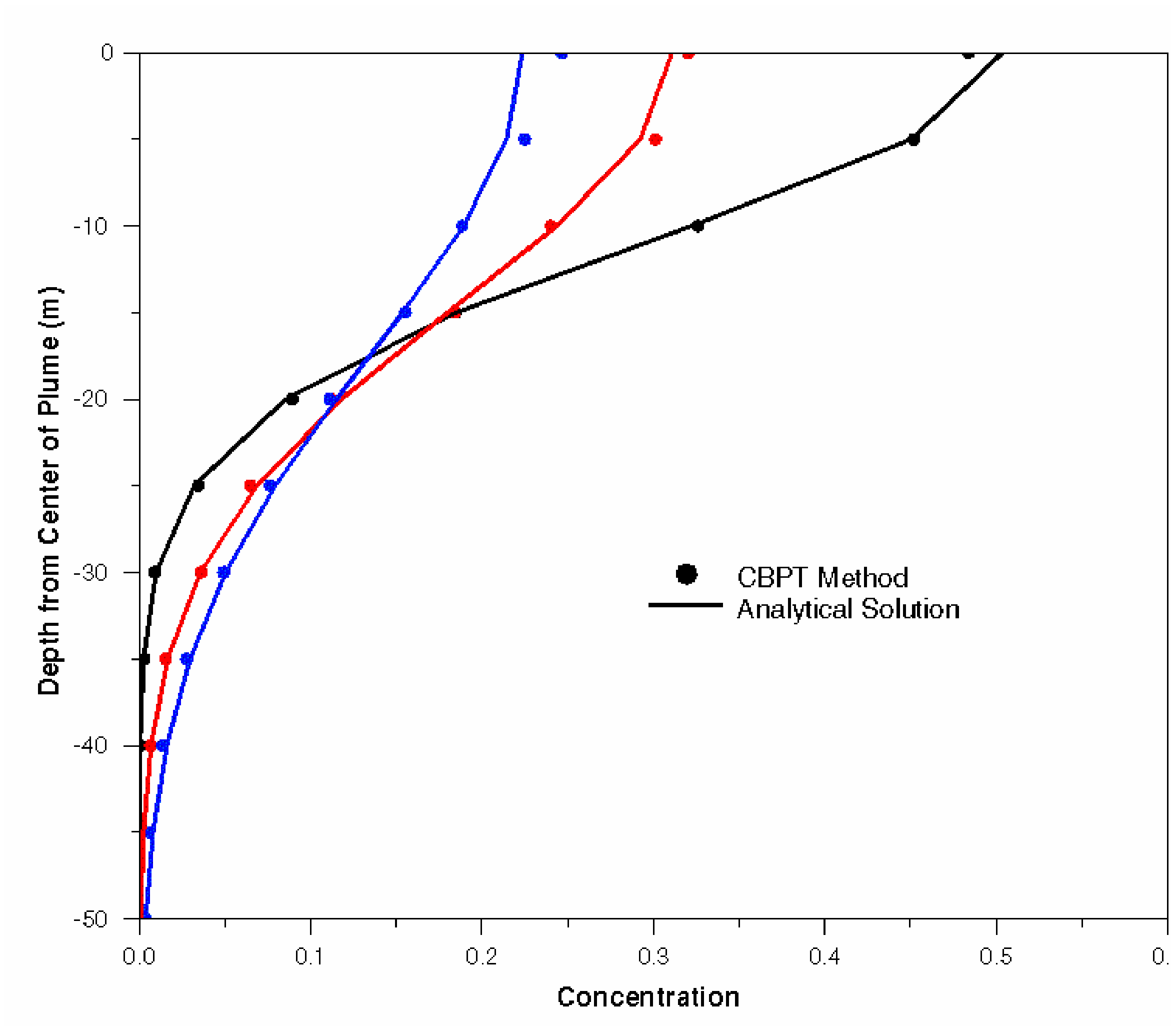

Figure 2. Comparison of the CBPT method, the conventional particle-based method, and the analytical solution of Leij et al. (1991). 10^6 particles are used for each particle-based method.

3.9.2 Single-Source Example, Resident Concentration at a Single Node Versus Time

The next simulations are calculations of the resident concentration at a single node versus time for a conservative solute, a solute that sorbs with a retardation factor of 3, and a solute that decays with a half life of 250 yr ($k = 7.590934\mathrm{x}10^{-6}$ day^{-1}). Figure 3 of shows that the model reproduces the analytical solution well, with the exception of a mismatch at the plateau, caused by the inherent inaccuracy of performing particle tracking simulations. Other realizations would be expected to perhaps overpredict the plateau; a greater number of particles is expected to yield a more accurate solution. As with all particle tracking based studies, simulations investigating the number of particles needed to achieve a convergent result are recommended.

The basic file setup for these three plumecalc runs are similar to the files listed above, with the following exceptions:

plumecalc.files: input and output file names are changed for each run.

```
plumecalc.files.resbtc1    plumecalc.files.resbtc2    plumecalc.files.resbtc1
 geometry2                  geometry2                  geometry2
 unformatted                unformatted                unformatted
 geometry2.stor             geometry2.stor             geometry2.stor
 binary                     binary                     binary
 plume3_fine.sptr2          plume3_fine.sptr2          plume3_fine.sptr2
 rock.macro                 rock_rfac3.macro           rock.macro
 plume3_resbtc1.sim         plume3_resbtc2.sim         plume3_resbtc3.sim
 plume3_resbtc1.output      plume3_resbtc2.output      plume3_resbtc3.output
```

Simulation file for the conservative, sorbing, and decaying solute runs:

```
plume3_resbtc1.sim         plume3_resbtc2.sim         plume3_resbtc3.sim
   1  0.                      1  0.                      1  7.590934e-6
   1                          1                          1
   1                          1                          1
   plume3.mdot                plume3.mdot                plume3.mdot
   3.6525e5        0          3.6525e5        0          3.6525e5        0
   46                         46                         46
      127838                     383512                     127838
      131490                     394470                     131490
      135142                     405428                     135142
      138795                     416385                     138795
      .                          .                          .
      .                          .                          .
      .                          .                          .
      281242                     843728                     281242
      284895                     854685                     284895
      288548                     865642                     288548
      292200                     876600                     292200
   node                       node                       node
   1                          1                          1
     87842                      87842                      87842
```

Rock property input file rock_rfac3.macro

```
rock
   1  0  0  2530. 3. 0.03361
```

In this case, output at a single node (number 87842) is requested at a series of times listed using the n_out_times = 0 option. Alternatively, a certain number of equally spaced times could have been chosen. However, by selecting only times during which the breakthrough at this location is non-zero (from knowledge of the problem, 46 such times were selected between 127838 and 292200 days for the conservative and decaying solutes and between 383512 and 876600 days for the sorbing solute), a minimum set of calculations are required. This brings up an aspect of the calculation of resident concentration that is important: the calculation at a particular time does not require the solution at previous times, as in a conventional finite difference solution that marches forward in time. Therefore, selection of the time(s) of the simulation can be made to maximize the efficiency.

For the decay simulation (plume3_resbtc3.sim) the only difference from the above listed simulation input file is the setting of the decay constant to 7.590934e-6 to incorporate decay.

For the sorption simulation (plume3_resbtc2.sim), the macro with rock properties is changed to rock_rfac3.macro, which sets a retardation of 3 everywhere in the model. The other difference is in the series of times chosen for output. These were increased by a factor of three in plume3_resbtc2.sim compared to plume3_resbtc1.sim so that times at which the mass is being computed contains the breakthrough curve.

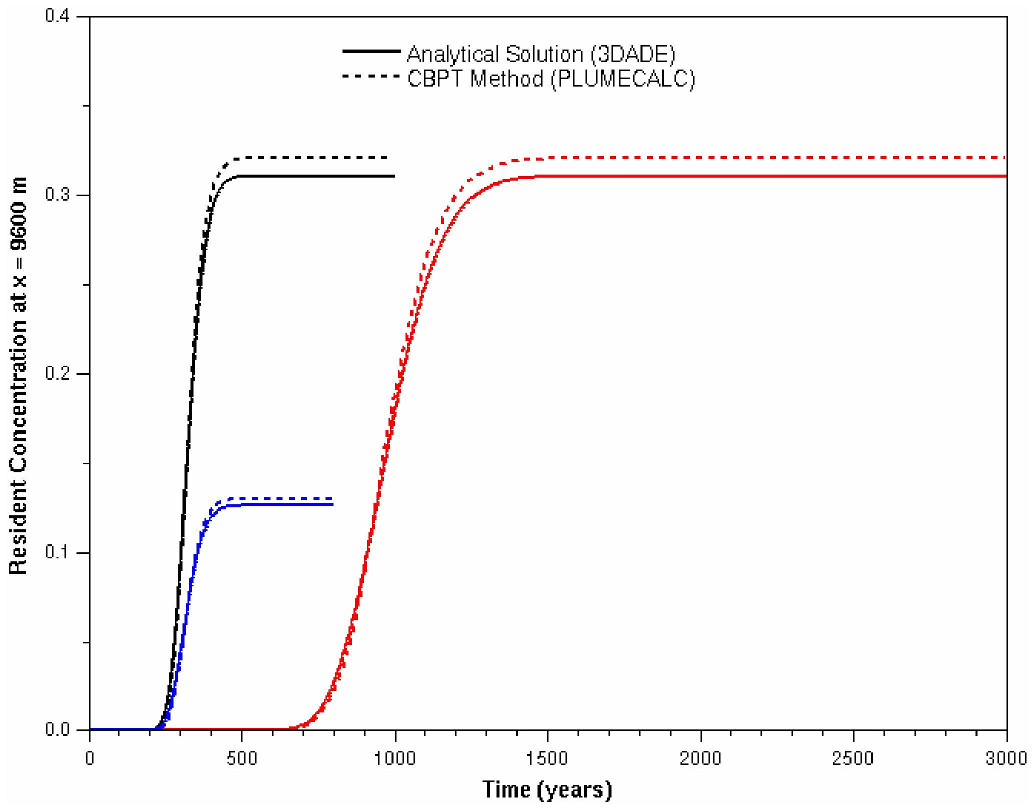

Figure 3. Comparison of the CBPT method and the Leij et al. (1991) analytical solution for resident concentration versus time at x = 9600 m, y = 0 m, z = -250 m. Conservative, sorbing (R_f = 3), and decaying ($t_{1/2}$ = 250 yr) solutes are simulated in response to a constant injection concentration.

