User's Guide for mod-PATH3DU

A groundwater path and travel-time simulator

S.S. Papadopulos & Associates, Inc.

UNIVERSITY OF WATERLOO

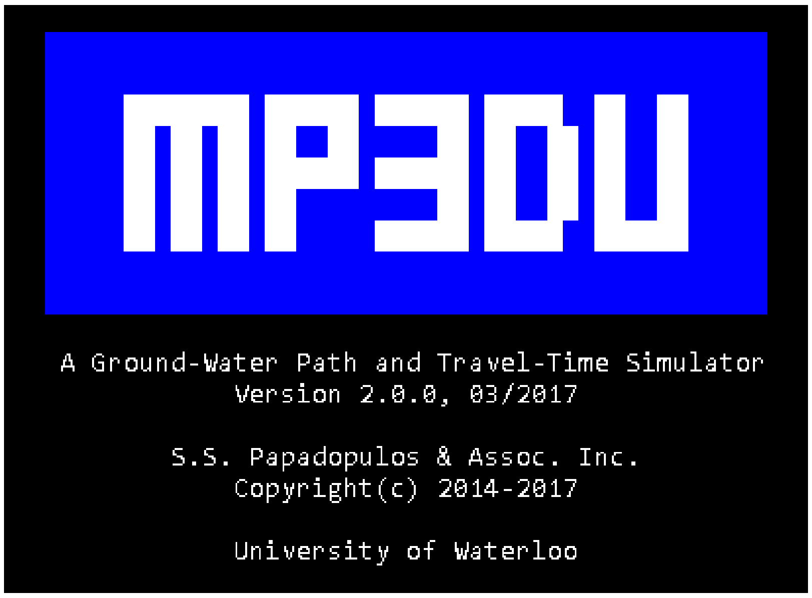
Christopher Muffels
Xiaomin Wang

Matthew Tonkin
Christopher Neville

Muhammad Ramadhan
James R. Craig

Bethesda, Maryland
Waterloo, Ontario

2017



MP3DU

A Ground-Water Path and Travel-Time Simulator
Version 2.0.0, 03/2017

S.S. Papadopulos & Assoc. Inc.
Copyright(c) 2014-2017

University of Waterloo

Support

S.S. Papadopulos & Associates Inc. in collaboration with the University of Waterloo developed mod-PATH3DU to support specific project applications, but is distributing it free-of-charge as a service to our industry. As such, only limited technical support can be provided by S.S. Papadopulos & Associates Inc. through the corresponding author.

Corresponding author:
Christopher Muffels
7944 Wisconsin Ave.
Bethesda, MD 20895
301-718-8900
cmuffels@sspa.com
www.sspa.com

Acknowledgements

The authors would like to thank Mike Kulbersh and the New England District Office of the U.S. Army Corps of Engineers for their support (Contract No. W912WJ-16-P-0040) in developing this latest version of mod-PATH3DU. A little goes a long way – and it is greatly appreciated.

The authors wish to acknowledge Graham Stonebridge, who, as an intern at SSP&A, helped to develop the proof-of-concept for the SSP&A tracking method and Karl Mihm, also a former intern, for his help in writing the first version of this software. Also, Robert Gordon, an intern, for his help testing the Waterloo method and for implementing and testing different location search algorithms.

We wish to thank Jim Rumbaugh, the developer of Groundwater Vistas, for his early adoption of mod-PATH3DU. Also, Alan Lemon and Michael Kennard with Aquaveo, developers of GMS for their early adoption and testing. And Wayne Hesch, formerly Senior Project Manager for Visual MODFLOW Flex, for his early testing. We thank Sorab Panday for his support of this development and his tireless pursuit of MODFLOW-USG. We also acknowledge all of our colleagues at S.S. Papadopulos & Assoc. Inc., especially Chunmiao Zheng, Charlie Andrews, Steve Larson, Jim Cousins and Doug Hayes. This software is a company wide effort drawing upon years, decades even, of research and testing in various forms and applications in support of our clients.

Quick Start Guide

1. Read this manual in its entirety.
2. Reread the IFACE section.
3. Download mod-PATH3DU and its support programs from www.sspa.com/software
4. Use writeP3DGSF.exe to create the necessary GSF file. See Appendix B for details.
5. Create the required mod-PATH3DU input files following the instructions in the Input Instructions section:
    a. Primary file – copy example from Input Instructions and modify as needed following the options listed in Appendix A
    b. Prepare the Per-cell property file as needed.
    c. Prepare starting locations point shapefile. Ensure it has attributes for starting location cell id, release time, and local-z elevation.
6. At a command prompt enter the name of the mod-PATH3DU executable, followed by the name of the primary input file. For example : `mp3du.exe NAME.json colorcode`
7. Review log and listing files to ensure the program executed as expected.
8. Convert the output binary file into a table or shapefile using writeP3DOutput.exe detailed in Appendix C.

Contents

Figures

Tables

Overview

mod-PATH3DU is a particle tracking code for calculating the three-dimensional flow pathlines and travel time of solute particles. It supports both MODFLOW-USG (Panday et al., 2013) and MEUK (Tonkin et al., 2016). MODFLOW-USG is an unstructured-grid version of MODFLOW, particles are tracked within the flow solution calculated by MODFLOW models using the Waterloo method (Ramadhan, 2015). The Waterloo method is a semi-analytical approach, akin to the Pollock method, that is based on an analytical Taylor series reconstruction of the intra-cell velocity. It is applicable to cells with arbitrary geometry, including quadpatch, quadtree, nested, and Voronoi grids. MEUK (multi-event universal kriging) is a method of interpolation rooted in universal kriging with hydrogeology-specific drift terms to map hydraulic-head. Particles are tracked on MEUK-calculated head surfaces using the SSP&A method (Tonkin and Larson, 2002). The SSP&A method evaluates the velocity components (*x* and *y*-directions) of a particle using Darcy's law, where the gradient in the vicinity of the particle is determined from the MEUK interpolated heads. mod-PATH3DU uses a higher order numerical Runge-Kutta scheme (Zheng, 1992) to move a particle given the velocities calculated by the Waterloo or SSP&A methods.

This document describes the particle tracking program mod-PATH3DU and provides instructions for using the program. Some of the required input files are created outside of MODFLOW-USG and it is important to read the instructions to make sure that all of the required input is specified. mod-PATH3DU does not include either a graphical user interface nor tools for visualizing the results. Users will need to devote care to preparing the input and analyzing the output. All of the files for the test cases detailed in the Examples section are available for download. These files illustrate the structure of the input and the procedures required to execute mod-PATH3DU.

Compatibility

MODFLOW

mod-PATH3DU uses information in both the cell-by-cell flow (CBB)[1] and head-save (HDS) files written by MODFLOW to calculate velocity. These files list the components of flow and hydraulic head for each cell. In structured grid mode, MODFLOW-USG writes these two files in a format identical to previous MODFLOW releases, as a result, mod-PATH3DU is compatible with previous versions of MODFLOW, specifically MODFLOW-2000 and MODFLOW-2005 (mod-PATH3DU has not been tested on earlier versions).

