Appendix G

NUCLEAR WASTE MANAGEMENT PROGRAM Sandia National Laboratories	**User's Manual Criteria Form**	**Form Number: NP 19-1-6** **Page 1 of 1**

Does the user's manual contain as appropriate:

1. **Software Name:** DTRKMF
2. **Software Version:** 1.00
3. **Document Version:** 1.00
4. **ERMS #:** 523246

5.	A statement(s) of functional requirements (consistent with those in the RD) and system limitations?	☒	Yes	☐	No	☐	N/A	☐ N/R
6.	An explanation of the mathematical model and numerical models?	☒	Yes	☐	No	☐	N/A	☐ N/R
7.	Physical and mathematical assumptions?	☒	Yes	☐	No	☐	N/A	☐ N/R
8.	The capabilities and limitations inherent in the software?	☒	Yes	☐	No	☐	N/A	☐ N/R
9.	Instructions that describe the user's interaction with the software?	☒	Yes	☐	No	☐	N/A	☐ N/R
10.	The identification of input parameters, formats, and valid ranges?	☒	Yes	☐	No	☐	N/A	☐ N/R
11.	Messages initiated as a result of improper input and how the user can respond?	☒	Yes	☐	No	☐	N/A	☐ N/R
12.	The identification and description of output specifications and formats?	☒	Yes	☐	No	☐	N/A	☐ N/R
13.	A description of any required training necessary to use the software?	☒	Yes	☐	No	☐	N/A	☐ N/R
14.	The identification of components of the code that were not tested?	☒	Yes	☐	No	☐	N/A	☐ N/R

15. David K. Rudeen **Code Team/Sponsor** (*print*)	David K Rudeen Signature	10-3-02 Date
16. Scott C. James **Technical Reviewer** (*print*)	Scott James Signature	10/03/2002 Date
17. for M. Kathryn Knowles **Responsible Manager** (*print*)	David Kessel Signature	4/3/03 Date
18. Rodger E. Coman **SCM Coordinator** (*print*)	Rodger E. Coman Signature	4/3/03 Date

for 4-3-03

Key for check boxes above:

Check **Yes** for each item reviewed and found acceptable	Check **No** for each item which requires further work
Check **N/A** for items not applicable	Check **N/R** for items not reviewed (multiple technical reviews)

Information Only

WIPP PA

User's Manual

for

DTRKMF Version 1.00

Document Version 1.00

EMRS # 523246

April 2003

This page intentionally LEFT BLANK.

Table of Contents

List of Figures

Intentionally left blank.

1.0 INTRODUCTION

The Waste Isolation Pilot Plant (WIPP) is located in southeastern New Mexico and is being developed by the U.S. Department of Energy (DOE) as a disposal facility for transuranic (TRU) waste (U.S. DOE, 1980, U.S. DOE, 1990). The WIPP must comply with various environmental regulations including 40 CFR 191, subpart B, Environmental Radiation Protection standards for the management and disposal of spent nuclear fuel and high-level and TRU radioactive wastes (U.S. EPA, 1985), and 40 CFR 268.6, which petitions to allow land disposal of waste prohibited under subpart C of part 268 (U.S. EPA, 1986). As part of the development process for the WIPP, a sequence of performance assessments (PAs) has been carried out by Sandia National Laboratories (SNL) to organize knowledge currently available about the WIPP and to provide guidance for future research and development efforts (WIPP PA, 1991; WIPP PA, 1992). The most recent iteration of these PAs was completed in the summer of 1996 and supports an application, designated the Compliance Certification Application, by DOE and the Environment Protection Agency (EPA) for the certification of the WIPP for the disposal of TRU waste (U.S. DOE, 1996). The next iteration of these PAs will be for the mandated five year Re-Certification (U.S. EPA, 1996).

In the previous WIPP PA, block centered, finite difference flow codes BRAGFLO and SECOFL2D were first run to compute flow fields and then the radionuclide transport codes NUTS and SECOTP2D were run using the predicted flow fields to compute transport solutions. The BRAGFLO/NUTS pair was used in the vicinity of the repository while the SECO pair was used in the Culebra. The transport solutions were subsequently used to determine if there were releases to the accessible environment. This process was both time consuming and expensive. Moreover, with the grids used, the solutions suffered from excessive numerical dispersion.

DTRKMF, Double precision particle TRacKing for MODFLOW 2000 (Harbaugh, 2000), is being developed to help streamline the PA process by abstracting 2-D or 3-D flow fields to 1-D particle tracks and represents the first phase of a two-phase development process. This version will get its input from MODFLOW 2000 input and output files. The second phase will be to map simplified transport solutions onto the 1-D tracks. The purpose of DTRKMF is to calculate particle tracks in 2-D or 3-D for steady state and time dependent, variably saturated flow fields. For time dependent, variably saturated flow fields, the particles are tracked until: (1) an arbitrary polygonal boundary in 2-D is reached or an arbitrary x-y polygonal edge bounded by a top and bottom z-plane in 3-D (prism) is reached; or, (2) a given simulation time is exceeded. For steady-state flow fields, the particles are tracked until either of the above stopping criteria is met or a particle is trapped in a cell.

This document serves as a User's Manual for the software DTRKMF. It describes the inputs, outputs as well as how the user interacts with and executes the code. There are only minor differences from the *DTRKCDB User's Manual* (WIPP PA, 2002) related to specifying the MODFLOW 2000 input files containing the grid description.

The following is an outline for the remainder of this document. In Section 2, the recommended background for a user is provided. A summary of the conceptual, theoretical and mathematical basis for the code is given in Section 3. In Section 4, the capabilities and limitations are presented. For completeness, the requirements (capabilities) have been included in Appendix A. Section 5 provides an overview of the execution of the program and a description of the command line arguments.

