LA-UR-11-02090



*Title:* User's Guide for the PLUMECALC Application with Subgridding

*Author(s):* Bruce A. Robinson
Zora V. Dash
Scott L. Painter

*Intended for:* Software Documentation




Form 836 (7/06)

# User's Guide for the PLUMECALC Application with Subgridding

# Version 2.3.2

**Bruce A. Robinson**[1]
**Zora V. Dash**[2]
**Scott L. Painter**[2]

**April 2011**

[1]**SPO-CNP: Civilian Nuclear Programs**
[2]**Earth and Environmental Sciences Division**
**LOS ALAMOS NATIONAL LABORATORY, LOS ALAMOS, NM 87544, USA**

**DISCLAIMER**

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors or their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The view and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

## REVISION HISTORY

| Revision | Date | Purpose of the Revision |
|---|---|---|
| 0 | 11/25/2008 | Initial Implementation |
| 1 | 10/19/2010 | Add discussion of modifications for mobile resident concentration calculations.<br><br>Incorporate discussion of subgridding and testing. |
| 2 | 04/05/2011 | Update input and output description to reflect code modifications.<br><br>Update examples to reflect code modifications for resident mobile concentration calculations.<br><br>Add appendix with additional discussion of matrix diffusion and testing.<br><br>Updated type curve data lookup tables provided for diffusion tests. |

# CONTENTS

**Page**

# FIGURES

# ABSTRACT


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

A numerical technique called the Convolution-Based Particle Tracking (CBPT) method was recently developed to simulate resident or flux-averaged solute concentrations in groundwater models. The method is valid for steady state flow, linear transport processes, and first order decay. Under these constraints, the principle of superposition and numerical convolution can be used to efficiently post-process results from particle tracking simulations involving pulse releases to account for the time variation of each input source function during the course of the calculation. Moreover, a residence time distribution approach can be used to account for various retardation processes such as sorption with a linear isotherm and diffusion into a matrix rock. The algorithms for carrying out the convolutions, superposition, and residence time calculations based on particle tracking results are very efficient. 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 demonstrate the use of PLUMECALC, the code which implements this post-processing 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 (Robinson, et al. 2010) and how to effectively install and use the PLUMECALC V2.3.2 software (Robinson, et al. 2011). This document also provides results of verification tests. This document only discusses PLUMECALC, a post-processor to particle-tracking simulations. Particle tracking simulations, which are required by PLUMECALC, are not discussed here. The code was initially developed to be compatible with the FEHM fluid flow and random-walk particle tracking model (Zyvoloski, et al. 1997, Dash, 2003a). Therefore, the code requires auxiliary input information related to the flow and transport model, such as the grid geometric information and particle times and positions, using a format compatible with FEHM input and output but does not preclude the use of other particle tracking models.

## 1.2 SOFTWARE IDENTIFICATION

PLUMECALC Version 2.3.2
LA-CC-11-029

## 1.3 DEFINITIONS

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

## 1.4 CONTACT

Zora Dash
zvd@lanl.gov
505-667-1923

# 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 subjects of ongoing research (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 with acceptably small levels 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 employs the principle of superposition in space and time to take advantage of efficiencies that can be gained under the assumption of linearity. The method is valid for linear transport processes and steady state flow, or, in the case of complex, transient flow, only those regions where the velocity field is time-invariant. For the general case of transient flow, a PLUMECALC implementation approach has been developed (Svrinivasan et al., 2011), but is not discussed in this manual. Contact the author for more information (gowri@lanl.gov). 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, $\theta$ 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 $\xi$ 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 $\xi$ 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 $\Omega$ requires that

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

where $\zeta(\xi, t)$ is the cumulative mass from source $\xi$ 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 $\xi$. Recognizing that under the assumptions stated above, the principle of superposition in time applies, the concentration $c(\xi,\tilde{x},\tau)$ at time $\tau$ can be computed using the following numerical convolution equation:

$$\theta c(\xi,\tilde{x},\tau) = \int_0^{\tau} \dot{m}(\xi,\tau-t)\hat{c}(\xi,\tilde{x},t)dt \qquad (3)$$

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 $\tau$. The requirement that the mass contribute to the calculation of $c(\xi,\tilde{x},\tau)$ is that it arrives at this location at time $\tau$; this can happen due to recently injected mass that quickly reaches $\tilde{x}$, or mass that was injected initially and took the full time $\tau$ 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 $\tau$.

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:

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

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. Flux-averaged concentration is only relevant for defining concentration in a water at a discharge location – it is not an appropriate metric within the flow system.

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. See, e.g. the textbook of Nauman and Buffham, 1983 for a summary of the theory and applications, and Painter et al. 2008 for recent applications to radionuclide transport. 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 $\xi$ 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 $\bar{c}(\xi,\tilde{x},\tau)$ exiting at $\tilde{x}$ due to injection at source region $\xi$ is given by the following numerical convolution equation:

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

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 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. One such scheme, as implemented in this version of PLUMECALC, is discussed in Section 4.

### 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 $\tau$ associated with the residence of this particle within the cell is proportional to

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

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 $\tau$, and the max functions ensure that only particles that have spent time in the cell at or before time $\tau$ are included. Then, the aqueous concentration calculation sums the contributions for all particles:

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

where $N_p(\xi)$ is the total number of particles associated with source location $\xi$, $\theta(\tilde{x})$ 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

Flux-averaged concentration (Eq. 6) is sometimes of interest at the location of a groundwater outflow boundary or pumping well. In this context, flux-averaged concentration corresponds to the concentration that would be measured in a mixing cell that collects the groundwater discharge. It is important to note, however, that flux-averaged concentration at locations not corresponding to a groundwater sink does not have a clear physical interpretation unless there is no dispersion, in which case it corresponds to resident concentration in mobile water (see Section 3). Thus, flux-averaged concentration is not recommended in general as a spatially distributed metric for radionuclide concentration.

The calculation of the flux-averaged concentration $\hat{c}(\xi,\tilde{x},t)$ using a particle tracking model of $f(\xi,\tilde{x},t)$ requires that a temporal averaging be performed, analogous to the spatial averaging carried out for the estimation of the resident concentration $\hat{c}(\xi,\tilde{x},t)$. Contributions to $f(\xi,\tilde{x},t)$ come in the form of incremental jumps in the output as individual particles leave a computational grid cell or cells at $\tilde{x}$, 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 $\Delta t_m$ 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 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 proportional to

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

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:

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

where the summation occurs over all particles that exit at $\tilde{x}$. Conceptually, each particle leaving at $\tilde{x}$ 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 $\rho_b$ 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 $\bar{c}(\xi, \tilde{x}, t)$, 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 distribution of solute mass in the primary porosity fluid, the secondary porosity fluid, and sorbed to both media. Care must be taken in defining concentration in this situation.

### 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 $\tau_{out}$ and $\tau_{in}$:

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

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 $\tau_{out}$ to $\tau_{in}$, 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 $\xi$, 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 $\xi$, 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. PLUMECALC REVISION TO CALCULATE MOBILE RADIONUCLIDE CONCENTRATION

Previous versions of PLUMECALC provided resident concentrations based on total radionuclide mass in a computational cell. For fracture-matrix systems the total resident concentration is not necessarily of primary interest because it reflects the mass and water volume over an entire grid cell including that which is sorbed or diffused into the immobile matrix, but not the concentration of the water moving in fractures – the dominant flow path. This limitation of previous PLUMECALC versions motivated the use of flux-averaged concentration because flux-averaged concentration approximates the quantity of interest (concentration in the mobile fluid) when the radionuclide mass flux is advection dominated. If the dispersive flux is not insignificant, however, the flux-averaged concentration does not have a clear physical meaning except at the location of groundwater discharge. To address this limitation, PLUMECALC 2.3.2 has a new option to directly calculate volume-averaged concentrations of radionuclides dissolved in mobile fluid (herein denoted “mobile concentration” or “mobile resident concentration”).

PLUMECALC calculates resident concentration at a specific computational cell at time τ due to a single source by summing contributions from individual particles (see Equation 13)

$$c(\xi,\tilde{x},\tau)=\sum_{p}c_{p}(\xi,\tilde{x},\tau) \tag{19}$$

where the summation is over the $N_p$ particles associated with that source and

$$N_{p}(\xi)\theta(\tilde{x})V(\tilde{x})C_{p}(\xi,\tilde{x},\tau)=\int_{\tau_{1,p}}^{\tau_{2,p}}\dot{m}(\xi,t)e^{-k(\tau-t)}dt \tag{20}$$

with $\tau_{2,p}=\max\left(0,\tau-t_{in,p}\right)$ and $\tau_{1,p}=\max\left(0,\tau-t_{out,p}\right)$. Here $\dot{m}(t)$ is the radionuclide mass source rate at time t, $\theta$ is the porosity, V is volume of the cell in question, and *k* is the radionuclide decay constant. In addition, $t_{in,p}$ and $t_{out,p}$ are the time the *p*-th particle entered and exited the cell in question, as calculated by FEHM. The concentration so-defined is "total resident", representing all radionuclide mass (sorbed, dissolved in mobile water, dissolved in immobile water) per unit volume of mobile water.

If matrix diffusion or other retardation/retention mechanisms are included, then the sequence of cell exit times is transformed within PlumeCalc by adding a random retention time to the groundwater residence time in each cell. The retention times are conditional on the groundwater residence time and sampled from known probability distributions based on the underlying physicochemical processes. If we keep the same notation, denoting the exit and entry times with retention as $t_{in,p}$ and $t_{out,p,}$ then equations 19 and 20 provide the total concentration (mass of all radionuclides in the cell divided by water volume associated with the porosity $\theta$).

To find the concentration of mobile radionuclides, note that the probability of finding a given particle in the mobile phase while it is in the cell in question is simply $\left(t_{out,p}^{adv}-t_{in,p}^{adv}\right)\Big/\left(t_{out,p}-t_{in,p}\right)$, i.e. the ratio of mobile transit time to total transit time. Here the superscript *adv* refers to particles traveling by advection only. Thus, the mobile concentration can be calculated as

$$C_{mobile}(\tau)=\sum_{p}\frac{\left(t_{out,p}^{adv}-t_{in,p}^{adv}\right)}{\left(t_{out,p}-t_{in,p}\right)}C_{p}(\tau) \tag{21}$$

It should be noted that matrix diffusion, equilibrium sorption or other retardation processes can be accounted for in PlumeCalc. Thus, a single FEHM particle-tracking run without retention is all that is required for a given flow field and dispersion model. Effects of different sources and retention processes/parameters can then be investigated by running PLUMECALC alone. Equation 21 is implemented in PLUMECALC V2.3.2.

# 4. SUBGRID PLUMECALC

Development of the subgrid module for PLUMECALC was motivated by a need to calculate concentrations at a scale smaller than the scale used for transport model particle tracking computations that form the starting point of the concentration computations performed by PLUMECALC. As given in Equations 11 and 13, concentration at a given cell requires a summation over those particles that cross the cell, of a convolution integral over the source function. The time limits of the integral depend on the type of averaging required (volume or flux), but involve the times at which a given particle enters and leaves the cell for which the concentration is being computed. It is clear that the boundaries of the cell will have a direct effect on value of this summation. In the case of volume-averaged concentrations, the equation (Eq. 11) also has in the denominator the volume of the cell under consideration - which depends directly on the size of the cell. In the case flux averaged concentrations, Eq. 13 has in the denominator the flux through the cell, which can either be a user specified quantity or the actual flux passing through the cell as computed by the flow simulation - which clearly depends directly on the size and shape of the cell. Because of these considerations, the concentrations computed on the scale of the grid size can underestimate concentrations that might exist locally within a cell. For example, suppose that all the particle tracks pass through only a portion of the volume being considered. Then if the grid spacing were fine enough, some of the cells would report a relatively high value of concentration while some of the cells would report a nearly zero value. But if the grid spacing is large so that this local variation is not captured, it will report a concentration that will be larger than zero but smaller than the maximum value reported for the fine grid. This distinction is not important when the regulatory criteria prescribed for performance evaluation of the system involve global quantities such as the breakthrough curves at a compliance boundary (for example YMP), however, the distinction between concentration values computed at different scales can be very important if the performance criteria depend on a maximum prescribed level of contaminant concentrations (e.g. UGTA).

While in principal it is possible to perform the flow and transport computations on a refined grid and then use PLUMECALC to calculate the concentrations on this refined scale, often there are practical limitations such as computer memory and computational time, especially when the numbers of nodes used even on a coarse scale are on the order of millions. Another practical consideration is that it is not always known a priori where the particle tracks will lie, so it is not known which portions of the grid need to be examined at a refined scale.

For these reasons, we have developed an extension of the code PLUMECALC to perform concentration calculations on a scale more refined than the scale of the grid used for transport calculations. A virtual subgrid is constructed in regions of the original grid traversed by the particle tracks computed in particle tracking model. The level of the subgridding is specified by the user, and can vary from cell to cell and also be different in different directions. The particle track information, and in case of flux averaging the flux information, is interpolated from the grid to the virtual subgrid to compute concentrations on the subgrid scale.

The virtual subgrid is constructed starting with the corner (left, bottom, back corner) and size (length, width, height of the box) information output for each node. The dimensions of the box are subdivided into the specified number of segments, and a virtual node is placed at the geometric center of each subbox. Note that due to this construction, for some grids, and boundary nodes, the locations of the virtual nodes may not coincide with actual node locations. The particle trajectory is interpolated as a straight line within the box using the entrance and exit locations and output times, and the entrance and exit locations and times for each subbox are calculated and saved for use by the rest of the PLUMECALC code.

# 5. USER INFORMATION

## 5.1 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 was initially developed to be compatible with the FEHM fluid flow and random-walk particle tracking model (Dash, 2003). 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.

## 5.2 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_sg_V2.3.2 [*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 5.3.2) 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.

## 5.3 INPUT SPECIFICATION

### 5.3.1 General information

PLUMECALC was initially implemented using streamline particle tracking output generated by the FEHM code. It is possible to use particle tracking data from other codes provided the file formats are consistent with those used by FEHM. The list of data files needed for a PLUMECALC simulation is given in Section 5.5. The particle tracking information needed includes 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, and when subgridding, the particle exit coordinate position. The particle tracking output used by PLUMECALC can use one of three possible formats:

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 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.

### 5.3.2 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 grid file for the particle tracking simulation. |
| (optional) | file_format | character*12 | Keyword “ascii”, “formatted”, or “unformatted” denoting the format of the storage coefficient file. If the keyword is omitted, ASCII formatting is assumed. |
| 2 | stor_file | character*100 | Name of the storage coefficient file for the particle tracking simulation. |
| (optional) | keyword | character*4 | Keyword “flux” to denote flux values will be read from a restart file. |

| Group number | Input Variable | Type | Definition |
|---|---|---|---|
| (optional) | file_format | character*12 | Keyword “ascii”, “formatted”, or “binary” denoting the format of the 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 restart file that contains steady state fluxes. |
| (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 of the particle tracking output. If the keyword is omitted, ASCII formatting is assumed. |
| 3 | sptr_file | character*100 | Name of the 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. |
| 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 for double porosity systems (optional). Note transfer function data is needed only for double-porosity simulations. |
| 8 | input_msg | character*6 | Optional Keyword “SUBGRID” to indicate subgridding should be invoked |

### 5.3.3 Input files from the Particle Tracking Simulation

The PLUMECALC simulation implicitly adopts all of the input parameters associated with a flow and particle tracking 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). It is the responsibility of the user to ensure that all particles have exited the system during the time allotted for the particle tracking simulation,
- 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).

### 5.3.4 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 solute ($day^{-1}$). If decay is not used a value of 0. must be entered. |
| Optional keyword<br>"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.<br>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.<br>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 | | |

| Input Variable | Type | Definition |
|---|---|---|
| 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 |
| 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<br>"favg" or "mflx" | | |
| conc_string | character*4 | Keyword denoting the type of calculation:<br>If 'favg' is input, flux-averaged concentration leaving a zone or cell is determined.<br>If 'mflx' is input, solute mass leaving a cell is determined.<br>Omitting this keyword means that the resident concentration is computed.<br>**Flux-averaged concentration or mass are only applicable at an outflow boundary and should not be used as a spatially distributed metric** (see Section 2.3.4). |
| 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):<br>nfavgzones<br>water_flux(i), for i = 1, nfavgzones<br>inodes_favg<br>nodes_favg(i), for i = 1 to inodes_favg<br>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 |

| Input Variable | Type | Definition |
|---|---|---|
| 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 |
| nodes_favg | integer array, size inodes_favg | List of nodes associated with this flux_averaged concentration zone |
| If conc_string = 'mflx' the following are input:<br>nfavgzones<br>nodes_favg(i), for i = 1 to inodes_favg | | |
| nfavgzones | integer | Number of nodes at which the mass leaving the cell is to be calculated |
| nodes_favg | integer array, size nfavgzones | List of nodes at which the solute mass leaving the cell is to be calculated |
| Group 5: total_time, n_out_times | | |
| total_time | real*8 | Total time of the plume concentration simulation (days) |
| n_out_times | real*8 | Time step parameter:<br>If >0: abs(n_out_times) is the total number of equally spaced times for which the calculation is performed<br>If < 0: the code uses this value as the time step, with appropriate rounding to ensure equally spaced time steps<br>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<br>ntimes, delta_time<br>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 [optout1] [optout2] [optout3] | | |

| Input Variable | Type | Definition |
|---|---|---|
| out_string | character*4 | Keyword denoting the type of nodal output when the code is computing resident concentration:<br>If 'pckd': output is a "packed" output containing concentrations of cells that have particles passing through them (the others are always 0).<br>If 'node': concentrations are output at nodes specified by the following input.<br>If 'tecp' or 'tecn': output uses tecplot style headers and formatting for packed or node output. |
| opt_out*n* | character | Optional keywords specifying output options (used only with keyword 'tecp'). |
| The following keywords may be used for opt_out and may be entered in any order: | | |
| **Keyword** | **Abbreviation** | **Definition** |
| sparse | N/A | Omit node numbers for output after initial output time. |
| Total_concentration | total | Output total resident concentration for each output node (resident concentration calculations only) |
| flux | N/A | Output water flux for each output node (flux averaged concentration calculations only) |
| mass | N/A | Integrate source input and output cumulative mass (moles) at each time. |
| Total_Cell_Mass | moles | Output total mass in cell at each time, integrate source input and output cumulative moles at each time. |
| Mobile_Cell_Mass | mobile | Output total and mobile mass in cell at each time, integrate source input and output cumulative moles at each time. |
| Note: Only one keyword for mass output (mass, moles, or mobile) should be entered. When a mass keyword is input, the source input will be integrated and an additional output file will be created. A name can be specified for this file by following the 'tecp' input line with the following: | | |
| keyword | character*4 | Keyword 'file' |
| mdot_out_file | character*100 | Name of the output file to use for the results for integration of the source input. Default filename is 'mdot_integral.dat' |
| The following are input only if out_string = 'node' or 'tecn'<br>noutnodes<br>ioutnode(i), for i = 1 to noutnodes | | |
| noutnodes | integer | Number of nodes at which output concentrations are specified |

| Input Variable | Type | Definition |
|---|---|---|
| ioutnode | integer array, size noutnodes | Array of nodes at which resident concentrations are to be output |
| When “SUBGRID” is specified in the PLUMECALC control file the following input is included at the end of the simulation control input file<br>subgrid, refine_type<br>sptr_velocity_file<br>refinement factors<br>There are three options for entering the refinement factors:<br>1) As a single value, where the same factor (scale_xyz or n_xyz) is used for the x, y, and z directions for all cells that have been accessed by particles.<br>2) As three values ( x_scale, y_scale, z_scale or n_x, n_y, n_z) where a value is entered for each direction, the same factors are applied to each cell that is accessed by particles<br>3) As multiple lines of data, where the cell to be subgridded is specified followed by the three factors (node# scale_x scale_y scale_z or node# n_x n_y n_z ) to be used to subgrid that cell . | | |
| keyword | character | Keyword “subgrid” |
| refine_type | character | Keyword specifying method to use for refinement: “scale_factor”, “refine_factor”<br><br>If the keyword "scale_factor" appears on the same line following the keyword "subgrid", then the dimensions of the cell in the coarse grid in each of the x, y, z directions are divided by the scale_factor (length, to the nearest integer) to create the number of subdivisions in that direction.<br><br>If the keyword "refine_factor" appears on the same line following the keyword "subgrid", then the cell in the coarse grid is divided into nx, ny, nz subcells in each of the x, y, z dimensions. |
| sptr_velocity_file | character*1024 | Name of file containing sptr velocities and geometric data |
| scale_xyz | real*8 | Scaling length in x, y, z directions (m) |
| x_scale | real*8 | Scaling length in x direction (m) |
| y_scale | real*8 | Scaling length in y direction (m) |
| z_scale | real*8 | Scaling length in z direction (m) |
| n_xyz | integer | Number of subcells in x, y, z direction |
| n_x | integer | Number of subcells in x direction |
| n_y | integer | Number of subcells in y direction |
| n_z | integer | Number of subcells in z direction |

| Input Variable | Type | Definition |
|---|---|---|
| node | integer | Node number where following factors will be applied |