MODFLOW-USG is built upon the MODFLOW-2005 framework and uses modified versions of its predecessor's subroutines to read input files. These modified routines are incorporated into mod-PATH3DU. As such, mod-PATH3DU does not support the global (GLO) package available with MODFLOW-2000. Further, the connected-linear network (CLN) package available with MODFLOW-USG supersedes the multi-node well (MNW) package in previous releases of MODFLOW. Because mod-PATH3DU relies on the CBB file for boundary and source/sink flow information, and not the input packages for these routines explicitly, it supports both the MNW and CLN packages. Currently, it does not track particles within a CLN network.

MODFLOW-USG does not require information about cell shapes or how cells are positioned in space (Panday et al. 2013). Spatial information about cells, however, is critical to a particle tracking model. Fortunately, a grid specification file (GSF) was decided upon that provides x, y and z coordinates for the vertices of a cell. mod-PATH3DU requires a modified version of this file for both structured and unstructured grid models. A pre-processing program, writeP3DGSF.exe, is included with mod-PATH3DU and is detailed in Appendix A. This program converts a GSF file in the original format into the mod-PATH3DU specific format. Additionally, it will write the required GSF file for a structured MODFLOW model defined by DELR and DELC. The format for the modified GSF file is provided in the Input Instructions section. This modified GSF format makes it easier for mod-PATH3DU to identify the shared face between two cells that are connected and is included as a pre-processing step to improve runtime as this identification need only be done once for a model grid.

[1] mod-PATH3DU requires the CBB file be written using the COMPACT BUDGET option in the MODFLOW output control (OC) file.

MEUK

The hydraulic-head maps written by MEUK are in the ArcMap/ArcInfo ASCII grid format[2] (lower-left corner designation for the origin) – mod-PATH3DU uses these maps, as well as information about the grid, hydraulic conductivity, porosity, and events and drift terms used in MEUK, to track particles. It supports the multi-event feature of MEUK by tracking on each event’s head surface for a specified duration. This approach is like how transient MODFLOW models are treated as a series of steady-state flow periods. Particles are stopped or considered captured if they travel within a user-specified distance of an extraction or injection well when forward or backward tracking, respectively, and when they cross a line-sink. The MEUK grid must be written to a GSF file prior to executing mod-PATH3DU; the pre-processing program writeP3DGSF.exe, detailed in Appendix A, can be used to do this. Hydraulic conductivity and porosity can be specified as constant across the entire MEUK domain, or be input from a file with non-uniform values. MEUK relies on a series of keywords to indicate the event and group a drift or observation set belong to, as well as for coordinates and other information required to calculate its maps; mod-PATH3DU uses the same keywords. Relating the MEUK to mod-PATH3DU keywords is described in Appendix A.

[2] http://resources.esri.com/help/9.3/arcgisengine/java/GP_ToolRef/spatial_analyst_tools/esri_ascii_raster_format.htm

Method

Advection

Zheng and Bennett (2002) provide an excellent discussion on advective transport, including particle tracking solution methods. We summarize, in part, this chapter here. It is recommended that users interested in additional detail or information review the relevant chapter in this book.

The pathlines of purely advective solute particles are governed by the following equation:

$$\frac{d\boldsymbol{X}_p}{dt} = \boldsymbol{V}\left(X_p, t\right) \tag{MA-1}$$

where $\boldsymbol{X}_p$ is the position of the particle, and $\boldsymbol{V}\left(X_p, t\right)$ is the seepage velocity of the particle at position $\boldsymbol{X}_p$ and time t. The solution to Equation MA-1 for particle location at time t is:

$$\boldsymbol{X}_p(t) = \boldsymbol{X}_p(t_0) + \int_{t_0}^{t} \boldsymbol{V}\left(X_p, t\right) dt \tag{MA-2}$$

where $\boldsymbol{X}_p(t_0)$ is the position at time t_0. Equation MA-2 can be integrated directly, but is typically solved numerically given the complexity of velocity fields in most groundwater systems. The Taylor series expansion of Equation MA-2 is given by:

$$\boldsymbol{X}_p(t + \Delta t) = \boldsymbol{X}_p(t) + \frac{d\boldsymbol{X}_p}{dt}\Delta t + \frac{d^2\boldsymbol{X}_p}{dt^2}\Delta t^2 + \cdots \tag{MA-3}$$

Thus, a simple first-order solution to Equation MA-2 is:

$$\boldsymbol{X}_p(t + \Delta t) = \boldsymbol{X}_p(t) + \boldsymbol{V}\left(X_p, t\right)\Delta t \tag{MA-4}$$

Equation MA-4 is referred to as Euler's method. It relies only on the velocity at the starting point of each tracking step; as such it requires sufficiently small tracking steps to be accurate. Accuracy can be increased by using higher order schemes, such as Runge-Kutta methods. The fourth-order Runge-Kutta method is commonly used and implemented in mod-PATH3DU. This Runge-Kutta method solves the particle tracking equation by advancing a particle over a time interval Δt by taking several Euler-like trial steps and combining the velocity information from these steps to match a fourth-order Taylor series expansion.

Runge-Kutta Numerical Scheme

The fourth-order Runge-Kutta (RK4) algorithm in mod-PATH3DU is the same one implemented in PATH3D (Zheng, 1989; Zheng, 1992).

The velocity is evaluated four times for each particle tracking step: once at the initial point (p_1), twice at trial midpoints of the step (p_2 and p_3) and once at a trial endpoint (p_4), as shown in Figure 1. Based on the velocities evaluated at the four points, the position of the particle at the beginning of the next step ($x_{n+1}, y_{n+1}, z_{n+1}$) is calculated as:

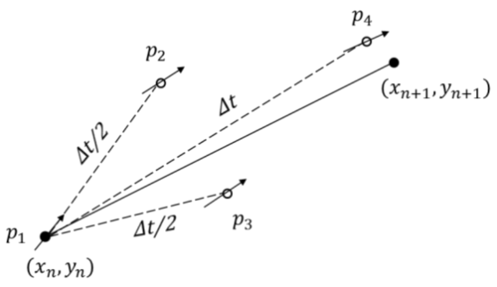


Figure 1 - Illustration of the fourth-order Runge-Kutta method

$$x_{n+1} = x_n + \frac{(k_1 + 2k_2 + 2k_3 + k_4)}{6} \quad \text{(RK-1a)}$$

$$y_{n+1} = y_n + \frac{(l_1 + 2l_2 + 2l_3 + l_4)}{6} \quad \text{(RK-1b)}$$

$$z_{n+1} = z_n + \frac{(m_1 + 2m_2 + 2m_3 + m_4)}{6} \quad \text{(RK-1c)}$$

where

$$\begin{aligned} k_1 &= \Delta t v_x(x_n, y_n, z_n, t_n) \\ k_2 &= \Delta t v_x(x_n + \frac{k_1}{2}, y_n + \frac{l_1}{2}, z_n + \frac{m_1}{2}, t_n + \frac{\Delta t}{2}) \\ k_3 &= \Delta t v_x(x_n + \frac{k_2}{2}, y_n + \frac{l_2}{2}, z_n + \frac{m_2}{2}, t_n + \frac{\Delta t}{2}) \\ k_4 &= \Delta t v_x(x_n + k_3, y_n + l_3, z_n + m_3, t_n + \Delta t) \end{aligned} \quad \text{(RK-2)}$$