Section 6 contains the user input data descriptions and instructions and provides the structure of the output data files. Section 7 describes the allowable data ranges for the few variables for which there are constraints, and Section 8 concludes with a listing and description of error messages.

1.1 Software Identifier

Code name:	DTRKMF	
WIPP Prefix:	DTRKMF	
Version:	1.00	
Build Information:	Executable file name:	*/h/WIPP/dtrkmf/dtrkmf_v0100*
	Build date/time:	September 23, 2002/ 13:04
	IMAGE FILE SIZE:	945442
Platform:	PC – i686;	
Operating System	Red Hat Linux release 7.2 (Enigma) Kernel 2.4.7-10	
Compiler	Lahey/Fujitsu Fortran 95 Compiler Release L6.10a	

1.2 Points of Contact

Code Sponsor:
David Rudeen
GRAM, Inc.
8500 Menaul Blvd. N.E.
Suite B-335
Albuquerque NM 87112
Voice: (505) 998-0046

SCMS Librarian:
Rodger E. Coman
Compaq Federal, LLC
recoman@sandia.gov

CVS Librarian:
David B. Hart
SNL Org. 6115
dbhart@sandia.gov

2.0 RECOMMENDED USER TRAINING AND/OR BACKGROUND

To run DTRKMF code successfully, the user needs a basic knowledge of variable saturated flow and more specifically MODFLOW 2000 flow code used in WIPP PA analyses. The user also needs a basic knowledge of the Unix and/or Linux operating systems. In addition, the user must have access to the WIPP Linux cluster or their equivalent.

3.0 DESCRIPTION OF MODELS AND METHODS

The information in this section is a summary of the theoretical overview found in the *DTRKMF Design Document* (WIPP PA, 2003a).

3.1 Conceptual Model

In block centered finite difference flow codes, nodes are at the center of a cell bounded by the faces determined from a rectilinear grid. As a result, these types of codes compute the velocities and/or flow rates at the cell face boundaries. Following Pollock (1989), who developed the MODPATH particle tracking code, the particles are tracked cell-by-cell using a semi-analytical solution (WIPP PA, 2003a). It is assumed that the velocities vary linearly between the cell faces as a function of the space coordinate and for time dependent cases, that the velocities at the faces vary linearly between time planes. The interested reader can find a discussion in Zheng and Bennett (1995). For steady-state problems the initial locations of the particles will be specified and for time dependent problems the initial locations will be specified and starting times will be specified from either the control input file or from the time data in the CDB input file. For efficiency, all particles are moved in a time step. The particles will be tracked until: (1) an arbitrary polygonal boundary in 2-D is reached or an arbitrary x-y polygonal edge bounded by a top and bottom z-plane in 3-D (prism) is reached; or, (2) a given simulation time is exceeded. For steady-state flow fields, the particles are tracked until either of the above stopping criteria is met or a particle is trapped in a cell.

3.2 Summary of Mathematical Model

In this section, a cell-by-cell semi-analytical solution is developed. For a continuous, time dependent, variably saturated flow field, the particle tracks are given by the solution to the equation:

$$s(\vec{\mathbf{x}},t)\phi(\vec{\mathbf{x}},t)\frac{d\vec{\mathbf{x}}}{dt} = \vec{v}(\vec{\mathbf{x}},t), \tag{3.1}$$

where s is the saturation, ϕ is the porosity, $\vec{\mathbf{x}}$ is the two- or three-dimensional location vector, t is time, and $\vec{\mathbf{v}}$ is the two- or three-dimensional velocity vector.

Now, for any given i,j,k cell, which will be denoted by the subscript c, (3.1) can be written as

$$s_c(t)\phi_c(t)\frac{dx_{c_i}}{dt} = \frac{\left[v_{c_i 2}(t) - v_{c_i 1}(t)\right]}{\Delta x_{c_i}}\left(x_{c_i} - x_{c_i 1}\right) + v_{c_i 1}(t), \tag{3.2}$$

where, the subscript i is the space coordinate component and ranges between 1 and the number of components (2 or 3), and the subscripts 1 and 2 refer to the (left, bottom, back) and (right, top, front) faces respectively, which are determined by the rectilinear grid, and

$$\Delta x_{c_i} = x_{c_i 2} - x_{c_i 1}. \tag{3.3}$$

Using the assumption that the face velocities vary linearly between time planes, a general face velocity component can be written as

$$v_c(t) = \frac{\left[v_c(t_1) - v_c(t_0)\right]}{\Delta t}(t - t_0) + v_c(t_0), \tag{3.4}$$

where t_0 and t_1 denote the time of the n and n+1 time steps respectively and

$$\Delta t = t_1 - t_0. \tag{3.5}$$

A note is in order regarding (3.2). If the either the porosity or the saturation is zero, then the face velocity components are assumed zero as well. For example, in the BRAGFLO code, when saturation is less than residual brine saturation then the velocity/flow rates are set to zero. However, in other codes, this may not be the case. If DTRKMF finds that the product of porosity and saturation is zero and the velocity is *not* zero an error is assumed; the code will write a warning message and the particle will be marked inactive.

Rearranging and combining (3.2) and (3.4) leads to an ODE of the form:

$$\frac{d\xi}{d\tau} = (A\tau + B)\xi + (C\tau + D), \tag{3.6}$$

where, ξ and τ are the normalized coordinate position and time, respectively, and the coefficients A, B, C, and D are functions of the face velocities, and the time and spatial increments. Equation (3.4) is solved using the method of integrating factors. Specific values and relative values of the coefficients lead to simplifications and special cases that are handled individually by the code. These special cases lead to analytical solutions that are functions of the complex error function.

