DCPT v1.0

-- New Particle Tracker for Modeling Transport in Dual-Continuum Media

User's Manual

Lehua Pan, Hui Hai Liu, Mark Cushey, and Gudmundu Bodvarsson

LBNL

April 2001

LBNL-42958

TABLE CONTENTS

ABSTRACT

DCPT (Dual-Continuum Particle Tracker) V1.0 is a new software for simulating solute transport in the subsurface. It is based on the random-walk method for modeling transport processes such as advection, dispersion/diffusion, linear sorption, radioactive decay, and fracture-matrix mass exchange (in fractured porous media). The user shall provide flow-field and other parameters in the form of input files. In Comparison to several analytical and numerical solutions for a number of test cases, DCPT shows excellent performance in both accuracy and efficiency.

This report serves as a user's manual of DCPT V1.0. It includes theoretical basis, numerical methods, software structure, input/output description, and examples.

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Transport through fractured porous media occurs in many subsurface systems and is of great importance in many scientific and engineering fields such as underground natural resource recovery, waster storage, soil physics, and environmental remediation. In fractured porous media, the fractures often occupy a tiny portion of the whole volume but the pore water velocity in the fractures can be orderly higher than that in the matrix (Wu et al., 2000). Meantime, especially under variably saturated condition, the global water flow in the matrix and the mass exchange between the fractures and the matrix also play important roles in the whole transport process in the fracture porous media as a complex system. These special features distinguish the fractured porous media from the general heterogeneous porous media or the pure fracture networks (without porous matrix), and make many numerical methods for treating heterogeneity unsuitable. Despite the progresses in understanding and modeling transport in fractured porous media along with advancement in computer resources in the past few decades, simulating transport of a tracer in heterogeneous fractured porous media is still a difficult challenge to a numerical method (Wu and Pruess, 2000). One of primary challenges is how to simulate the mass transfer between fractures and matrix, two sub-domains often having very different velocities. A discrete fracture network model with detailed matrix, which requires great detailed description of the spatial structure of a fracture-matrix transport system, is not feasible in practice for the real-world problems because of its high requirements in both computer resources and field measured data (parameters). Therefore, a dual-continua approach has often been used to numerically simulate the fracture-matrix transport system (Huyakorn et al., 1983; van Genuchten and Dalton, 1986; Wu and Pruess, 2000;

Liu et al., 2000). The major advantage of the dual-continuum approach is its capability to capture the major features of flow and transport in fractured porous rock (i.e., a fast fracture sub-system combined with a slow matrix sub-system) with minimum computational resources.

The random-walk particle method (RWPM) presents an attracting alternative to simulate transport in fractured porous media because of its lack of numerical mixing and its potential of highly computational efficiency. Although significant progresses have been made in simulating the transport in heterogeneous porous media using RWPM (Labolle et al., 1996, 2000; Tompson and Gelhar, 1990; Uffink, 1985; Semra et al., 1993 and many others), they are not readily applicable to the fractured porous media for two reasons. One is that the detailed geometry of fracture-matrix interfaces is hardly available in real-world applications or too complicated to be described with affordable computational resources. Therefore, the methods to deal with the property discontinuity at the fracture-matrix interface (Labolle et al., 2000; Uffink, 1985; Semra et al., 1993) become useless because of lacking of required information. The other is that the fractures often occupy a tiny space comparing to the matrix (e.g., a thin film network), which makes it extremely inefficient to simulate mass transfer between fractures and matrix with conventional RWPM. Liu et al. (2000) proposed an approach of using the particle-transfer probability to describe the mass flow between fractures and matrix under a framework of hybrid RWPM and the dual-continuum approach. The original scheme of calculating the particle-transfer probability proposed by Liu et al.(2000) has been modified to remove the limitation on the size of time step. The modified scheme has been integrated with a new time stepping technique and an efficient 3-D irregular grid-tracker into a Dual-Continuum-Particle-Tracker (DCPT) v1.0. The objective of DCPT v1.0 is to provide an efficient simulator of tracer transport in complex fractured porous media with advantages of RWPM such as less numerical dispersion, more efficiency, higher spatial resolution, and straightforward to track migration pathway of solute.

DCPT v1.0 can be used to simulate transport processes of an arbitrary number of independent tracers with sorption, radioactive decay, advection, hydrodynamic dispersion, and molecular diffusion in variable saturated fractured porous media with regular/irregular numerical grids of arbitrary sizes. Both flow fields and the grids must be provided by users as the input files. The authors highly recommend that WinGridder v1.0 or higher is used to generate the grid and that the TOUGH2 v1.4 is used to generate the flow fields because these two codes can readily generate several output files that DCPT v1.0 can directly use as input files.

2. MATHEMATICAL MODEL AND NUMERICAL METHODS

2.1 Modeled Processes and Assumptions

DCPT V1.0 can model the transport of an arbitrary number n of tracers in dual-continuum media, accounting for

(a) Advection,

(b) Hydrodynamic dispersion (with full 3-D tensor representation), including molecular diffusion,

(c) Sorption (equilibrium),

(d) Radioactive decay,

(e) Tracking up to n-1 daughter products of radioactive decay.

(f) Mass transfer through the fracture-matrix interface.

The major assumption is that the concentration of the tracers is so low that no measurable effect on the flow regime should be considered. The transport simulation can then be performed based on the flow field provided by the users. The flow field can be defined on a dual-continuum grid.

3.2 Framework of Dual-Continuum RWPM

The dual-continuum RWPM used in DCPT v1.0 simulates the particle movements in each continuum with regular (single-continuum) RWPM and uses a random continuum-switch process to simulate the particle transfer between two continua (Pan et al., 2001). A random number is drawn from the uniform distribution U[0, 1] and compared to the particle-transfer probability after each movement. If the random number is larger than the particle-transfer probability, the other continuum will be selected as the medium in which the particle will travel at the next time step. Otherwise, the current continuum is still the selected medium. In this way, particles will travel randomly in either fractures or matrix.

The dual-continuum RWPM tracks both the location (including which continuum) of each particle and the time spent to reach that location. As a result, we can derive a breakthrough curve by calculating the total mass carried by the particles passing a specified observation location (e.g., at water table) at different times. If all particles are released at the same time, the breakthrough curve will correspond to a tracer pulse injection. By defining appropriately the particle release times at the source, we can calculate the breakthrough curves for any given tracer injection with variable concentration.

3.3 Advection and Dispersion in a Particular Continuum

When the continuum has been selected, the new location of a particle after a time step Δt is a random vector and can be calculated as (LaBolle et al. 1996, neglecting the two additional terms related to the divergence of D and the gradient of the volumetric water content for simplicity)

$$X(t+\Delta t) = X(t) + V\Delta t + \sqrt{\Delta t}BW \qquad (1)$$

where B is a tensor with BB^T = 2D, while D is the local dispersion coefficient tensor in the continuum. W is a random vector, each component of which observes the $N(0,1)$ distribution. V is the local velocity vector in the continuum. As shown in Sections 7 and 8, this approximation is acceptable for advection-dominant transport processes in the dual-continuum media, i.e., in the case of the unsaturated zone in Yucca Mountain site. In a given continuum, the mean displacement vector is VΔt, while the variance tensor is 2DΔt.