and

$$\begin{aligned} l_1 &= \Delta t v_y(x_n, y_n, z_n, t_n) \\ l_2 &= \Delta t v_y(x_n + \frac{k_1}{2}, y_n + \frac{l_1}{2}, z_n + \frac{m_1}{2}, t_n + \frac{\Delta t}{2}) \\ l_3 &= \Delta t v_y(x_n + \frac{k_2}{2}, y_n + \frac{l_2}{2}, z_n + \frac{m_2}{2}, t_n + \frac{\Delta t}{2}) \\ l_4 &= \Delta t v_y(x_n + k_3, y_n + l_3, z_n + m_3, t_n + \Delta t) \end{aligned} \quad \text{(RK-3)}$$

and

$$m_1 = \Delta t v_z(x_n, y_n, z_n, t_n) \quad \text{(RK-4)}$$

$$m_2 = \Delta t v_z(x_n + \frac{k_1}{2},\ y_n + \frac{l_1}{2},\ z_n + \frac{m_1}{2},\ t_n + \frac{\Delta t}{2})$$
$$m_3 = \Delta t v_z(x_n + \frac{k_2}{2},\ y_n + \frac{l_2}{2},\ z_n + \frac{m_2}{2},\ t_n + \frac{\Delta t}{2})$$
$$m_4 = \Delta t v_z(x_n + k_3,\ y_n + l_3,\ z_n + m_3,\ t_n + \Delta t)$$

The velocity components at points p_1, p_2, p_3 and p_4 are calculated using either the Waterloo or SSP&A methods. The series of calculations is repeated to move a particle step-by-step until the particle reaches a discharge point or a termination criterion is met.

Compared with the Pollock (1994) semi-analytical method, the fourth-order Runge-Kutta method is generally more computationally intensive and may introduce numerical truncation errors. However, the Runge-Kutta method is more general, in that it is not limited to linear interpolation, but applicable to any velocity calculation scheme.

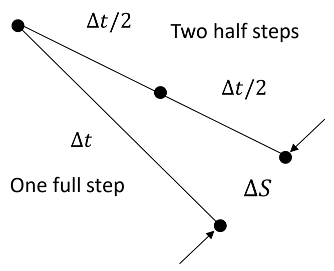


Figure 2 - Illustration of the adaptive step size control procedure

Like the Euler method, the accuracy of the Runge-Kutta method depends on the tracking step size Δt. If Δt is too large the calculation of the velocity components may be inaccurate and the particle tracking pathline may divert from the actual flow path. If Δt is too small significant computational effort may be required to move a particle over a given distance. Thus, it is important to determine an appropriate step size Δt for the particle tracking process. The adaptive step size control procedure used in the PATH3D code (Zheng, 1989; Zheng, 1992), "step doubling", is also implemented in mod-PATH3DU. The tracking step Δt is taken twice: once as a full step and once as two half steps, illustrated in Figure 2. If Δt is sufficiently brief for accurate tracking, the difference between the particle endpoint locations calculated with a full step and by taking two half steps, denoted as ΔS, will be small. Because the Runge-Kutta is a fourth-order accurate method, ΔS can be scaled as $(\Delta t)^5$:

$$(\frac{\Delta t_0}{\Delta t})^5 = \frac{\Delta S_0}{S} \tag{RK-5}$$

where ΔS_0 and ΔS are the differences in particle locations with respect to time steps Δt_0 and Δt. To estimate the step size Δt_0, a "safety factor", f_s, is assigned in Equation (RK-5) and rearranging the equation to yield,

$$\Delta t_0 = f_s \Delta t(\frac{\Delta S_0}{S})^{0.2} \tag{RK-6}$$

The safety factor has a value slightly smaller than unity (e.g., 0.9). Equation (RK-6) is implemented in every tracking step to estimate the required step size adjustment. If the value of ΔS calculated with respect to the step size Δt is larger than the required accuracy specified by ΔS_0, the step size is then reduced to Δt_0 and the tracking calculation is repeated for that step. If ΔS is smaller than ΔS_0, the tracking calculation based on the step size Δt is acceptable, and the initial step size for the next step is taken as Δt_0. The difference in particle locations ΔS is treated as an indicator for the step size adjustment and is a vector in *x*-, *y*- and *z*- directions which can be expressed in terms of a single error criterion, ε, as,

$$\begin{aligned}\Delta X_0 &= \varepsilon \times XMAX \\ \Delta Y_0 &= \varepsilon \times YMAX \\ \Delta Z_0 &= \varepsilon \times ZMAX\end{aligned} \qquad \text{(RK-7)}$$

where $XMAX$, $YMAX$ and $ZMAX$ are scaling factors in the three directions and are the maximum lengths of the flow domain in *x*-, *y*- and *z*- directions. Given an error criterion, ε, as specified by the user, the maximum allowed error in all directions can be calculated according to Equation (RK-7).

Advection and Dispersion

In one dimension, the advection-dispersion equation is (e.g. Zheng and Bennett, 2002):

$$\frac{\partial C}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial C}{\partial x}\right) - \frac{\partial}{\partial x}(vC) \qquad \text{(AD-1)}$$

where C is concentration, D is dispersion, and v is velocity. The analytical solution to equation AD-1 following an instantaneous injection of mass at x=0 and assuming uniform flow is:

$$C(x,t) = \frac{C_0}{\sqrt{4\pi Dt}} e^{-\frac{(x-vt)^2}{4Dt}} \qquad \text{(AD-2)}$$

As Prickett et al. (1981) and others note the spreading of solutes by dispersion can be described by a Gaussian or normal distribution, whose mean, μ, is the advective movement of a particle and whose standard deviation, σ, is the dispersive movement; the density function of which is:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad \text{(AD-3)}$$

Equating equations AD-2 and AD-3 is the foundation of the "Random Walk" solute transport model described by Prickett et al. (1981). In most natural systems groundwater flow is not uniform, as such, the Pricket et al. (1981) model is only approximate.

Kinzelbach (1990) and others demonstrated that for non-uniform flow fields, specifically when dispersion is not constant, the appropriate density function that satisfies equation AD-1, for many particles, is akin to the Fokker-Planck equation, which in one-dimensional form is expressed as (e.g. Zheng and Bennett, 2002):

$$\frac{\partial f}{\partial t} = \frac{\partial}{\eth x}\left(D\frac{\eth f}{\eth x}\right) - \frac{\partial}{\partial x}\left[\left(v - \frac{\partial D}{\partial x}\right) f\right] \qquad \text{(AD-4)}$$

mod-PATH3DU uses many particles to solve equation AD-1 following the density function of equation AD-4. Generally, solute particles are advected at the average groundwater velocity and dispersed randomly, per (e.g. Salamon, 2006):

$$\boldsymbol{X}_p(t+\Delta t) = \boldsymbol{X}_p(t) + \boldsymbol{A}(X_p, t)\Delta t + \boldsymbol{B}(\boldsymbol{X}_p, t)\boldsymbol{Z}\sqrt{\Delta t} \qquad \text{(AD-5)}$$