3.9.3 Multiple Source Example, Calculation of Flux-Averaged Concentration

In this example, a complex multiple source term function for a model similar to that developed in the test problem of the previous section is applied. This example assumes that to accurately characterize the time varying mass flux input at the upstream end of the model, two adjacent source terms are needed. The center of the source region is located in the same place as in the previous simulations (*y = 0 m, z = -250 m*), and the width is still *1000 m*, but the height is assumed to be *100 m*, with an upper and lower region of *50 m* height each. Figure 4 shows the input solute mass flux versus time for each source. The upper source is a constant input mass flux for *200 yr*, followed by a reduction to 0 thereafter. The lower source decays exponentially from an initial value of *30 moles/yr* with a drop to one half its current value every *100 years*. Because these sources are located at different spatial locations, the resulting solute plume will have a complex structure governed by the rate of transport through the model.

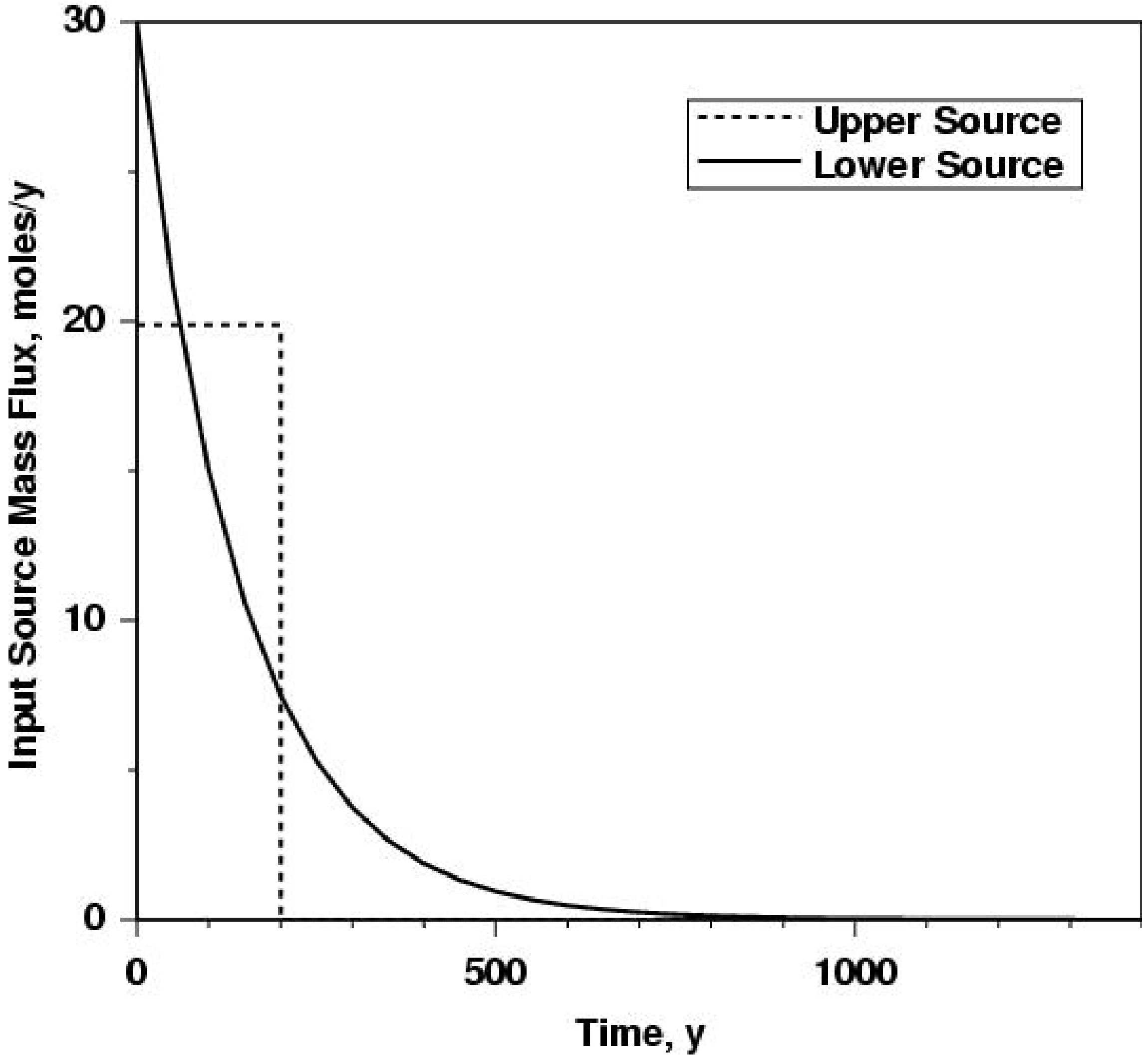

Figure 4. Input mass flux versus time for the two solute sources in the multiple source example problem.

Isoconcentration plots for a conservative, non-decaying solute at various times during the simulation are shown in Figure 5 (plots utilize a 10X vertical exaggeration to aid in the visualization). Figures 6a-e show the plume isoconcentration surface for C=0.0034 moles/l at 200, 400, 600, 800, and 1000 yr. For reference, the mean travel time to reach the exit plane is about 694 yr. At 200 yr (Figure 5a), both sources have injected mass continuously, and the plume has progressed roughly one third of the way through the model. The plots at 400 and 600 years (Figures 5b and c) exhibit a surface that is influenced by the continued injection of mass in the lower source zone, but without additional injection in the upper zone. Therefore, the plume near the source region is narrower in the vertical direction because the upper source has been turned off. At 800 and 1000 years (Figure 5d and e), the majority of the initially injected mass has left the system. Nevertheless, the continued injection of mass into the lower source region at much lower levels means that a plume of lower concentration persists at long times.

Figures 5f-h, which plot the isoconcentration surface of lower value (C = 0.00015 moles/l), show that the method can accurately reproduce this portion of the plume evolution at the tail end of a simulation that previously predicted much higher concentrations. This feature of the CBPT method is very difficult to reproduce using a conventional method using particles.

This example simulation, shown in Figure 6 is the flux-averaged concentration determined at the exit plane of the model, assuming that the solute mixes with the entire fluid flow rate leaving the model. Both a conservative solute and a solute that undergoes sorption and decay are simulated. The concentrations are presented using a log axis to illustrate that the method is capable of accurately reproducing a wide range of concentrations in the source term during the course of a simulation. The behavior after about 1000 yr is essentially log-linear, reflecting the influence of only the lower source (the mass input in the upper source is completely through the model after this time), which decays exponentially.