3.4 Sorption and Decay

For an adsorbing tracer, only a portion of particles is in fluid and move as described by Equation (1). The probability of a particle being in fluid, P_r, can be defined as

$$P_r = \frac{\theta}{\theta + (1-\phi)K_d\rho_R} = \frac{1}{R} \qquad (2)$$

where K_d , ϕ, and ρ_R are the distribution coefficient (m^3/kg), the porosity (m^3/m^3) and the rock density (kg/m^3) of the particular continuum, respectively, while R is called as retardation factor. For implementation of a particle tracking sorption process, we can take P_r, in a deterministic way, as the percentage of the particle mass that engages in moving. Therefore, the effective displacement of the particle will be P_r times the original displacement, which can be implemented by simply multiplying A and B in Equation (1) by P_r.

To simulate the radioactive decay, we calculate the mass of each particle as a function of time, t:

$$\mu\,(t+\Delta t)=\mu\,(t)\;2^{-\Delta t/t_{0.5}} \tag{3}$$

where $t_{0.5}$ is the half-life and Δt is the time step.

3.5. Particle-transfer Probability: Mass Transfer between Fracture and Matrix

The particle-transfer probability is here defined as the probability of a particle entering another continuum during a given time interval. Because the fracture-matrix interaction is described at the cell level in the dual-continuum model, the particle-transfer probability is also calculated at the cell level for each continuum. The net mass transfer from the fracture continuum to the matrix continuum during the time interval Δt is the difference between the mass transferred from the fractures to the matrix (m_{fm}) and the mass transferred from the matrix to the fractures (m_{mf}) during the same time interval. It can be related to the particle-transfer probability as follows:

$$m_{fm}-m_{mf}=N_f\;\mu\,P_{fm}-N_m\;\mu\,P_{mf} \tag{4}$$

where, μ is the mass carried by each particle while P_{fm} and P_{mf} are the particle-transfer probabilities for particles in the fracture and the matrix continua at time t, respectively. The subscript indicates the transfer direction (e.g., “fm” means from the fracture continuum to the matrix continuum). N_f and N_m are the numbers of particles in the fracture continuum and the matrix continuum, respectively, for a given cell at time t. Unlike finite-difference methods or finite-element methods, the particle-tracking method does not directly calculate the net mass transfer between two continua. Instead, it simulates the mass transfer in two directions separately by tracking the transfer of the particles between two continua. For example, particle transfer occurs even if the net mass transfer equals zero between the two continua, which only means that the number of particles transferred in two directions is the same. As revealed in Equation (4), the particle-transfer probability from one continuum to the other can be calculated as the ratio of the mass entering the other continuum during the time interval, Δt, to the mass in the current continuum at the beginning of the time interval. Finding a proper scheme to estimate the mass transfer between two continua (m_{fm} or m_{mf}) is essential. The mass transfer consists of two components: advection and diffusion/dispersion. The component due to advection can be directly calculated as the product of the amount of the water entering the other continuum and the concentration of that water. The component due to diffusion/dispersion, however, needs to be calculated based on some assumptions about the concentration distribution within the matrix because only a single grid block is used to represent the matrix in a dual-continuum model. By assuming a second-order polynomial distribution of concentration away from the fracture-matrix interface into the matrix, Liu et al. (2000) proposed a simple scheme to approximate the diffusion component of the mass transfer and the corresponding particle transfer probabilities for both directions. However, the scheme may fail if Δt is large. To overcome the problem, we have derived an improved scheme by considering the temporal feature of the average concentration (Appendix A). The new schemes of P_{fm} and P_{mf} for a given time interval, Δt, can be expressed as (5a) and (5b), respectively (see Appendix A).

$$P_{fm} = \frac{F_{fm}}{Q_f + F_{fm}}[1 - \exp(-\Delta t/\tau_f)] \qquad (5a)$$

$$P_{mf} = \frac{F_{mf}}{Q_m + F_{mf}}[1 - \exp(-\Delta t/\tau_m)] \qquad (5b)$$

where F and Q describe the strength of advection and dispersion/diffusion processes through the fracture-matrix interface and the interfaces to adjacent grid cells in the same continuum, respectively. The subscripts indicate the direction of those processes. The parameters τ_f and τ_m are the characteristic times of the fracture continuum and the matrix continuum, respectively. The detailed expressions of these parameters are listed below:

$$F_{fm} = \max(q_{fm}A_{fm}, 0) + \frac{D_m A_{fm}}{S_{fm}} \qquad (6a)$$

$$Q_f = \sum_{i=1}^{N}\left[\max(q_f A_i, 0) + \frac{Df_i A_i}{S_i}\right] \qquad (6b)$$

$$\tau_f = \frac{\mathrm{V_f}\ \mathrm{R_f}}{\mathrm{F_{fm}} + \mathrm{Q_f}} \tag{6c}$$

$$\mathrm{F_{mf}} = \max(-\mathrm{q_{fm}} \mathrm{A_{fm}}, 0) + \frac{\mathrm{D_m} \mathrm{A_{fm}}}{\mathrm{S_{fm}}} \tag{6d}$$

$$\mathrm{Q_m} = \sum_{i=1}^{N} \left[\max(\mathrm{q_m} \mathrm{A_i}, 0) + \frac{\mathrm{Dm_i} \mathrm{A_i}}{\mathrm{S_i}} \right] \tag{6e}$$

$$\tau_m = \frac{\mathrm{V_m}\ \mathrm{R_m}}{\mathrm{F_{mf}} + \mathrm{Q_m}} \tag{6f}$$

where, A_{fm} and A_i are the area of the fracture-matrix interface within the grid cell and the interface area to the ith adjacent grid cell, respectively. The q_{fm} (positive if flow from the fracture to the matrix) and q_i (outwardly positive) are the water flux through the fracture-matrix interface within the grid cell and the interface to the ith adjacent grid cell, respectively. The parameters D_m , Df_i, and Dm_i are effective dispersion coefficients of the matrix within the grid cell, the fracture continuum at the interface to the ith adjacent cell, and the matrix continuum at the interface to the ith adjacent cell, respectively. S_i is the distance between the center of the cell and the ith adjacent cell, while S_{fm} is the characteristic length of the fracture-matrix system (e.g., 1/6 of the fracture spacing for a parallel fracture system as suggested by Liu et al., 2000). V and R are the volume of water and the retardation factor, respectively, with the subscripts indicating the continuum. Note that for a very small time step ($\Delta t/\tau_f$ or $\Delta t/\tau_m$ is closing to zero), the particle-transfer probabilities defined in Equations (5a) and (5b) will reduce to the same one proposed by Liu et al. (2000).

3.6 Adaptive Time Steps

Particle-tracking time steps used in DCPT are adaptive to the local flow field, cell size, and other transport parameters. For a given type of solute, each cell has two time steps corresponding to the fracture continuum and the matrix continuum, respectively. For either continuum of a cell, the time step is calculated as follows:

$$\Delta t = \min(0.05\frac{\Delta xy}{|Vxy|}, 0.05\frac{\Delta z}{|Vz|}, 0.25t_c\ ) \tag{7}$$

where Δxy and $|Vxy|$ are the lateral size of the cell and the magnitude of the lateral component of the pore velocity within the cell, respectively, and Δz and $|Vz|$ are the height of the cell and the magnitude of the vertical component of the pore velocity, respectively. The main advantage of the time-stepping technique based on Equation (7) is to use small time steps only where it is necessary (e.g., at location having high pore-water velocity). If sorption exists, effective velocities are used in Equation (7) by multiplying the original pore velocities by the factor P_r (see Section 3.4).