where $\boldsymbol{BZ}$ is the (random) displacement due to dispersion, $\boldsymbol{B}$ is the displacement matrix and is related to dispersion $\boldsymbol{D}$ by:

$$2\boldsymbol{D} = \boldsymbol{BB}^T \qquad \text{(AD-6)}$$

where $\boldsymbol{D}$, in three-dimensions, is defined following Burnett and Frind (1987):

$$\boldsymbol{D} = \begin{bmatrix} \frac{1}{|v|}\left(\alpha_L v_x^2 + \alpha_T^H v_y^2 + \alpha_T^V v_z^2\right) & (\alpha_L - \alpha_T^H)\frac{v_x v_y}{|v|} & (\alpha_L - \alpha_T^V)\frac{v_x v_z}{|v|} \\ (\alpha_L - \alpha_T^H)\frac{v_x v_y}{|v|} & \frac{1}{|v|}\left(\alpha_T^H v_x^2 + \alpha_L v_y^2 + \alpha_T^V v_z^2\right) & (\alpha_L - \alpha_T^V)\frac{v_y v_z}{|v|} \\ (\alpha_L - \alpha_T^V)\frac{v_x v_z}{|v|} & (\alpha_L - \alpha_T^V)\frac{v_y v_z}{|v|} & \frac{1}{|v|}\left(\alpha_T^V(v_x^2 + v_y^2) + \alpha_L v_z^2\right) \end{bmatrix} \qquad \text{(AD-7)}$$

and

$$B = \begin{bmatrix} \frac{v_x}{|v|}\sqrt{2(\alpha_L|v|)} & -\frac{v_x v_z\sqrt{2(\alpha_T^V|v|)}}{|v|\sqrt{v_x^2+v_y^2}} & -\frac{v_y}{\sqrt{v_x^2+v_y^2}}\sqrt{2\left(\frac{\alpha_T^H(v_x^2+v_y^2)+\alpha_T^V v_z^2}{|v|}\right)} \\ \frac{v_y}{|v|}\sqrt{2(\alpha_L|v|)} & -\frac{v_y v_z\sqrt{2(\alpha_T^V|v|)}}{|v|\sqrt{v_x^2+v_y^2}} & \frac{v_x}{\sqrt{v_x^2+v_y^2}}\sqrt{2\left(\frac{\alpha_T^H(v_x^2+v_y^2)+\alpha_T^V v_z^2}{|v|}\right)} \\ \frac{v_z}{|v|}\sqrt{2(\alpha_L|v|)} & \sqrt{2\left(\frac{v_x^2+v_y^2}{|v|^2}\right)(\alpha_V^T|v|)} & 0 \end{bmatrix} \tag{AD-8}$$

and

$$\boldsymbol{A}(X_p,t) = \boldsymbol{V}(X_p,t) + \nabla \cdot \boldsymbol{D}(X_p,t) \tag{AD-9}$$

$\boldsymbol{A}(X_p,t)$ is termed the "drift" vector, it is a function of the seepage velocity $\boldsymbol{V}(X_p,t)$ and the gradient of the dispersion coefficient. mod-PATH3DU uses the solution to AD-5 given by Labolle et al. (2000). Their generalized stochastic differential equation (GSDE) solution assumes constant porosity and isotropic dispersion and is given by the equations:

$$\boldsymbol{X}_p(t+\Delta t) = \boldsymbol{X}_p(t) + \boldsymbol{V}(X_p,t)\Delta t + \boldsymbol{B}(\boldsymbol{X}_p + \Delta\boldsymbol{Y}, t)\boldsymbol{Z}\sqrt{\Delta t} \tag{AD-10a}$$

$$\Delta\boldsymbol{Y} = \boldsymbol{B}(\boldsymbol{X}_p, t)\boldsymbol{Z}\sqrt{\Delta t} \tag{AD-10b}$$

Implementation of this approach is a two-step process. First, equation AD-10b is evaluated to determine the particle displacement due to dispersion, and second, the velocity at this point is used to determine the dispersion displacement in AD-10a.

Intra-cell Velocity Algorithms

The Waterloo Method

The Waterloo method is detailed by (Ramadhan, 2015). It is based upon reconstructing the intra-cell velocity field using local analytical solutions to the Laplace and Poisson equations for each cell. Solutions to Laplace's equation can be formulated in terms of a complex potential function, Ω (e.g. Haitjema, 2005):

$$\Omega(z) = \Phi(z) + i\Psi(z) \tag{WM-1}$$

The real and imaginary parts of Ω are the discharge potential Φ, and the stream function Ψ, respectively, and *z* is the complex coordinate, where $z = x + iy$. The Waterloo method makes use of complex discharge potential W, defined as (e.g. Haitjema, 1995; Ramadhan, 2015):

$$W = -\frac{d\Omega}{dz} = -\frac{\partial\Phi}{\partial x} - i\frac{\partial\Psi}{\partial x} = Q_x - iQ_y \tag{WM-2}$$

where Q_x and Q_y are the volumetric discharge (L^3/t) in the *x* and *y*-directions, respectively. A special property of complex coordinates is that any function of *z*, $f(z)$, has real and imaginary parts that automatically satisfy the Laplace equation. The Waterloo method, for example, uses two such functions, the complex Taylor series solution of a circular inhomogeneity (Jankovic and Barnes, 1999b) to account for flow through the sides of a cell, and the complex logarithmic function to represent flow to a point sink or source. These functions are respectively:

$$\mathrm{f}(z) = \sum_{n=0}^{\infty} a_n z^n \tag{WM-3a}$$

and

$$\mathrm{f}(z) = \frac{Q}{2\pi}\ln(z) \tag{WM-3b}$$

where α_n are unknown complex coefficients. These functions form the basis by which the Waterloo method calculates a local exact solution to the groundwater flow equation for any cell. The form of the solution in terms of a complex potential is (Ramadhan, 2015):

$$\Omega(Z) = \Phi(Z) + i\Psi(Z) = \sum_{n=0}^{N} a_n Z^n - \frac{q_v}{2} Re(RZ)^2 + \frac{Q_w}{2\pi} log(R|Z - Z_w|) \qquad \text{(WM-4)}$$

where $Z = \frac{z-z_c}{R}$ is the local complex coordinate of the particle, *z*, relative to the complex location of the cell center z_c, *R* is the maximum radius of a circle centered at the cell center that fully encompasses the cell, *N* is termed the order of approximation (Jankovic and Barnes, 1999a), q_v is the vertical flow to the cell from recharge, creeks or streams, cells above or below, etc., Q_w is the pumping rate of a well in the cell located at $Z_w = \frac{z_w-z_c}{R}$. The only unknowns in equation WM-4 are the *N* a parameters. These parameters are solved for using the overspecification principal discussed by Jankovic and Barnes (1999a and 1999b).

The SSP&A Method

The SSP&A method was originally developed by Tonkin and Larson (2002) to track particles and estimate hydraulic capture on a mapping of ground water level data under the influence of pumping. Later it was used by Karanovic et al. (2009) to calculate capture frequency maps. It is a grid independent approach and therefore, sufficiently flexible to be used with either structured or unstructured grids. The SSP&A method evaluates the velocity components of a particle in the *x* and *y*-directions using the interpolated head based on the distribution solved for by MEUK, for example. The head interpolation is implemented at points in the immediate vicinity of a particle using universal kriging, from which the velocity components in the *x*- and *y*- directions are calculated according to Darcy's law (Figure 3).