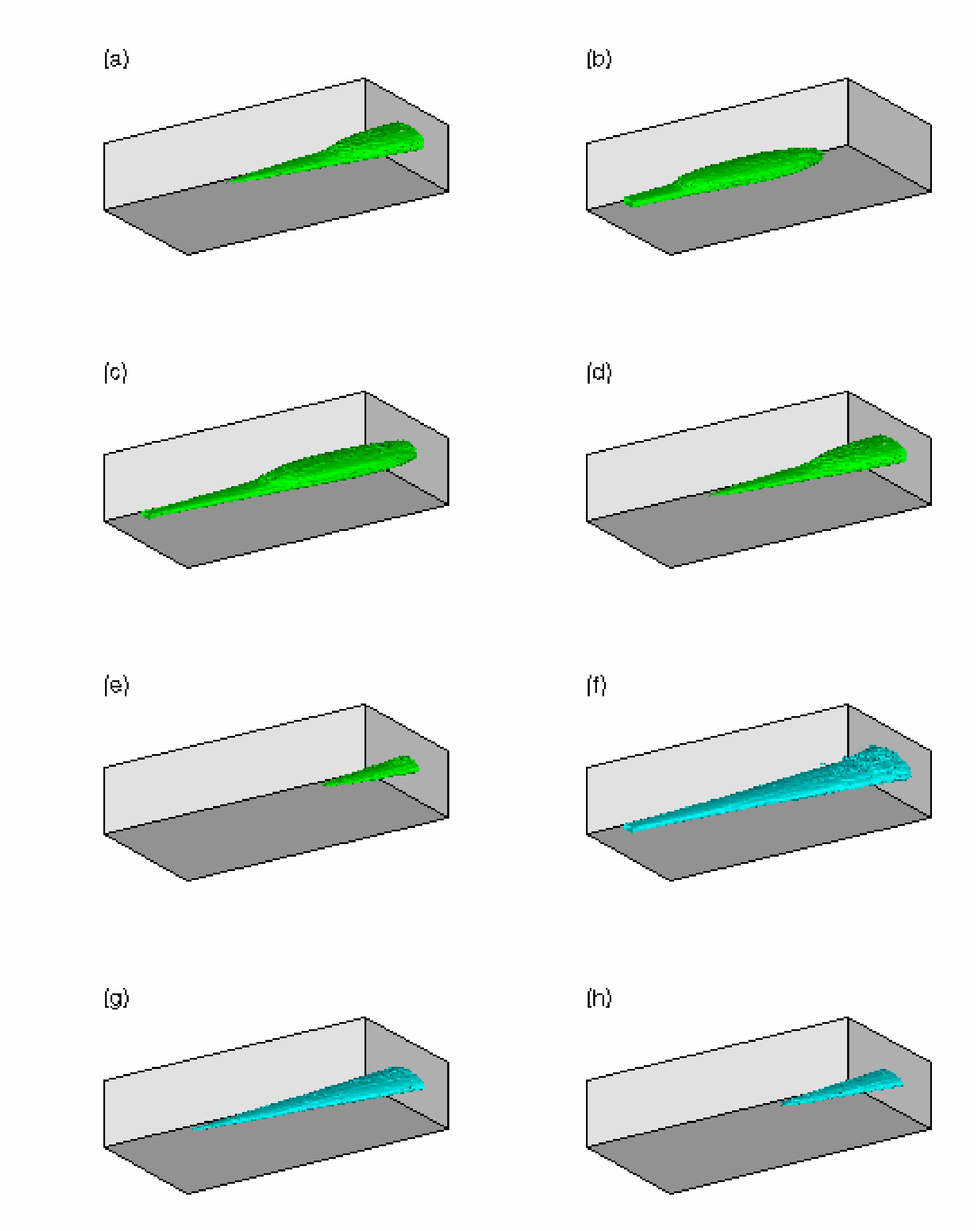

Figure 5. Isoconcentration surfaces predicted using the CBPT method for the multiple source example problem (10X vertical exaggeration). Surfaces in a-e (green) are for C=0.0034 moles/l, and surfaces f-h (blue) are for C=0.00015 moles/l. a) t=200 yr; b) t=400 yr; c) t=600 yr; d) t=800 yr; e) t=1000 yr, C=0.0034; f) t=1000 yr, C=0.00015; g) t=1200 yr, h) t=1400 yr.

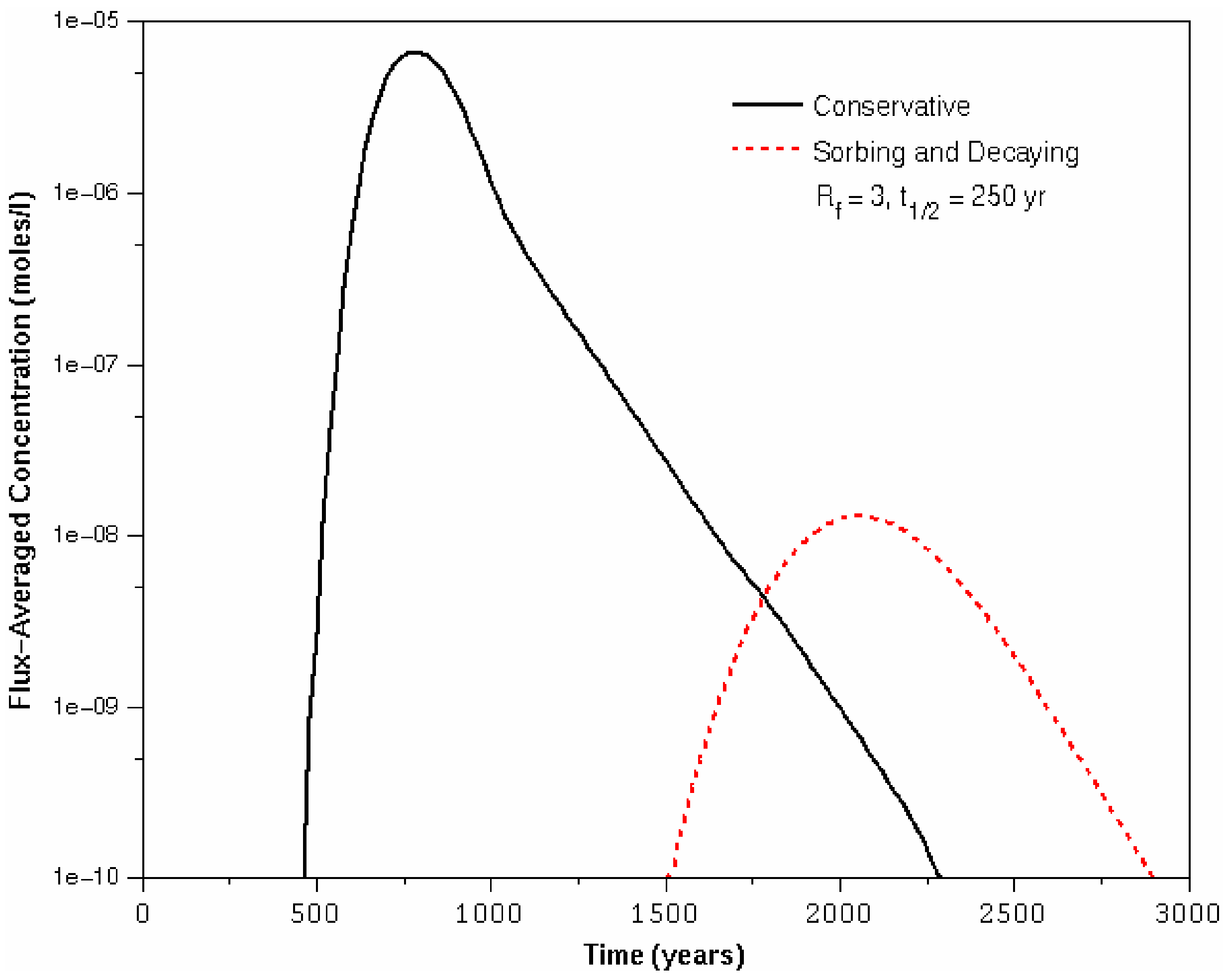

Figure 6. Flux-averaged concentration versus time at the exit of the model for the multiple source example problem. A conservative solute and a solute that sorbs and decays ($R_f = 3$, $t_{1/2} = 250$ yr) are simulated.

The input files for this simulation are discussed below:

plumecalc.files (plumecalc.files.example)

plumecalc.files.example

```
   geometry2
   geometry2.stor
   bin
   example_plume.sptr2
   rock.macro
   example2_favg.sim
   example2_favg.output
```

plumecalc.files.examplerfd

```
   geometry2
   geometry2.stor
   bin
   example_plume.sptr2
   rock_rfac3.macro
   example2_favgrfd.sim
   example2_favgrfd.output
```

Simulation file: example2_favg.sim

```
2  0.
1       50087
1       1
```

```
example1_source1.mdot
example1_source2.mdot
favg
1
1.2732e7
1739
     101      202      303      404      505      606      707      808
909    1010
.
.
.
  174831   174932   175033   175134   175235   175336   175437   175538
175639
1.09575e6       150
pckd
```

Solute mass flux input files:

example1_source1.mdot

```
0.       19.87482
7.305e4  19.87482
7.305e4  0.
1.e7     0.
```

example1_source2.mdot

```
0     30
18262.5  21.2132
36525    15
54787.5  10.6066
73050    7.49996
91312.5  5.30326
109575   3.74997
127838   2.65162
146100   1.87498
164362   1.32581
182625   0.937487
200888   0.662902
219150   0.468742
237412   0.33145
255675   0.23437
273938   0.165725
292200   0.117185
310462   0.0828621
328725   0.0585923
346988   0.0414309
365250   0.0292961
438300   0.00732397
511350   0.00183098
600000   0.
1095750. 0.
```

This example invokes the multiple source option (2 sources), and specifies that the particles numbered 1 to 50086 belong to the first zone, and 50087 to 100489 belong to the second zone. The mass flux curves for the two zones are in example1_source1.mdot and example1_source2.mdot, respectively. The input follows the format specified in Section 3.3.6 above. The keyword ‘favg’ denotes a flux-averaged concentration calculation at a single zone, in this case the entire outlet plane. The flow rate leaving that plane, 1.2732e7 liters/day, is obtained from the FEHM flow simulation result. This plane contains 1739 nodes (some of which are listed above – all are listed in the actual input file, of course). These nodes were copied from the FEHM .chk file that lists all nodes associated with that zone, since it was defined in that model run so that the outlet boundary condition could be applied. After the listing of all nodes, the total simulation time is set, and the calculation is specified to consist of 150 time steps. In contrast to the calculation of resident concentration, time steps starting from t = 0 are advisable, and in fact the current version of the code requires it. The reason for this restriction is to ensure that the global mass balance is maintained – although the calculation would be accurate even if it started at some arbitrary time and used uneven time steps, performing a calculation from time 0 ensures this.