3.7 Tracking particles in an irregular grid

DCPT is designed for simulating transport in dual-continuum and heterogeneous media represented by a three-dimensional, irregular grid. All transport parameters including the flow velocities are given at grid cell level for each continuum and can vary with the cells due to heterogeneity. As a result, the grid cells in which a particle is traveling have to be tracked besides the abstract coordinates in 3-D space of the particle and the continuum in which the particle travels. In other words, an efficient algorithm for locating the cell of a 3-D irregular grid based on given coordinates of a walking particle is essential for assigning proper transport parameters used in the calculations of a random-walk step. In a regular grid, we can simply determine it by dividing the coordinates by some known node spacing. However, this is not the case in an irregular grid whose cells may have varied sizes and shapes. The question has two parts: has a particle exited from the current cell after a random walk step and which neighboring cell it would enter if it had exited from the current cell? To address this problem, an efficient approach has been developed and implemented in DCPT. First, we define a data object for any grid cell such that it knows all its faces (i.e., the geometry configuration) and the corresponding neighboring cells (i.e., the relative position in the grid). Second, connecting the starting and ending points of a particle during a time step into a trail segment. In this way, to find the next right cell becomes to find which face of a known cell (currently contains the particle, cell A) intersects with the trail segment of the particle (Figure 1). If none of those faces intersects the movement segment, the particle is still in the same cell (the cell A). If face AB intersects the trail segment, the next right cell is cell B in Figure 1. Note that this algorithm is totally localized because the calculations only depend on the geometry of the current cell (i.e., the number of its faces or neighbors). Therefore, its performance will not suffer when the number of cells becomes very large. This feature is particularly important for the cases where the number of grid cells becomes very large (e.g., in large-scale field problems).

4 DESIGN OF DCPT

DCPT is written in FORTRAN90 following the Object-Oriented-Programming principles. The followings are the descriptions of the major structures of the DCPT code.

4.1 Major Data Structures and Common Modules

Two major objects are used in DCPT. One is called *CELL*, which has all the information of the media (e.g., the geometry, local velocities, dispersion coefficient tensor, and other parameters for both fracture and matrix). The other is called *PARTICLE*, which has properties describing the current status of a particle, including the current time, current XYZ position, the current cell, and the current continuum (fracture or matrix). Other related objects and subobjects are *COLLUMN*, *SEGMENT*, *CONNECTION*, *ParticleGroup*, *CellGeometry*, *CellProperty*, *ROCK*, *RockProperty*, and *ParticleProperty*.

There are two common modules in DCPT: *ComData* and *GeoMetry*. *ComData* includes the definitions of all objects used, the common arrays of these objects, and the functions/subroutines defining operations performed on them. *GeoMetry* includes all geometric calculation function/subroutines. Both *ComData* and *GeoMetry* are available to the subroutines that cite them.

4.2 List of Source Code Files and Function/Subroutines

DCPT consists of the following source code files:

Main program

(a) Program *ParticleTracking* (in "Pt.for") – the main program

Input subroutines/Functions

(a) *InputProp* (in "Input.f") – Read data from two *.tec files generated by the flow simulator (TOUGH2)

(b) *InputConIndex* (in "Inputgrid.for") – Input the connection configurations of the grid and the flow rates per connections;

(c) *InputGrid* (in "Inputgrid.for") – Input information of cells, columns, and segments;

(d) *InputParams* (in "Inputgrid.for") – Call InputProp(), and input rock properties, particle properties, and rock index if multiple species simulation or input the sorption coefficient (K_d) if single species simulation;

(e) *InputInitDis* (in "Inputgrid.for") – Input the initial distributions of the particles;

(f) *MainInput* (in "Maininput.for") – Read in the information in the control file.

Particle tracking subroutines and functions (advection)

(a) *STATUS* (in "Status.f") – Determine if a particle will switch to the other continuum at next time level;

(b) *Rans* (in "Status.f") – Generate a uniform-distributed random number within (0, 1), modified from Numerical Recipes (Press et al. 1992).

(c) *Tracking* (in “Tracking.for”) – Simulate the movement of a particle;

(d) *DrawBackXY* (in “Tracking.for”) – Adjust the time step based on the given (X,Y) coordinates;

(e) *DrawBackZ* (in “Tracking.for”) – Adjust the time step based on the given Z;

(f) *GetCell* (in “Tracking.for”) – Get the cell that contains the particle among the splitting cells (in case of a local refined mesh);

(g) *SearchDown* (in “Tracking.for”) – Find the cell that contains the particle in a given column down from the current cell;

(h) *SearchUP* (in “Tracking.for”) -- Find the cell that contains the particle in a given column up from the current cell;

(i) *SearchColumn* (in “Tracking.for”) -- Find the cell that contains the particle in a given column from top to bottom;

(j) *OutColumn* (in “Tracking.for”) – Check if a particle went out of the current column and determine which neighboring column it went if so;

(k) *FoundColumn* (in “Tracking.for”) – Search the whole grid to find the column that contains a given particle.

Common functions for advection, dispersion, and sorption

(a) *SetFMTransP* (in “Comdata.for”) –Calculate the parameters of the particle-transfer-probability for every cell;

(b) *SetTimeStep* (in “Comdata.for”) –Setting time steps for every cell;

(c) *SetFlowRate* (in "Comdata.for") -- Calculate flow for every cell and determine the global direction of the flow per connection;

(d) *RetardVelocity* (in "Comdata.for") – Scale the velocity by the retardation factor;

(e) *SetVelocity* (in "Comdata.for") – Calculate the vertical component of the averaged velocity for every cell;

(f) *CalDTensor* (in "Comdata.for") – Calculate the dispersion tensors (D) based on the molecular diffusion coefficients and longitude and transverse dispersivities etc., and decompose the D into B with Cholosky decomposition method;

(g) *Choldc* (in "Comdata.for")—Perform Cholosky decomposition (from Numerical Recipes, Press et al. 1992)

(h) *Distance* (in "Comdata.for") –Calculate the distance between two cells (nodes).

(i) *CalParaforCell* (in "Comdata.for") --Calculate the parameters of a particular cell for a particular particle type.

<u>Dispersion and diffusion subroutine and functions</u>

(a) *Disper* (in "Dispersion.f") – Calculate the random steps;

(b) *gasdev* (in "Dispersion.f") – Generate a random number that observes the normal distribution N(0,1), modified from Numerical Recipes (Press et al. 1992);

(c) *ran2* (in “Dispersion.f”) -- Generate a uniform-distributed random number within (0, 1), modified from Numerical Recipes (Press et al. 1992).

Radioactive decay subroutine and function

(a) *Decay* (in “Decay.for”) – Calculate the mass left for a particle after one time step and determine if the particle will become its daughter product if any;

(b) *ranDecay* (in “Decay.for”) – Generate a uniform-distributed random number within (0, 1), modified from Numerical Recipes (Press et al. 1992).