### 5.3.5 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.<br>Group 1: rock | | |
| dummy_string | character*4 | Keyword "rock" designating rock property information follows |
| Group 2: ja, jb, jc, denr, kdp, ps [rpor] | | |
| 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. (kg-fluid / kg-rock). The retardation factor for sorption will be computed. This parameter is used for single-porosity systems only. Group 3 input describes input for dual porosity systems. |
| ps | real*8 | Porosity of the medium (under unsaturated conditions, this is the volumetric water content) |

| Input Variable | Type | Definition |
|---|---|---|
| rpor | real*8 | Reference porosity used in the FEHM particle tracking simulation (optional). The reference porosity is used to compute a scaling factor (the ratio of the PLUMECALC model porosity to FEHM model porosity) for time corrections using $vcf = ps / rpor$. If omitted the correction factor has a default value of 1. In previous versions of PLUMECALC the scaling factor (called the velocity correction factor) was entered directly. For compatibility with previous versions of PLUMECALC, the code will compute the factor if the value entered for *rpor* is less than 1. otherwise it will interpret the value as the scaling factor. |
| 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.<br>Group 1: diff [keyword] | | |
| dummy_string | character*4 | Keyword "diff" designating diffusion model information follows |
| dummy_string | character*8 | Optional keyword "fracture" or "matrix" to indicate that the total resident concentration should be computed using fracture or matrix porosity where a double porosity model is specified. The default is to use the total porosity defined as $\phi_{total} = \phi_f + (1-\phi_f) \cdot \phi_m$. Note that the fracture porosity for a double porosity model is the porosity (*ps*) defined in the rock macro. |
| 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<br>Group 3: kd, diffmfl, rd_frac, matrix_por, spacing_primary | | |
| kd | real*8 | Sorption coefficient in the matrix (kg-fluid / kg-rock). |
| diffmfl | real*8 | Molecular diffusion coefficient in the rock matrix ($m^2/s$) |
| rd_frac | real*8 | Retardation factor in the fracture system |
| matrix_por | real*8 | Matrix porosity |

| Input Variable | Type | Definition |
|---|---|---|
| spacing_primary | real*8 | Length scale, fracture aperture (2b), in the primary porosity for the diffusion model (m). If a negative value is input for this parameter, an error function solution is used to determine 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 |

### 5.3.6 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.<br>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 |

## 5.4 OUTPUT SPECIFICATION

The output file contains the following output from the simulation. Unless otherwise specified, when resident concentration calculations are invoked, the default output will be the mobile resident concentration. For tecplot output that includes coordinates, the coordinates are only output for the initial output time.

| Output Variable | Type | Definition |
|---|---|---|
| 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. (output option “pckd” or “tecp”). |
| touched_cells | integer | List of the nodes that have non-zero concentrations or if subgridding is used the subgrid node. |
| noutnodes | integer | Number of nodes being output (output option “node” or “tecn”). |
| ioutnode | integer | List of specified output nodes. |
| nfavgzones | integer | Number of zones for which flux averaged concentrations are computed or number of nodes for which solute mass is output. |
| nodes_favg | integer | List of solute mass output nodes. |
| index_favg | integer | Index to mass solute node output in cfavg array. |
| current_time | real*8 | Current simulation time (days). |
| x | real*8 | X coordinate of node (m) |
| y | real*8 | Y coordinate of node (m) |
| z | real*8 | Z coordinate of node (m) |
| conc_mobile | real*8 | Cell mobile concentration at current time (moles/l). |
| concentration | real*8 | Total cell concentration at current time (moles/l) |
| cfavg | real*8 | Flux averaged concentration of the output zone (moles/l) or solute mass (moles) at current time. |
| water_flux | real*8 | Water flux of the output zone (l/day) |
| conc_total | real*8 | Total moles in cell at current time (resident) or total moles leaving cell (flux averaged) |
| conc_mobtot | real*8 | Total mobile moles in cell at current time (resident) |

| Output Variable | Type | Definition |
|---|---|---|
| id_parent_sg | integer | Parent node number (original node number in the model grid) |
| n_sources | integer | Number of solute sources input for the simulation. |
| r1 | real*8 | Solute mass input for the ith source during the current time interval. The time interval is calculated as the current_time - previous_output_time (moles). |
| r2 | real*8 | Solute mass decayed for the ith source during the current interval (moles). |
| mdot_total | real*8 | Cumulative solute mass input or decayed for the ith source (moles). |
| result | real*8 | Cumulative solute mass undecayed for the ith source (moles). |
| r1_sum, r2_sum | real*8 | Sum of the solute mass (input or decayed) for all sources during the current interval (moles). |
| result_sum | real*8 | Cumulative solute mass undecayed for all sources (moles). |

Resident concentration calculations:

| |
|---|
| **Output Option 'pckd'**<br>Group 1: ntimes, n_touched_cells<br>Group 2: touched_cells(i), for i = 1 , n_touched_cells<br>Groups 3 and 4 are repeated ntimes<br>Group 3: current_time<br>Group 4: conc_mobile(i) for i = 1 to n_touched_cells |
| **Output Option 'node'**<br>Group 1: ntimes, n_touched_cells<br>Groups 2 and 3 are repeated ntimes, with Group 3 repeated for each output node<br>Group 2: current_time<br>Group 3: ioutnode(i), conc_mobile(ioutnode(i)), for i = 1 to noutnodes |

**Output Option ‘tecn’**

Header: variables= "X (m)", "Y (m)", "Z (m)", "Node", "Mobile_Concentration (moles/l)"

Groups 1 and 2 are used for the first output time

Group 1: zone t="time current_time days"

Group 2: x(i), y(i), z(i), ioutnode(i), conc_mobile(ioutnode(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 days", VARSHARELIST = ([1-4]=1)

Group 4: conc_mobile(ioutnode(i)), for i = 1 to noutnodes

**Output Option ‘tecp’**

Header: variables= "X (m)", "Y (m)", "Z (m)", "Node", "Mobile_Concentration (moles/l)"

Groups 1 and 2 are used for the first output time

Group 1: zone t="time current_time days"

Group 2: x(i), y(i), z(i), touched_cells(i), conc_mobile(i), for i = 1 to n_touched_cells

Groups 3 and 4 are repeated for each subsequent output time (up to ntimes):

Group 3: zone t="time current_time days", VARSHARELIST = ([1-3]=1)

Group 4: touched_cells(i), conc_mobile(i), for i = 1 to n_touched_cells

**Output Option ‘tecp sparse’**

The Header and Groups 1 and 2 are the same as above, Groups 3 and 4 are repeated for each subsequent output time (up to ntimes) and Group 4 output does not include node numbers.

Group 3: zone t="time current_time days", VARSHARELIST = ([1-4]=1)

Group 4: conc_mobile(i) for i = 1 to n_touched_cells

**Output Option ‘tecp total’**

Header: variables= "X (m)", "Y (m)", "Z (m)", "Node", "Total_Concentration (moles/l)", "Mobile_Concentration (moles/l)"

Groups 1 and 2 are used for the first output time

Group 1: zone t="time current_time days"

Group 2: x(i), y(i), z(i), touched_cells(i), concentration(i), conc_mobile(i), for i = 1 to n_touched_cells

Groups 3 and 4 are repeated for each subsequent output time (up to ntimes):

Group 3: zone t="time current_time days", VARSHARELIST = ([1-3]=1)

Group 4: touched_cells(i), concentration(i), conc_mobile(i), for i = 1 to n_touched_cells

**Output Option ‘tecp total sparse’**

The Header and Groups 1 and 2 are the same as above, Groups 3 and 4 are repeated for each subsequent output time (up to ntimes) and Group 4 output does not include node numbers.

Group 3: zone t="time current_time days", VARSHARELIST = ([1-4]=1)

Group 4: concentration(i), conc_mobile(i), for i = 1 to n_touched_cells

**Output Options including ‘moles’ or ‘mobile’**

For option ‘moles’, variable title "Total_Cell_Mass (moles)" is appended to the header line and Groups 2 and 4 include conc_total(i)

For option ‘mobile’, variable titles "Total_Cell_Mass (moles)" , "Mobile_Cell_Mass (moles)" are appended to the header line and Groups 2 and 4 include conc_total(i), conc_mobtot(i)

Flux-averaged concentration calculation:

**Output Option ‘pckd’** when concentrations are computed for zones

Header: Time (days) Zone0001 . . . ZoneNNNN

Group 1 is repeated ntimes

Group 1: current_time, cfavg(i), i=1,nfavgzones

**Output Option 'tecp'** when concentrations are computed for zones

Header: variables= "Time (days)" "Zone 0001" . . . "Zone NNNN"

Group 1 is repeated ntimes

Group 1: current_time, cfavg(i), i=1,nfavgzones

**Output Option 'tecp'** when fluxes are read from restart file

Note that each node/cell is treated as a zone and output will use the same format as resident time concentrations when using the "tecp" option.

Header: variables= "X (m)", "Y (m)", "Z (m)", "Node", "Flux_Averaged_Concentration (moles/l)"

Groups 1 and 2 are used for the first output time

Group 1: zone t="time current_time days"

Group 2: x(i), y(i), z(i), touched_cells(i), cfavg(touched_cells(i)), for i = 1 to n_touched_cells

Groups 3 and 4 are repeated for each subsequent output time (up to ntimes):

Group 3: zone t="time current_time days", VARSHARELIST = ([1-3]=1)

Group 4: touched_cells(i), cfavg(touched_cells(i)), for i = 1 to n_touched_cells

**Output Option 'tecp sparse'**

The Header and Groups 1 and 2 are the same as above, Groups 3 and 4 are repeated for each subsequent output time (up to ntimes) and Group 4 output does not include node numbers.

Group 3: zone t="time current_time days", VARSHARELIST = ([1-4]=1)

Group 4: cfavg(touched_cells(i)), for i = 1 to n_touched_cells

**Output Option 'tecp flux'**

Header: variables= "X (m)", "Y (m)", "Z (m)", "Node", "Flux_Averaged_Concentration (moles/l)", "Water_Flux (l/day)"

Groups 1 and 2 are used for the first output time

Group 1: zone t="time current_time days"

Group 2: x(i), y(i), z(i), touched_cells(i), cfavg(touched_cells(i)), water_flux(touched_cells(i)) for i = 1 to n_touched_cells

Groups 3 and 4 are repeated for each subsequent output time (up to ntimes):

Group 3: zone t="time current_time days", VARSHARELIST = ([1-3, 6]=1)

Group 4: touched_cells(i), cfavg(touched_cells(i)), for i = 1 to n_touched_cells

**Output Option ‘tecp flux sparse’**

The Header and Groups 1 and 2 are the same as above, Groups 3 and 4 are repeated for each subsequent output time (up to ntimes) and Group 4 output does not include node numbers.

Group 3: zone t="time current_time days", VARSHARELIST = ([1-4, 6]=1)

Group 4: cfavg(touched_cells(i)), for i = 1 to n_touched_cells

**Output Options including ‘moles’**

For option ‘moles’, variable title "Total_Cell_Mass (moles)" is appended to the header line and Groups 2 and 4 include conc_total(i)

Solute mass output:

**Output Option ‘tecp’**

Header: variables= "X (m)", "Y (m)", "Z (m)", "Node", "Solute_Mass (moles)"

Groups 1 and 2 are used for the first output time

Group 1: zone t="time current_time days"

Group 2: x(i), y(i), z(i), nodes_favg(i), cfavg(index_favg(nodes_favg(i))), for i = 1 to nfavgzones

Groups 3 and 4 are repeated for each subsequent output time (up to ntimes):

Group 3: zone t="time current_time days", VARSHARELIST = ([1-3]=1)

Group 4: nodes_favg (i), cfavg(index_favg(nodes_favg(i))), for i = 1 to nfavgzones

**Output Option ‘tecp sparse’**

The Header and Groups 1 and 2 are the same as above, Groups 3 and 4 are repeated for each subsequent output time (up to ntimes) and Group 4 output does not include node numbers.

Group 3: zone t="time current_time days", VARSHARELIST = ([1-4]=1)

Group 4: cfavg(index_favg(nodes_favg(i))), for i = 1 to nfavgzones

Output with subgridding:

When subgridding is invoked the subgrid parent node is output in addition to the subgrid node number. The output would be modified as follows:

Header: variables= "X (m)", "Y (m)", "Z (m)", "Node_parent", "Node_subgrid", . . .

Group 2: x(j), y(j), z(j), id_parent_sg(j), touched_cells(i), . . .

Group 4: id_parent_sg(j), touched_cells(i), . . .

and variables are indexed by touched_cells(i)

For flux averaged concentration with keyword “flux” Group 3 is also modified:

Group 3: zone t="time current_time , VARSHARELIST = ([1-3, 7]=1)"

When the optional keyword “sparse” is used Group 3 is modified as follows and Group 4 output does not include any node numbers.

Group 3: zone t="time current_time , VARSHARELIST = ([1-5]=1)"

Or with keyword “flux”

Group 3: zone t="time current_time , VARSHARELIST = ([1-5, 7]=1)"

Mass summary output:

**Output Options including ‘mass’ “moles’ or ‘mobile’**

The output data are the same as described above for the various “tecp” options but will include the following summary for each output time:

text = "Total moles in system = *cum_moles* "
text = "Total moles from source = mdot_total(n_sources + 1, 1) "
text = "Total moles that exited = *cum_produced*"
Or for the ‘mflux’ option
text = "Total moles in output nodes= *cum_moles* " (‘mflx’ option)
text = "Total moles from source = mdot_total(n_sources + 1, 1) "

Or if decay is included:

text = "Total moles in system = *cum_moles* , mobile moles = *cum_mobile* "
text = "Total moles from source = mdot_total(n_sources + 1, 1) , moles decayed = mdot_total(n_sources + 1, 2), , moles undecayed = *mdot_undecayed* "
text = "Total moles that exited = *cum_produced* (undecayed) "
Or for the ‘mflx’ option
text = "Total moles in output nodes= *cum_moles* "
text = "Total moles from source = mdot_total(n_sources + 1, 1) , moles decayed = mdot_total(n_sources + 1, 2), , moles undecayed = *mdot_undecayed* "

In addition, the source term data will be integrated and output to the file specified after the output option in the simulation input file or to ‘mdot_integral.dat’.

Integrated source term data (mdot_integral.dat):

Simulation without decay

Header: variables = "Source #" "Moles Input" "Cumulative Moles"

Groups 1, 2 and 3are repeated for each output time (up to ntimes):

Group 1: zone t="Time current_time days"

| |
|---|
| Group 2 is repeated for each solute source (isource = 1, n_sources)<br>Group 2: isource, r1, mdot_total(isource, 1)<br>Group 3: text t = "Total r1_sum, mdot_total(n_sources + 1, 1)” |
| Simulation with decay<br>Header: variables = "Source" "Moles Input" "Cumulative Moles" "Moles Decayed" "Cumulative Decayed" "Cumulative UnDecayed"<br>Groups 1 and 2 are repeated for each output time (up to ntimes):<br>Group 1: zone t="Time current_time days"<br>Group 2 is repeated for each solute source (isource = 1, n_sources)<br>Group 2: isource, r1, mdot_total(isource, 1), r2, mdot_total(isource, 2), result<br>Group 3: text t = "Total r1_sum, mdot_total(n_sources + 1, 1), r2_sum, mdot_total(n_sources + 1, 2), result_sum |

## 5.5 DATA FILES

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

- I/O input file for PLUMECALC. This file contains the input and output file information (see Section 5.3.2).
- Grid file for the particle tracking simulation.*
- Storage coefficient file for the particle tracking simulation.* This file may be formatted or unformatted.
- Restart file containing steady-state fluxes used for the particle tracking simulation.* This file may be formatted or unformatted.
- Particle tracking output file to be used in the calculation of plume concentrations.* This file may be formatted, unformatted, or binary.
- Simulation control input file for the PLUMECALC simulation (see Section 5.3.4).
- Rock property input file (see Section 5.3.5). An optional zone macro data file may be used within the rock property information file for the zone input.*
- Solute mass flux input files (see Section 5.3.6).
- Transfer function curve data file.*
- Output file for the PLUMECALC simulation (see Section 5.4).
- Error conditions and messages output file, plumecalc.err.

* See Appendix B and the FEHM users manual for a description of the file format (Dash, 2003a).

## 5.6 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.

## 5.7 ERRORS

Error / warning 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 / Warning Condition** | **Message** |
|---|---|
| I/O file error | |
| File does not exist or may not be opened for reading / writing. | ERROR opening *FILENAME*<br>STOPPING execution |
| File cannot be read as written. | ERROR reading coefficient storage file<br>STOPPING execution |
| An error was encountered while trying to read flux data from the FEHM restart file. | ERROR reading flux data<br>STOPPING execution |
| Format wrong for input file needed for subgridding | Wrong format for sptr_corn_file<br>STOP |
| Missing input data in particle file used for subgridding | Program stopped, XYZ data not found in sptr2 file: *filename* |
| Input Error | |
| An illegal keyword was input for out_string in Group 6 of the simulation control input file. | Unrecognized output option: *OUT_STRING*<br>use keyword pckd, tec, or node instead<br>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<br>STOPPING Execution |
| Illegal data has been entered in the rock property input file for macro “rock”. | Fatal error - for array number *IARRAY*<br>macro - *MACRO*<br>Group number - *IGROUP*<br>Something other than a real or integer has been specified<br>Line number - *INUMBER*<br>Bad input, check this line |
| Error found in transfer function curve input data. | Stopping in svdcmp_new<br>Fatal error in transfer function |
| Wrong number of parameters found in transfer function curve data file. | Error in particle tracking interp.<br>STOPPING execution |
| Error found in transfer function curve input data. | Decreasing data found in type curve, stop<br>point: *M* conc= *CONC* |
| Bad data in svdcmp weighting | Stopping in svdcmp_new<br>Fatal error in transfer function |
| Missing input | Dispersion type curve data not input |

| | |
|---|---|
| Missing time input for calculating flux averaged concentration | ERROR - delta_time must be entered for flux averaged concentrations<br>STOPPING execution |
| Calculation option not supported for subgridding | ERROR - keyword "mflx" cannot be used with SUBGRID<br>STOPPING execution |
| Output otion not supported for subgridding | ERROR - "node" output cannot be used with SUBGRID<br>STOPPING execution |
| A zero subdivision has been input for subgridding | zero subdivisions in boundary_planes<br>stop |
| Missing input | read_subgrid_info: Inconsistent input:<br>ERROR: Keyword SUBGRID found.<br>ERROR: XYZ data was not found in SPTR2 input file.<br>ERROR: Subgrid calculations require SPTR2 XYZ data.<br>STOPPING execution |
| Bad input for subgrid | read_subgrid_info: Unrecognized input string<br>Expected Real or Integer, Found Character string<br>STOPPING execution |
| Programming error | |
| Illegal call to initdata2 routine. | Fatal error, too many real inputs to initdata2<br>STOPPING Execution |
| Illegal call to initdata2 routine. | Fatal error, too many integer inputs to initdata2<br>STOPPING Execution |
| Error in subgrid calculations | |
| Number of subdivisions is less than 1 for subgridding | ERROR Subgrid Cell Refinement ERROR<br>cell dx = dx scale factor = sfx sg nx= nx<br>cell dy = dy scale factor = sfy sg ny= ny<br>cell dz = dy scale factor = sfz sg nz= nz<br>ERROR-STOP in compute_nxyz-ERROR |
| Number of touched cells is greater than number of points in grid | ERROR: fill_ip_touched_cells<br>n_touched_cells = n_touched_cell<br>n_grid_points = n_grid_points<br>ERROR: fill_ip_touched_cells |
| Cell size for subgridding is 0. | zero cell size in global_local for cell# *i*<br>STOP |
| Time to cross subcell too small | delta-t < epsilon in line_plane_t.STOP<br><br>xin,yin,zin,tin,alam,amue,anue= xin yin zin tin alam amue anue |
| Time to cross subcell is 0 | zero delta-t in parametric_line<br>STOP |
| Number of subdivisions is less than 1 for subgridding | some of nx,ny,nz = 0 for ig= ig subroutine xyz_ijk. STOP |

| Warning Conditions | |
|---|---|
| A zero flux has been found for a node with solute. | `Flux of 0. found for cell with solute output will be negative total moles` |
| Diffusion model used values outside of defined parameter space | `Diffusion model used values outside of defined`<br>`Retention time set to sigma/omega for values less than` *`minsigma`*<br>`or`<br>`Retention time approximates error function solution`<br>`parameter space:`<br>`min sigma =` *`minsigmap`* `max sigma =` *`maxsigmap`* `min omega =` *`minomegap`* `max omega =` *`maxomegap`*<br>`Range needed:`<br>`min sigma =` *`minsigma`* `max sigma =` *`maxsigma`* `min omega =` *`minomega`* `max omega =` *`maxomega`* |
| Values selected will result in subgridding to larger extent than practicle | `Subgrid Cell Refinement WARNING`<br>`Input will result in a large number of subgrid refined cells.`<br>`cell dx = dx scale factor = sfx sg nx= nx`<br>`cell dy = dy scale factor = sfy sg ny= ny`<br>`cell dz = dy scale factor = sfz sg nz= nz` |
| Bad input format for subgrid data | `read_subgrid_info: WARNING`<br>`WARNING: Found a blank line instead of a keyword.`<br>`WARNING: Attempt to continue after the blank line.` |

## 5.8 HARDWARE/SOFTWARE ENVIRONMENTS

No special hardware features or environments are required by the software. The code will run on Linux 2.4.21 or higher workstations. For other platforms please contact LANL. 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.