For the simulation with decay and sorption, example2_favgrfd.sim differs from the file listed above by incorporating the decay constant of 7.590934e-6 day^{-1}, and the rock macro file rock_rfac3.macro is used in plumecalc.files, rather than rock.macro.

4. INSTALLATION

4.1 INSTALLATION AND INSTALLATION VERIFICATION

4.1.1 Installation

Obtain the distribution media and/or archive file for the target platform. The archive will be made available to the user via electronic distribution. Create a directory in which to install the executable and associated input files. Move the *plumecalc* archive from the distribution media or directory to the destination installation directory.

4.1.2 Installation Verification

To verify the installation of plumecalc, execute plumecalc using input from example problem 1. Copy plumecalc.files.fine to plumecalc.files and execute the code. The output file, plume3_fine.out should contain the following output (using the node option, see Section 3.4). The concentration values may vary slightly depending on which platform the test was executed:

```
1 49955
365250.0
13054 0.0E+0
16791 0.0E+0
20528 0.0E+0
24265 0.0E+0
28002 0.0E+0
31739 0.0E+0
35476 0.0E+0
39213 0.0E+0
42950 0.0E+0
46687 0.0E+0
50424 0.0E+0
54161 0.0E+0
57898 3.222469618299163E-4
61635 2.9356705899396076E-3
65372 9.048939533971727E-3
69109 0.03456926553741998
```

```
72846 0.08936625393911019
76583 0.18456763798707948
80320 0.32599427161462585
84057 0.4520755077007287
87794 0.48375502457454316
13078 0.0E+0
16815 0.0E+0
20552 0.0E+0
24289 0.0E+0
28026 0.0E+0
31763 0.0E+0
35500 0.0E+0
39237 0.0E+0
42974 0.0E+0
46711 4.21692944812662E-4
50448 4.1950767816114026E-4
54185 1.8298209562923855E-3
57922 6.704049246328586E-3
61659 0.0154177901162852
65396 0.03615280507601377
69133 0.06517587156154586
72870 0.11159597389317874
76607 0.18427023366423864
80344 0.2401263428683089
84081 0.30118656374729646
87818 0.3203243488064638
13102 0.0E+0
16839 0.0E+0
20576 0.0E+0
24313 0.0E+0
28050 0.0E+0
31787 0.0E+0
35524 1.8264030258654615E-4
39261 3.073595373027541E-4
42998 4.7951556467601766E-4
46735 1.3164482333844508E-3
50472 3.638514819292967E-3
54209 5.73528391985197E-3
57946 0.013737203207546686
61683 0.02782264356451133
65420 0.04945847869263525
69157 0.07645340926806499
72894 0.11119902046740071
76631 0.155146303090421
80368 0.18854192333222177
84105 0.22485665252780964
87842 0.24661457538016473
```

4.2 VALIDATION TESTS

The example problems presented in Sections 3.9.1 and 3.9.2 also serve as validation tests demonstrating the ability of PLUMECALC to compute flux-averaged and resident concentrations. In addition, tests of PLUMECALC simulations with diffusion and computing on an unstructured grid are included below.

4.2.1 Tests of the Diffusion Model

The PLUMECALC validation tests for diffusion are based on test case 2.23.4.1 of the Validation Test Plan (VTP) for the FEHM Application Version 2.21 (Dash, 2003). This is the same model used for the example tests described in Section 3.9, however the grid has less resolution in x and y and fixed grid spacing. The discretization consists of 128,775 nodes, 51 in the x direction (20 km model length), 25 nodes in the y direction (9.6 km model width), and 101 nodes in the z direction (500 m thickness). The grid spacings are, $\Delta x = \Delta y = 400m$ and $\Delta z = 5m$. Transport properties are the same as for the examples in Sections 3.9.1 and 3.9.2. The solute mass is input into the domain at the upstream end in a patch 3000 m wide and 12.5 m thick, centered in the middle of the plane.

Two particle tracking output files are generated for the plumecalc runs: sptr_long3ndd.sptr2, and sptr_long3dsp.sptr2. The FEHM simulation that generated particle tracking output files sptr_long3ndd.sptr2 and sptr_long3dsp.sptr2 were generated using no dispersion and a longitudinal dispersion (α_{long}) of 500 m, respectively. For the tests breakthrough was computed for a single node at x = 15200 m. It should be noted that FEHM records particle times for breakthrough when a particle first enters a cell while plumecalc records times when the particle leaves the cell thus breakthrough was recorded for node 3201 in FEHM while plumecalc used node 3200.

Figure 8 shows the comparison of plumecalc generated breakthrough curves for the case of no dispersion (sptr_long3ndd.sptr2) with and without diffusion and Figure 9 for the case with longitudinal dispersion (sptr_long3dsp.sptr2) with and without diffusion. These tests illustrate the results from using different diffusion model options – error function solution (infinite fracture spacing) or transfer function curve interpolation. It should be noted that although transfer function curves of two different formats were used, both sets of transfer function curves use the Sudicky & Frind diffusion model. For the regular spaced Sudicky & Frind transfer function curves, diffusion values are determined using a 4-point interpolation. For the “free” format curves a 3-point interpolation is used.

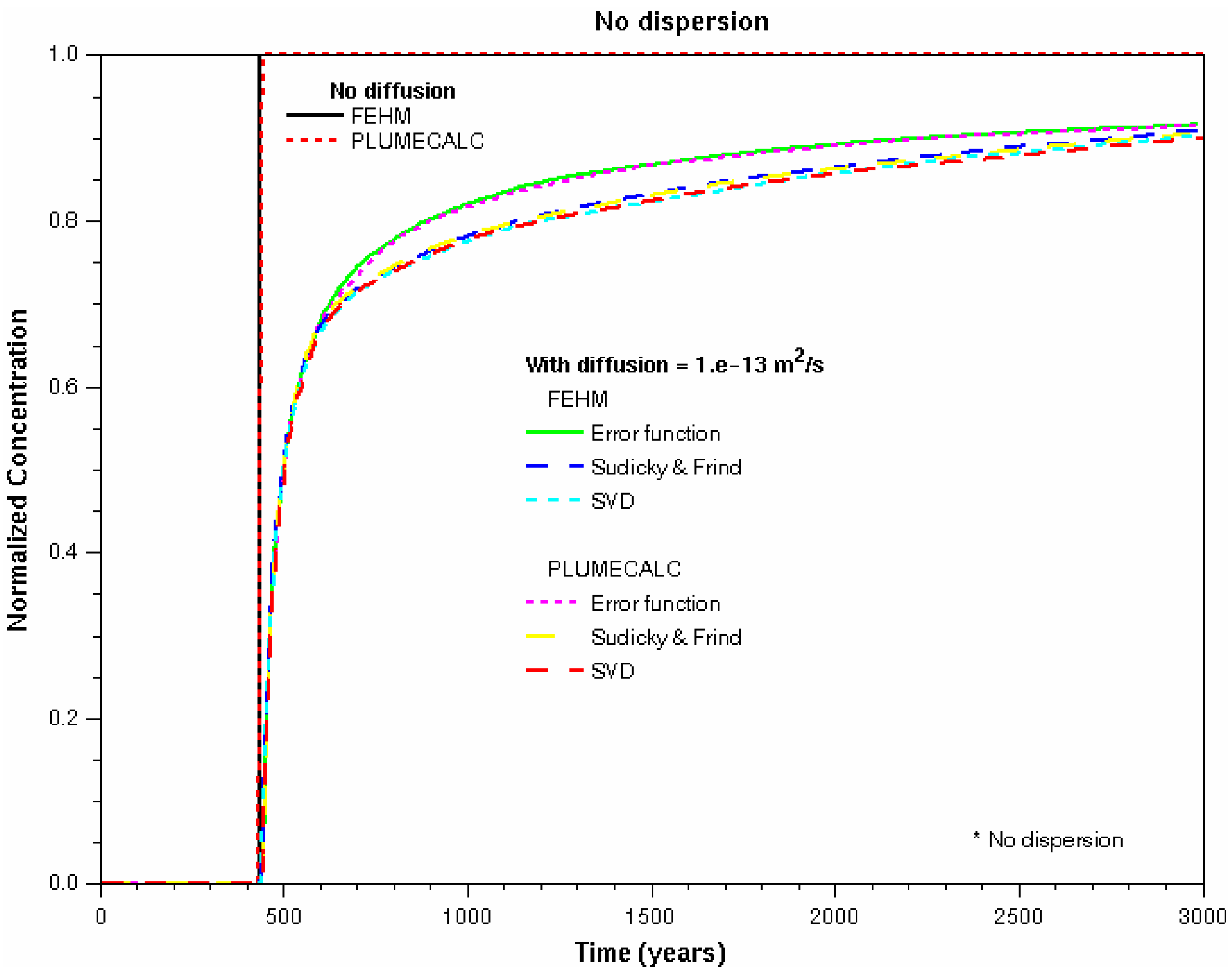

Figure 8. Comparison of plumecalc simulation of breakthrough with output from FEHM for the case of no dispersion.