Geometric functions

(a) *IntersecP* (in “Geometry.for”) – Determine if two segments intersect and return the coordinates of the intersection point if any;

(b) *SegInColumn* (in “Geometry.for”) – Check if a segment is inside of a column (polygon);

(c) *SegCrossPoly* (in “Geometry.for”) – Check if a segment is across a polygon;

(d) *InPolygon* (in “Geometry.for”) – Check if a point is inside of a polygon;

(e) *InOnPolygon* (in “Geometry.for”) – Check if a point is inside of or on the boundary of a polygon;

(f) *OnPolygon* (in "Geometry.for") – Check if a point is on the boundary of a polygon;

(g) *OnSegment* (in "Geometry.for") – Check if a point is on a segment;

(h) *GetNeibor* (in "Geometry.for") – Get the connected cell of the neighboring column

Output subroutines

(a) *SaveAParticle* (in "Output.for") – Write the information of the final status of a particle to a disk file;

(b) *SaveFinalDis* (in "Output.for") – Write the final distribution of particles per cell.

4.3 Information flow in DCPT

DCPT can be run with two options: a single species simulation without decay and a multiple species simulation with full capabilities. The information flow is slightly different as shown in Figures 2a & 2b (main flow chart) and Figures 2c & 2d (flow chart for one particle at one time step). The difference is mainly about when the input data are processed to become intermediate parameters that can be directly used in tracking calculations (e.g., the particle-transfer probability and time step for each continuum of each cell). With Option 1, DCPT calculates those parameters for all cells of a grid right after the input data have been read (Figure 2a). With Option 2, however, it calculates those parameters for a cell only when the cell first time becomes a host for particles (Figure 2d) or the parameters need to be updated.

5. DATA INPUT AND OUTPUT

5.1 General Description of Input and Output files

Several data files are used as input and output files by DCPT V1.0. Only the control file has a fixed name (i.e., the name that cannot be specified by the users). The users can choose one of the two input/output file options by specifying the keyword (i.e., MULTI) in the control file. Note that Option 1 has limited capabilities.

5.2 Input files:

Option 1 – Single Species Simulation without Decay

Control File (Fixed name: PTInput.txt):

Data	Format/Range
Title*	Any String
Maximum simulation time	Real (>0)
Filename of the TEC file (matrix)	String in “ ”, any valid filename
Filename of the TEC file (fracture)	
Filename of the mesh file	
Filename of the initial particle distribution	
Filename of the final status of particles	
Filename of the numbers of particles in each cell	
Filename of the flow rates of each connection	
Filename of the connection configurations	
Filename of the sorption parameters (Kd and bulk density)	
Flag indicating whether the dispersion process is considered	Logical (.true. or .false.)
Flag indicating whether the mass exchange between the fracture and the matrix is considered	Logical (.true. or .false.)
Flag indicating whether or not the sorption process is considered	Logical (.true. or .false.)
Longitude dispersivities in the matrix and the fracture continua	Real, separated by a comma (>=0)
Transverse dispersivities in the matrix and fracture continua	
Molecular diffusion coefficient in the matrix and fracture continua	
Dispersivity on the interface between the fracture and the matrix	Real (>=0)

* The first five characters in the control file can be anything but MULTI for Option 1.

TEC file (matrix):
This file can be generated by TOUGH2 V1.4 EOS9. The first three rows will be skipped by DCPT V1.0. Each following row contains information of a cell (matrix part) in the format of "a1, a1, a3, 8x, 3(f10.3, 1x), 15(e14.6)". The variables are *Name, X, Y, Z, P, SL, PCap, Vx, Vy, Vz, Fx, Fy, Fz, Ff-m, Af-m, S, phi, Vn,* and *Tau*. Among these variables, only those listed in the following table are used by DCPT:

Variable	**Format /Range**
Name	String in " ", (the first character must be "M")
X	Real (any real number)
Y	
Z	
SL (liquid saturation)	Real (>=0)
Vx (x-component of pore velocity)	Real (any real number)
Vy (y-component of pore velocity)	
Vz (z-component of pore velocity)	
Ff-m (liquid flow rate from fracture to matrix)	
Af-m (area of the fracture/matrix interface)	Real (>=0)
S (characteristic distance of the fracture-matrix system)	Real (>0)
Phi (porosity)	Real (>=0)
Vn (bulk volume)	
Tau (tortursity)	Real (>=0 and <=1)

TEC file (fracture):

This file can be generated by TOUGH2 V1.4 EOS9. The first three rows will be skipped by DCPT V1.0. Each following row contains information of a cell (fracture part) in the format of "a1, a1, a3, 8x, 3(f10.3, 1x), 15(e14.6)". The variables are *Name, X, Y, Z, P, SL, PCap, Vx, Vy, Vz, Fx, Fy, Fz, Ff-m, Xf-m, S, phi, Vn, Tau,* and *Rf-m*. Among these variables, only those listed in the following table are used by DCPT:

Variable	**Format/Range**
Name	String in " ", (the first character must be "F")
X	Real (any real number)
Y	
Z	
SL (liquid saturation)	Real (>=0)
Vx (x-component of pore velocity)	Real (any real number)
Vy (y-component of pore velocity)	
Vz (z-component of pore velocity)	
Ff-m (liquid flow rate from fracture to matrix)	
Rf-m (reduction factor to the area of the fracture/matrix interface)	Real (>=0 and <=1)
Phi (porosity)	
Vn (bulk volume)	Real (>=0)
Tau (Tortursity)	Real (>=0 and <=1)

Mesh file:
This file contains three sections of information: cells (Section 1), columns (Section 2), and segments (Section 3).

Section 1: the first row is the number of cells (integer). Each cell occupies two rows. The first is the cell *ID* (integer). The second contains *ColumnID* (integer), *CellType* (one character in “”), *x* (real), *y* (real), *z_top* (real), *z_bottom* (real), and *CellSize* (real), separated by a comma.

Section 2: the first row is the number of columns (integer). Each column occupies two rows. The first contains the *ColumnID* (integer) and the number of its neighboring columns (integer). The second contains all neighboring *ColumnID*s (integers), the top cell *ID* (integer) and the bottom cell *ID* (integer) of this column, *x* (real), and *y* (real), separated by a comma.

Section 3: the first row is the total number of the segments (integer). Each segment occupies two rows. The first contains the *segmentID* (integer). The second contains two neighboring *ColumnID*s (integers) and *x-y coordinate*s (real) of two ends of the segment.

This file can be generated using WinGridder V1.0 and up (click File→Save mesh for particle tracking), provided that the 3-D mesh is generated by WinGridder V1.0 and up.

Initial particle distribution file:
This file provides information on the particles to be simulated. It starts with the total number (integer) of the particle groups (the first row). The rows followed are data that

describes each group of particles. Each group occupies one row, including sequentially the number of the particles in the group (integer, >0), the *ID* (integer) of the cell that initially hosts the particles, the *ratio* (real, >0) of the particles that reside in the fracture continuum to the total of the particles in the cell, and the *InitialTime* (the time of the particles entering the domain, real, ≥ 0).

The file of the flow rates per connections:
These data are part of a TOUGH2 output file (Pruess et al., 1996). This file lists all flow rates per connection in an order that is consistent with that of the connection configuration file. The format and the data ranges are the same as those used by TOUGH2 V1.4. The first 7 rows are skipped by DCPT, and then three rows are skipped every 60 rows because of the format used by TOUGH2 V1.4. The data between columns 32 and 46 are in each eligible row used by DCPT.

The connection configuration file:
This file starts with the total number of connections between the cells. Each following row contains the description of one connection, including sequentially the two neighboring cells' IDs (integers, >0) and the area of the connection (the interface). The listing order of the connections must be consistent with that used in the file of the flow rates per connections. Note that there are two flow rates: fracture-flow and matrix-flow rates in the file of flow rate per connection corresponding to one connection in this file because TOUGH2 code splits it into two connections.