As an alternative to the analytic solution, the SLATEC(DEPAC) routine DDBEAM has also been implemented (Shampine and Watts, 1979; Haskell, et al., 1980). DDBEAM is a double precision routine that solves initial value problems in ordinary differential equations using the Adams-Bashforth-Moulton Predictor-Corrector formulas of orders one through twelve to integrate a system of first order ordinary differential equations.

3.3 Boundary Crossing Model

The boundary in the x-y plane is made up of piecewise straight-line segments that form a simply connected closed curve. To determine if a particle has exited the boundary, a coordinate system is set up on each of the straight-line segments with the origin at the beginning of the segment. If the points are entered in a clockwise direction, the new coordinate system has an outward pointing positive axis. Particle locations are transformed into the local coordinate system and then a simple check can be made to evaluate whether the particle has crossed the particular boundary segment. Details can be found in *DTRKMF Design Document* (WIPP PA, 2003a).

4.0 CAPABILITIES AND LIMITATIONS

The capabilities of the DTRKMF code are best summarized by the list of functional requirements contained in Appendix A and detailed in *DTRKMF Requirements Document* (WIPP PA, 2003b).

DTRKMF uses FORTRAN 90 dynamic memory management to size arrays at execution time based on the current problem configuration. The primary variables affecting array dimensions are the grid and time dimensions of the flow field and the number and locations (in time and space) of the particles that will be dropped into the flow field. Therefore, there is essentially no problem size limitation beyond the storage and CPU availability of the computer used for the analysis.

One exception is the number of time points along a path. Because it is not possible to accurately predict the number, it is estimated using the number of time steps on the flow field file and the number of subcycle steps across a cell. A maximum of 1000 has been selected. If a particle path reaches the calculated maximum number of points, a warning is printed and the particle is marked as inactive.

DTRKMF will only input flow fields from files generated by MODFLOW 2000 or files with the same format. This format is detailed in Section 6.2 of this report.

Boundary segments in 2D (or boundary planes in 3D) are described by line segments in the x-y plane and minimum and maximum z-values. Boundary planes are, therefore, either perpendicular to the x-y plane or parallel to it.

5.0 EXECUTION OF DTRKMF

5.1 Overview

A typical DTRKMF execution requires two input files and will create up to two output files. The input files are: (1) a text control file that contains user specified control parameters described below; and, (2) the flow field/budget file created by MODFLOW also described below. Generally, the MODFLOW flow fields are saturated and either transient or steady state. A third input file containing the input grid description file used by MODFLOW is specified in the text control file.

The output files are: (1) an optional text file containing the information describing the calculated particle paths (described in Section 6.3.1); and, (2) a diagnostic or summary text file containing an echo of the input control file, a brief summary of the execution and possibly extensive, user controllable, debugging information. A user settable flag controls the echo of summary and error information to the computer screen. This flag should be turned off for batch type production runs.

The debug or summary file contains a table that summarizes the particle path information by tabulating each particle's initial position and drop time, final arc length, position and exit time. This table is a good place to start an analysis of results. It gives the user an idea of which particles might be worth examining in more detail.

The optional text file containing particle path information is created at the end of the calculation. Therefore, they will not be created for an execution that terminates abnormally. However, during the execution a scratch file named *fort.21* (system dependent default for unit 21) containing particle path information up to the time of abnormal termination is created which can be used as a debugging aide. This file is deleted automatically if the program terminates normally.

5.2 Command Line

To execute DTRKMF, specify the executable with four file names as command line arguments ($ is a configuration dependent prompt):

```
$ dtrkmf    control.inp    -
            mf2k.bud       -
            particle.out   -
            dtrkmf.dbg
```

where,

`control.inp` – The input control file (text file) containing particle tracking control parameters.

`mf2k.bud` – The input MODFLOW flux or budget file.

`particle.out` - The output particle path text file. If NULL the file is not created.

`dtrkmf.dbg` – The output debug text file containing summary information from the execution of the DTRKMF program including an echo of the input control file.

Note that for production, the files and directories will be different, but the structure of the command line will be the same.

6.0 INPUT AND OUTPUT SPECIFICATIONS

6.1 Input Control File

The input control file is a keyword driven ASCII text file. Major sections of the input file are bounded by records containing *KEYWORD and *END keywords. The three section keywords are *CONTROL, *PARTICLE and *BOUNDARY. Any number of comment records (or blank records) can precede a section *KEYWORD record (or appear after a *END record). Between the *KEYWORD and *END records the data records are order dependent and are free formatted – any number of blanks can be placed between values on a record. Currently, blanks are the only acceptable value separator. Blanks cannot be used within a character string. Character string data should not be delimited with ' or ". Within the *CONTROL section there are two secondary keywords, *GRID and *FLOW, which mark the beginning of the grid variable name specifications and the flow variable name specifications, respectively. A detailed summary of all records follows. An example is provided in Figure 6.1.

*CONTROL
Record 1: Flow_Flag, Steady_State_Flag, Data_Echo_Flag, Interface_Area_Flag, History_Steps_Flag, ODE_Flag, Debug_Flag

Flow_Flag – Must be set to MODFLOW .

Steady_State_Flag – If set to Steady_State then steady state flow is assumed using the flow field from the time step specified on the next record, otherwise transient flow is assumed.

Data_Echo_Flag – If set to ECHO then data and errors are echoed to the computer screen as well as written to the debug file, otherwise they are only written to the debug file.