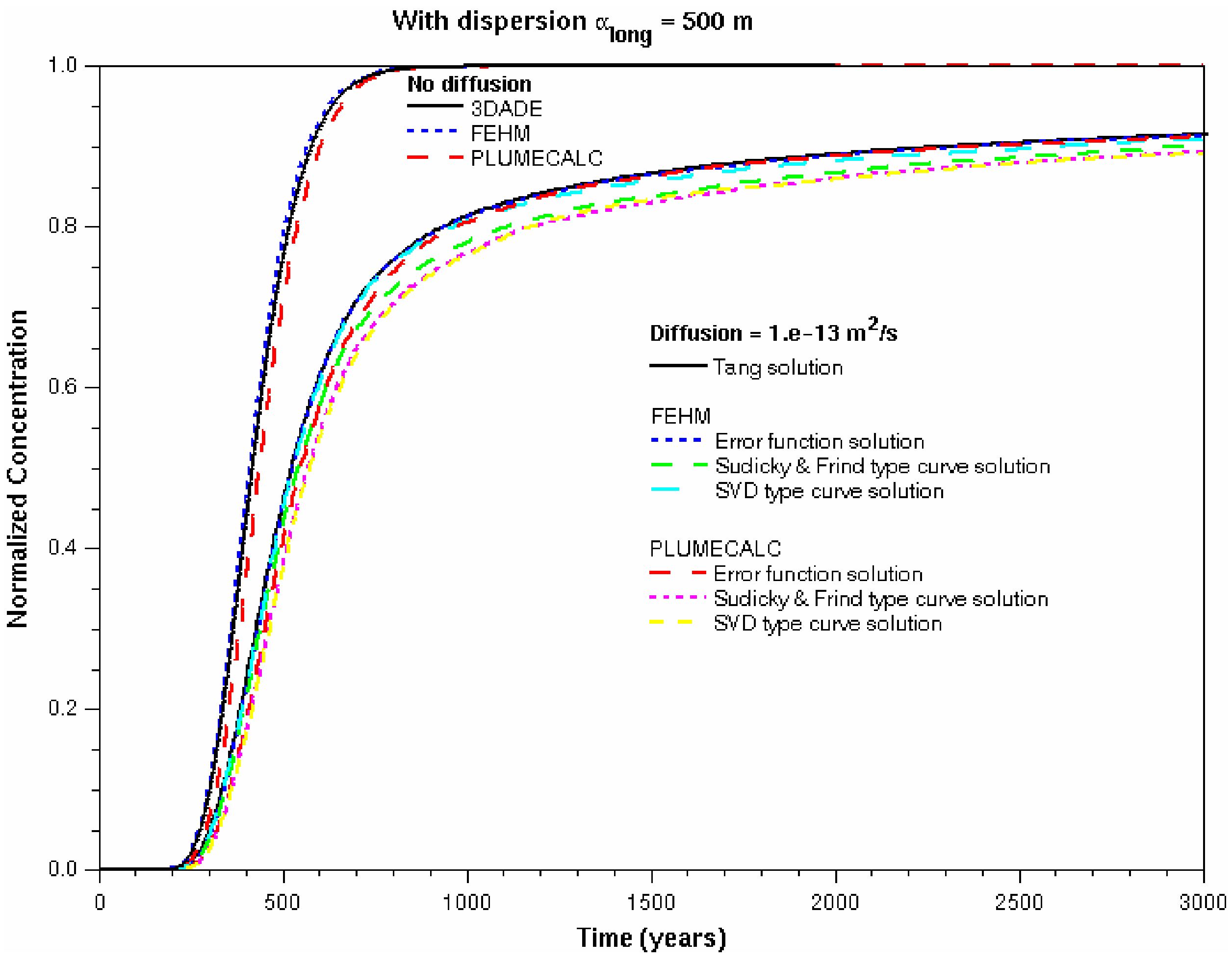

Figure 9. Comparison of plumecalc simulation of breakthrough with output from FEHM. Longitudinal dispersion (α_{long}) was set to 500 m. For the case without diffusion the 3DADE analytical solution is shown and for the case with diffusion the Tang analytical solution is shown.

4.2.2 Test of PLUMECALC with an unstructured grid

To test the functioning of PLUMECALC when using an unstructured grid, a model using a cube 8800 m x 8800 m x 8800 m was developed. Two grids were used for this test, a regular grid and one that used OMR refinement. The discretization of the regular grid consists of 1728 nodes, with uniform grid spacing, $\Delta x = \Delta y = \Delta z = 800m$. The OMR grid refinement added 1299 nodes, for a total of 3027 nodes, which consisted of adding a plane of refinement from $z = 4000$ to 4800 m, with 400m spacing. Constant head boundaries are applied on the top (*z =8800 m*) and bottom (*z = 0 m*) planes, and no flow conditions are assumed on the other sides, resulting in uniform, steady state flow in the z direction (perpendicular to the plane of refinement in the OMR grid). The solute mass is input into the domain at the top surface of the model and distributed uniformly over the entire surface. PLUMECALC was run to produce steady-state concentrations. The setup of the simulation should result in equal concentrations over the entire domain.

In FEHM to compute particle tracks on an OMR grid, the control volumes associated with OMR nodes are adjusted to compute the velocities associated with the cells. The semi-analytical particle tracking solution that is used in FEHM requires brick shaped control volumes and in the OMR regions of the grid this condition is not satisfied. Thus FEHM assigns approximate brick-shaped control volumes to each OMR node. In Figure 10 – 15, contours of concentration are shown for a horizontal plane at 4000m and a vertical region from x = 4000 – 4800 m. Figures 10 and 13 show that for the regular grid uniform concentrations are achieved, as is expected for this simple test problem. Figures 11 and 14 illustrate the problem of using the "non"-brick shaped control volumes when computing the concentrations. Figures 12 and 15 show the improved performance for the OMR grid when the modified control volumes are used. Except for edge effects at the bottoms of these plots, uniform concentrations are obtained within the computational grid (the desired result) for the OMR grids. This test case demonstrates that if the revised storage information is obtained from FEHM at the time the particle tracking simulation is performed (see Section 3.3.3), good results should be obtained on these grids.

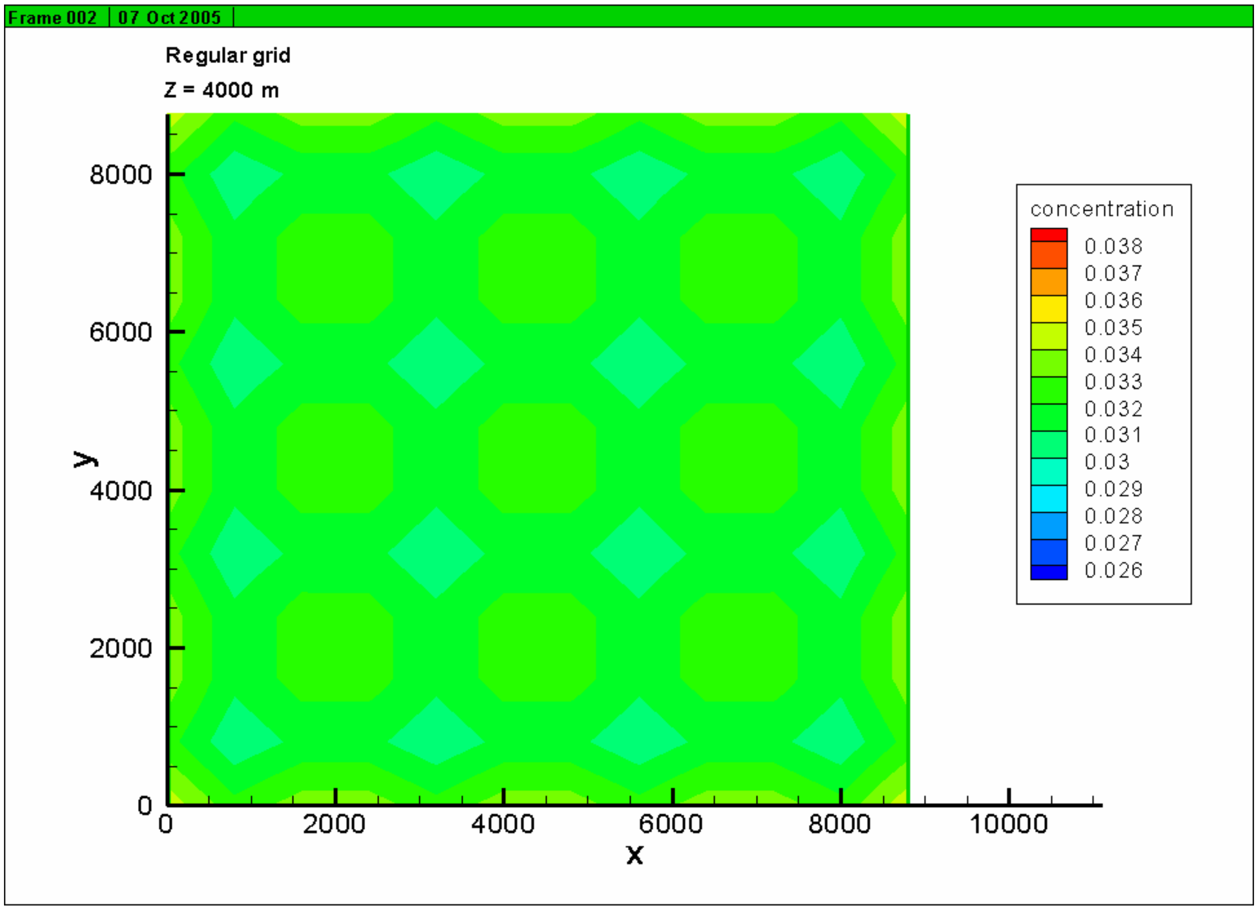

Figure 10.