The Kd file:
This file includes adsorption-related parameters for each cell. Each row is for one cell, including sequentially the cell ID, the sorption coefficient (K_d, real, >=0) for matrix, rock

density (real, >0) for matrix, K_d (real, >=0) for fracture, and rock density (real, >0) for fracture.

<u>Option 2 – Multiple Species Simulation with Full Capabilities</u>

All files are user-specified except those noted.

Control file (Fixed filename: “PTInput.txt”):

Data	Format/Range
Title*	String in “ ”, (the first five characters must to be MULTI)
Maximum simulation time, Number of species	Real (>0), Real (>0)
Filename of the TEC file (matrix)	String in “ ”, any valid filename
Filename of the TEC file (fracture)	
Filename of the mesh file	
Filename of the initial particle distribution	
Filename of the final status of particles	
Filename of the numbers of particles in each cell	
Filename of the rock properties	
Filename of the flow rates of each connection	
Filename of the connection configurations	
Filename of the rock index for each cell	
Filename of the particle properties	
Flag indicating whether the dispersion process is considered	Logical (.true. or .false.)
Flag indicating whether the mass exchange between the fracture and the matrix is considered	
Flag indicating whether the sorption process is considered	
Flag indicating whether the colloid facilitate process is considered	
Dispersivity on the interface between the fracture and matrix continua	Real (>=0)

*The first five characters have to be “MULTI” to use Option 2.

TEC file (matrix): same as Option 1

TEC file (fracture): same as Option 1

Mesh file: same as Option 1

Initial particle distribution file: same as Option 1

Flow rates per connections: same as Option 1

Connection configuration file: same as Option 1

Rock property file:

This file contains data for each rock card sequentially, except for the first row that contains the number of species. Each rock card starts with a row that includes rock *ID* (integer, positive), *name* (character), *density* (real, >0), *pore size* (real, >0), *concentration of colloids* (real, >=0), and *fraction of colloids filtered* (real, >=0 and <=1). Then the following parameters: *longitude dispersivity* (real, >=0), *transverse dispersivity* (real, >=0), K_d (real, >=0), K_d *for colloids* (real, >=0), and *dispersivity for fracture-matrix interface* (real, >=0), are repeated for each species.

Rock index file:

This file identifies the fracture and matrix rock cards for each cell. Each row represents one cell, including sequentially the cell ID (integer, positive), the rock ID of matrix continuum (integer, positive) and the rock ID of fracture continuum.

Particle property file:

This file includes data for each group of particles. The first row contains the descriptions of parameters and will be skipped by DCPT. Starting from the second row, each row represents one type of particles, includin sequentially particle name (character), particle ID (integer, positive), daughter particle ID (integer, positive), particle size (real, >0), decay rate (real, >=0), molecular diffusion coefficient (real, >=0), and type of particle (character).

5.3 Output Data Files

There are two options for outputs that correspond to the two options for input (see Section 5.2).

Option 1 – Single Species Simulation without Decay

The final status of particles:

This file lists the status of all particles by either the maximum simulation time or the time when the particle exited the domain. The following information will be output for each particle in one row:

Output	**Format**	**Explanation/Range**
Media index – either in fracture or matrix	Real	0=in fracture; 1=in matrix.
Exit status	Single character	'b'=exit through bottom boundary; 't'=exit through top boundary; 'I'=still inside the domain when the maximum time reached
Start time	Real	The time of the particle entering the domain, ≥0
End time	Real	The time of the particle leaving the domain or the maximum simulation time, ≥0 and ≥Start time
X	Real	x-coordinate of the particle, any real number
Y	Real	y-coordinate of the particle, any real number
Z	Real	z-coordinate of the particle, any real number
Starting cell	Integer	The ID of the cell where the particle entered the domain, positive and valid cell ID
Ending cell	Integer	The ID of the cell where the particle resides at the end time, positive, valid cell ID

The final distribution of particles per cell:

This file lists sequentially the x, y, and z coordinates (any real numbers) of every cell in the grid and the number of particles (positive integer) in those cells at the end time. One row represents one cell.

Variable	Format	Explanation
X	Real	x-coordinate of the cell
Y	Real	y-coordinate of the cell
Z	Real	z-coordinate of the cell
Np	Integer	Number of particles in the cell at Tend

Option 2 – Multiple Species Simulation with Full Capabilities

Final status of particles file:

This file lists the status of all particles by either the maximum simulation time or when the particle exited the domain. The following information is output for each particle (one row representing one particle):

Output	Format	Explanation/Range
Media Index – either in fracture or matrix	Real	0=in fracture; 1=in matrix.
Exit status	Single character	‘b’=exit through bottom boundary; ‘t’=exit through top boundary; ‘I’=still inside the domain when the maximum time reached.
Start time	Real	The time of the particle entering the domain, ≥0
End time	Real	The time of the particle leaving the domain or the maximum simulation time, ≥0 and ≥Start time
X	Real	x-coordinate of the particle, any real number
Y	Real	y-coordinate of the particle, any real number
Z	Real	z-coordinate of the particle, any real number
Starting cell	Integer	The ID of the cell where the particle entered the domain, positive and valid cell ID
Ending cell	Integer	The ID of the cell where the particle resides at the end time, positive and valid cell ID
Mass	Real	Mass of the particle, ≥0
Particle type	Single character	‘C’=Colloid, ‘F’=Filtered colloid, ‘N’=Normal, ‘A’=Adhere to Colloid
Parent	Integer	ID of the parent particle if this particle is a daughter product, valid particle ID (positive), a negative number if it is not a daughter product.

Final distribution of particles per cell file: Same as Option 1

6. ANTICIPATED ERROR MESSAGES

A description of error and user responses are provided below:

Error	Response
Error in reading Input Control file.	Check the control file and correct any errors
Incorrect mesh file.	Check the mesh file and correct any errors
Error in tec file, Kd file, rock file, or particle property file.	Check the corresponding files and correct any errors
Error in connection configuration file or flow rates per cell file.	Check the corresponding files and correct any errors, especially the consistency between two files
Error in initial particle distribution file.	Check the control file and correct any errors
Fail to set Flow rate – Unsuccessful in calling SetFlowRate().	Check the values in the mesh file, the tec files, the connection configulation file, and the flow rate file and correct any errors
Fail to set vertical velocity – Unsuccessful in calling SetVelocity().	
Fail to retard the velocity field – Unsuccessful in calling RetardVelocity().	Check the Kd file (option 1) or the rock file (option 2) as well as the mesh file, and correct any errors
Fail in calculation FM interaction – Unsuccessful in calling SetFMTransP().	Check the mesh file, especially the connections between the fracture and the matrix and correct any errors
Fail in setting time step – Unsuccessful in calling SetTimeStep().	Check the input data of the velocities to see if all velocity components are zero for any single cell
Fail in decomposition of D tensors – Unsuccessful in calling CalDTensor().	Check the input data of the dispersivity and diffusion coefficient and make sure that they are physically correct and the diffusion coefficient is not exactly zero if the dispersion Flag in the control file is set to be true.
Mesh Error – The representative node of the cell is not within the representative polygon of the column.	Check the mesh file and correct any errors
Starting point for particle not in current cell – the initial elevation of the particle is not between the top and the bottom elevations of the cell that is identified as containing the particle.	Check the mesh file and make sure the node's z-coordinate of a cell is between the top elevation and the bottom elevation of the cell

7. TEST PROBELMS

We tested DCPT v1.0 in four problems. It is compared with analytical solutions for 1-D and 2-D cases, as well as numerical solutions using T2R3D for a 1-D case and the full 3-D case of the Yucca Mountain UZ model.