## 5.9 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. Unless otherwise noted, the examples, presented here and in Section 6, used resident concentration calculations and the output presented is for the mobile resident concentration. 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 $\Delta z = 5m$.

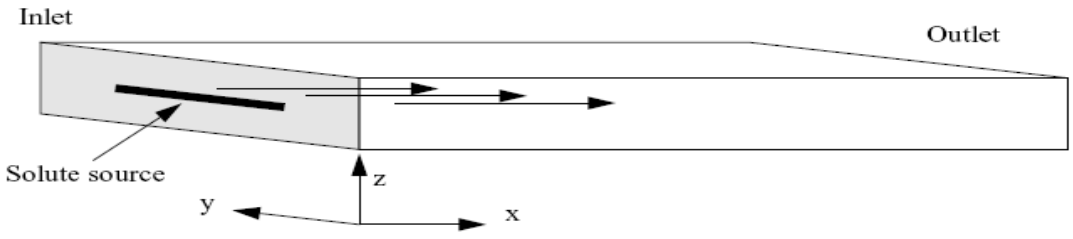


**Figure 1. Schematic of the transport test problem used to demonstrate the CBPT method.**

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 29 *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 $\alpha_L$ of 100 *m*, transverse horizontal dispersivity $\alpha_T^H$ of 10 *m*, and transverse vertical dispersivity $\alpha_T^V$ 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.

### 5.9.1 Single-Source Example, Steady State Resident Concentration

In the first example, 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 $10^{-3}$ mole/liter.

Two FEHM particle tracking simulations were run that used a particle distribution at the inlet over a two-dimensional patch at *x = 0 m*, *y = –500 to 500 m*, and *z = –242.5 to –-257.5 m*. to generate the particle tracks. For the first simulation 50x20 particles (a total of 1000 particles, output file plume3_fine1k.sptr2), while for the second 500x200 particles (a total of 100000 particles, output file plume3_fine100k.sptr2) were used. This defined 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 $10^{-3}$ mole/liter for the injected fluid in the patch, the fluid flow rate through the region defined by the patch is determined. The concentration ($10^{-3}$ 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 (148.15 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 40 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 in the simulation control file plume3_fine.sim. 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 m.

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. For a complete view of the input files, please check the actual input files supplied with the executable. The simulation was run twice, once for each particle input file, **to illustrate the importance of using enough particles to yield reasonable results**.

I/O input file: plumecalc.files

```
fine.grid
unformatted
fine.stor
binary
plume3_fine_NUM_.sptr2
rock.macro
plume3_fine.sim
plume3_fine_NUM_.dat
```

where *NUM* is 1k or 100k

Simulation control input file: plume3_fine.sim

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

Rock property input file: rock.macro

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

Solute mass flux input file: plume3.mdot

```
1.     40.0
1.e7   40.0
```

The downstream concentrations at various *z* values are shown for *x = 4800 m, y = 0 m, x = 9600 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) when 100000 particles are used. Dispersion horizontally and vertically have reduced the concentration to a fraction of its inlet value as particles move away from the center of the plume. For 1000 particles the results are more scattered and illustrate the need to use enough particles to minimize jaggedness in the concentration calculations. .

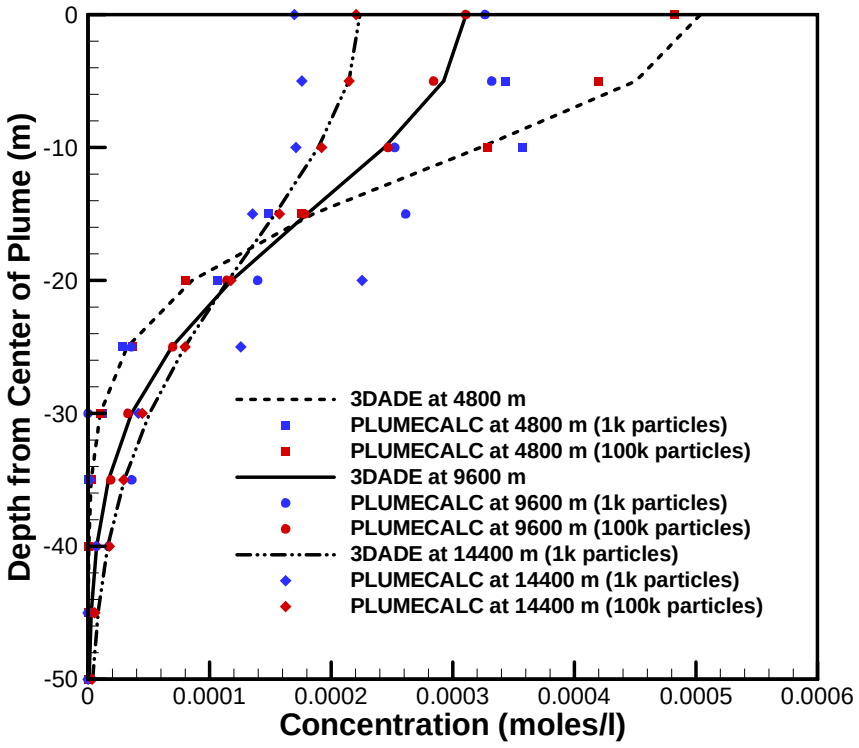


**Figure 2. Comparison of the CBPT method and the analytical solution of Leij et al. (1991)) at t = 1000 years.**

### 5.9.2 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 (kd = $2.66 \times 10^{-2}$), and a solute that decays with a half life of 250 yr (k = $7.590934 \times 10^{-6}$ $day^{-1}$). As before each simulation was run using both particle input files.

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
```
fine.grid
unformatted
fine.stor
binary
plume3_fine_NUM_.sptr2
rock.macro
plume3_resbtc1.sim
plume3_resbtc1_NUM_.dat
```

plumecalc.files.resbtc2
```
fine.grid
unformatted
fine.stor
binary
plume3_fine_NUM_.sptr2
rock_rfac3.macro
plume3_resbtc2.sim
plume3_resbtc2_NUM_.dat
```

plumecalc.files.resbtc1
```
fine.grid
unformatted
fine.stor
binary
plume3_fine_NUM_.sptr2
rock.macro
plume3_resbtc3.sim
plume3_resbtc3_NUM_.dat
```

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

plume3_resbtc1.sim
```
1  0.
1
1
plume3.mdot
3.6525e5        0
46
   127838
   131490
   135142
   138795
   .
   .
   .
   281242
   284895
   288548
   292200
node
1
  87818
```

plume3_resbtc2.sim
```
1  0.
1
1
plume3.mdot
3.6525e5        0
46
   383512
   394470
   405428
   416385
   .
   .
   .
   843728
   854685
   865642
   876600
node
1
  87818
```

plume3_resbtc3.sim
```
1  7.590934e-6
1
1
plume3.mdot
3.6525e5        0
46
   127838
   131490
   135142
   138795
   .
   .
   .
   281242
   284895
   288548
   292200
node
1
  87818
```

Rock property input file rock_rfac3.macro
```
rock
   1  0  0  2530. 3.2.66e-2   0.03361
```

In this case, output at a single node (number 87818) 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 shows that the model reproduces the analytical solution well, with the exception of a mismatch at the plateau for the simulations using 1000 particles, caused by the inherent inaccuracy of performing particle tracking simulations. Other realizations would be expected to perhaps under predict the plateau. A greater number of particles, as expected, yields 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**.

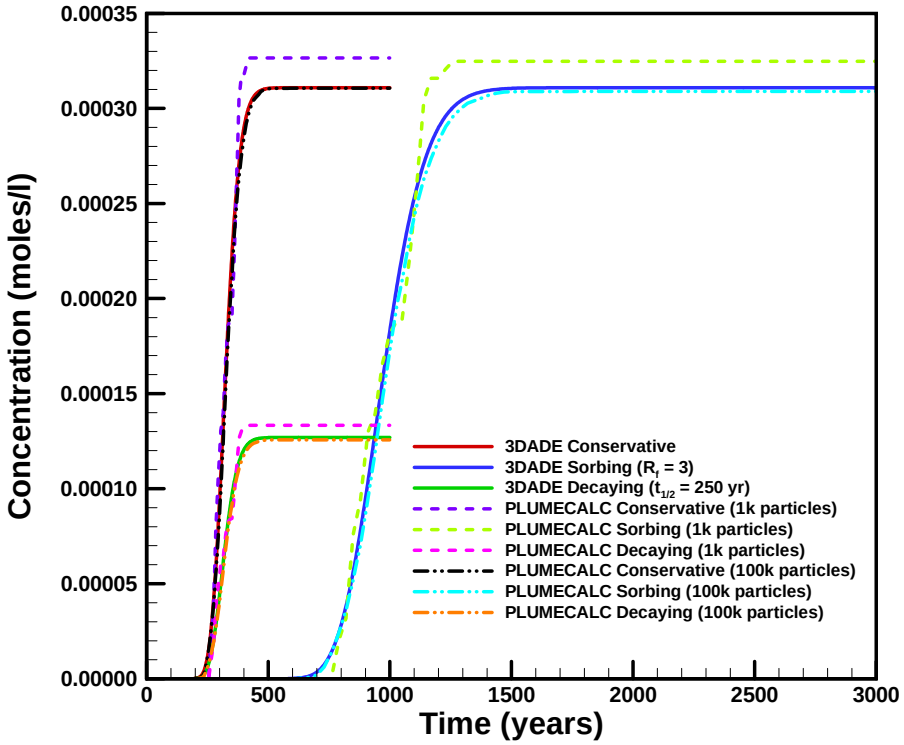


**Figure 3. Comparison of the CBPT method and the Leij et al. (1991) analytical solution for mobile resident concentration versus time at x = 9600 m, y = 0 m, z = -250 m. Conservative, sorbing ($R_f$ = 3), and decaying ($t_{1/2}$ = 250 yr) solutes are simulated in response to a constant injection concentration.**

### 5.9.3 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. The simulation input file 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.

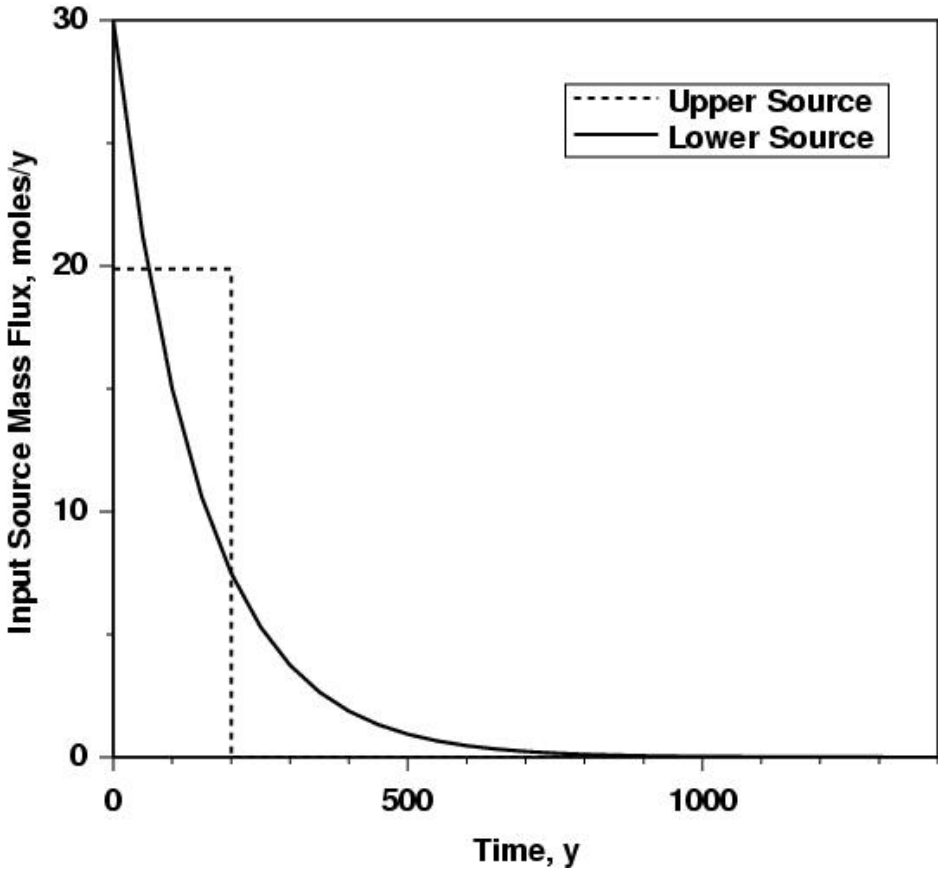


**Figure 4. Input mass flux versus time for the two solute sources in the multiple source example problem.**

The input files for this simulation are discussed below:

I/O file: plumecalc.files

Resident calculations

```
fine.grid
unformatted
fine.stor
bin
example_plume.sptr2
```

Flux averaged concentrations

```
fine.grid
unformatted
fine.stor
bin
example_plume.sptr2
```

```
rock.macro
example2_res*.sim
example2_res*.dat
```

Where res* represents the 3 cases run: res (source 1 and source 2), res_s1only and res_s2only

```
rock.macro
example2_favg*.sim
example2_favg*.dat
```

Where favg* represents the 2 cases run: favg (conservative) and favgrfd (sorption and decay)

Simulation files:

example2_res.sim

```
2 0.
1  50087
1  1
../plume_runs/input/example1_source1.mdot
../plume_runs/input/example1_source2.mdot
1.09575e6   0
7
73000
146000
219000
292000
365000
438000
511000
tecp
```

Note that `example1_source0.mdot` replaces the second source file for example2_res_s1only,sim and the first source file for example2_res_s2only.sim.

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
```

Note that 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.

Solute mass flux input files:

example1_source0.mdot

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

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.
```

The plume for the combined sources was generated using the resident concentration calculation. Two additional runs were made for the resident concentration calculations such that one of the source functions was set to zero (example0_source2.mdot) so the single source plumes could be compared to the plume that developed with the combined sources. The input follows the format specified in Section 5.3.6 above. For the final calculations, 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. 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.

Figure 5 presents an isoconcentration plot for a conservative, non-decaying solute at 200 years where C = 1.0e-5 moles/l. The plot which shows the domain for this and subsequent illustrations, utilizes a 10X vertical exaggeration to aid in the visualization. Although the maximum concentration computed in this simulation (1.14e-4 moles/l) is about 1 order of magnitude greater than this value, this value was chosen to better

illustrate the extent of the plume, as were the values chosen for the subsequent isoconcentration plots. Concentration profiles (at x = 10000 m and y = 0 m) for this plume are shown in Figure 6.

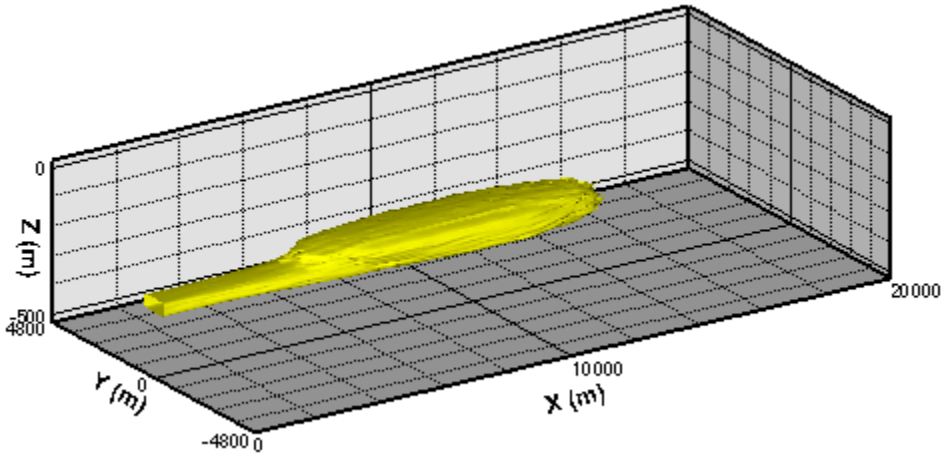


**Figure 5. Isoconcentration surface predicted using the CBPT method for the multiple source example problem (with 10X vertical exaggeration) at 200 years for C = 1.0e-5 moles/l.**

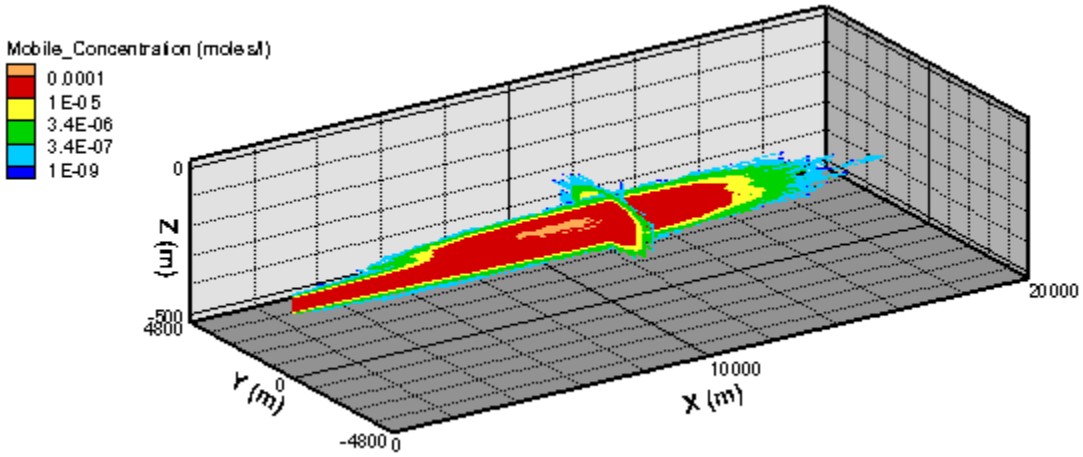


**Figure 6. Concentration profiles at 200 years for the plume predicted using the CBPT method for the multiple source example problem.**

Isoconcentration plots for each individual source and the combined sources are shown in Figure 7 and Figure 8. The plume isoconcentration surfaces plotted are for C = 3.4e-6 moles/l at 200, 400 and 800 years, and C = 3.4e-7 moles/l at 1000 years. For reference, the mean travel time to reach the exit plane is about 694 years. At 200 years (Figures 7a, b and c), both sources have injected mass continuously, and the plume has progressed roughly one third of the way through the model. The plots at 400 years (Figures 7d, e and

f) 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 combined plume near the source region is narrower in the vertical direction because the upper source has been turned off.