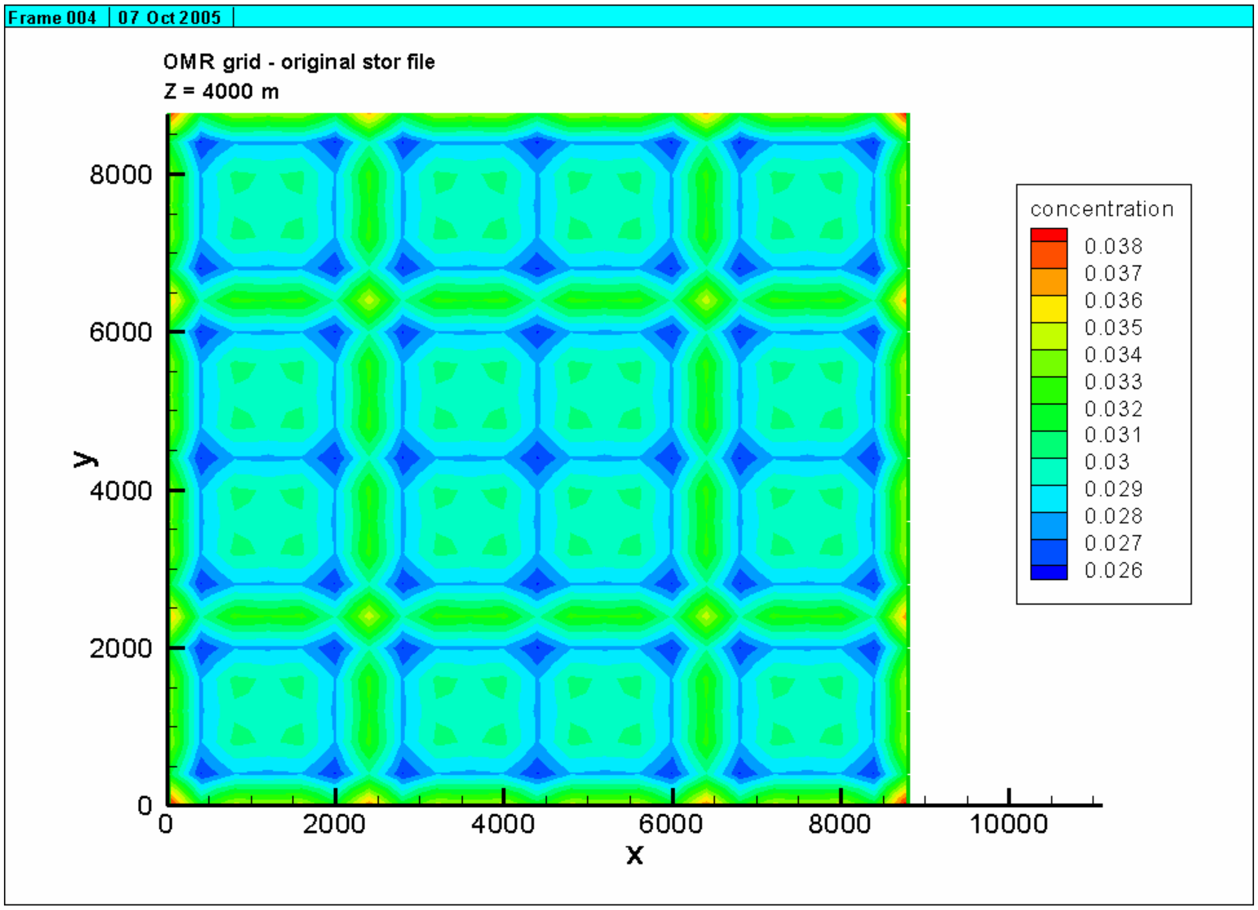
Frame 004
07 Oct 2005
OMR grid - original stor file
Z = 4000 m
concentration
0.038
0.037
0.036
0.035
0.034
0.033
0.032
0.031
0.03
0.029
0.028
0.027
0.026
y
x
0
2000
4000
6000
8000
10000

Figure 11.

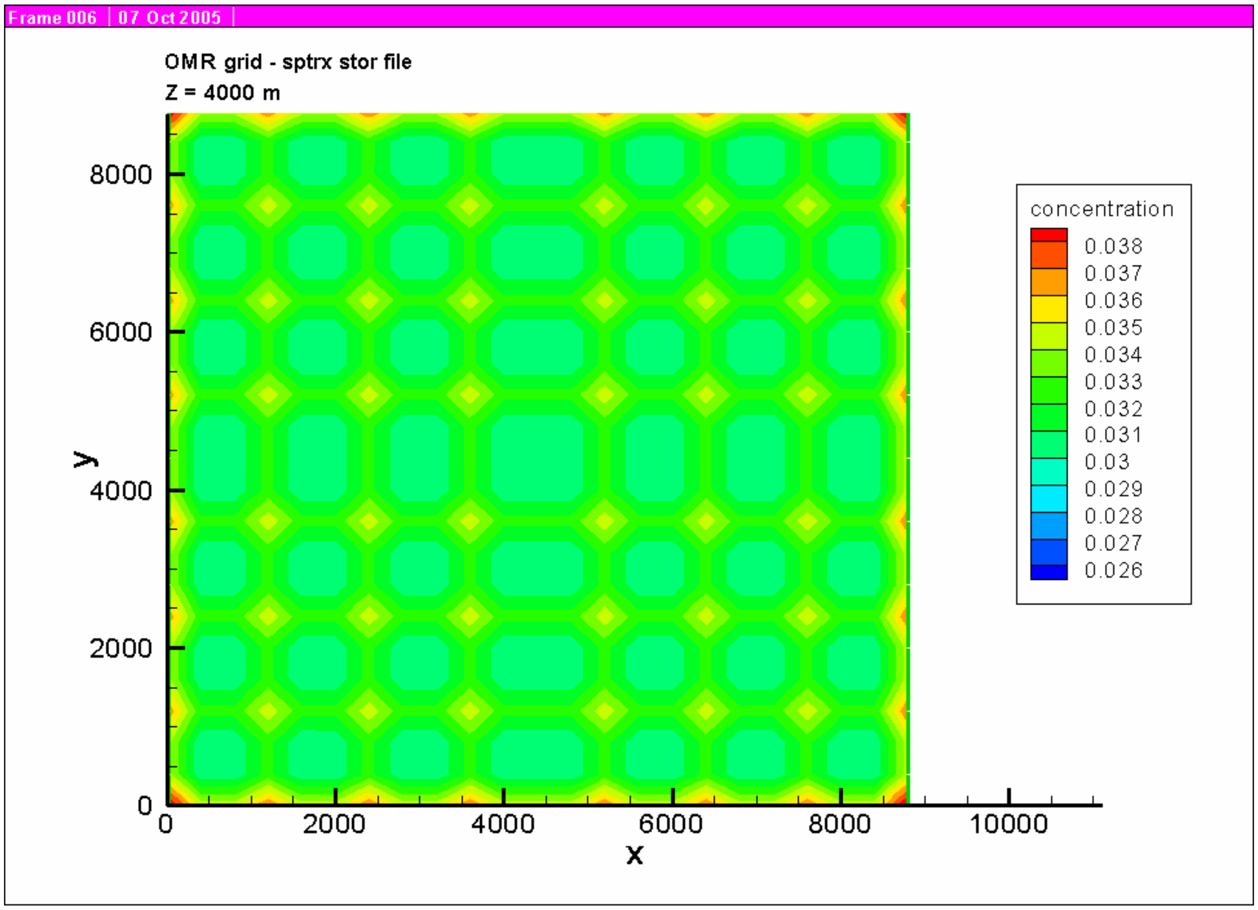
Frame 006 | 07 Oct 2005 |
OMR grid - sptrx stor file
Z = 4000 m
concentration
0.038
0.037
0.036
0.035
0.034
0.033
0.032
0.031
0.03
0.029
0.028
0.027
0.026
y
0
2000
4000
6000
8000
x
0
2000
4000
6000
8000
10000

Figure 12.