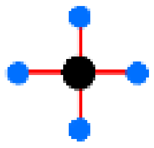


Figure 3 - Schematic of SSP&A method velocity calculation

The SSP&A method uses bilinear interpolation to determine the gradients, except near a pumping well, where kriging is used. Kriging is often used to interpolate irregularly spaced measurement data to unsampled locations (typically a grid of points suitable for contouring). The values at unknown locations are then estimated by minimizing the error variance of predicted values based on their spatial distribution. In the context of mod-PATH3DU, the heads calculated by MEUK that are used to determine velocity constitute the "measured" data, while the position of a particle is the “unsampled” location. Kriging is an exact interpolator in the absence of measurement error or co-located data. Two popular forms of kriging are simple and ordinary. In simple kriging, the mean of the data is assumed to be constant everywhere and its value known *a-priori*. With ordinary kriging, the mean is assumed to be unknown, but is estimable using some function of the measured data. In addition, ordinary kriging can support a spatially varying mean that is not only a function of the data, but includes some trend or “drift”. This form of ordinary kriging is called universal kriging. Universal kriging is a means of incorporating the shape associated with different hydrologic features into the estimate at an unsampled location. For example, near a pumping well, a point sink/source of known strength, derived from the Thiem equation, can be used to incorporate the logarithmic drawdown shape (Tonkin and Larson, 2002). The mathematical details on kriging have been well documented by Journel and Huijbregts (1992), Cressie (1993) and Deutsch and Journel (1998). The kriging algorithm used in the SSP&A method is adapted from the public domain software Geostatistical Software Library (GSLIB) (Deutsch and Journel, 1998) and Skrivan and Karlinger (1980).

Additional Considerations

Boundary Package Flow Accounting and IFACE

Like the Pollock method, the Waterloo method computes the velocity vector using the flow components of the cell faces calculated by MODFLOW. The Waterloo method is predicated on a net flow of zero for each cell, thus it is important that the flux across each cell face include boundary condition flows when appropriate. The auxiliary variable IFACE (Pollock, 2012) is the mechanism by which boundary flows are assigned to a particular cell face and included in the correct term in equation WM-4. Assigning ALL boundary package flows to the correct term on the right-hand side (RHS) of equation WM-4 via IFACE is very important. For example, consider recharge, its flux is vertical and enters a cell through the top face. If the recharge flux is not assigned to the top face, the Waterloo method will not account for it in the vertical flow term of equation WM-4 and erroneous particle paths will be calculated or the program will terminate in error because the unknown coefficients in WM-4 cannot be solved for. Additionally, the velocity in the *z*-direction calculated by mod-PATH3DU would be incorrect for the upper layer and a particle could only be reliably tracked within layers below.

In MODPATH, IFACE ranges from 0 to 6, where 1 through 4 represent the different side faces of a cell, 5 and 6, denote the bottom and top faces, respectively, and 0 is used to specify an internal sink/source (i.e. flow is not assigned to a face). mod-PATH3DU supports a limited number of IFACE values – boundary package flows cannot be assigned to side faces. A list of supported IFACE values is given in the table below. The default IFACE for packages not assigned is 6.

Table 1 Supported IFACE values and their meaning

IFACE	Meaning	Example
0	Internal sink/source; flow for which is included in the Q_w term in equation WM-4.	Extraction or injection well modeled using the WEL, MNW or CLN package
2	Implicit side-face; flow for which is not explicitly assigned, rather mod-PATH3DU distributes the flow to side faces for which flow is not specified for inclusion in the first term on the RHS of equation WM-4.	Constant head specified along boundary of 2D model domain. The corresponding flow represents lateral flow into the domain from a side face.
5	Same as MODPATH. Flow through the bottom face of a cell; flow for which is included in the q_v term in equation WM-4 and in the calculation of velocity in the *z*-direction.	
6	Same as MODPATH. Flow through the top face of a cell; flow for which is included in the q_v term in equation WM-4 and in the calculation of velocity in the *z*-direction.	Recharge, or a drain or river cell whose flow leaves or enters through the top face of a model cell.

An excellent discussion of the impact and importance of IFACE on MODPATH results is given by Abrams et al. (2013). Generally, incorrect assignment of IFACE for a boundary condition will manifest as particle tracks that do not make sense in that they abruptly change direction or oscillate between two cells. If a particle is oscillating between two cells mod-PATH3DU will take much longer to run and result in large output file size because a particle is effectively stagnated and unable to terminate. Also, if AUX IFACE is entered as an option in a boundary package file ensure it is supplied correctly; if AUX IFACE is specified but that variable is not present in the cell listing section, mod-PATH3DU may not perform as expected. The following are suggestions for assigning IFACE for different boundary conditions.

1. General-head : { "HEAD DEP BOUNDS" : 2 }; In 2D, this will distribute GHB flow to any side faces that bound implicit or explicit no-flow cells.
2. Constant-head : { "CONSTANT HEAD" : 2 }; In 2D, if the constant-head cell is connected to an implicit or explicit no-flow cell, IFACE=2 will distribute the flow for this CHD to the sides that are shared, or
3. Constant-head : { "CONSTANT HEAD" : 6 }; If the constant-head cell does not share a face with a no-flow cell, IFACE=6 will assign this flow to the top face of the cell so it is included in the vertical velocity calculation.
4. Pumping well : { "MNW2" : 0 }, or { "WELLS" : 0 }, or { " GWF TO CLN " : 0 }; IFACE=0 invokes the local analytic well correction term and provide implicit weak sink support.
5. Drain : { "DRAINS" : 6 }; this assigns the flow to the drain cell to the top face of the cell so that it is included in the vertical velocity calculation.
6. River : { "RIVER LEAKAGE" : 6 }; this assigns the flow to the drain cell to the top face of the cell so that it is included in the vertical velocity calculation.
7. Recharge : { "RECHARGE" : 6 }; this assigns the flow to the drain cell to the top face of the cell so that it is included in the vertical velocity calculation.

Weak Sinks and Sources

A limitation of the Pollock method is the treatment of weak sinks and sources. For a weak sink, some of the water flowing to the cell containing it discharges to the sink and some passes through the cell (Pollock, 1994). The Waterloo method does not differentiate between weak or strong sinks – sinks are explicitly represented by the well analytic element term in equation WM-4 (third term on the RHS), and accurately tracked to or around.

Backward Tracking and Time Concepts

mod-PATH3DU tracks in the simulation time defined by the flow model. That is, all times input to and output by mod-PATH3DU correspond to the simulation time defined in the flow model. For example, if a MEUK or MODFLOW model has a single stress period 1000 days long and a particle's starting time is specified as day 500, mod-PATH3DU will track forwards from day 500 to day 1000, and track backwards from day 500 to day 0. It is noted here this is different than MODPATH – which tracks according to a “tracking time” which is 0 at a specified “reference time” relative in the MODFLOW “simulation time”. Technically, mod-PATH3DU does something similar, but this accounting is handled internally and hidden from the user.