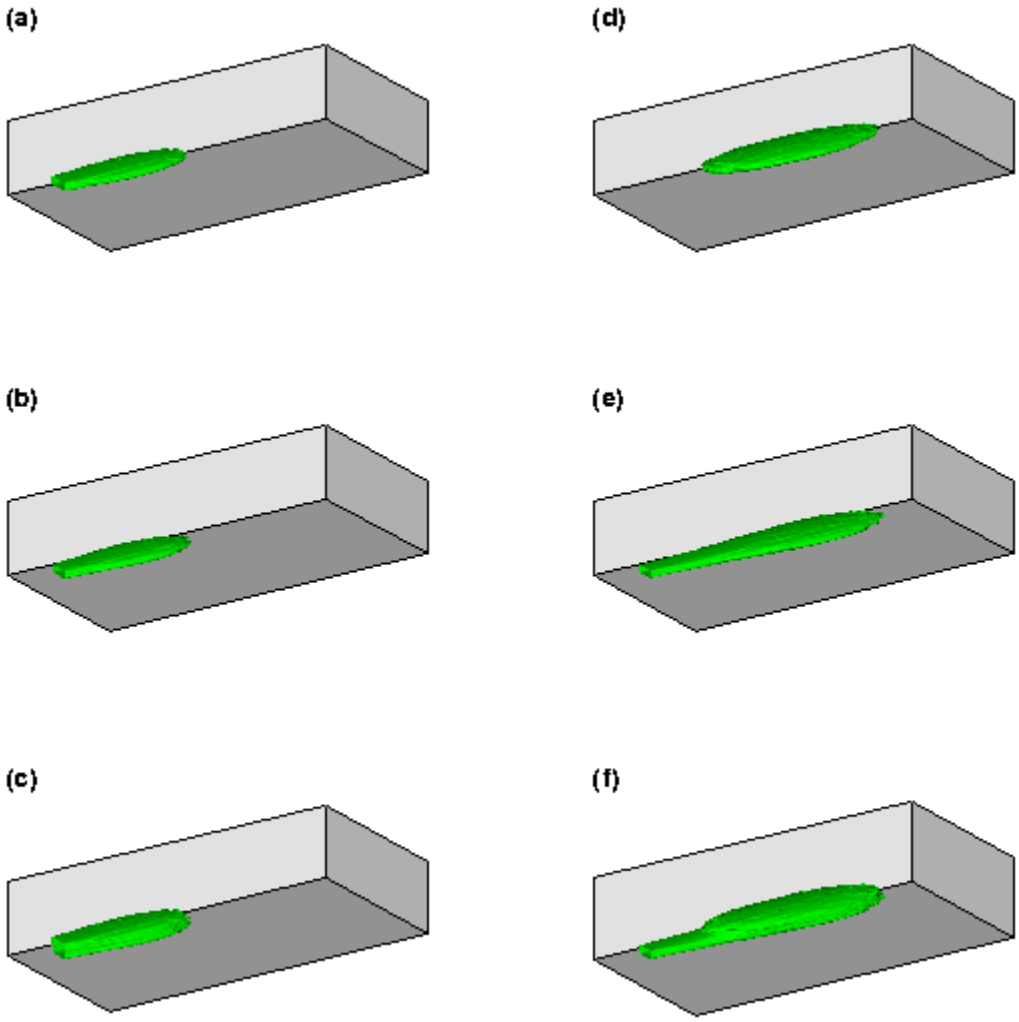


**Figure 7. Isoconcentration surfaces predicted using the CBPT method for the multiple source example problem. Isosurfaces are for C = 3.4e-6 moles/l. Surfaces a and d represent the upper source, b and e the lower source, and c and f the combined sources at 200 and 400 years respectively.**

At 800 years and 1000 years (Figure 8), 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 8d, e and f, which plot the isoconcentration surface of a lower value, show that the method can reproduce this portion of the plume evolution at the tail end of a simulation. This feature of the CBPT method is very difficult to reproduce using a conventional method using particles.

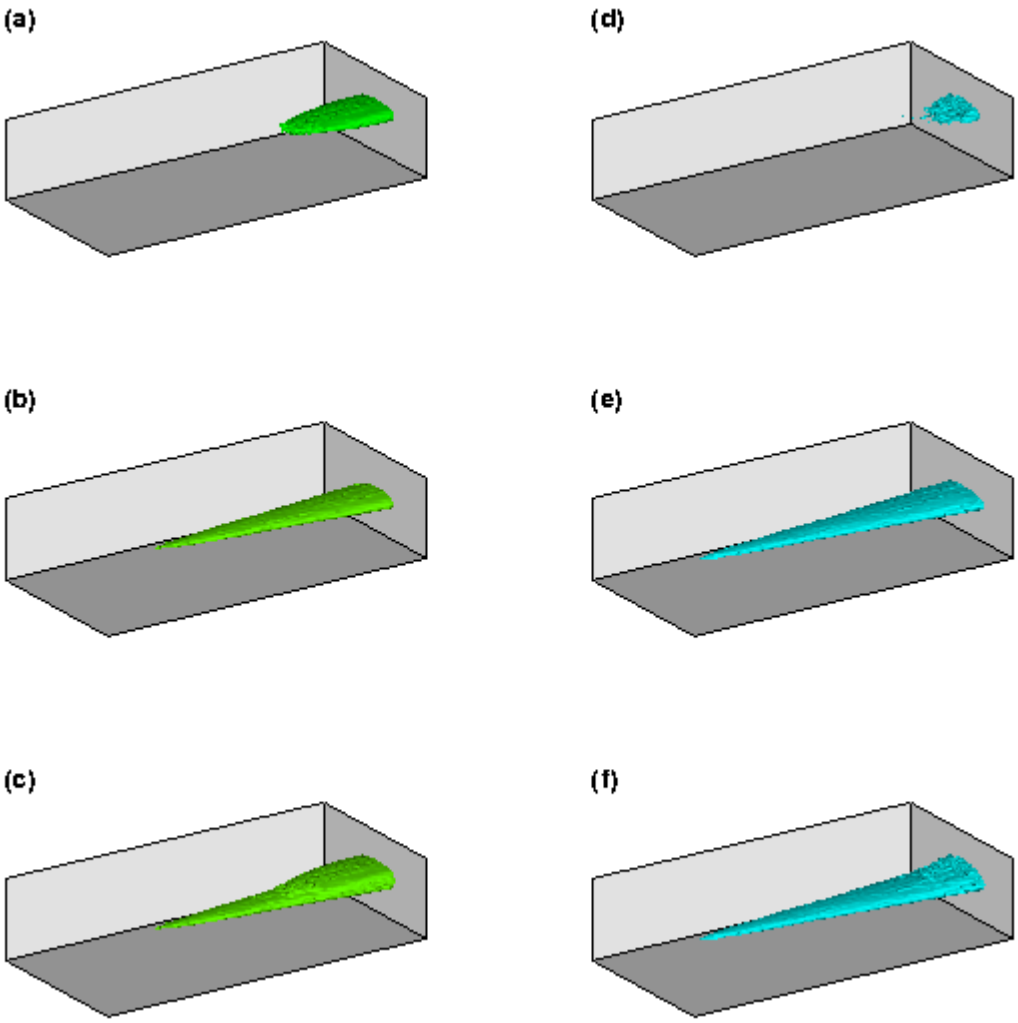


**Figure 8. Isoconcentration surfaces predicted using the CBPT method for the multiple source example problem. Isosurfaces are for C = 3.4e-6 moles/l at 800 years and C = 3.4e-7 moles/liter at 1000 years, respectively. Surfaces a and d represent the upper source, b and e the lower source, and c and f the combined sources.**

Figure 9 presents 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.

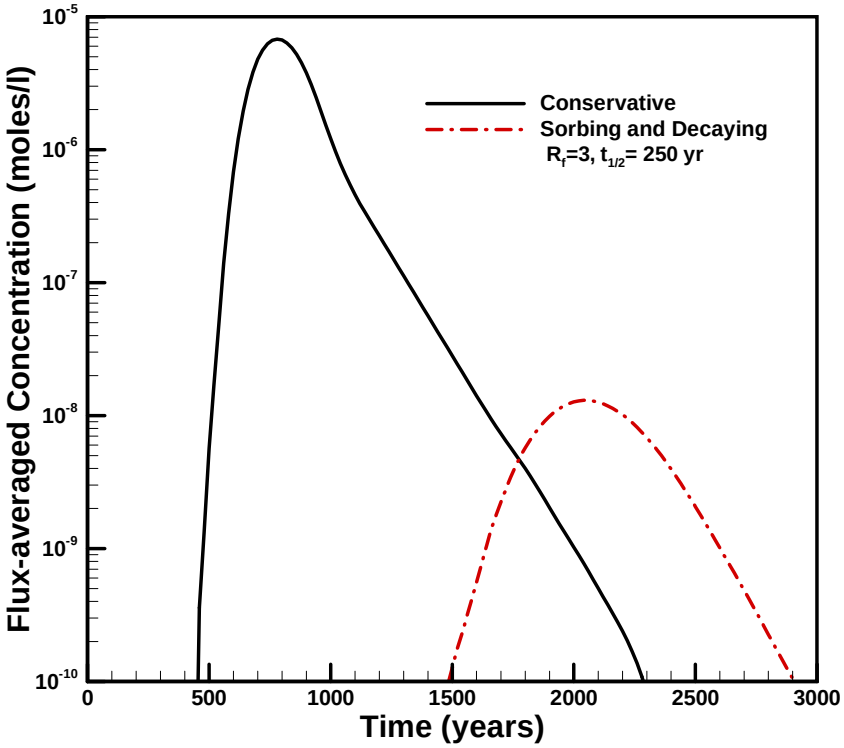


**Figure 9. Flux-averaged concentration versus time at the exit of the model for the multiple source example problem. A conservative solute and a solute that sorbs and decays ($R_f$ = 3, $t_{1/2}$ = 250 yr) are simulated.**

# 6. INSTALLATION

## 6.1 INSTALLATION AND INSTALLATION VERIFICATION

### 6.1.1 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.

### 6.1.2 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 for resident mobile concentration (using the node option, see Section 5.4). The concentration values may vary slightly depending on which platform the test was executed:

```
1        52124
182625.000
13054   0.00000000
```

```
16791   0.00000000
20528   0.00000000
24265   0.00000000
28002   0.00000000
31739   0.00000000
35476   0.00000000
39213   0.00000000
42950   0.00000000
46687   0.00000000
50424   0.00000000
54161  0.955568175E-06
57898  0.105737302E-05
61635  0.347376132E-05
65372  0.108570803E-04
69109  0.338225480E-04
72846  0.902014877E-04
76583  0.183095868E-03
80320  0.304497713E-03
84057  0.414230545E-03
87794  0.484733350E-03
13078   0.00000000
16815   0.00000000
20552   0.00000000
24289   0.00000000
28026   0.00000000
31763   0.00000000
35500   0.00000000
39237   0.00000000
42974  0.134651958E-06
46711  0.469073422E-06
50448  0.160691139E-05
54185  0.344961792E-05
57922  0.713647960E-05
61659  0.169681127E-04
65396  0.395285967E-04
69133  0.679952741E-04
72870  0.117095476E-03
76607  0.171625291E-03
80344  0.230319547E-03
84081  0.294963923E-03
87818  0.305239329E-03
13102   0.00000000
16839   0.00000000
20576   0.00000000
24313   0.00000000
28050   0.00000000
31787   0.00000000
35524   0.00000000
39261  0.274879187E-06
42998  0.834442280E-06
46735  0.132840457E-05
50472  0.473978938E-05
54209  0.759414188E-05
57946  0.156807056E-04
61683  0.304717958E-04
65420  0.541233650E-04
69157  0.798494512E-04
```

```
72894  0.121522870E-03
76631  0.151980166E-03
80368  0.186317896E-03
84105  0.204160495E-03
87842  0.216205000E-03
```

## 6.2 VALIDATION TESTS

The example problems presented in 5.9 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, computing on an unstructured grid, and subgridding are included below.

### 6.2.1 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, 2003b, 2003c). This is the same model used for the example tests described in Section 5.9, 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 (excluding diffusion) are the same as for the examples in Sections 5.9.1 and 5.9.2. 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 ($\alpha_{long}$) 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 10 shows the comparison of PLUMECALC mobile resident concentration breakthrough curves for the case of no dispersion (sptr_long3ndd.sptr2) with and without diffusion and Figure 11 for the case with longitudinal dispersion (sptr_long3dsp.sptr2) with and without diffusion. A diffusion factor of 1.e-13 $m^2/s$ was used. 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.

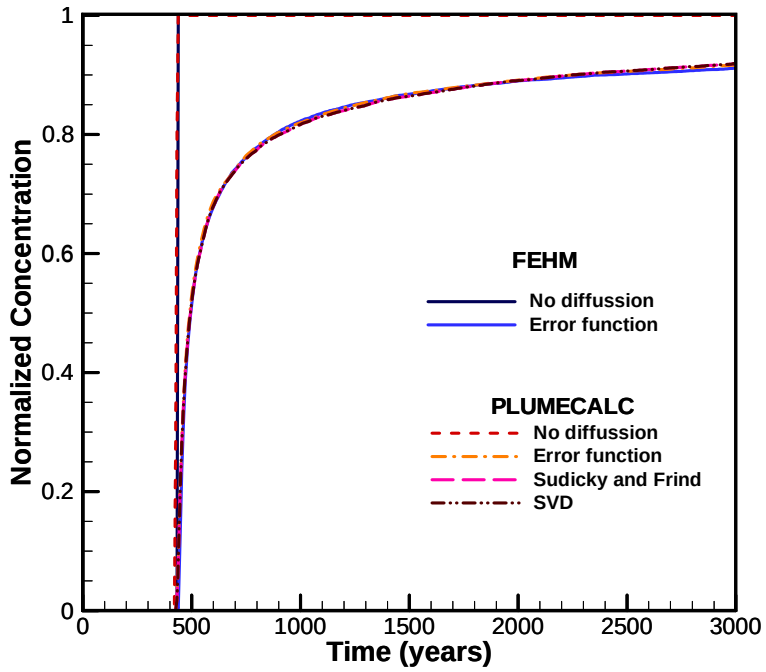


**Figure 10. Comparison of PLUMECALC simulation of breakthrough with output from FEHM for the case of no dispersion.**

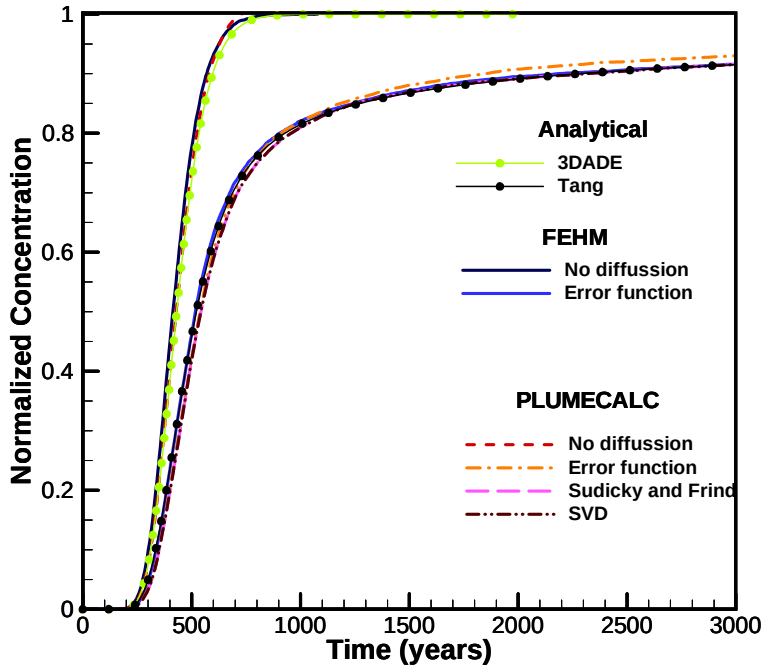


**Figure 11. Comparison of PLUMECALC simulation of breakthrough with output from FEHM. Longitudinal dispersion ($\alpha_{long}$) was set to 500 m. For the case without diffusion the 3DADE analytical solution is shown and for the case with diffusion the Tang analytical solution is shown.**

### 6.2.2 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 (316 x 316 particles). 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 Figures 12 to 17, contours of concentration are shown for a horizontal plane at z = 4400m (within the zone of refinement for the OMR grid) and a vertical region for x = 4400 m. Figure 12 and Figure 15 show that for the regular grid uniform concentrations are achieved, as is expected for this simple test problem. Figures 13 and 16 illustrate the problem of using the “non”-brick shaped control volumes when computing the concentrations. Although Figure 13 and Figure 14 are very similar, a comparison of Figure 16 and Figure 17 shows the improved performance for the OMR grid when the modified control volumes are used. Except for edge effects 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.