Interface_Area_Flag – If set to AVERAGE then the flow velocity at the interface is a function of the average of the cell cross sectional areas on either side of the interface. Otherwise, the actual cell cross sectional area is used to calculate the component of the flow velocity within the cell.

History_Steps_Flag – If set to HISTORY_STEPS then the times from the history only steps on the input CDB are used to control the time steps for the particle tracking. Otherwise, the times from the whole time steps from the CDB are used. Ignored, but retained for consistency with DTRKCDB.

ODE_Flag – Set to ANALYTIC if the semi analytic solution technique is desired or set to ODE if the SLATEC ODE solver is desired.

Debug_Flag – Set to DEBUG*n*, (*n* = 0,1,2 or 3) for printing debug information. *n* is a level number for controlling the amount of debug information printed. *n*=blank is the same as *n*=0 and is the lowest level of debug output and consists of time step and cell transit time information only. Level 1 adds considerable information from the controlling subroutine CONTROL. Level 2 adds details from lower level subroutines like BUILDCOEF and COORDOFTIME. Levels 1 and 2 can output a considerable amount of information and are recommended only for a single particle track.

Record 2: Sim_Time, Tolsv, SS_Step, NoSS, Nsubcl, Nintgrl

Sim_Time – The maximum simulation time in years. For a transient flow field, Sim_Time is set by the program to the minimum of Sim_Time and the maximum time on the input flow field file.

Tolsv – The relative change in particle position required for saving particle path data. If zero, all data is saved. If 0.1, then data is saved after the particle has moved 0.1 of a cell width.

SS_Step – The flow field step from which to read the steady state flow field. It is not used for transient flow.

NoSS – The number of steps to take between 0.0 and Sim_Time for steady state. NoSS is actually used to calculate the maximum timestep for the steady state solution. It is not used for transient particle flow. Not Recommended – set to 1. Nsubcl is the preferred way to control sub-cycling particle tracking across a single cell.

Nsubcl – The number of steps to subcycle particle tracking across a cell. A value of 5 implies that the travel time to the exit boundary divided by 5 is the approximate time increment for subcyling particle tracking across a cell.

Nintgrl – The number of intervals to use in the numerical integration algorithm used to estimate the travel time to the exit boundary for transient flow. A value of 100 is typically adequate.

`*GRID` – Secondary keyword .

Record 1: Name of MODFLOW Discretization File containing grid description (Harbaugh, 2000, pp. 45–49)

*FLOW – Secondary keyword.

Record 1: Porosity array in one of the four MODFLOW 2000 array formats listed below (Harbaugh 2000, pp86)

(1) CONSTANT *porosity comment*

(2) INTERNAL *scale_factor format print_flag comment*
porosity(i), i=1, n

(3)EXTERNAL *unit scale_factor format print_flag comment*
data file *fort.unit* contains formatted data

(4) OPEN/CLOSE *porosity_file scale_factor format print_flag comment*
data file *porosity_file* contains formatted porosity data

There is a read statement for each row in each layer that reads *ncol* data values. Therefore *nlay*nrow* records are read, where *nrow, ncol, nlay* are number of row, columns and layers in the grid, respectively.

Examples:

(1)CONSTANT 0.2 porosity of rock

(2)INTERNAL 1.0 (FREE) 0 porosity of rock
0.2 0.2 0.2, 17*0.2
20*0.2
...

(3) EXTERNAL 73 1.0 (FREE) 1 porosity of rock
data file *fort.73* contains free formatted porosity data

(4)OPEN/CLOSE *porosity.dat* 1.0 (FREE) 0 porosity of rock
data file *porosity.dat* contains free formatted porosity data

*END - – End of *CONTROL section of input file.

*/ Place any number of comment or blank records here.

*PARTICLE

Record 1: NStart, Ndrop

NStart – The timestep, at which to start dropping particles into the flow field. Nstart determines the time to start tracking particles. Particles are dropped at all locations specified on record 3.

NDrop – The increment in time steps for which to drop particles into the flow field. Zero specifies that particles will not be dropped based on time steps. A value of 3 means that particles will be dropped every third step starting at step NStart. If NDrop NwDrop and TDropI are zero then particles will be release at the time specified on Record 3.

Record 2: Tdrop0, TDropI

TDrop0 – The initial time to drop particles into the flow field based on a time increment.

TDropI – The increment in time for which to drop particles into the flow field. Zero specifies that particles will not be dropped based on time. A value of 100 means that particles will be dropped every 100 years starting at time TDrop0. If NDrop, NwDrop, and TDropI are all zero then particles will be release at the time specified on Record 3.

Record 3: Logical_Flag, Part_Times(ip,1), Part(ip,1,1), Part(ip,2,1), Part(ip,3,1)

Record 3 is repeated for all particle location and start time combinations. Terminate with an *END record.

Logical_Flag – Set to L (no quotes) if initial coordinate positions are to be specified in logical i, j, k index space coordinates. Set to X if coordinate positions are to be specified in x, y, z coordinates. If Logical_Flag is L then the coordinate (3.5, 4.5) is the center of cell with indices i=3 and j= 4; 0.5 is the center of the first cell

Part_Times(ip,1) – Time, in years, to start tracking particle ip. If using the particle drop controls on records 1 and/or 2, Part_Time is ignored, but still must be specified.

Part(ip,1,1) – Initial x-coordinate of particle ip.

Part(ip,2,1) – Initial y-coordinate of particle ip.

Part(ip,3,1) – Initial z-coordinate of particle ip. Specified for 3-D problems only.

*END - End of *PARTICLE section of input file.