To track backwards, mod-PATH3DU uses negative tracking time steps. The user supplies the initial tracking step and mod-PATH3DU multiplies this number by -1 when tracking backwards. Tracking time limits (when moving from one time step to another in the simulation model) is handled internally relative to simulation time so the forward and backward algorithms are identical. This, again, differs from MODPATH which reverses the sign of all the velocity components.

Distorted Vertical Discretization and the Water Table

The way distorted vertical discretization is handled in mod-PATH3DU is like MODPATH. In each cell, elevation, z, is transformed to a local coordinate system, z_L, that ranges from 0 at the bottom of a cell to 1 at the top or water table elevation. By scaling the velocity in the vertical direction by the saturated thickness of the cell, a particle can be tracked in this transformed space if z_L is updated when a particle changes layers. The “Non-Rectangular Vertical Discretization” and “Water Table Layers” sections of the MODPATH manual (Pollock, 1994) provide excellent discussions of tracking in this transformed space, as does Zheng (1994).

Quasi Three-dimensional Representation of Confining Layers

Implicit confining layers are not supported now. The program is setup to handle these layers, but at the time of this writing the option was not tested. If there is interest in this option, please contact the corresponding author.

Connected Linear Networks (CLN)

Currently, mod-PATH3DU only supports the CLN package if it is used to simulate a well. It does not track particles within networks; however, it is hoped a future release will – please contact the corresponding author if you are interested in this feature and supporting its development. To track particles near a CLN well a default IFACE of 0 must be assigned to the CBB file property “GWF TO CLN”. Currently, it is assumed all networks in the CLN package represent a pumping well. This assumption will be rectified in a future release – if interested in supporting this development, please contact the corresponding author.

Multithreading and Reproducibility

Limitations

The Waterloo method, because it is semi-analytical, has similar limitations to the Pollock Method. For a detailed description of the Waterloo Method, including limitations, see Ramadhan (2015).

The applicability of the SSP&A method to tracking problems is limited by 1) the assumptions in the underlying tracking scheme, 2) the interpolation method it employs, and 3) limitations in the groundwater flow model, including discretization and boundary effects. The accuracy of the universal kriging interpolation is dictated by (a) grid discretization, (b) severe heterogeneities, and (c) proximity to certain boundaries. The refinement obtained by adding more cells spaced closer together will provide the approach with better information to calculate velocity. Nevertheless, the use of universal kriging can overcome many discretization and boundary effects provided an appropriate drift term is used. However, because the coefficients of these drift terms are determined through a regression, they can be made zero or their sign reversed (although this is unlikely). For example, when pumping in the presence of a strong regional gradient, the pumping well signal (its contribution to the hydraulic-head in a cell) may be overwhelmed by the signal from the regional gradient and thus made zero in the regression. Drift terms to overcome hydraulic conductivity dichotomies between two cells and to account for additional boundaries, including no-flow and streams/rivers, are possible, but are not yet available in this version of mod-PATH3DU.

The accuracy of the Runge-Kutta scheme depends on the tracking step size Δt. If Δt is too large, the calculation of the velocity components may be inaccurate and the particle tracking pathline may divert from the actual flow path. However, mod-PATH3DU mitigates this error by implementing the adaptive step size control procedure used in PATH3D called “step doubling”. For more information on the Runge-Kutta scheme please see Zheng (1989;1992;1994) and Zheng and Bennett (2002).

Because the Pollock method is used to calculate velocity in the *z*-direction, vertical sub-discretization is not supported now. Further, the cell structure in each layer must be identical.

Input Instructions

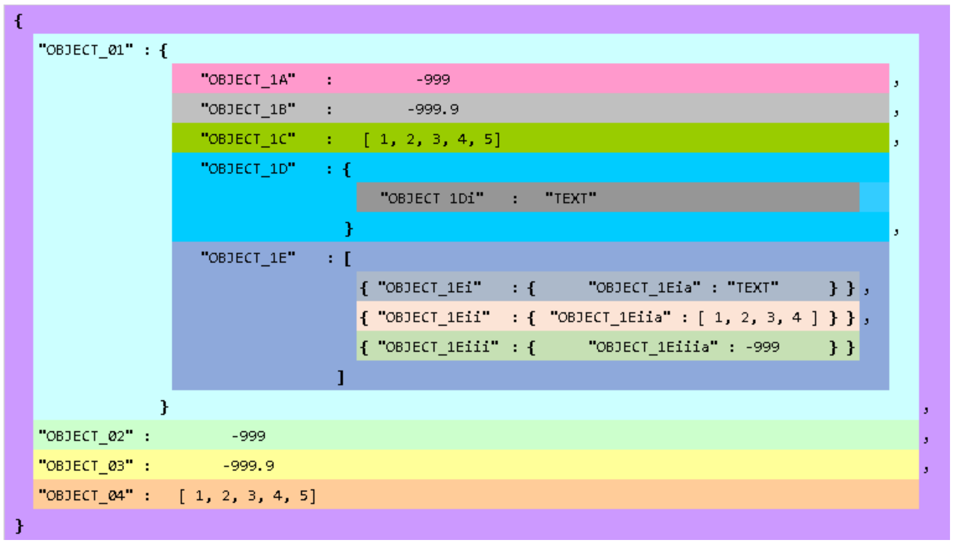
mod-PATH3DU v.2.0.0 is a rewrite. It is not backward compatible with previous releases. There are four mod-PATH3DU specific input files:

1. Primary file,
2. Particle starting location shapefile,
3. Per-cell property file, and
4. Grid specification file (GSF)

The following sections detail these files.

JSON

The format of the primary input files for mod-PATH3DU and its support programs follow the structure and rules of the lightweight data-interchange JSON[3] format. This format has few and simple rules and lends itself to being a flexible and dynamic human readable file.



```
{
  "OBJECT_01" : {
                  "OBJECT_1A"   :        -999                                  ,
                  "OBJECT_1B"   :        -999.9                                ,
                  "OBJECT_1C"   :   [ 1, 2, 3, 4, 5]                           ,
                  "OBJECT_1D"   : {
                                     "OBJECT 1Di"   :   "TEXT"
                                  }                                            ,
                  "OBJECT_1E"   : [
                                   { "OBJECT_1Ei"   : {     "OBJECT_1Eia" : "TEXT"        } } ,
                                   { "OBJECT_1Eii"  : {  "OBJECT_1Eiia" : [ 1, 2, 3, 4 ] } } ,
                                   { "OBJECT_1Eiii" : {     "OBJECT_1Eiiia" : -999        } }
                                  ]
                }                                                                ,
  "OBJECT_02" :        -999                                                      ,
  "OBJECT_03" :       -999.9                                                     ,
  "OBJECT_04" :   [ 1, 2, 3, 4, 5]
}
```

Figure 4 - Example JSON file. Each color represents a unique *key*:*value* pair, or object. Purple indicates the primary object (for which a key is not required). Pay particular attention to the use of commas and brackets in this example.