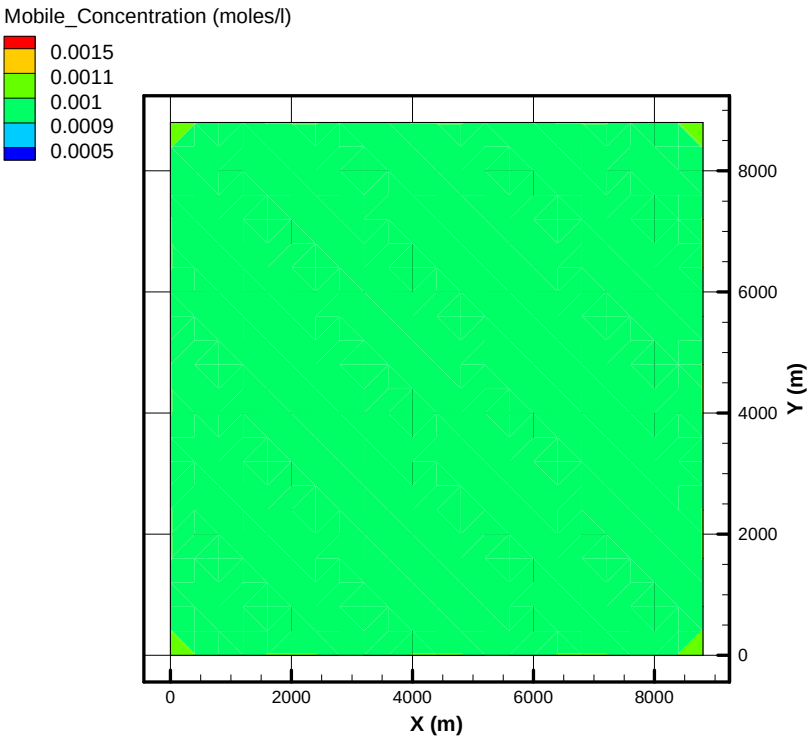


**Figure 12. Concentration at z = 4400 m for the structured grid.**

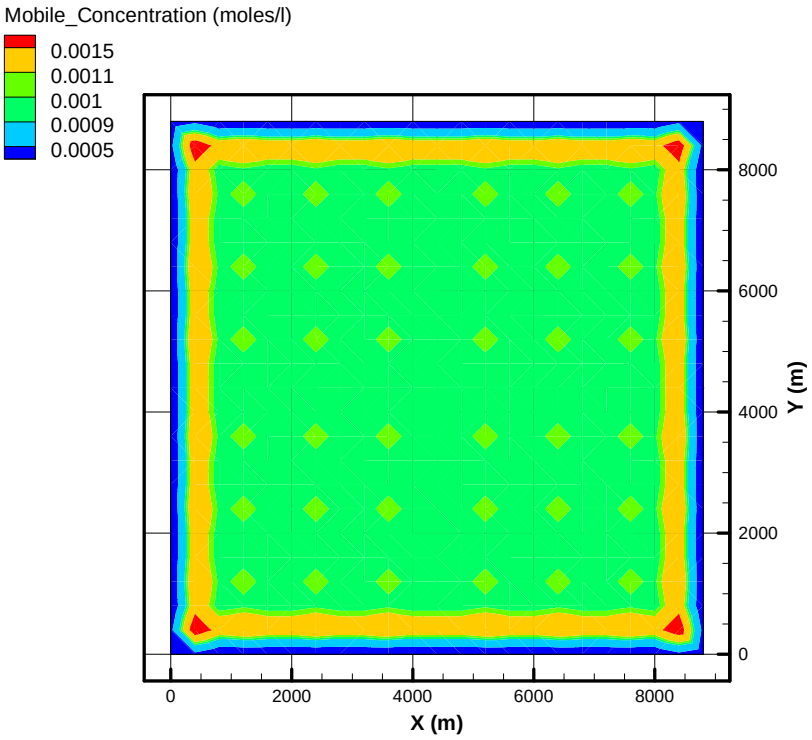


**Figure 13. Concentration at z = 4400 m for the omr grid using FEHM node volumes.**

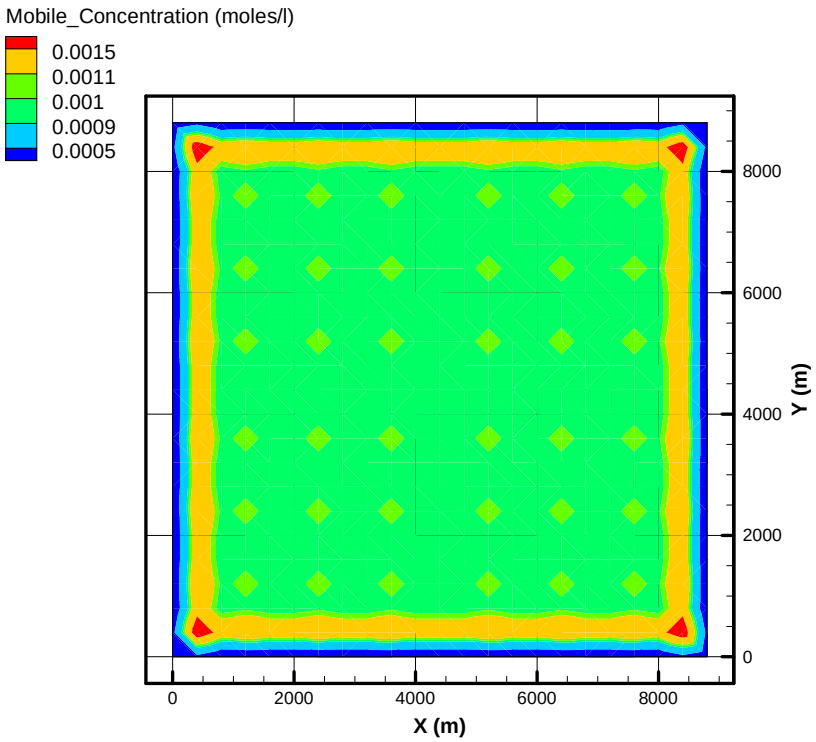


**Figure 14. Concentration at z = 4400 m for the omr grid using approximate brick-shaped control volumes.**

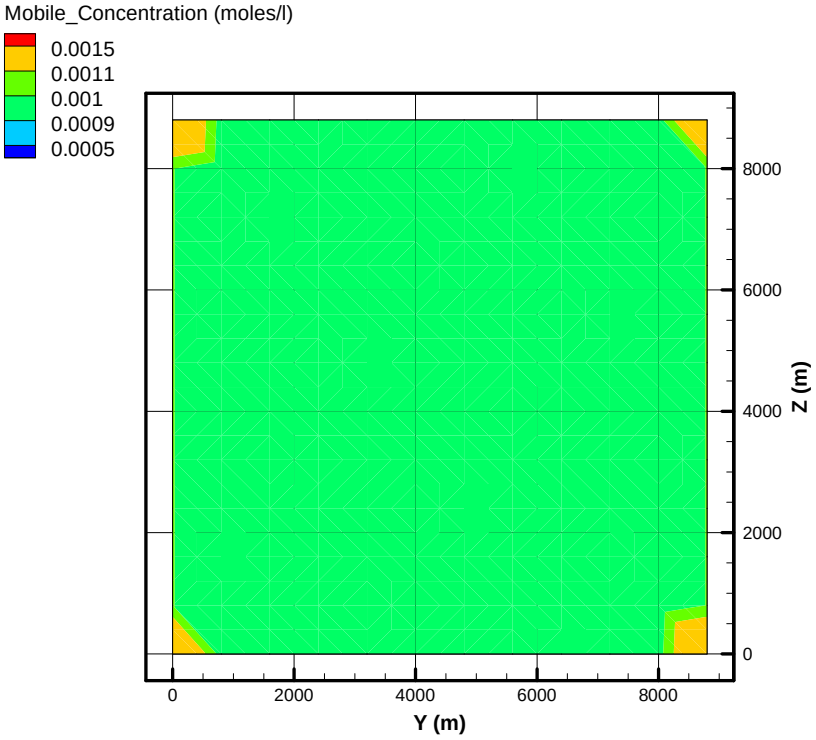


**Figure 15. Concentration at x = 4400 m for the structured grid.**

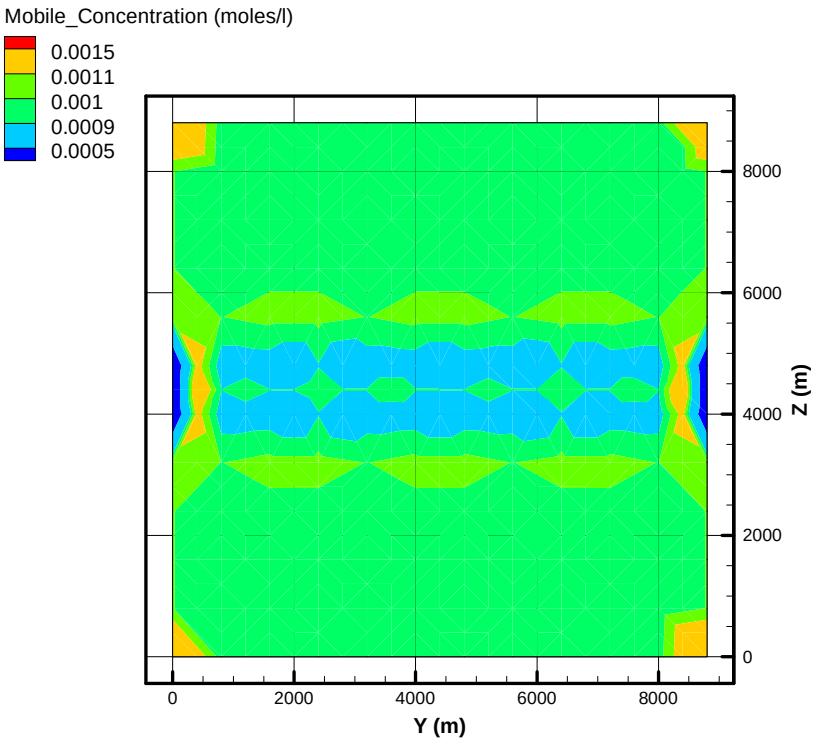


**Figure 16. Concentration at x = 4400 m for the omr grid using FEHM node volumes.**

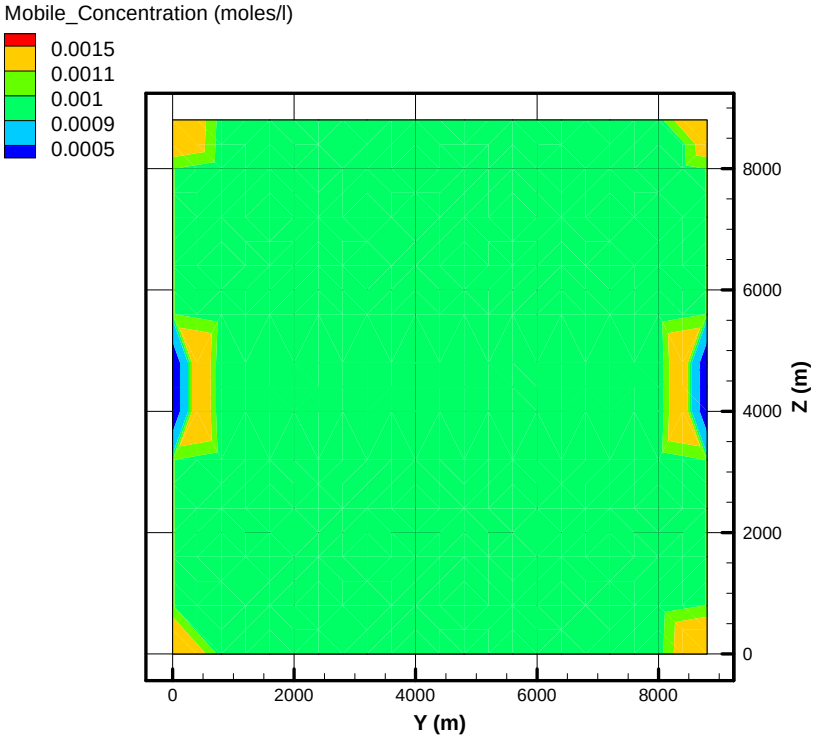


**Figure 17. Concentration at x = 4400 m for the omr grid using approximate brick-shaped control volumes.**

## 6.3 SUBGRID EXAMPLES

### 6.3.1 Subgrid cube example

A simple three-dimensional model (160 m cube) with uniform properties and one dimensional flow in the z direction was used. The problem domain is represented by a 3-D tetrahedral grid (Figure 18), with 9x9x9 grid points (Figure 19). Figure 20 and Figure 21 show the steady-state pressure and velocity fields. For the streamline particle tracking simulation, 6561 particles are uniformly distributed over a centrally located patch, encompassing 9 grid points, on the top surface of the cube (Figure 22) with an area of 160 $m^2$ (Figure 23).

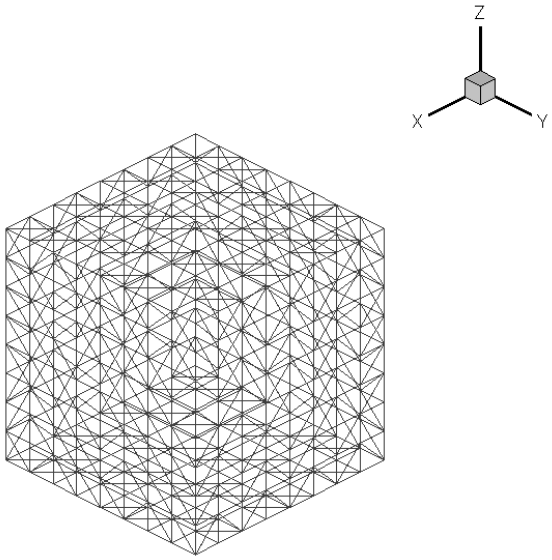


**Figure 18. Subgrid test problem, 9x9x9 tetrahedral grid.**

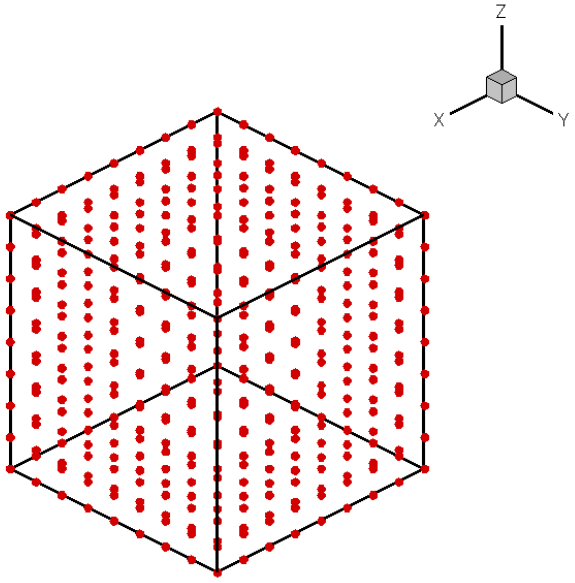


**Figure 19. Subgrid cube test problem, gridpoints of the 9x9x9 tetrahedral grid.**

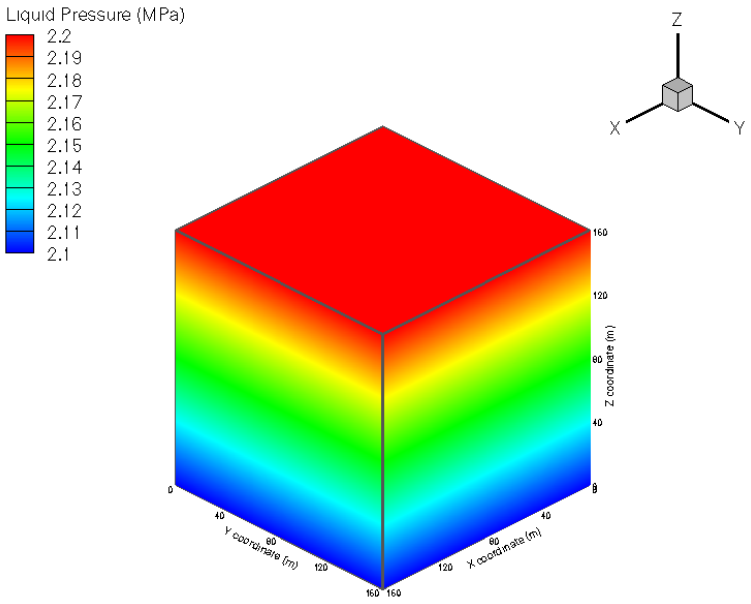


**Figure 20. Subgrid cube test problem steady-state pressures.**

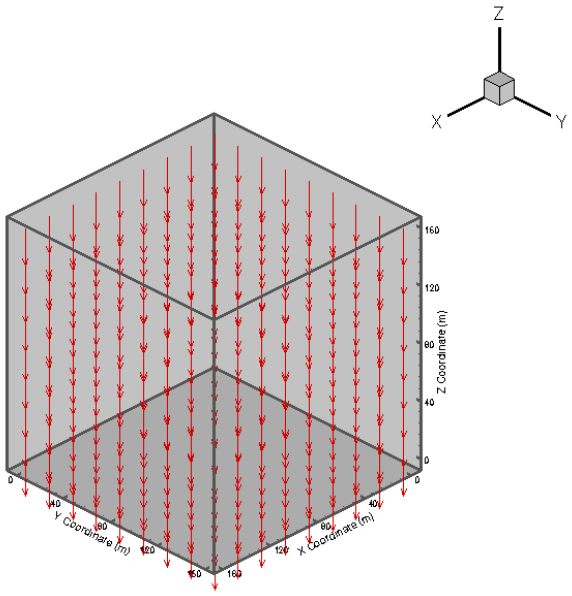


**Figure 21. Subgrid cube test problem velocity field.**

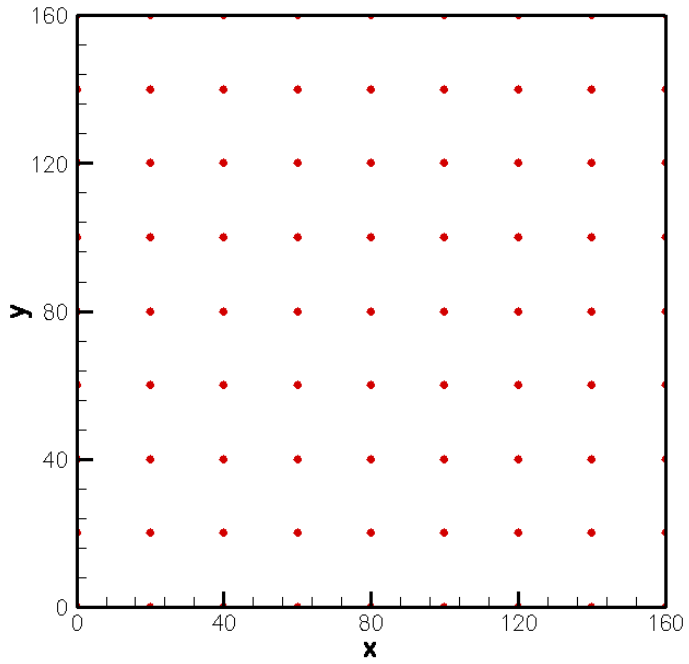


**Figure 22. Subgrid cube test problem, grid points on particle injection surface.**

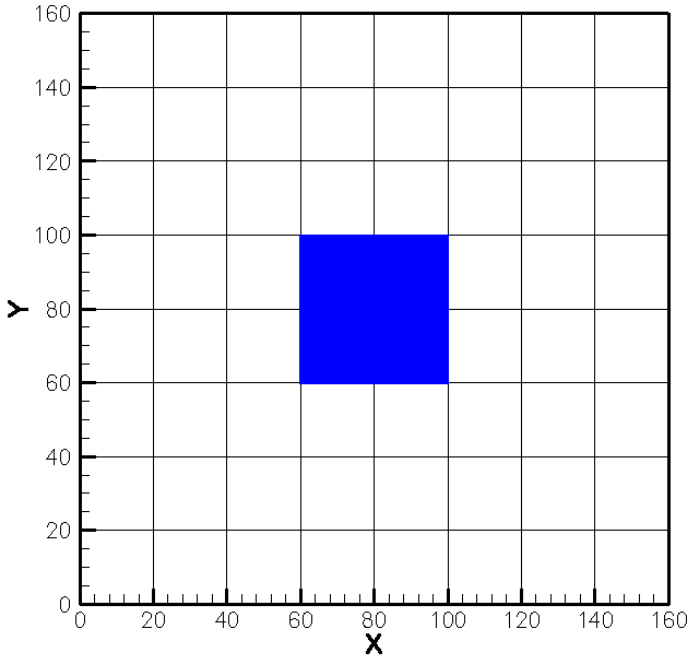


**Figure 23. Subgrid cube test problem, particle injection patch. The area of injection encompasses the 9 grids points in the center of the top surface.**

Using the results from the streamline particle tracking simulation, the following PLUMECALC runs were performed with two different mass flux models using both flux averaged and resident concentration calculations: 1) PLUMECALC without subgridding, 2) PLUMECALC with subgridding, 1x1x1 refinement, 3) PLUMECALC with subgridding, 2x2x2 refinement, and 4) PLUMECALC with subgridding, 4x4x4 refinement. The first mass flux model used a constant solute source (steady.mdot), and the second a variable source (variable_source.mdot). A comparison was made of the results for PLUMECALC without subgridding and PLUMECALC with subgridding using a 1x1x1 refinement. The 1x1x1 refinement although spacially the same as a run without refinement, causes the code to utilize the subgridding routines. Particle velocities across the cell are interpolated based on the particle entry and exit locations using the Pollock control volumes, determined in the FEHM run, which can lead to minor differences with the run without subgridding. Mobile concentration results for the runs with and without subgridding were numerically compared and the maximum difference in concentration between the runs was less than 1% for both the constant and variable solute source.

The mass flux input files used for these tests are shown below:

```
steady.mdot
0.       40.
1.e10    40.
```

```
variable_source.mdot
#Time       moles/day
0           60
18262.5     42.4264
36525       30
54787.5     21.2132
73050       14.9999
91312.5     10.6065
109575      7.49994
127838      5.30324
146100      3.74996
164362      2.65162
182625      1.87497
200888      1.3258
219150      0.937484
237412      0.6629
255675      0.46874
273938      0.33145
292200      0.23437
310462      0.165724
328725      0.117185
346988      0.0828618
365250      0.0585922
438300      0.0146479
511350      0.00366196
600000      0
1.09575e+06 0
```

Figures 24 – 27 present the results for the subgridding example simulations. Figure 24 illustrates the concentration profile obtained for PLUMECALC without subgridding. The concentration in the middle of the profile is as would be expected with dilution as you move to the edge of the injection zone because the size of the control volumes along the edges are greater than the area that the particles actually traverse. Figure 25 shows the mobile resident concentrations across the exit plane for a run without subgridding, a

1x1x1 subgrid run which is basically the same as the simulation without subgridding, a 2x2x2 subgrid run, and 4x4x4 subgrid run. With no subgridding or a 1x1x1 subgrid, the concentration at the center matches the concentration one would expect for the constant flow and the concentrations around the edges are diluted due to the fact that the control volumes exceed the boundary of the injection patch as noted above. When 2x2x2 and 4x4x4 subgridding are used, a uniform concentration is obtained because the subgrid area matches the injection patch. The slight variations in concentration seen for the 4x4x4 subgrid are due to distribution of particles in the subgrid and can be reduced by increasing the number of particles used. It also should be noted that the concentrations plotted are represented by bands of color which also enhance this effect. Figure 26 and Figure 27 present the mobile resident concentration results for the variable flux simulation at two simulation times over the extent of the grid (x, y = 40 – 100 m, and z = 0 – 160 m). As with the uniform flux case, it can be seen that the concentrations of the plume are diluted along the outer edges for the 1x1x1 subgrid runs while they are fairly uniform for the 2x2x2 and 4x4x4 subgrid runs.