*/ Place any number of comment or blank records here.

*BOUNDARY

Record 1; Logical_Flag, Zbot, Ztop

Record 1 is used for 3-dimensional problems only, but must be specified for both 2-D and 3-D.

Logical_Flag – Set to L if the z boundary planes are to be specified in logical i, j, k index space coordinates, including fractional part. Set to Z (or x) if boundary endpoints are to be specified in x, y, z coordinates. If Logical_Flag is L then the coordinate (3.5, 4.5) is the center of cell with indices i=3 and j=4; 0.5 is the center of the first cell.

Zbot – z-coordinate plane value for the bottom of the 3-D prism.

Ztop – z-coordinate plane value for the top of the 3-D prism.

Record 2: Logical_Flag, xb(ib), yb(ib)

Record 2 is repeated for all boundary segment endpoints. Terminate with a *END record. For 3-D problems, the segments represent planes that are perpendicular to the x,y plane. Endpoints must be specified clockwise. The first and last endpoints should be the same for a closed boundary.

Logical_Flag	– Set to L if the boundary endpoints are to be specified in logical i, j, k index space coordinates, including fractional part. Set to X if boundary endpoints are to be specified in x, y, z coordinates. If Logical_Flag is L then the coordinate (3.5, 4.5) is the center of cell with indices i=3 and j=4; 0.5 is the center of the first cell.
xb(ib)	– x-coordinate of boundary segment endpoint ib.
yb(ib)	– y-coordinate of boundary segment endpoint ib

*END – End of boundary input

```
*/=====================================================
*/ File:    dtrkmf_control_t1.inp
*/ Purpose: DTRKMF Test Case #1
*/ Date:    September 2002
*/ By:      David K Rudeen
*/=====================================================
*/
*/ Sim_Time, Tol_Save, SS_Step, No_SS, Nsubcl, Nintgrl
*CONTROL
MODFLOW steady noecho avearge_area no_history_steps analytic nodebug
100000.   0.     1   1    5    100
*GRID
dtrkmf_mf2k.dis
*FLOW
CONSTANT 1.0 Porosity
*END

*PARTICLES
0    0
1000.     0.
x 0.      600.      950.
X 0.      650       950.
L 0.      8.5       17.5
*END

*BOUNDARY
Z   0.      1.
X   100.    100.
X   100.    958.13
L  18.     18.
L  18.      5.
L   8.      1.
L   1.      1.
*END
```

Figure 6-1. Sample Input Control File

6.2 Input Flow Field File

There are certain assumptions made about the structure of the data on the input low field file. Even though it is planned that MODFLOW 2000 be used to supply the input flow field, any file that follows these assumptions can be used. The assumptions are:

- Time is in seconds.
- Fluid flux between adjacent cells is defined for the interface on the higher index side of the cell. In MODFLOW 2000 nomenclature, this would be the right and front interfaces. Units are volume per second.

The binary file format is best described by the FORTRAN code fragment listed below:

```
text='FLOW RIGHT'
rite(66)kstp,kper,text,ncol,nrow,nlay
write(66)((Qx(i,j),i=1,ncol),j=1,nrow)

text='FLOW FRONT'
write(66)kstp,kper,text,ncol,nrow,nlay
write(66)((Qy(i,j),i=1,ncol),j=1,nrow)
```

where, `kstp` is a step number within the time period, `kper`. For steady state flow, `kstp` and `kper` are 1. `ncol`, `nrow`, and `nlay` are the number of columns, rows and layers in the computational grid, respectively. For two-dimensional problems, `nlay` is 1. `Qx` and `Qy` are the volumetric fluid fluxes in the *x* and *y* directions, respectively.

6.3 Output

6.3.1 Output Particle Path Text File

An output particle path text file is created if the corresponding file name is specified on the command line (not NULL). The text file contains the following two records for each particle:

Record 1: ip, num_part_times(ip)

ip	– Particle number.
num_part_times(ip)	– Number of time points on the path.

Record 2: part_times, star,(pl(ic),ic=1,num_coord), (part(ic),ic=1,num_coord), arclength, speed, por_mid, sat_mid

Record 2 is repeated for all time points for the current particle.

Part_Times	– Time, in years
Star	– An asterisk is printed beside each time that comes from a whole step on the input CDB file
Pl(ic)	– Logical *i*-, *j*-, *k*-coordinates of the particle, including fractional part. The *k* coordinate is written only for 3-D problems. A logical *i*-coordinate of 4.75 implies the particle is located ¾ of the way across cell 4 in the *i* or *x* direction. Indices start with 0.
Part(ic)	– Cartesian coordinates of the particle. The *z* coordinate is written only for 3-D problems.
ArcLength	– The total path length or distance traveled by the particle.
Speed	– The fluid velocity at the current location.
Por_Mid	– The time averaged porosity of the cell containing the particle.
Sat_Mid	– The time averaged water saturation of the cell containing the particle.

7.0 ALLOWABLE/PRESCRIBED RANGES FOR INPUT/OUTPUT

Allowable ranges or limits for input and output values are not restrictive. It is expected that the porosity and saturation data lie between 0.0 and 1.0. As stated previously, if either the porosity or the saturation is zero, then the velocity must be zero as well. Otherwise, a warning message is written and the particle is marked inactive.

Coordinate positions for particle dropping and boundary segment endpoints must be specified within the tracking boundaries or an error message is written and the execution is terminated.

The simulation time for a transient flow field is set to the minimum of the user specified value and the maximum simulation time on the input flow field file.

The input and output data are unit specific as discussed above.