[3] www.json.org

JSON is an organized sequence of *key*:*value* pairs. A *key* must be a string (enclosed in quotation marks), for example "OBJECT_01", and a *value* can be one of three types, 1) an integer, real, or string (denoted by double-quotes) value, 2) a vector of values, or 3) an object (listing of additional *key*:*value* pairs). Vectors are denoted by square-braces, [ ], objects by curly-braces, { }, and single values by the absence of braces. Multiple objects can be listed for a *value*, each being separated by a comma. The sections that detail the primary input file present only the required keywords for each file. Appendix A details all of the keywords recognized by mod-PATH3DU and its support programs. Most of these keywords are optional – mod-PATH3DU will use a default value if a keyword is not specified or terminate in error if it is required. Additionally, the input instruction section for these primary files contain several examples. Because the files are human readable, it is hoped these examples make clear the file requirements and flexibility in using the JSON format. In the examples, pay attention to the brackets and commas – this is where most mistakes with the file are made. JSON is supported by a number of open-source libraries that facilitate the reading and writing of files - mod-PATH3DU incorporates JSONCPP[4] for this purpose. Additionally, there are tools available that allow a user to debug the JSON formatting of a file, some of which are online[5].

Primary File

The primary input file for mod-PATH3DU details the setup, options, and settings for the particle tracking. The following are required keywords:

KEY	VALUE
FLOW_MODEL_TYPE	Object, MODFLOW or MEUK
SIMULATIONS	Vector, listing any number and combination of PATHLINE or ENDPOINT

Keywords are detailed in Appendix A. Example files for MODFLOW and MEUK follow.

[4] https://github.com/open-source-parsers/jsoncpp
[5] e.g.: http://www.jsoneditoronline.org/

Example Primary File for MODFLOW

```
{
   "FLOW_MODEL_TYPE" : {
      "MODFLOW" : {
         "THREAD_COUNT" : 10,
         "GSF_FILE" : {
           "TYPE" : "GSF_V.1.1.0",
           "FILE_NAME" : "mp3du.gsf"
         },
         "NAME_FILE" : "mf2k_WEL.nam",
         "OUTPUT_PRECISION" : "DOUBLE",
         "IFACE" : [ { "WELLS" : 0 }, { "CONSTANT HEAD" : 2 } ]
      }
   },
   "SIMULATIONS" : [
      { "ENDPOINT" : {
          "NAME" : "DISPWELDEMO_23363",
          "THREAD_COUNT" : 10,
          "INITIAL_STEPSIZE" : 0.1,
          "MAX_DT" : 1000.,
          "ADAPTIVE_STEP_ERROR" : 1.e-6,
          "DIRECTION" : "FORWARD",
          "CAPTURE_RADIUS" : 95.0,
          "SIMULATION_END_TIME" : 1.e12,
          "OPTIONS" : [ "DISPERSION", "TRACK_TO_TERMINATION" ],
          "PARTICLE_START_LOCATIONS" : {
            "REPEAT" : 5000,
            "SHAPEFILE" : {
              "FILE_NAME" : "PartStart_Intersect1.shp",
              "CELLID_ATTR" : "P3D_CellID",
              "TIME_ATTR" : "TREL",
              "ZLOC_ATTR" : "ZLOC"
            }
          }
      } },
      { "PATHLINE" : {
          "NAME" : "DISPWELDEMO_23363",
          "DIRECTION" : "FORWARD",
          "PARTICLE_START_LOCATIONS" : {
            "SHAPEFILE" : {
              "FILE_NAME" : "PartStart_Intersect1.shp",
              "CELLID_ATTR" : "P3D_CellID",
              "TIME_ATTR" : "TREL",
              "ZLOC_ATTR" : "ZLOC"
            }
          }
      } }
   ]
}
```

This section is the FLOW_MODEL_TYPE. It is specified as MODFLOW, for which the NAM file and corresponding GSF file are listed. Also, the IFACE assignations are made. Porosity and other properties are input through the per-cell property file.

This section lists the SIMULATIONS. There are two simulations listed, an ENDPOINT and a PATHLINE. In the ENDPOINT listing the required and optional options and settings are listed, while the PATHLINE only lists the required parameters. More details on these KEYWORDs are detailed in Appendix A.

Example Primary File for MEUK

```
{
   "FLOW_MODEL_TYPE" : {
      "MEUK" : {
         "GRID" : { "NCOLS" : 201, "NROWS" : 201 },
         "GSF_FILE" : {
           "TYPE" : "GSF_V.1.1.0", "FILE_NAME" : "meuk.gsf"
         },
         "HHK" : { "CONSTANT" : 100. },
         "POROSITY" : { "CONSTANT" : 0.3 },
         "DISPERSION" : {
           "LONGITUDINAL" : 100.0,
           "TRANSVERSE_HORIZONTAL" : 10.0
         },
         "DRIFTS" : [
             { "WELL" : {
                "FILE_NAME" : "MEUK_WELLS.csv",
                "XCOORDS" : "XCOORDS", "YCOORDS" : "YCOORDS",
                "VAL" : "VAL", "TERM" : "TERM",
                "NAME" : "NAME", "EVENT" : "EVENT"
             } }
         ],
         "EVENTS" : [
            { "SteadyState" : {
               "DURATION" : 50000.,
               "FILE_NAME" : "Heads_for_MEUK.asc"
            } }
         ],
         "SEARCH_RADIUS" : 400.
      } },
   "SIMULATIONS" : [
      { "ENDPOINT" : {
          "NAME" : "MEUK_50000",
          "THREAD_COUNT" : 10, "INITIAL_STEPSIZE" : 0.1,
          "ADAPTIVE_STEP_ERROR" : 1.e-6, "CAPTURE_RADIUS" : 95.,
          "DIRECTION" : "FORWARD",
          "OPTIONS" : [ "DISPERSION" ],
          "PARTICLE_START_LOCATIONS" : {
             "REPEAT" : 5000,
             "SHAPEFILE" : {
                "FILE_NAME" : "PartStart_Intersect1.shp",
                "CELLID_ATTR" : "P3D_CellID",
                "TIME_ATTR" : "TREL", "ZLOC_ATTR" : "ZLOC"
             }
          }
      } }
   ]
}
```

This section is the FLOW_MODEL_TYPE. It is specified as MEUK. Input for MEUK is much different than MODFLOW because MEUK does not have a formal input file. As such, tracking properties, like porosity, are specified in this section. Further, additional aquifer properties, like hydraulic conductivity (HHK) not required by MEUK are specified here as well. The DRIFTS section lists the different drifts used in MEUK model, including the keyword correspondence between mod-PATH3DU and MEUK input files. The EVENTS section lists the kriging output file and the duration of each from which tracking velocity is calculated.

Particle Starting Locations File

The mod-PATH3DU particle starting location file is a shapefile. This shapefile must be a POINT shapefile and contain attributes indicating the starting cell id, local z elevation, and release time. Each mod-PATH3DU “SIMULATION” must include a “PARTICLE_START_LOCATIONS” section. This section requires a “SHAPEFILE” keyword through which the user supplies the filename and the attribute headers corresponding to the required “CELLID_ATTR”, “TIME_ATTR”, and “ZLOC_ATTR” keywords. See examples in the Primary File section above.