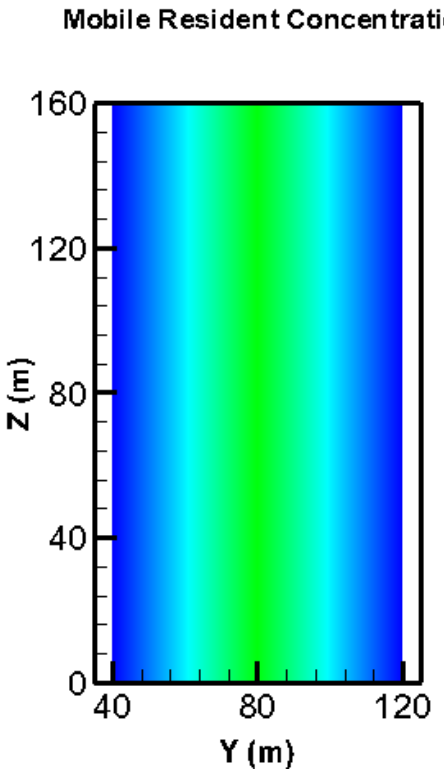


**Figure 24. Steady state concentrations for a constant mass flux rate PLUMECALC run without subgridding.**

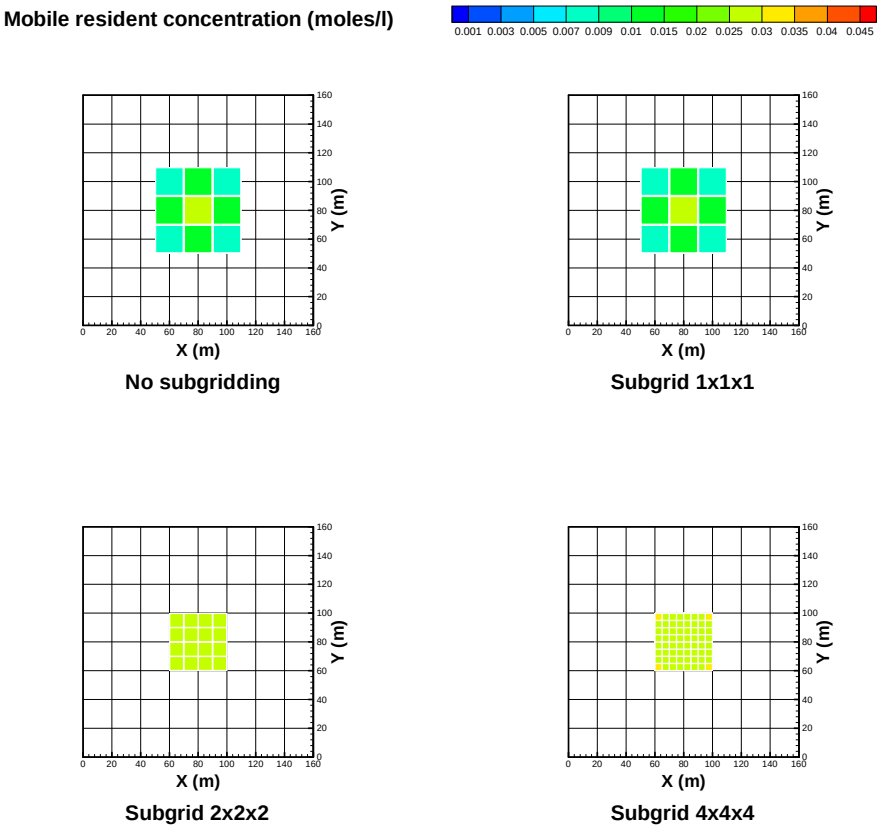


**Figure 25. Steady state resident concentration at exit plane from subgrid PLUMECALC run using no refinement, 1x1x1 (equivalent to regular PLUMECALC), 2x2x2, and 4x4x4 refinement. Each square represents the control volume associated with the grid point.**

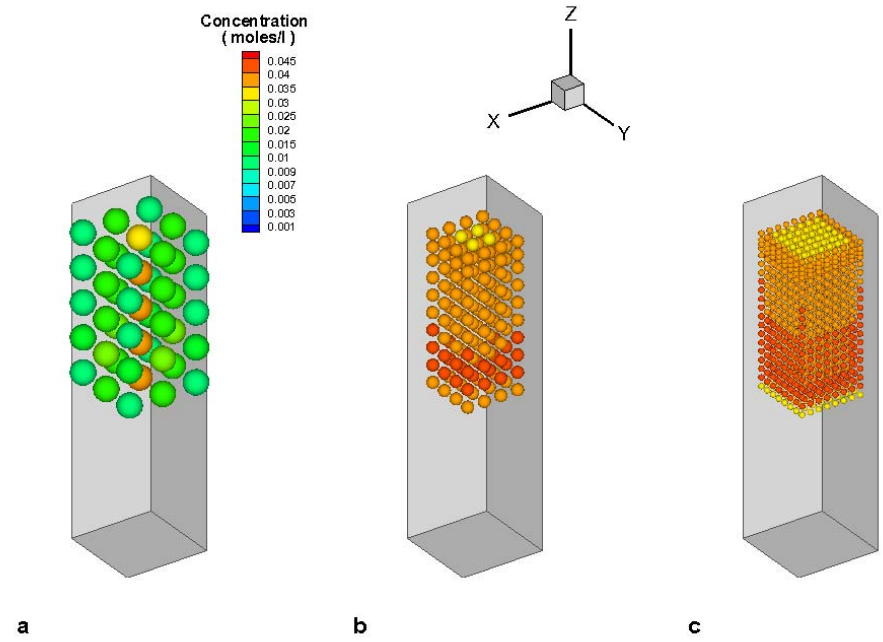


**Figure 26. Subgrid PLUMECALC mobile resident concentrations at 1.e4 days for a variable mass flux rate run: a) 1x1x1 refinement, b) 2x2x2 refinement, c) 4x4x4 refinement. The sphere size represents the relative size of the control volume.**

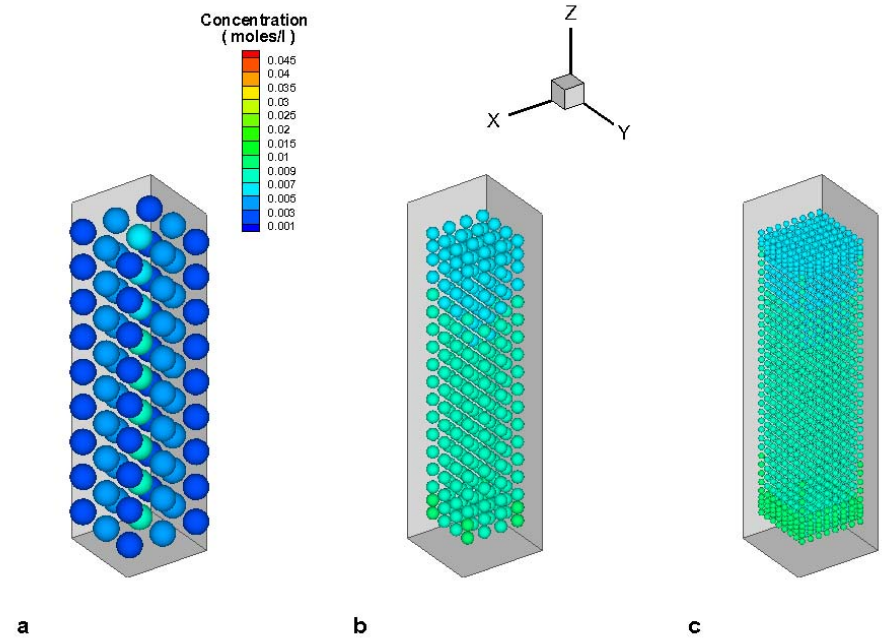


**Figure 27. Subgrid PLUMECALC mobile resident concentrations at 1.e5 days for a variable mass flux rate run: a) 1x1x1 refinement, b) 2x2x2 refinement, c) 4x4x4 refinement.**

### 6.3.2 Complex grid subgrid example.

A streamline particle tracking run (without dispersion) was set up using the Frenchman Flat grid to demonstrate subgridding for a more complex grid. 20000 particles were injected in a region just below the water table over an area perpendicular to the projected flow (Figure 28). The resulting particle positions for the streamline particle tracking run are shown in Figure 29 colored by travel time. These results were used for the following PLUMECALC runs using the mass flux input for $^{14}C$ shown in Figure 30: 1) PLUMECALC without subgridding, 2) PLUMECALC with 1x1x1 refinement, 3) Subgrid PLUMECALC with 2x2x2 refinement, and 4) Subgrid PLUMECALC with 5x5x5 refinement.

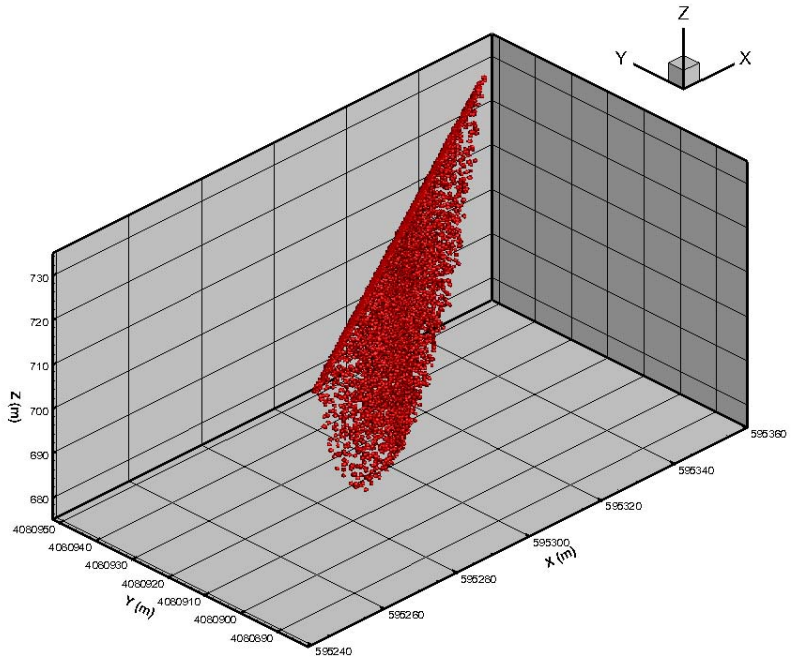


**Figure 28. Particle injection positions for the Complex grid streamline particle tracking test run.**

Resident concentrations were output for 14 times from 10 to 1100 years for PLUMECALC without subgridding and subgrid PLUMECALC run with 1x1x1 refinement. The subgrid PLUMECALC runs using 2x2x2 and 5x5x5 refinement were output for a single time representing concentrations at 500 years. The concentration results for the regular PLUMECALC runs were numerically compared to the subgrid PLUMECALC runs with 1x1x1 refinement to demonstrate that the subgrid technique will work on a complex grid. Figure 31 shows the PLUMECALC output cell locations. The circles on the plot represent the locations where concentration differences were greater than 1%. The differences between regular PLUMECALC without subgridding and subgrid PLUMECALC with 1x1x1 refinement are due to the slight differences that result between the node locations and subgrid calculated cell center positions. Only a small

number of locations (19 total) had concentration differences that exceeded 1%, and the locations with the greatest differences were near the injection plane (Figure 31).

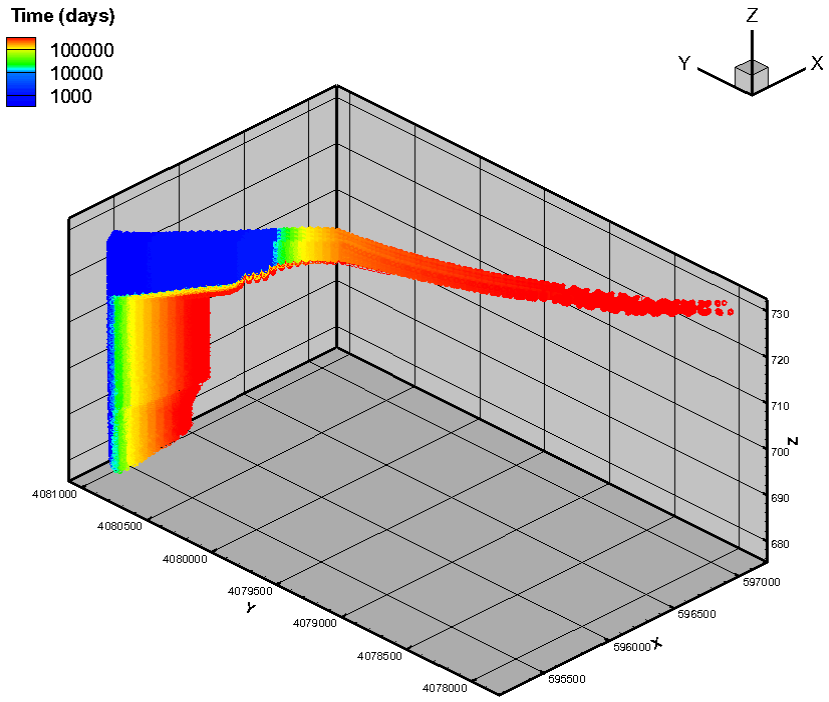


**Figure 29. Particle positions with time for the streamline particle tracking test run.**

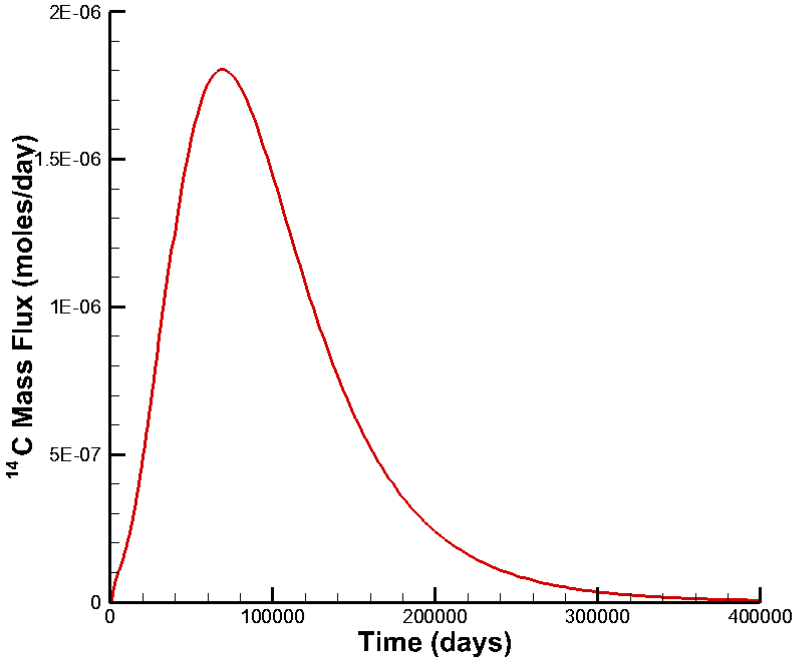


**Figure 30. $^{14}$C Mass flux input function used for the PLUMECALC model runs.**

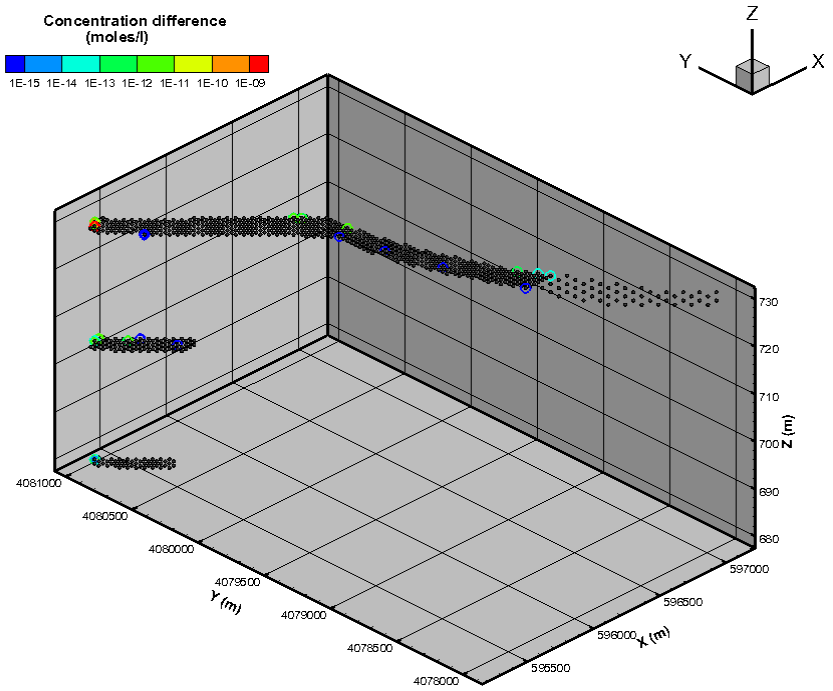


**Figure 31. PLUMECALC positions for output of resident concentration. The circles represent locations where the numerical differences with the subgrid PLUMECALC concentrations were greater than 1%.**

Figures 32 through 34 show the output concentrations for the 3 subgrid cases at 500 years. The sphere size on each plot is proportional to the output cell control volume. As can be seen from Figure 33 (subgrid PLUMECALC with 2x2x2 refinement) and Figure 34 (subgrid PLUMECALC with 5x5x5 refinement) the increasing refinement more closely captures the particle distribution seen in the streamline particle tracking run (Figure 29).

### 6.3.3 Complex grid subgrid example with dispersion, diffusion, and retardation.

An additional streamline particle tracking run was made to include dispersion and provide a basis for illustrating mobile resident concentration when dispersion, diffusion and retardation are used. The following dispersion parameters were used for the new particle tracking run: $\alpha_L$ = 10. m, $\alpha_{TH}$ = 1. m, and $\alpha_{TV}$ = 0.1 m. All other data remained the same. PLUMECALC runs were made using the same subgrid cases as before. For runs including diffusion, diffusion parameters from rock_vsq_diff.macro were used. For runs with retardation, a factor of 2 was used. Results are shown in Figures 35 through 39 for the 5x5x5 subgrid cases at 500 years. Dispersion causes spreading of the plume (Figures 37 – 39). Adding diffusion results in a reduction in the overall size of the plume (Figure 35 and Figure 38). With sorption the extent of the plume is somewhat reduced and the center of the plume has higher concentrations for a greater period of time (Figure 36 and Figure 39).