7.1 Case 1: Transport in a 1-D Parallel Fracture System

The first test case is a one-dimensional solute transport in fractured porous media with parallel fractures, for which the particle-transfer-probability approach and the sorption model of DCPT can be tested against an analytical solution (Sudicky and Frind 1982). The analytical solution is based on the assumption that chemical transport between fractures and matrix occurs through matrix diffusion in the direction perpendicular to the fracture only. Matrix advection and diffusion in the vertical direction is ignored. Furthermore, the initial solute concentration is zero in the system, and the concentration at inlets of fractures ($z = 0$) is constant for time $t > 0$. For simplicity, the diffusion/dispersion in the fractures is also ignored. The other parameters are shown in Table 1.

Figure 3 shows the predicted breakthrough curve at the outlet. The results from DCPT v1.0 agree well with the analytical solutions. This implies that the particle-transfer-probability approach used in DCPT of diffusive mass exchange between fracture and matrix is representative for this transient case. The CPU time used in simulation by DCPT on a PC (Pentium II 300) is about 10 seconds, excluding the time used for reading/writing files. Filenames are given in Appendix B.

7.2 Case 2: Two-dimensional Transport in Porous Media

The second test case is a two-dimensional solute transport in porous media (no fracture) with a dispersion tensor, for which the advection and dispersion model of DCPT can be tested against an analytical solution. Table 2 shows the case specifications while all parameters are dimensionless.

The problem is actually simulated as a 3-D transport problem with no discretization in the *y*-direction. Solutes are released at time zero in the form of a point pulse source. At dimensionless time = 10, the relative concentration along the dimensionless *x* = 10.25 is calculated within the specific slice. Figure 2 compares these results with the analytical solution (Bear 1979):

$$M(z)/M_0 = \tfrac{1}{M_0}\int_{-\infty}^{+\infty}\int_{10.24}^{10.26} C(x,y,z,T)\,dx\,dy = \frac{1}{4\pi\, T\sqrt{D_x D_z}}\exp\left[-\frac{(z-V_z)^2}{4D_z T}\right] \tag{8}$$

where *T* is the dimensionless monitoring time (= 10), and D_x and D_z are dispersion coefficients corresponding to *x*-direction and *z*-direction, respectively. The concentration distribution simulated by DCPT v1.0 is consistent with the analytical solution. Two million particles were used in the simulation (Figure 4), and the CPU time used is about 10 minutes on a PC with a Pentium II 300 processor.

All input and output filenames are given in Appendix B.

7.3 Case 3: One-dimensional Transport in Fractured Porous Rocks

This one-dimensional transport problem is designed to further test the capabilities of the particle-tracking models against the numerical solutions provided by T2R3D. The main focus is on simulations of the fracture-matrix mass exchange and sorption processes in existing of water flow through the fracture continuum and the matrix continuum as well as the interface between them. For such a complicated transport system, no analytical solution is available. The case involves the column near borehole SD-9 extracted from the 1997 FY 3-D model of the Yucca Mountain site. The radionuclides are released at the repository at time zero as a pulse. A steady-state flow field is assumed and determined using TOUGH2 version 1.4. The transport parameters used in simulations are shown in Table 3. Radioactive decay is ignored in the simulations for simplicity. A total of 2,000 particles are used in DCPT simulation. The CPU time used is about 10 seconds for both DCPT and T2R3D while DCPT is executed on a Pentium II PC and T2R3D is executed on a DEC ALPHA (Unix). A total of 27 cells are used.

In this case, significant mass flow occurs as a result of advection and dispersion between fracture and matrix. Figure 5 shows the relative mass through the water table versus time. The relative mass is defined as the cumulative mass flowing out to ground water divided by the total mass released at the repository horizon. In both cases, the results are very similar except that the DCPT shows fewer numerical mixing effects than the T2R3D, especially in the case of a smaller molecular diffusion coefficient (Figure 5a).

7.4 Case 4: a Full 3-D Problem of Transport in Yucca Mountain Site

The full 3-D model of the Yucca Mountain unsaturated zone is a comprehensive, mountain-scale model. It includes all aspects of flow and transport processes in the fractured porous media that exist there, which provides a comprehensive test case for the particle-tracking simulator and other numerical simulators. Although no analytical solution is available for such a complex system, comparison of the particle tracker (DCPT) with the numerical solutions (T2R3D) provides insights into these methods.

The three-dimensional model domain and the corresponding irregular numerical grid of this mountain-scale problem are shown in plan view in Figure 6. The model domain covers a total area of approximately 40 km^2. Domain thickness varies from 500 to 700 m, depending on local topography. The domain is filled with layered tuffs of spatially varied thickness complicated by faults. From the land surface downward, these layers are the Tiva Canyon welded (TCw) unit, the Paintbrush nonwelded (PTn) unit, the Topopah Spring welded (TSw) unit, the Calico Hills nonwelded (CHn), and the Crater Flat undifferentiated (Cfu) units. Fracture density varies with the different layers, mainly depending on the degree of welding. The potential repository is located in the middle of the domain, an approximate area of 1,000 m (east-west) by 5,000 m (north-south), surrounded by faults (Figure 6). The repository is represented by a number of 5 m thick grid cells. A steady-state flow field corresponding to the present-day infiltration map is assumed and calculated using the TOUGH2 code (Wu et al., 2000). The dual-permeability grid used in calculation has 104,156 grid cells and 421,134 connections. A pulse release of two radionuclides into fractures with technetium as a conservative tracer and neptunium as a reactive (i.e., adsorbing) tracer is considered. Table 4 shows the parameters used in the simulation. Radioactive decay and mechanical dispersion are ignored for simplicity.

A total of 1680 particles are used in the simulations using DCPT. The corresponding CPU time used for each run is about 310 seconds using DCPT on a Pentium II PC and about 3600 seconds using T2R3D on a DEC ALPHA, respectively.

The cumulative mass fractions entering ground water versus time are depicted on Figures 7 (technetium) and 8 (neptunium). In the figures, the relative mass is defined as the cumulative mass of the radionuclides passing through the water table over time, normalized by the total mass released at the repository. The results of DCPT agree very well with the results of T2R3D, which rigorously solves the advection-dispersion equation of radionuclide transport at the Yucca Mountain site. Its performance will not diminish as the size of the grid (number of cells) increases, a feature particularly important in large-scale models such as the UZ Model of the Yucca Mountain.

8 SUMMARY AND CONCLUSIONS

A new software, DCPT V1.0, has been developed. Comparisons between DCPT simulation results with analytical solutions and results obtained from T2R3D show that DCPT could be used to simulate chemical transport processes associated with linear sorption in fractured porous media without numerical dispersion. DCPT provides results nearly identical to those of T2R3D for the time frames and scenarios considered. It can effectively simulate complex transport processes of radionuclides in dual continuum media. It is an efficient simulator, in terms of computational requirements, especially when only a cumulative breakthrough curve is required. Its performance will not suffer as the number of grid cells increases, a feature particularly important in large-scale models.