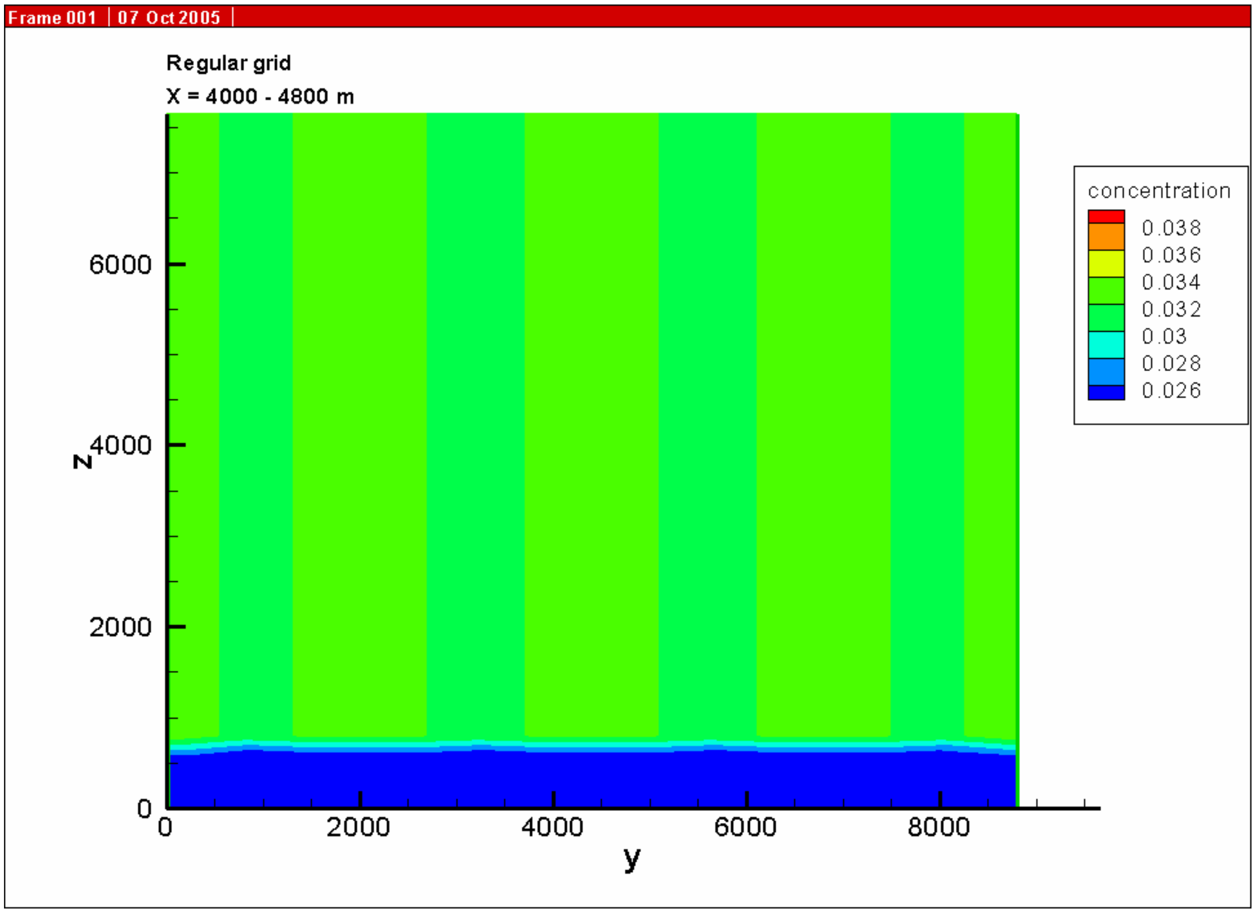
Frame 001 | 07 Oct 2005
Regular grid
X = 4000 - 4800 m
concentration
0.038
0.036
0.034
0.032
0.03
0.028
0.026
z
y
0
2000
4000
6000
8000

Figure 13.

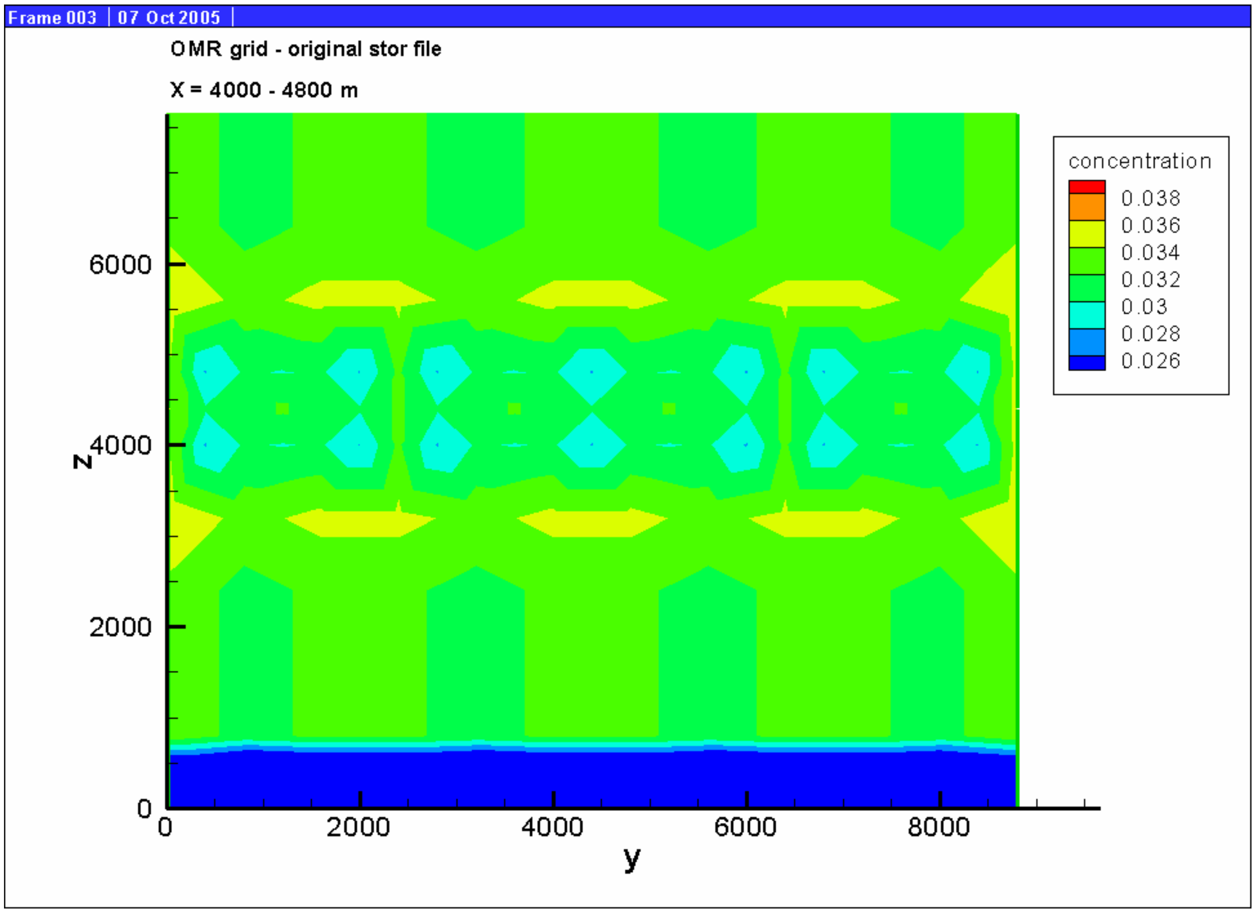
Frame 003 | 07 Oct 2005 |
OMR grid - original stor file
X = 4000 - 4800 m
concentration
0.038
0.036
0.034
0.032
0.03
0.028
0.026
z
y
0
2000
4000
6000
8000

Figure 14.