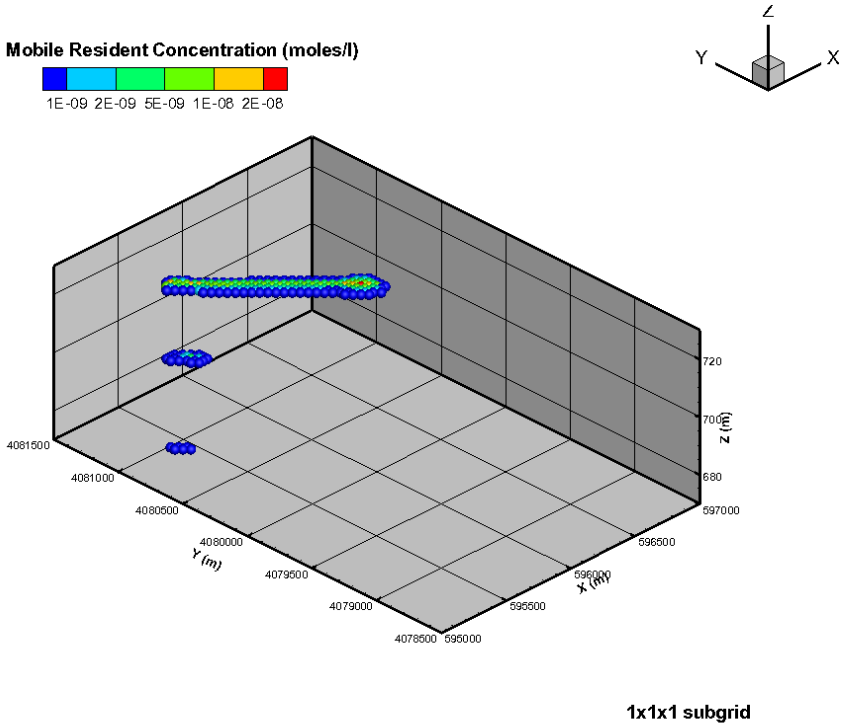


**Figure 32. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 1x1x1 refinement.**

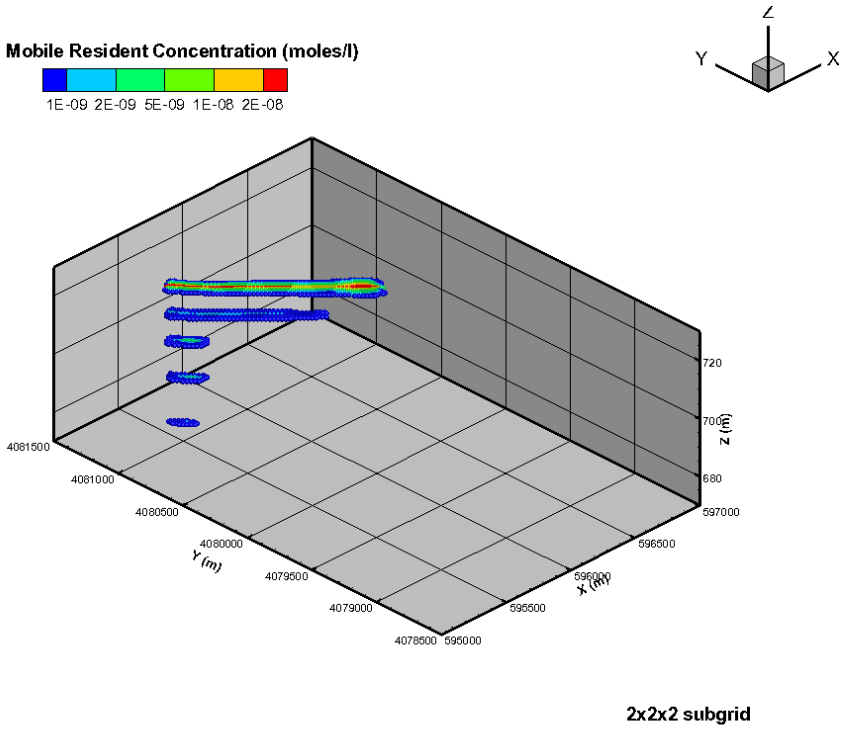


**Figure 33. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 2x2x2 refinement.**

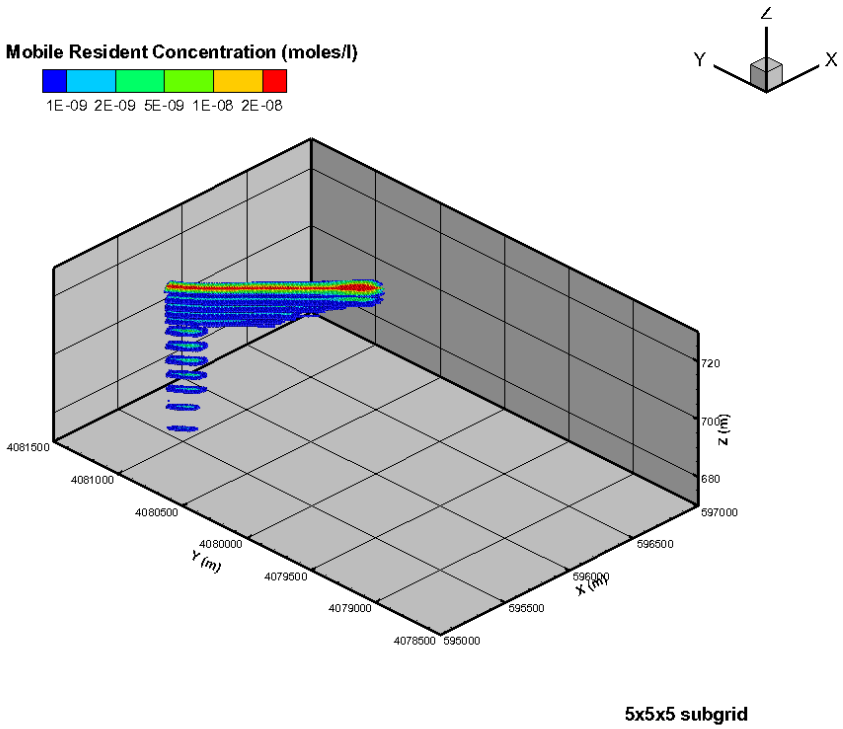


**Figure 34. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 5x5x5 refinement.**

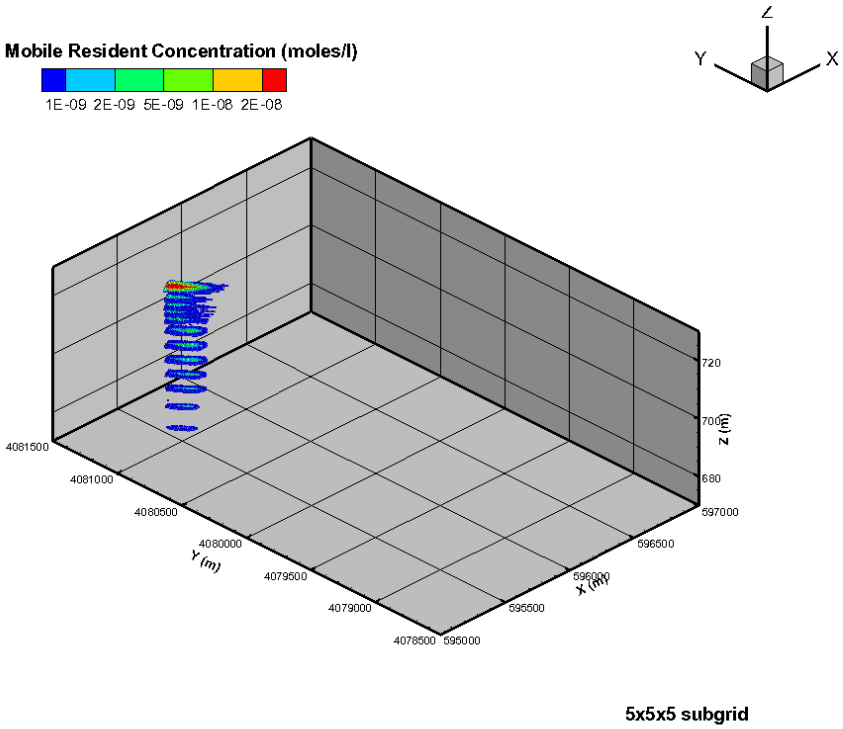


**Figure 35. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 5x5x5 refinement with diffusion.**

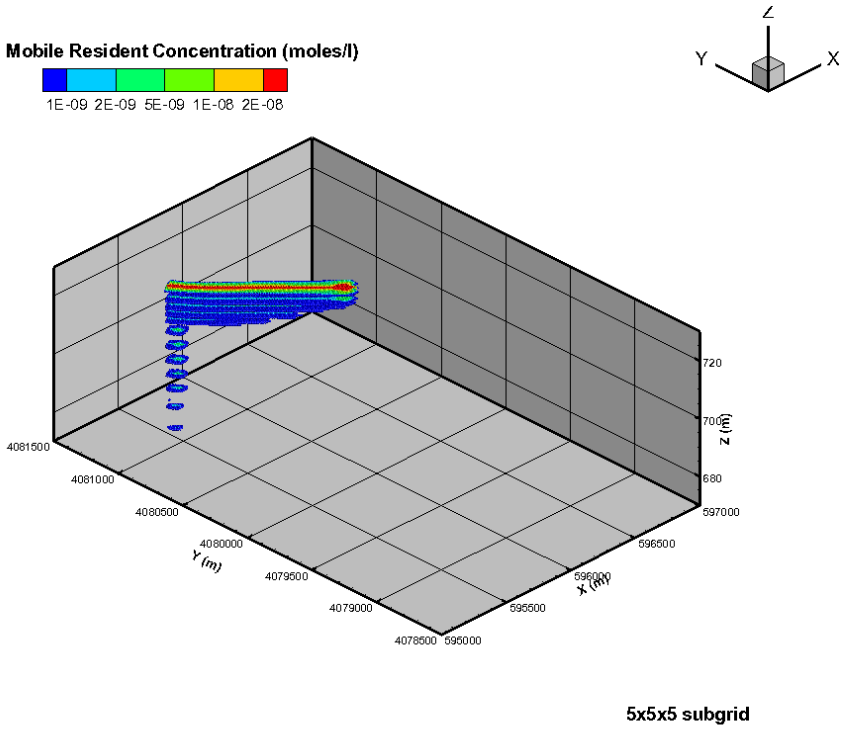


**Figure 36. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 5x5x5 refinement with retardation.**

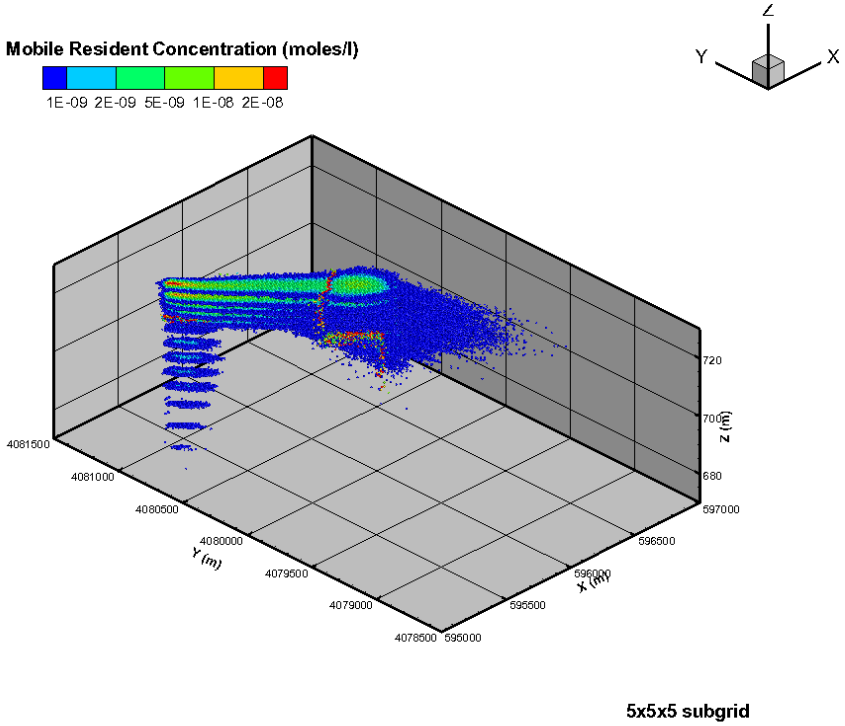


**Figure 37. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 5x5x5 refinement with dispersion.**

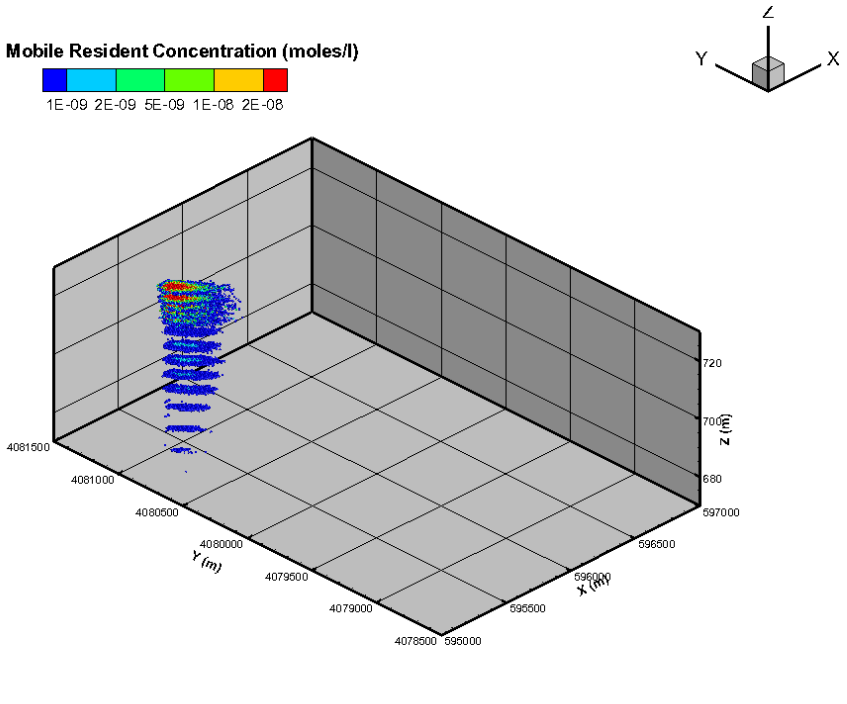


**Figure 38. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 5x5x5 refinement with dispersion and diffusion.**

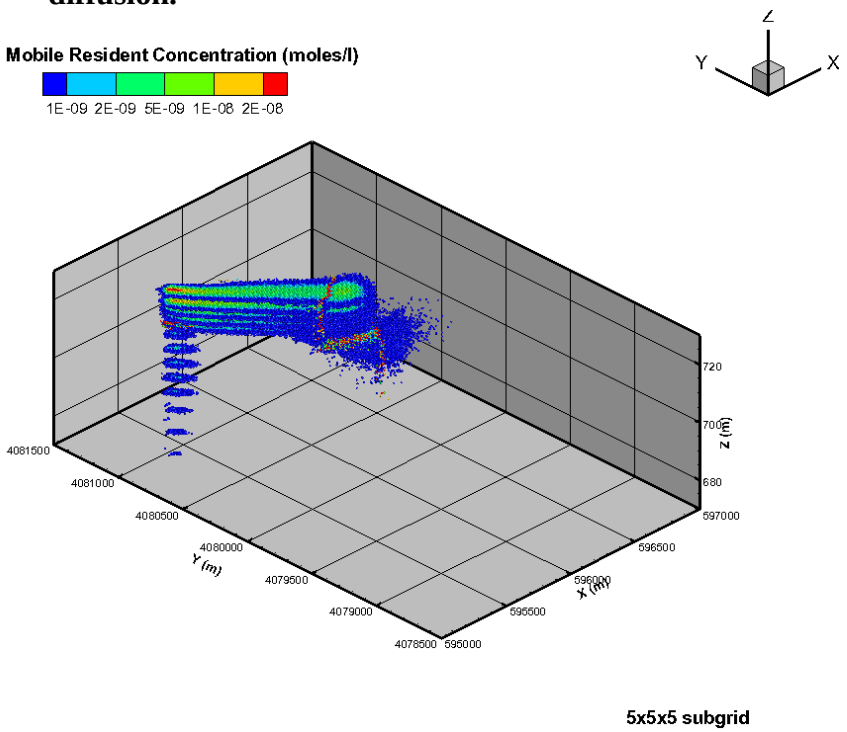


**Figure 39. Complex grid mobile resident concentrations for the subgrid PLUMECALC run with 5x5x5 refinement with dispersion and retardation.**

## 7. 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. 2003a. “Software Users Manual (UM) for the FEHM Application Version 2.21 Document ID: 10086-UM-2.21-00, MOL.20031031.0266.

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

Dash, Zora. 2003c. “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.

Painter, S., V. Cvetkovic, J. Mancillas, O. Pensado, 2008. "Random Walk Particle Tracking Methods for Simulating Transport with Retention and First-order Transformation." *Water Resources Research*, 44, doi:10.1029/2007WR005944.

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.

Robinson, B. A., Z. V. Dash, and G. Srinivasan, 2010. "A particle tracking transport method for the simulation of resident and flux-averaged concentration of solute plumes in groundwater models," *Comput Geosci* **14** (4), 779–792.

Robinson, B. A., Z. V. Dash, C. Gable, S. Kelkar, S. Painter, and G. Srinivasan, 2011. PLUMECALC V2.32, LA-CC-11-029.

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.

Srinivasan, G., E. Keating, Z. V. Dash, B. A. Robinson, and J. D. Moulton, 2011, "Convolution Based Particle Tracking Method for Transient Flow", in review *Comput Geosci*.

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.

## Appendix A. Additional Details and Verification Tests for the Matrix Diffusion Model

PLUMECALC uses a post-processing strategy to represent the effects of matrix diffusion into a finite-sized matrix block. In this strategy, a particle tracking code is first used to generate exit times from each finite-volume cell for a set of non-sorbing, non-diffusing tracers. PLUMECALC uses these advective/dispersion transit times for each cell to sample from a retention time distribution. The retention time represents time spent in the immobile water in the matrix pore space and time sorbed onto matrix minerals.

The total residence time $t_{res}$ that a particle spends in a cell can be written as [Painter et al. 2008]

$$t_{res} = \omega^2 t_* + R_f \tau \qquad \text{(A-1)}$$

where $\omega = \theta_m \tau \sqrt{R_m D_m} / b$ , $R_f$ is the fracture retardation factor, and $\tau$ is the advective/dispersive travel time as calculated by the particle tracking code. Here, 2*b* is fracture aperture, and $R_m$, $D_m$ and $\theta_m$ are retardation factor, diffusion coefficient and porosity in the matrix, respectively. The scaled time $t_*$ has a known probability distribution that depends on a single parameter grouping $\eta = \dfrac{b(B-b)}{\theta_m D_m \tau}$ where 2B is fracture spacing. The cumulative distribution of the scaled residence time $F(t_*;\ \eta)$ has Laplace transform [Painter et al. 2009]

$$\widehat{F}(s;\eta) = \frac{1}{s}\exp\left[-s^{1/2}\tanh\left(\eta s^{1/2}\right)\right] \qquad \text{(A-2)}$$

where *s* is the Laplace variable complementary to time.

The parameter $\eta$ is the ratio of Sudicky and Frind's $\sigma$ and $\omega$ parameters and provides a dimensionless measure of the size of the matrix block (fracture spacing). When $\eta << 1$ the characteristic diffusion time in the matrix block is much smaller than the characteristic transport time and the diffusion process approximates equilibrium sorption. In this situation, the (scaled) retention time distribution approximates a dirac-delta function centered at $\eta$. When $\eta >> 1$ the characteristic diffusion time in the matrix block is much smaller than the characteristic transport time and the diffusion process approximates diffusion into an infinite half-space.

In PLUMECALC, the cumulative distribution of scaled retention time is saved as "type curve" lookup tables that relate scaled retention time and quantiles of the distribution for different values of $\eta$. The retention time distribution is then sampled by sampling a random quantile *q* between 0 and 1, and then obtaining a value of $t_*$ by interpolating the lookup table $F^{-1}(q;\ \eta)$. The lookup tables were calculated as follows

- For $\eta < 0.01$, $F^{-1}(q;\ \eta) \approx \eta$, corresponding to equilibrium sorption.

- For $0.01 \le \eta < 1$, $F^{-1}(q;\ \eta)$ was computed by numerical quadrature of the integrals in Eq. 35b of Sudicky and Frind (1982) with no fracture retardation and no decay.
- For $1 \le \eta \le 1000$, $F^{-1}(q;\ \eta)$ was computed by numerical inversion of Equation A-2.
- For $\eta > 1000$, the $F^{-1}(q;\ \eta) = \left[2\,\mathrm{erfc}^{-1}(q)\right]^{-2}$ where $\mathrm{erfc}^{-1}$ is the inverse of the complementary error function. This distribution corresponds to diffusion into an infinite half space.

Different numerical approaches were used for large and small values of $\eta$ for numerical reasons. The numerical Laplace inversion has difficulty converging when $\eta < 1$; hence the analytical solution of Sudicky and Frind (1982) was used. For $\eta > 1$ the numerical Laplace inversion is more reliable than direct integration of the integrals in Eq. 35b of Sudicky and Frind (1982). The $F^{-1}(q;\ \eta)$ for various values of are shown $\eta$ in Figure A-1.

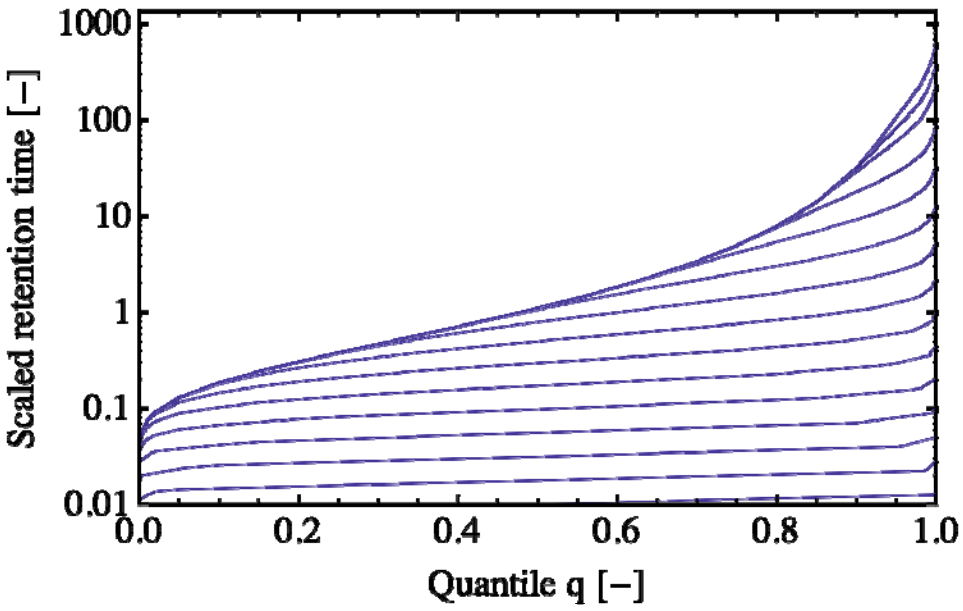


*Figure A-1. Distribution of scaled retention time $t_*$ shown as $t_*$ versus quantile for different values of $\eta$. These curves are sampled in PLUMECALC to represent matrix diffusion.*

Several verification tests were used to test PLUMECALC's matrix diffusion model. The flow configuration is a one-dimensional flow-through configuration of length 20,000 meters in the x-direction. Flow is parallel to the x-direction. The cross-section of the flow domain is 100 m by 100 m. The domain is discretized into $101 \times 4 \times 4$ cells.