8.0 ERROR MESSAGES

This section lists error and warning messages that could occur during execution. There are two categories of errors: (1) those that are internal errors that result from numerical or logic problems that probably require sponsor action; and, (2) user errors which usually are either bad input values or bad file names. Bold text is the error or warning message and italicized text represent values.

******** *nnn* **Memory Errors in DTRKMF ******

The dynamic memory manager has detected memory errors. This error should not occur during normal execution. Check input data for bad parameter value. Call sponsor to debug.

Particle *ip* trapped in cell

A warning that a particle has reached a cell with zero velocity and the particle will be marked as inactive. This occurs only for steady state calculations.

Particle *ip* **trapped on cell boundary**

For a transient calculation, it means that the tracking algorithm has had trouble evaluating velocities on either side of a cell boundary usually due to roundoff problems. Contact sponsor to debug.

CONV_FLOW - corrected snew, *ic,icell(ic),test,sz*

Warning only. The CONV_FLOW subroutine has detected that a particle has crossed a convergent flow divide – a point in a cell where the flow is zero. The particle position has been corrected back to the location of the divide.

ERROR in CONV_FLOW

An error has occurred detecting a convergent flow divide. Contact sponsor to debug.

Particle *ip* will exceed time dimension: *ntdim*
Particle will be marked inactive

Warning only. The number of time points along a particle path has exceeded the time dimension. The time dimension cannot be calculated directly so it is estimated based on the number time steps on the flow field file and the number of subcycling steps per cell. The minimum is 1000. Tracking will stop for this particle and it will be marked inactive. Typically indicates that the algorithm is struggling with the flow field. Contact sponsor to debug or to raise limit.

```
An error occurred for exit s2 face calculation
error, particle, component = ier_fzero,ip,ic,
S1, s2       : s1(ic), s2(ic)
Old location: (sold(iic),iic=1,3)
New location: (snew(iic),iic=1,3)
t0p,t1p,dts : t0p,t1p,dts(ic)
ijk of cell : (icell(iic),iic=1,3)
v10, v20     : v10(ic), v20(ic)
v11, v21     : v11(ic), v21(ic)
```

The algorithm cannot determine the time of exit of a particle from a cell, usually due to numerical problems. This error message can occur for either the s1 or s2 face (left/right, top/bottom). Contact sponsor. The error flag *ier_fzero* is returned from the SLATEC routine DFZERO, which finds roots to equations on the interval (B,C). *Ier_fzero* is defined as follows:

1- B is within the requested tolerance of a zero. The interval (B,C) collapsed to the requested tolerance, the function changes sign in (B,C), and F(X) decreased in magnitude as (B,C) collapsed.

2- F(B) = 0. However, the interval (B,C) may not have collapsed to the requested tolerance.

3- B may be near a singular point of F(X). The interval (B,C) collapsed to the requested tolerance and the function changes sign in (B,C), but F(X) increased in magnitude as (B,C) collapsed, i.e. ABS(F(B out)) .GT. MAX(ABS(F(B in)),ABS(F(C in))).

4- No change in sign of F(X) was found although the interval (B,C) collapsed to the requested tolerance. The user must examine this case and decide whether B is near a local minimum of F(X), or B is near a zero of even multiplicity, or neither of these.

5- Too many (.GT. 500) function evaluations used.

```
New cell of particle ip, is out of data range
The new cell is,(ijkcells(ip,ii),ii=1,num_coord),
Particle will be marked inactive
```

The particle location specified by the user is outside the grid domain. The user should correct the input location and re-run or the particle will be marked inactive.

```
Number of coordinates exceeds dimension: ncdim
```

This indicates an internal error. The dimension of the grid in the subroutine INITIAL is larger than the value scanned in sub routine SCAN_INPUT. Contact sponsor.

Requested nx,ny,nz sizes exceed dimensions: *nvidim,nvjdim,nvkdim*

This indicates an internal error. The dimension of the grid in the subroutine INITIAL is larger than the value scanned in subroutine SCAN_INPUT. Contact sponsor.

Simulation time exceeds maximum simulation time, Set to maximum

Warning only. The simulation time specified by the user exceeds the simulation time on the input flow field file. The simulation time is set to the maximum time on the file.

Number of particles exceed particle dimension: *npdim*

This indicates an internal error. The number of particles read in INITIAL is larger than the value scanned in subroutine SCAN_INPUT. Contact sponsor.

Number of boundary points exceed dimension: *nbdim*

This indicates an internal error. The number of boundary points read in INITIAL is larger than the value scanned in subroutine SCAN_INPUT. Contact sponsor.

WARNING por_sat=0, v0= *v0(ix,iy,iz,ic),*
ix,iy,iz,ic= *ix,iy,iz,ic*

Warning only. The program has detected a cell with the product of porosity and saturation equal to zero, but the fluid velocity is not zero. The velocity is set to zero and the calculation continues.

An error occurred for the exit boundary calculation
For particle number *ip,* **ier_fzero =** *,ier_fzero*
t0p,t1p,delta_t = *t0p,t1p,delta_t*
ijk of cell = *(icell(iic),iic=1,num_coord)*

The algorithm cannot determine the time of exit of a particle across a boundary segment, usually due to numerical problems. Contact sponsor. Error is returned from the SLATEC routine DFZERO, which finds roots to equations on the interval (B,C). *Ier_fzero* is defined as follows:

1- B is within the requested tolerance of a zero. The interval (B,C) collapsed to the requested tolerance, the function changes sign in (B,C), and F(X) decreased in magnitude as (B,C) collapsed.

2- F(B) = 0. However, the interval (B,C) may not have collapsed to the requested tolerance.