9 REFERENCES

Bear, J. 1972. *Dynamics of Fluid in Porous Media.* New York, New York: Dover Publications.

Bear, J. 1979. *Hydraulics of Groundwater.* New York, New York: McGraw Hill.

LaBolle, Eric M.; Graham E. Fogg; and Tompson, Andrew F.B. 1996. "Random-Walk Simulation of Transport in Heterogeneous Porous Media: Local Mass-Conservation Problem and Implementation Methods." *Water Resour. Res. (32)*, 583-593. Washington, D.C.: American Geophysical Union.

Liu, H.H., G.S. Bodvarsson, and L. Pan. 2000. "Determination of particle transfer in random walk particle methods for fractured porous media." *Water Resour. Res. (32)*, 583-593. Washington, D.C.: American Geophysical Union.

Press, H. W., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. 1992. Numerical recipes in FORTRAN: The art of scientific computing (Second edition). Cambridge University Press.

Pruess, K. 1991. *TOUGH2-A General Purpose Numerical Simulator for Multiphase Fluid and Heat Flow.* Report LBL-29400, UC-251. Berkeley, California: Lawrence Berkeley National Laboratory.

Sudicky, E.A.; Frind, E.O. 1982. "Contaminant Transport in Fractured Porous Media: Analytical Solutions for a System of Parallel Fractures." 1634-1642. Washington, DC: American Geophysical Union.

Tang, D.H. et al. 1981. "Contaminant Transport in Fractured Porous Media: Analytical Solution for a Single Fracture." 555-564.

Wu, Y.S.; Ahlers, C.F.; Fraser, P.; Simmons, A.; and Pruess, K. 1996. *Software Qualification of Selected TOUGH2 Modules.* Report LBNL-39490, UC-800. Berkeley, California: Lawrence Berkeley National Laboratory.

Wu, Y. S and K. Pruess. 1998. A 3-D Hydrodynamic Dispersion Model for Modeling Tracer Transport in Geothermal Reservoirs, *Proceedings of the Twenty-third Workshop, Geothermal Reservoir Engineering*, 139-146. Stanford, California: Stanford University.

Wu, Y. S., J. Liu, T. Xu, C. Haukwa, W. Zhang, H.H., Liu, and C.F. Ahlers, UZ flow models and submodels. Analysis/Model Report MDL-NBS-HS-000006 REV00G, CRWMS M&O, 2000.

ACKNOWLEDGEMENTS

This work was supported by the Director, Office of Civilian Radioactive Waste Management, U.S. Department of Energy, through Memorandum Purchase Order EA9013MC5X between Bechtel SAIC Company, LLC and the Ernest Orlando Lawrence Berkeley National Laboratory (Berkeley Lab). The support is provided to Berkeley Lab through the U.S. Department of Energy Contract No. DE-AC03-76SF00098.

APPENDIX A: DERIVATION OF THE PARTICLE-TRANSFER PROBABILITY

In the framework of the dual-continuum model, the mass of particles initially in continuum j will be partitioned into three portions after a duration [0, t]. They are:

(a) Mass entering the same continuum of neighboring cells (m^j_{out});

(b) Mass entering the other continuum of the same cell (m_{jk});

(c) Mass staying in the same continuum of the same cell (m^j_t).

Therefore, the particle-transfer probability, at which a single particle in continuum j at t = 0 will be in continuum k at time t, can be defined as the following ratio:

$$P_{jk} = \frac{m_{jk}}{m_{jk} + m^j_{out} + m^j_t} = \frac{m_{jk}}{m^j(0)} \qquad (\text{A1})$$

where $m^j(0)$ is the mass of particles initially in continuum j.

The mass-balance equation for the particles in continuum j at t = 0 for a given cell can be expressed as:

$$C_j(0)(Vol_j + Kd_j m_j) = C_j(t)(Vol_j + Kd_j m_j) + \int_0^t F_{jk}\, C_j(\tau)\, d\tau + \int_0^t Q_j C_j(\tau)\, d\tau \qquad (\text{A2})$$

where C_j, Vol_j, Kd_j, and m_j are the concentration, volume of liquid, adsorption coefficient, and mass of rock in continuum j, respectively. F_{jk} and Q_j describe the strength of advection and dispersion/diffusion processes through the fracture-matrix interface and the interfaces to adjacent grid cells in continuum j, respectively. The detailed expressions of these two parameters are shown in Equations (6a) and (6[illegible]) for fracture continuum (j = f), and Equations (6d) and (6e) for matrix continuum (j = m), respectively. Both F_{jk} and Q_j are assumed to be constant within the time interval (0, t). Note that handling F_{jk} and Q_j in this way is equivalent to the way for representing the net mass flux through an interface in a finite-difference method. For example, the net mass flux from fracture continuum to matrix continuum can be expressed as (replacing j with f and k with m):

$$F_{fm}\, C_f - F_{mf}\, C_m = \max(q_{fm} A_{fm}, 0)\, C_f - \max(-q_{fm} A_{fm}, 0)\, C_m + \frac{D_m A_{fm}}{S_{fm}} \left[C_f - C_m\right] \qquad (\text{A3})$$

In Equation (A3), the sum of first two terms on the right side represents the advection flux while the last term on the right side is simply the finite-difference approximation of the diffusion flux at the fracture-matrix interface.

Taking derivatives on both sides of Equation (A2) with respect to t, we have a first order ordinary differential equation:

$$\frac{dC_j(t)}{dt} + \frac{1}{\tau_j} C_j(t) = 0 \qquad \text{(A4)}$$

where, τ_j is the characteristic time of continuum j for the given cell, and is defined as :

$$\tau_j = \frac{Vol_j + Kd_j\, m_j}{F_{jk} + Q_j} \qquad \text{(A5)}$$

The solution of Equation (A4) is readily obtained as

$$C_j(t) = C_j(0)\ \exp\ (-t / \tau_j) \qquad \text{(A6)}$$

Therefore, the probability of a particle being transferred from continuum j to continuum k during (0, t) can be calculated as:

$$P_{jk} = \frac{F_{jk}\int_0^t C_j(\tau)d\tau}{C_j(0)\left(Vol_j + Kd_j\, m_j\right)} = \frac{F_{jk}}{Q_j + F_{jk}}\left[1 - \exp\left(-t/\tau_j\right)\right] \qquad \text{(A7)}$$

Note that, in the above derivation, we use t instead of Δt for symbolic simplicity. The t = 0 actually indicates the starting point of a time step. The particle-transfer probability from fracture continuum to matrix continuum defined in Equation (5a) are obtained by replacing t with Δt, j with f, and k with m in (A7), respectively. Expression (5b) can be obtained in the similar way.

APPENDIX B: INPUT/OUTPUT FILES USED IN TESTING OF DCPT

List of Files Used in Case 1

Filename	Description
“FM1DR.in”[1]	Control file. List of all input/output files and parameters used by DCPT (copied to PTInput.txt before use)
"FM1D_m.tec"	List of the flow field and other transport parameters of the matrix for each cell
"FM1D_f.tec"	List of the flow field and other transport parameters of the fracture for each cell
“FM1D.txt”	Mesh file (cells, columns, and segments)
“FM1Dini.txt”	List of initial distribution of particles
“FM1DoutR.txt”	Output file, list of the final status of particles
“FM1D.wb3”	A “Corel Quattro Pro 7” file which contains all post-process results and comparisons with the analytical solutions

[1] Four files omitted in the control file because the related information has been hard-incorporated into the code of DCPT (commented out for other simulations) by taking advantages of uniform media. So the adsorption coefficients.