Two sets of particle tracking results were used. The first set comes from the Walkabout code (Painter, 2011). The second set is synthetic. The synthetic particle tracking results were

generated by sampling arrival time distributions at cell boundaries using well-known analytical results. Specifically, each sample of the advective/dispersive travel time τ across a control volume cell was produced by generating a random number *R*, uniformly distributed between 0 and 1, and solving for τ from $\tau = F^{-1}(R)$ where *F* is the right side of Equation 7 in Kreft and Zuber (1978). Note the arrival time distribution is based on injection in flux into a semi-infinite bed with detection in flux. Other boundary conditions are incorrect and in general would give erroneous results (see Kreft and Zuber, 1978).

A benchmark solution was developed by numerically solving the following system of partial differential equations

$$R_f \frac{\partial C_f}{\partial t} + V \frac{\partial C_f}{\partial x} - \alpha V \frac{\partial^2 C_f}{\partial x^2} = -\lambda R_f C_f + \frac{\theta_m D_m}{b} \frac{\partial C_m}{\partial z}\bigg|_{z=b} \qquad 0 \le x \le \infty$$

$$R_m \frac{\partial C_m}{\partial t} - D_m \frac{\partial^2 C_m}{\partial z^2} = -\lambda R_m C_m \qquad b \le z \le B$$

$$C_f(x,0) = 0$$

$$C_f(0,t) - \alpha \frac{\partial C_f}{\partial z}\bigg|_{z=0} = \frac{\dot{m}(t) B}{AbV}$$

$$C_f(\infty,t) = 0$$

$$C_m(x,z,0) = 0$$

$$C_m(x,b,t) = C_f(x,t)$$

$$\frac{\partial C_m(x,z,t)}{\partial z}\bigg|_{z=B} = 0$$

Here *x* is distance along the fracture, *z* is distance perpendicular to the fracture [reversed from Sudicky and Frind's (1982) notation], *t* is time, $C_m(x,z,t)$ and $C_f(x,t)$ are concentrations in the matrix and fracture pore waters, $R_m$ and $R_f$ are fracture and matrix retardation factors, V is groundwater velocity, α is dispersivity, A is cross-sectional area of the domain, 2b is fracture aperture, 2B is fracture spacing, $D_m$ and $\theta_m$ are diffusion coefficient and porosity in the matrix, respectively. Note that diffusion coefficient is defined so that diffusive flux in the matrix is proportional to diffusion coefficient *and porosity* and should not be confused with effective diffusion coefficient $D_{eff} \equiv \theta_m D_m$. The inlet boundary condition is a specified mass injection rate $\dot{m}(t)$, which is consistent with PLUMECALC and different from the Sudicky and Frind (1982) assumption. The NDSolve option of Mathematica™ was used to solve the equations. NDSolve uses very sophisticated adaptive algorithm selection and error control.

Figures A-2 – A-5 compare results form PLUMECALC with the numerical benchmark solution. Unless otherwise noted, the full aperture is 0.862 mm, fracture porosity is 1.01 $10^{-3}$ (corresponding to a fracture spacing of 85.3 cm), the velocity is 1.97 m/day, the matrix porosity is 0.279, the diffusion coefficient is 2.56 $10^{-10}$ $m^2/s$, and there is no sorption in the matrix. The agreement is good over the entire range tested. The testing spans an $\eta$ range of 7.8 $10^{-3}$ to 750 (from equilibrium sorption to diffusion into an infinite half-space). The grid Peclet number spans a range of 5 to infinity, and the half-life spans a range of 10 years to infinity.

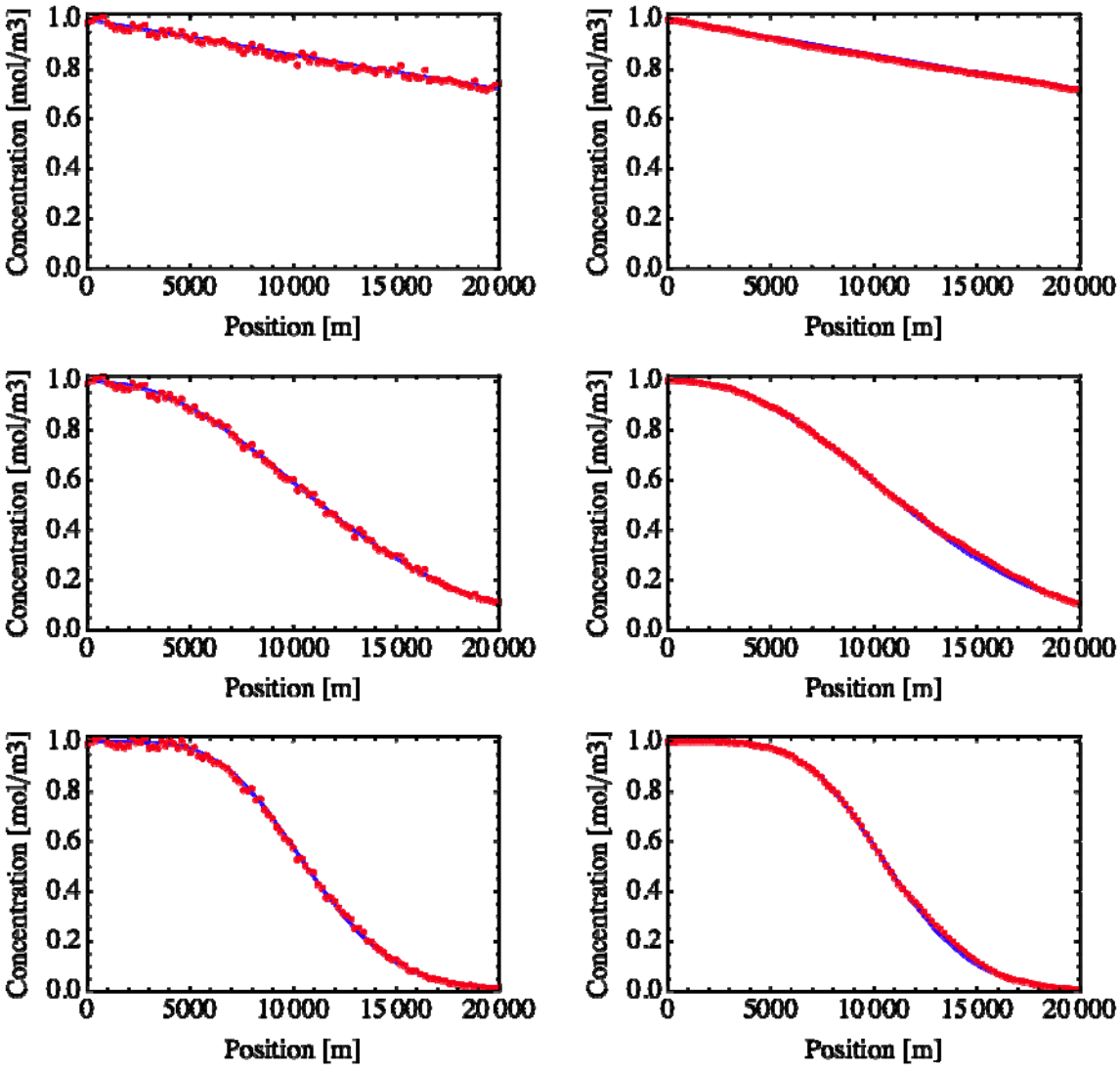


Figure A-2. Result of testing using PLUMECALC using synthetic particle tracking data as input. PLUMECALC results are shown as red dots, numerical solutions to the above PDE system are shown as blue curves (obscured by the dots). Shown are concentration (mols/$m^3$) version position (meters) at 4000 years for different values of matrix diffusion coefficient and dispersivity. Plots on the right have no dispersion. Plots on the left have longitudinal dispersivity

of 40 m. From top to bottom, the diffusion coefficients are $10^{-11}$ $m^2/s$, 3 $10^{-12}$ $m^2/s$ and 1 $10^{-13}$ $m^2/s$.

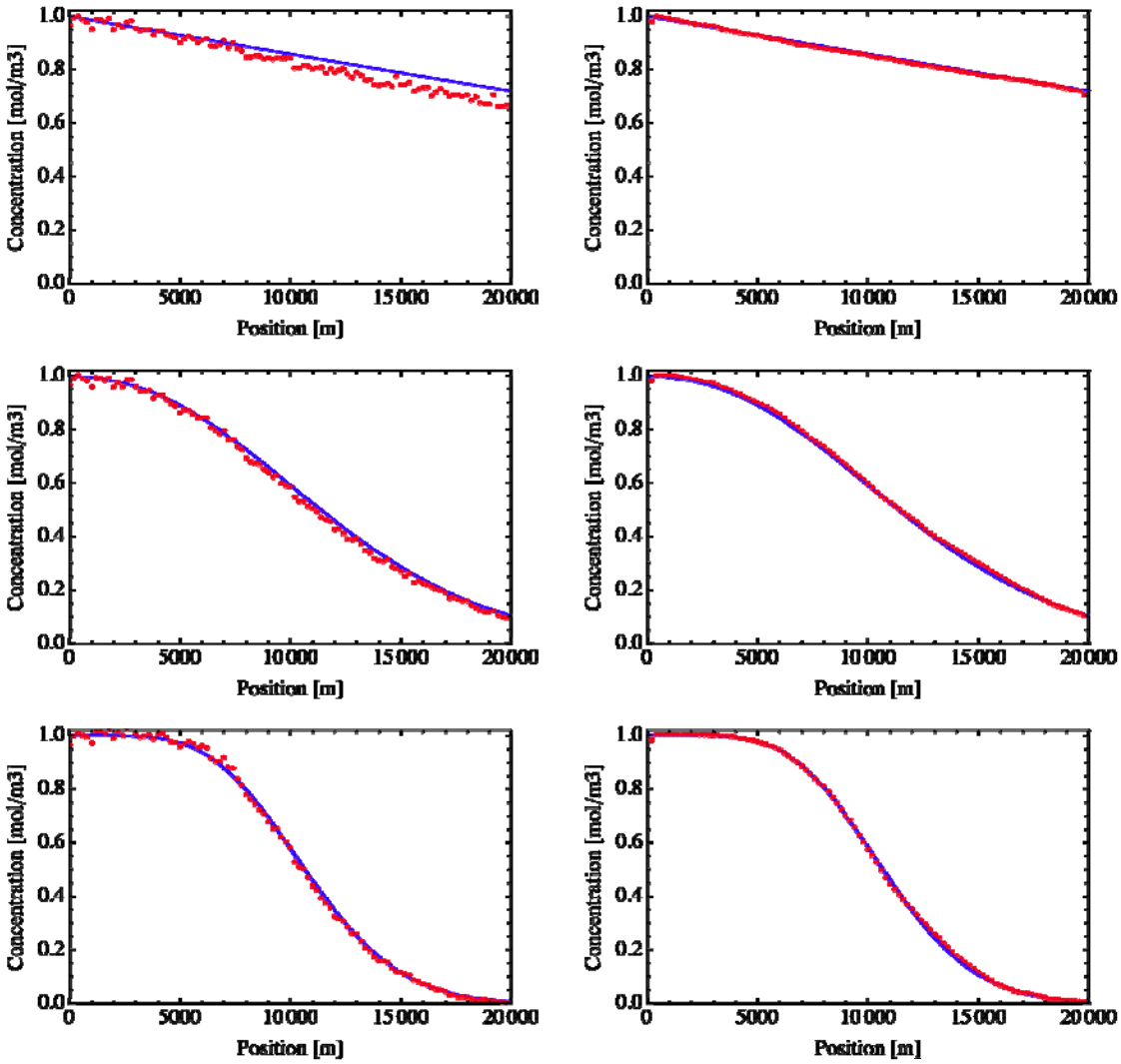


Figure A-3. Result of PLUMECALC testing using Walkabout particle tracking data as input. PLUMECALC results are shown as red dots, numerical solutions to the above PDE system are shown as blue curves. Shown are concentration (mols/$m^3$) version position (meters) at 4000 years for different values of matrix diffusion coefficient and dispersivity. Plots on the right have no dispersion. Plots on the left have longitudinal dispersivity of 40 m. From top to bottom, the diffusion coefficients are $10^{-11}$ $m^2/s$, 3 $10^{-12}$ $m^2/s$ and 1 $10^{-13}$ $m^2/s$.

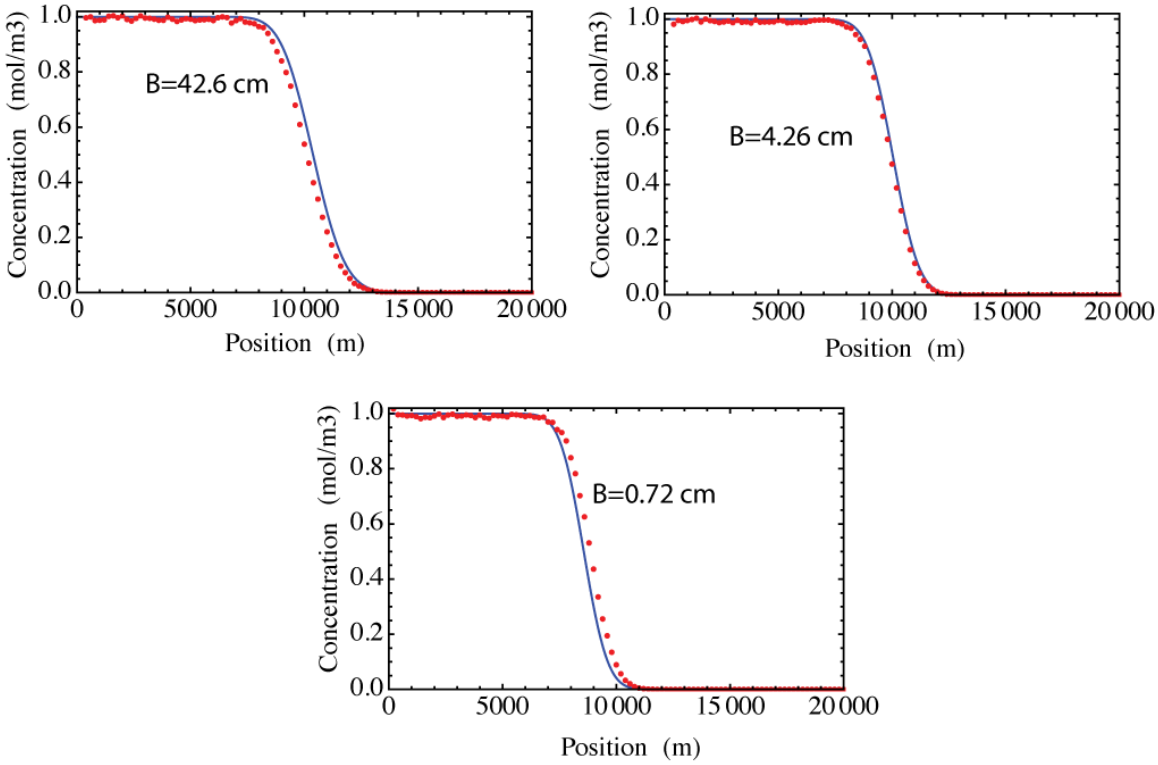


Figure A-4. Result of PLUMECALC testing using synthetic particle tracking data as input. PLUMECALC results are shown as dots, numerical solutions to the above PDE system are shown as curves. Shown are concentrations (mols/$m^3$) versus position (meters) at 4000 years for different values of the fracture spacing 2B. The longitudinal dispersivity is 40 m.

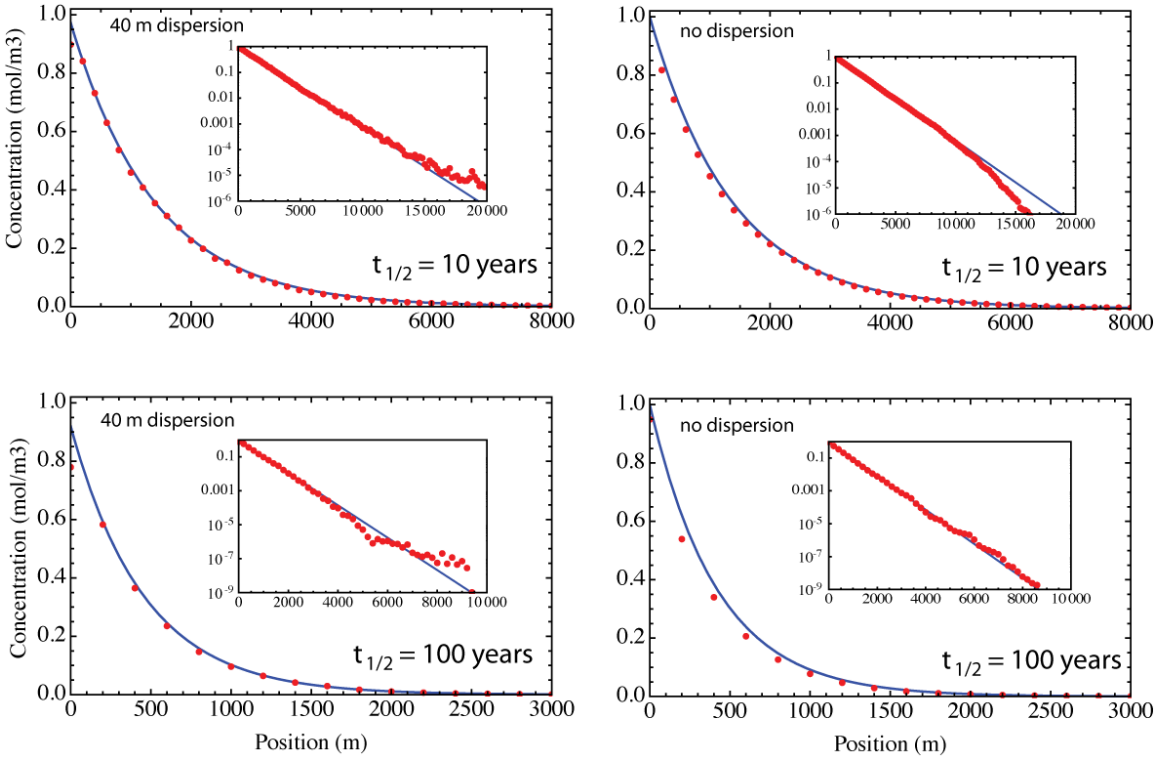


Figure A-5. Result of PLUMECALC testing using Walkabout particle tracking data as input. PLUMECALC results are shown as dots, numerical solutions to the above PDE system are shown as curves. Shown is concentration (mols/$m^3$) version position (meters) at 4000 years for different values of solute half-life. The plots on the left have longitudinal dispersivity of 40 m; the plots on the right have no dispersion. Each inset shows the same plot on a log-linear scale. The diffusion coefficient is 3 $10^{-12}$ $m^2/s$

References

Painter, S., V. Cvetkovic, J. Mancillas, O. Pensado, 2008. “Random Walk Particle Tracking Methods for Simulating Transport with Retention and First-order Transformation.” *Water Resources Research*, 44, doi:10.1029/2007WR005944.

Painter, S., 2011. “Users Manual for Walkabout Version 1.0”. Los Alamos National Laboratory Report LA-UR 11-01952, Los Alamos National Laboratory.

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.

Sudicky, E.A. and E.O. Frind, 1982. “Contaminant Transport in Fractured Porous Media: Analytical Solutions for a System of Parallel Fractures.” Water Resources Research 18(6), 1634–1642.

## Appendix B. FEHM Files Used for a Particle Tracking Simulation

The 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, the streamline particle tracking (sptr2) output file, sptr control volume output file, and sptr auxillary data output file. 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 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, and when subgridding, the particle exit coordinate position.

The FEHM model used for PLUMECALC with subgridding needs to be run using the “xyz” parameter option in the sptr macro so that the particle coordinate position is output to the “sptr2” file when a particle exits a cell. A sptr velocity file also needs to be output (the “omr” keyword is used and the name of a file for the sptr auxiliary data should be provided) and the transport porosities need to be written to the end of that file.

The virtual subgrid is constructed starting with the corner (left, bottom, back corner) and size (length, width, height of the box) information output by FEHM for each node in the auxiliary data file. The dimensions of the box are subdivided into the specified number of segments, and a virtual node is placed at the geometric center of each sub-box. Note that due to this construction, for some grids, and boundary nodes, the locations of the virtual nodes may not coincide with any of the FEHM node locations. The particle trajectory is interpolated as a straight line within the FEHM box using the entrance and exit locations and times output by FEHM, and the entrance and exit locations and times for each subbox are calculated and saved for use by the rest of the PLUMECALC code.