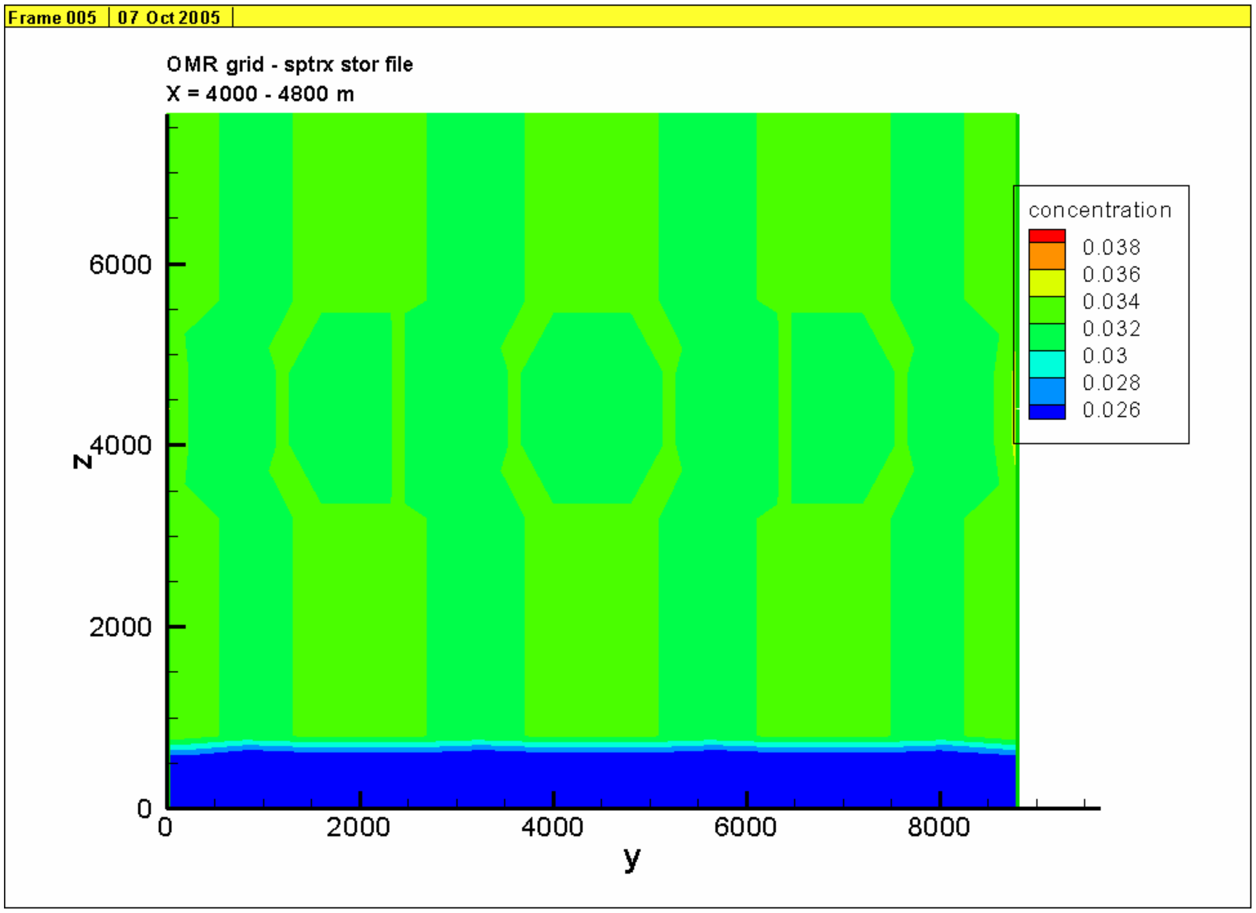

Figure 15.

5. REFERENCES

Arnold, B. W., S. P. Kuzio, and B. A. Robinson, 2003. “Radionuclide transport simulation and uncertainty analyses with the saturated-zone site-scale model at Yucca Mountain, Nevada.” *J. Contam. Hydrology* **62 –63**, 401-419.

Bagtzoglou, A. C., A. F. B. Tompson, and D. E. Dougherty, 1992. “Projection Functions for Particle-Grid Methods.” *Numerical Methods for Partial Differential Equations*, **8**, 325-340.

Burnett, R.D. and Frind, E.O. 1987. “Simulation of Contaminant Transport in Three Dimensions: 2. Dimensionality Effects.” *Water Resources Research*, **23** (4), 695–705.

Chiang, C.Y.; Wheeler, M.F.; and Bedient, P.D. 1989. "A Modified Method of Characteristics Technique and Mixed Finite Elements Method for Simulation of Groundwater Solute Transport." *Water Resources Research*, **25** (7), 1541-1549.

Cordes, C. and Kinzelbach, W. 1992. "Continuous Groundwater Velocity Fields and Path Lines in Linear, Bilinear, and Trilinear Finite Elements." *Water Resources Research*, **28** (11), 2903-2911.

Cox, R. A., and T. Nishikawa, 1991. “A New Total Variation Diminishing Scheme for the Solution of Advective-Dominant Solute Transport.” *Water Resources Research*, **27** (10), 2645-2654.

Danckwerts, P. V., 1953. “Continuous Flow Systems: Distribution of Residence Times,” *Chemical Engineering Science*, **2**, 1-13.

Dash, Zora. 2003. “Software Users Manual (UM) for the FEHM Application Version 2.21 Document ID: 10086-UM-2.21-00, MOL.20031031.0266.

Dash, Zora. 2003. “Validation Test Plan (VTP) for the FEHM Application Version 2.21 Document ID: 10086-VTP-2.21-00, MOL.20031031.0264.

Dash, Zora. 2003. “FEHM V2.21 Validation Test Plan – Attachment 1, Validation Test Plan (VTP) Results for the FEHM Application Version 2.21 Document ID: 10086-VTP-2.21-00, MOL.20031031.0265.

Gelhar, L.W. 1997. “Perspectives on Field-Scale Application of Stochastic Subsurface Hydrology.” *Subsurface Flow and Transport: A Stochastic Approach*. Dagan, G., and Neuman, S.P., eds. 157-176. New York, New York: Cambridge University Press.

Kapoor, V., and L. W. Gelhar, 1994. “Transport in three-dimensionally heterogeneous aquifers 1. Dynamics of concentration fluctuations.” *Water Resources Research*, **30** (6), 1775-1788.

Kinzelbach, W 1988. *The Random Walk Method in Pollutant Transport Simulation.* Groundwater Flow and Quality Modelling. p. 227-245. Norwell, Massachusetts: D. Reidel Publishing Company.

Kitanidis, P. K., 1994. “The concept of the dilution index.” *Water Resources Research*, **30** (7), 2011-2026.

Kreft, A., and A. Zuber, 1978. “On the Physical Meaning of the Dispersion Equation and its Solutions for Different Initial and Boundary Conditions.” *Chemical Engineering Science*, **33**, 1471-1480.

Labolle, E. M., G. E. Fogg, and A. F. B. Tompson, 1996. “Random-Walk Simulation of Transport in Heterogeneous Porous Media: Local Mass-Conservation Problem and Implementation Methods.” *Water Resources Research*, **32** (3), 583-593.

Leij, F.J.; Skaggs, T.H.; and van Genuchten, M.T. 1991. “Analytical Solutions for Solute Transport in Three-Dimensional Semi-Infinite Porous Media.” *Water Resources Research*, **27**(10), 2719–2733.

Lichtner, P. C., S. Kelkar, and B. A. Robinson, 2002. “New Form of Dispersion Tensor for Flow in Axisymmetric Media with Implementation in Particle Tracking,” *Water Resources Research* **38**(8), 21-1 to 21-16.

Nauman, E.B. and Buffham, B.A. 1983. *Mixing in Continuous Flow Systems.* New York, New York: John Wiley & Sons.

Neupauer, R. M., and J. L. Wilson, 2001. “Adjoint-Derived Location and Travel Time Probabilities for a Multidimensional Groundwater System.” *Water Resources Research*, **37** (6), 1657-1668.

Parker, J. C., and M. Th. van Genuchten, 1984. “Flux-Averaged and Volume-Averaged Concentrations in Continuum Approaches to Solute Transport.” *Water Resources Research*, **20** (7), 866-872.

Pollock, D.W. 1988. “Semianalytical Computation of Path Lines for Finite-Difference Models.” *Ground Water,* **26** (6), 743-750.

Prevost, M., M. G. Edwards, and M. J. Blunt, 2001. “Streamline Tracing on Curvilinear Structured and Unstructured Grids,” SPE Reservoir Simulation Symposium, Houston, Texas, 11-14 February.

Robinson, B. A., and G. Y. Bussod, 2000. “Radionuclide transport in the unsaturated zone at Yucca Mountain: Numerical model and preliminary field observations.” in Faybishenko, B., P. A. Witherspoon, and S. M. Benson (eds.), *Dynamics of Fluid in Fractured Rock*, American Geophysical Union, Geophysical Monograph **122**, 323-336.

Robinson, B. A., C. Li, and C. K. Ho, 2003. “Performance assessment model development and analysis of radionuclide transport in the unsaturated zone, Yucca Mountain, Nevada.” *J. Contam. Hydrology* **62-63**, 249-268.

Schafer-Perini, A.L. and Wilson, J.L. 1991. "Efficient and Accurate Front Tracking for Two-Dimensional Groundwater Flow Models." *Water Resources Research,* **27** (7), 1471-1485.

Tompson, A. F. B., 1993. “Numerical Simulation of Chemical Migration in Physically and Chemically Heterogeneous Porous Media.” *Water Resources Research,* **29** (11), 3709-3726.

Tompson, A.F.B. and Gelhar, L.W. 1990. “Numerical Simulation of Solute Transport in Three-Dimensional, Randomly Heterogeneous Porous Media.” *Water Resources Research,* **26** (10), 2541–2562.

Tompson, A.F.B.; Vomvoris, E.G.; and Gelhar, L.W. 1987. *Numerical Simulation of Solute Transport in Randomly Heterogeneous Porous Media: Motivation, Model Development, and Application*. UCID 21281. Livermore, California: Lawrence Livermore National Laboratory.

Uffink, G. J. M., 1989. “Application of Kolmogorov’s Backward Equation in Random Walk Simulations of Groundwater Contaminant Transport,” in *Contaminant Transport in Groundwater,* H. E. Kobus and W. Kinzelbach (eds.), 283-289, A. A. Balkema, Brookfield, Vermont.

Wolfsberg, A.V. and Freyberg, D.L. 1994. "Efficient Simulation of Single Species and Multispecies Transport in Groundwater with Local Adaptive Grid Refinement." *Water Resources Research,* **30** (11), 2979-2991.

Zyvoloski, G.A.; Robinson, B.A.; Dash, Z.V.; and Trease, L.L. 1997. *Summary of the Models and Methods for the FEHM Application—A Finite-Element Heat- and Mass-Transfer Code.* LA-13307-MS. Los Alamos, New Mexico: Los Alamos National Laboratory.