3- B may be near a singular point of F(X). The interval (B,C) collapsed to the requested tolerance and the function changes sign in (B,C), but F(X) increased in magnitude as (B,C) collapsed, i.e., ABS(F(B out)) .GT. MAX(ABS(F(B in)),ABS(F(C in)))

4- No change in sign of F(X) was found although the interval (B,C) collapsed to the requested tolerance. The user must examine this case and decide whether B is near a local minimum of F(X), or B is near a zero of even multiplicity, or neither of these.(Generates a WARNING rather than an error.)

5- Too many (>500) function evaluations used.

Error from dcerf: ierr = *ierr*
kode,signa,signboa = *kode,signa,signboa*
z = *z*

Subroutine DCERF has detected an internal error in calculating the erf(z). Contact sponsor.

FILE SPECIFICATION ERROR(S)

Subroutine PREPRO has detected an error with the command line specified file names. One of the user specified files probably does not exist or the user does not have access to one of the directories. The user should closely check spelling , path and version of all specified files.

Initial position of particle *ip* **is out of data range.**
The initial cell is *(ijkcells(ip,ii),ii=1,num_coord)*

The user specified initial position of a particle is out side the range of the input CDB. User should check input data.

IN CERF, KODE NOT 1 OR 2

Subroutine DCERF has detected an error with argument KODE. Contact sponsor.

KODE=1 returns values for erf(z)

KODE=2 Returns values for exp(z**2)*erfc(z), where erfc(z)=1.0-erf(z)

IN CERF, OVERFLOW, CABS(Z2) EXCEEDS** *valmax*

Subroutine DCERF has detected an internal error. Z is too large. There is possibly a problem with the flow field. Contact sponsor.

9.0 REFERENCES

Amos, D. E. and S. L. Daniel, *1977. AMOSLIB, A Special Function Library Version 9/77.* SAND77-1390, Sandia National Laboratories, Albuquerque, NM.

Harbaugh, A. W., R. E. Banta, M. C. Hill, and M. G. McDonald, 2000. MODFLOW-2000, *The U.S. Geological Survey Modular Ground-Water Model – User Guide to Modularization Concepts and the Ground-Water Process.* OFR 00-92, USGS, Reston, VA.

Haskell, K. H., W. H. Vandevender, and E. L. Walton, 1980. *SLATEC Common Mathematical Library – SNLA Implementation.* SAND80-2792, Sandia National Laboratories, Albuquerque, NM.

Pollock, D., 1989. *"Documentation of Computer Programs to Compute and Display Pathlines Using Results from the U. S. Geological Survey Modular Three-Dimensional Finite-Difference Ground-Water Flow Model."* OFR 89-381, USGS.

Shampine, L. F. and H. A. Watts, 1979. *DEPAC – Design of a User Oriented Package of ODE,* SAND79-2374. Sandia National Laboratories, Albuquerque, NM.

U.S. DOE, 1980. *"Final Environmental Impact Statement: Waste Isolation Pilot Plant."* DOE/EIS-0026. Washington, D.C.

U.S. DOE, 1990. *"Final Supplement Impact Statement: Waste Isolation Pilot."* DOE/EIS-0026-FS. Washington, D.C.

U.S. DOE, 1996. *"Title 40 CFR Part 191 Compliance Certification Application for the Waste Isolation Pilot Plant."* DOE/CAO-1996-2184. Carlsbad, NM: U. S. Department of Energy, Waste Isolation Pilot Plant, Carlsbad Area Office.

U.S. EPA. 1985. *"40 CFR Part 191: Environmental Standards for the Management and Disposal of Spent Nuclear Fuel, High-Level and Transuranic Radioactive Wastes: Final Rule."* Federal Register Vol. 50: p. 38066–38089.

U.S. EPA. 1986. *"40 CFR Part 268: Land Disposal Restrictions, as amended and published in the most recent Code of Federal regulations."* Washington, D. C.: Office of the Federal Register, National Archives and Records Administration.

U. S. EPA, 1996. *"40 CFR Part 194: Criteria for the Certification and Re-Certification of the Waste Isolation Pilot Plant's Compliance with 40 CFR Part 191 Disposal Regulations: Final Rule."* Federal Register, Vol. 61, no. 28, 5224–5245.

WIPP PA, 1991. *Preliminary Comparison with 40 CFR Part 191, Subpart B for the Waste Isolation Pilot Plant.* SAND91-0893/1-4. Sandia National Laboratories/NM, Albuquerque, NM.

WIPP PA, 1992. *Preliminary Performance Assessment for the Waste Isolation Pilot Plant.* SAND92-0700/1-5. Sandia National Laboratories/NM, Albuquerque, NM.

WIPP PA, 1996b. *User's Manual for NUTS (Version 2.02).* WPO #37927, Sandia National Laboratories/NM, Albuquerque, NM.

WIPP PA, 2002. *WIPP PA User's Manual for DTRKCDB, Version 1.00.* EMRS #515090 Sandia National Laboratories/NM, Albuquerque, NM.

WIPP PA, 2003a. *WIPP PA Design Document for DTRKMF, Version 1.00.* EMRS #523244 Sandia National Laboratories/NM, Albuquerque, NM.

WIPP PA, 2003b. *WIPP PA Requirements Document for DTRKMF, Version 1.00.* EMRS #523242, Sandia National Laboratories/NM, Albuquerque, NM.

Zheng, C. and G. D. Bennett, 1995. *Applied Contaminant Transport Modeling: Theory and Practice.* Van Nostrand Reinhold, New York.

INTENTIONALLY LEFT BLANK.