List of Files Used in Case 2

Filename	Description
“Analy3D.in”	Control file. List of all input/output files and parameters used by DCPT (Copied to PTInput.txt before use)
“Analy3D_m.tec”	List of the flow field and other transport parameters of the matrix for each cell
“Analy3D_f.tec”	List of the flow field and other transport parameters of the fracture for each cell
“Analy3D.txt”	Mesh file (cells, columns, and segments)
“Ana3Dtextini.txt”	List of initial distribution of particles
“Analy3Dout.txt”	Temporary output file, list of the final status of particles
“Analy3D2M.out”	Output file, distribution of particles along the specific line in space (y=0)
“Analy3D.wb3”	A “Corel Quattro Pro 7” file which contains all post-process results and comparisons with the analytical solutions

List of Files Used in Case 3

Filename	Description
"UZ97_1D.in"	Control file. List of all input/output files and parameters used by DCPT for the case without sorption (Technetium) (Copied to PTInput.txt before use)
"UZ97_1Dm.tec"	List of the flow field and other transport parameters of the matrix for each cell
"UZ97_1Df.tec"	List of the flow field and other transport parameters of the fracture for each cell
"UZ97_1DR.mesh"	Mesh file (cells, columns, and segments)
"UZ97_1DPT.ini"	List of initial distribution of particles
"UZ97_1Dout.txt"	Temporary output file, list of the final status of particles. The results are loaded into the spreadsheet file before another run of DCPT
"UZ97_1Dfmd.dat"	List of the characteristic distances of the fracture systems in each cell
"UZ97_1D.flow"	List water flow rates (via both fracture and matrix) per connections of neighboring cells (part of TOUGH2 output)
"UZ97_1Dcon.dat"	Configuration of cell connections in the grid
"UZ97_1D.kd"	List of values of Kd and bulk density of related cells
"UZ97_1DR.in"	Control file. List of all input/output files and parameters used by DCPT for the case with sorption (Neptunium) (Copied to PTInput.txt before use)
"UZ971D.wb3"	A "Corel Quattro Pro 7" file which contains all post-process results and comparisons with the numerical solutions for the case without sorption (Technetium)
"UZ971DR.wb3"	A "Corel Quattro Pro 7" file which contains all post-process results and comparisons with the numerical solutions for the case with sorption (Neptunium)

List of Files Used in Case 4

Filename	Description
"UZ99.in"	Control file. List of all input/output files and parameters used by DCPT for the case without sorption (Technetium)
"UZ99_m.tec"	List of the flow field and other transport parameters of the matrix for each cell
"UZ99_f.tec"	List of the flow field and other transport parameters of the fracture for each cell
"UZ99mesh.txt"	Mesh file (cells, columns, and segments)
"UZ99Ptini.txt"	List of initial distribution of particles
"UZ99out.txt"	Output file, list of the final status of particles
"UZ99.flow"	List water flow rates (via both fracture and matrix) per connections of neighboring cells (part of TOUGH2 output)
"UZ99mesh.con"	Configuration of cell connections in the grid
"UZ99.kd"	List of values of Kd and bulk density of related cells
"UZ99DR.in"	Control file. List of all input/output files and parameters used by DCPT for the case with sorption (Neptunium)
"UZ99.wb3"	A "Corel Quattro Pro 7" file which contains all post-process results and comparisons with the numerical solutions for the case without sorption (Technetium)

APPENDIX C: REQUIREMENTS AND SYSTEM LIMITATIONS

Hardware, Operating system, and Compiler Requirements:

DCPT V1.0 has been tested on the following platforms:

Platform	Operating System	Compiler
PC Pentium II	Windows 95/98	Digital Visual Fortran V5.0.A

User Requirements:

The correct implementation, setup, problem formulation, and interpretation of the results of DCPT require knowledge of the basic principles of fluid flow and transport in variably saturated, fractured porous media. The user must understand numerical methods in general and the random-walk particle tracking methods and dual-continuum approaches used in DCPT in particular. In addition, an in-depth understanding of the transport system being modeled and of the data being input to the model is crucial for interpreting the results of DCPT.

POST-PROCESSING:

The output of DCPT lists only the final status (e.g., location and travel time) of simulated particles. The user needs post-processing programs or software to calculate either breakthrough curves or spatial distribution of mass depending on the particular purposes.

Table 1 Parameters Used for the Transport Problem in a Parallel Fracture System

Parameter	**Value**
Molecular diffusion coefficient (D_m)	2.5×10^{-11} m^2 /s
Fracture spacing (2B)	1.0 m (Case 1) or 10.0 m (Case 2)
Fracture aperture (2b)	2×10^{-5} m
Retardation factor (R)	30
Volumetric Water content in matrix	0.1 m^3/ m^3
Velocity in fracture	1.1574×10^{-5} m/s
Grid spacing	0.5 m
Matrix volume per cell	0.25 m^3 (Case 1) or 2.5 m^3 (Case 2)
Fracture volume per cell	0.5×10^{-5} m^3
Fracture-matrix interface area	0.5 m^2
Distance from the source of tracer to the bottom	36.5 m

Table 2. Parameters of the 2D Transport Problem with a Dispersion Tensor

Parameters	Value
Domain dimension (x, y, z)	20.5 × 20.5 × 30.5
Pore velocity	Vx = Vy = 0, Vz = 1
Dispersivity	$\alpha_L = 0.05$, $\alpha_T = 0.01$
Diffusion coefficient	0.0
Grid spacing	$\Delta x = \Delta z = 0.5$; $\Delta y = 20.5$
Plume location at t = 0	X = 10.25, y = 10.25, and z = 0.0
Monitoring location at t = 10	X = 10.25
Monitoring resolutions	$\delta x = 0.02$ and $\delta z = 0.01$

Table 3. Parameters Used for 1-D Radionuclide Transport

Parameter	Value
Molecular diffusion coefficient of technetium	3.2×10^{-11} m^2 /s
Molecular diffusion coefficient of neptunium	1.6×10^{-10} m^2 /s
Fracture longitudinal and transverse dispersivity	20 m and 2 m
Matrix longitudinal and transverse dispersivity	Set to be zero
Fracture-matrix dispersivity	Set to be zero
Fracture and matrix tortuosities	0.7 and 0.7
Temperature	25 °C
Distribution coefficients (K_d) of technetium	Set to be zero in both fracture and matrix
Distribution coefficients (K_d) of neptunium	Set to be zero in fracture and matrix of TCw, PTn, TSw units; 4×10^{-3} kg/m^3 and 1×10^{-3} kg/m^3 in matrix of zeolitic rock and vitric rock in CHn unit, respectively.

Table 4 Parameters for the mountain-scale transport problem

Parameters	**Technetium** (Tc)	**Neptunium**(Np)
Molecular diffusion coefficient (m^2/s)	3.2×10^{-11}	1.6×10^{-10}
K_d (cm^2/s)	0.0 for all fractures and matrix	4.0 for zeolitic matrix in CHn; 1.0 for vitric matrix and fault matrix in CHn; 1.0 for matrix in TSw; 0.0 for else

Note: CHn—Calico Hills non-welded unit; TSw—Tiva Canyon welded unit.