APPENDIX A:
REQUIREMENTS

A.1 Functional Requirements

DTRKMF is required to perform the following functions:

R.1 Calculate particle tracks for a steady state, 2-D flow field until: (1) an arbitrary polygonal boundary in 2-D is reached; (2) a particle is trapped in a cell; or, (3) a given simulation time is exceeded.

R.2 Calculate particle tracks for a steady state, 3-D flow field until: (1) an arbitrary *x-y* polygonal edge bounded by a top and bottom *z*-plane in 3-D (prism) is reached; (2) a particle is trapped in a cell; or, (3) a given simulation time is exceeded.

R.3 Calculate particle tracks for a time dependent, variably saturated, 2-D flow field until: (1) an arbitrary polygonal boundary in 2-D is reached; or, (2) a given simulation time is exceeded.

R.4 Calculate particle tracks for a time dependent, variably saturated, 3-D flow field until: (1) an arbitrary *x-y* polygonal edge bounded by a top and bottom *z*-plane in 3-D (prism) is reached; or, (2) a given simulation time is exceeded.

R.5 For steady-state flow fields, read in particle starting locations from an ASCII input file.

R.6 For time dependent flow fields, read in particle release times and starting locations from an ASCII input file.

R.7 For time dependent flow fields, read in particle starting locations from an ASCII input file and read particle release times from an input binary file (see **R.9**) as every n^{th} time step starting from an initial time step. The initial time step and n are inputs.

R.8 For each particle, calculate a time step such that no more than one grid cell is traversed in a given step.

A.2 External Interface Requirements

R.9 DTRKMF will read MODFLOW 2000 input and output files for its coordinate data, porosity and saturation data, velocity/flow field data and, optionally, release times.

R.10 DTRKMF will produce the following files: (1) an ASCII debug file; and, (2) an ASCII particle time and position file.

APPENDIX B:
DESCRIPTIONS OF MAJOR SUBROUTINES FOR DTRKMF

Note that filenames, in general, are subroutine names with a "DTRKMF_" prefix.

- DTRKMF is the FORTRAN77 main program that sets up dynamic memory allocation, reads input from the command line, scans the input files to determine array sizes, dynamically allocates memory, and starts the calculations.

- Subroutine PREPRO is an input preprocessor that performs QA functions, parses the command line input, and opens the files found on the command line.

- Subroutine SCAN_INPUT reads the input files for array sizing parameters.

- Integer Function LNBLNK finds the first trailing blank character.

- Subroutine FINDKW finds records beginning with a keyword.

- Subroutine CONTROL provides the overall control of the calculation. It is responsible for initializing the data arrays, reading the data using subroutines, determining the initial particle locations and particle initial times, reading the boundary data, computing and storing the boundary transformations, moving the particles in a time step, computing the arc length and speed along a trajectory, and outputting the results.

- Subroutine GET_MF_GRID reads the input control file to find the MODFLOW discretization file name, reads the grid point coordinates, computes the face coordinates, calculates the DX, DY, DZ delta grid data, and computes appropriate cross sectional areas.

- Subroutine PARTINIT initializes the particle arrays, determines the initial cell of a particle, and checks that the particle is in the range of the data.

- Subroutine GET_MF_DATA reads fluxes, computes Darcy velocities, and reads porosity and saturation data.

- Subroutine BUILDCOEF builds the component coefficients.

- Subroutine COORDOFTIME computes the coordinate location given the above coefficients and time using either the semi analytic solution or an ODE solver, DDEABM.

- Subroutine DDEABM is the a SLATEC(DEPAC) library routine for solving a system of ODEs.

- Subroutine DCERF is the double precision, complex error function routine from the AMOSLIB library that is used to implement the complex erf() and erfc() functions (Amos and Daniel, 1977).

- Subroutine TIMEOFEXIT_GEN computes the time required to exit a cell in a given coordinate direction or the time required to cross a boundary for the general case.

- Subroutine TIMEOFEXIT_SPC computes the time required to exit a cell in a given coordinate direction or the time required to cross a boundary for special cases.

- Subroutine TIMEOFEXIT_SS computes the time required to exit a cell in a given coordinate direction or the time required to cross a boundary for steady state flow.

- Subroutine TIMEOFEXIT_NI computes the time required to exit a cell in a given coordinate direction or the time required to cross a boundary for transient flow using numerical integration. Good for timestep estimation only. TIMEOFEXIT_GEN is used when more accuracy is required.

- Subroutine DFZERO is the double precision zero finding routine from the SLATEC library, (Haskell et al., 1980). The routine uses a bisection algorithm coupled with the secant method for efficiency. @F represents the external function that DFZERO calls. The routines to the right of DFZERO in the chart are from the SLATEC library. D1MACH provides double precision machine specific information regarding data representation ranges, special constants, etc. I1MACH provides the integer information. The other routines deal with error handling and are not important for this discussion.

- Double precision function EXITTIME is the function of time, F(T), that provides the equation to be solved. The equation is the equation of a plane whose coordinates are provided by calls to the COOROFTIME routine. The precise equation solved is determined by the coefficients of the passed to EXITTIME.

- Integer Function MINLOC determines the index of the minimum component of a vector.

- Subroutine VELOCITY_SS sets up flow arrays (velocity, porosity and saturation) for steady state flow. Controls the reading the data from the MODFLOW budget file using GET_MF_DATA.

- Subroutine VELOCITY_TR sets up flow arrays (velocity, porosity and saturation) for transient flow. Controls the reading the data from the MODFLOW budget file using GET_MF_DATA.

- INCLUDE files handle the typing of variables and the declaration of COMMON blocks.