Per-cell Property File

The per-cell property file lists values for properties required by the particle tracking that may vary in each cell, such as porosity and dispersion This file is only required for MODFLOW, and must be included in the NAME file with the PATH type. Only later versions (after 2015) of MODFLOW-USG ignore the PATH type; older versions terminate in error when unknown types are listed in a NAME file. Thus, a user may have to comment out mod-PATH3DU’s property file when running the flow model.

Below is an example input file followed by instructions.

```
(1)   # mod-PATH3DU per-cell property input file (P3D)
(2)
(3)     CONSTANT       3          (10f12.4)        -1 VELOCITY METHOD L1
(4)     CONSTANT     0.1          (10f12.4)        -1 POROSITY L1
(5)     CONSTANT     1.0          (10f12.4)        -1 RETARDATION L1
(6)     CONSTANT     1.0          (10f12.4)        -1 Long. DISP L1
(7)     CONSTANT     0.1          (10f12.4)        -1 Trans-H. DISP L1
(8)     CONSTANT     0.0          (10f12.4)        -1 Trans-V. DISP L1
```

Table 2 – Per-cell property input instructions

Item	Variable[1]	Type[2]	Options / Values	Description
1	Comment	C	Optional. Must be preceeded by # in first column.	Comment
2	-	-	-	Required blank line.
			Repeat item 3 for every layer	
3	Velocity method	I	2 - SSPA 3 - Waterloo	The velocity method, SSPA or Waterloo. Currently this field is ignored, mod-PATH3DU will

				always use the Waterloo method for MODFLOW models, and the SSP&A method for MEUK models. Read using U2DREL.
			Repeat item 4 for every layer	
4	Porosity	R	-	Porosity. Read using U2DREL.
			Repeat item 5 for every layer	
5	Retardation	R	-	Retardation. Ignored. Read using U2DREL.
			Repeat item 6 for every layer	
6	α_L	R	-	Longitudinal Dispersivity. Read using U2DREL.
			Repeat item 7 for every layer	
7	α_T^H	R	-	Horizontal transverse dispersivity. Read using U2DREL.
			Repeat item 8 for every layer	
8	α_T^V	R	-	Vertical transverse dispersivity. Read using U2DREL.

1 Square brackets indicate optional or dependent variables

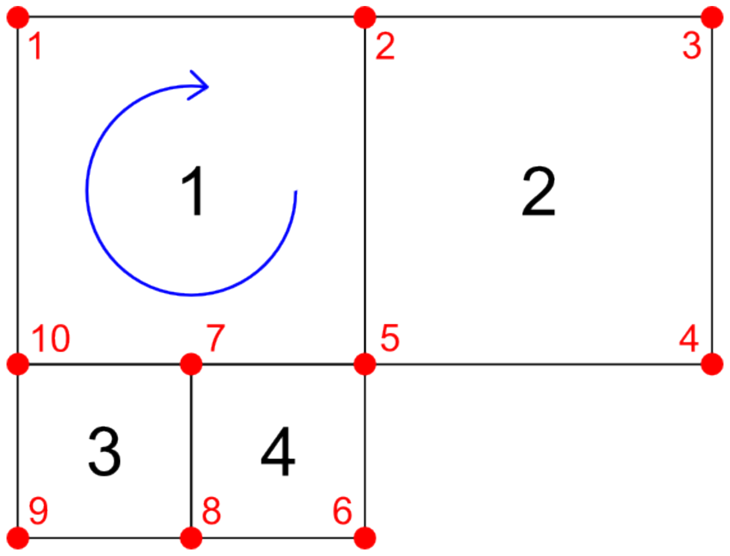
2 C Character
I Integer
R Real

Grid Specification File (GSF)

The grid specification file provides *x* and *y*-coordinates for the vertices of each cell in a MODFLOW-USG grid. The format of this file required by mod-PATH3DU v.1.1.0 and later is a modified form of the original format required for previous mod-PATH3DU releases. The mod-PATH3DU-specific GSF format is less general than the original format – the restrictions are:

1. Duplicate vertices are not listed. For example, between two identical square cells, for example 1 and 2 in the figure below, there are two shared nodes, 2 and 5, those shared nodes are listed only once – and referenced by both cells.
2. The z-coordinate is not needed when listing vertex coordinates. mod-PATH3DU gets elevations from the discretization and head-save files.

3. For each cell, the vertex nodes must be listed clockwise. It does not matter which node is listed first for a cell, but the subsequent nodes must be listed clockwise from that one. For example, for cell 2 in the figure below, its nodes are 4,5,2,3 or 2,3,4,5 etc.
4. For quad-based grids – if a cell has more than one connection on a side, all of the shared vertices must be listed for that cell. For example, for cell 1 in the figure below, its nodes are 1,2,5,7,10 – node 7 must be included.
5. For each vertex pair making a cell "face", list the connected cell that shares that face. For example, for cell 1, if the vertices are listed as 5,7,10,1,2, the cell that shares the face given by vertices 5 and 7 is 4, and for the face given by 7 and 10 is 3. For the face given by 10 and 1 a -999 is used to indicate there is not a connected cell, and the same for the face define by 1 and 2. The face between nodes 2 and 5 share cell 2.



At this point, most of the popular MODFLOW interfaces support the original GSF file. A utility is provided with mod-PATH3DU that will convert a GSF file in the original format into the mod-PATH3DU specific one. This program is described in Appendix B of this manual. This program, WriteP3DGSF, will:

1. Check for and exclude duplicate nodes – its tolerance is 1.e-4; any nodes within a distance of 1.e-04 units are considered identical.
2. Remove any reference to z-coordinates.
3. List vertices for each cell clockwise.
4. Ensure all vertices are listed for any cells with a side connected to more than one other cell. For example, in the figure above, if the GSF in the original format only lists nodes

1,2,5,10 for cell 1, WriteP3DGSF will automatically add node 7 to those listed for this cell. It does this by buffering each cell and identifying additional nodes inside this buffer.

5. Determines the connected cell that shares each face of a cell.
6. Write a new GSF file in the required format.

The mod-PATH3DU GSF file, corresponding to the figure above, is given in Figure 5.

```
(1) mod-PATH3DUv110
(2)                    10
(3)                     1          0.0         7.5
(3)                     2          5.0         7.5
(3)                     3         10.0         7.5
(3)                     4         10.0         2.5
(3)                     5          5.0         2.5
(3)                     6          5.0         0.0
(3)                     7          2.5         2.5
(3)                     8          2.5         0.0
(3)                     9          0.0         0.0
(3)                    10          0.0         2.5
(4)      1    2.5     5.0      5     5      7    10    1     2     4     3 -999   -999    2
(4)      2    7.5     5.0      4     4      5     2    3           -999     1 -999   -999
(4)      3   1.25    1.25      4     8      9    10    7           -999  -999    1      4
(4)      4   3.75    1.25      4     6      8     7    5           -999     3    1   -999
```

Figure 5 - Example GSF file

Table 3 - GSF input instructions

Item	Variable[1]	Type[2]	Options / Values	Description
1	Version	C	“mod-PATH3DUv110”	Version of the file. It is possible this file will change over time and this will allow for backward compatibility.
2	NVERT	I	-	Total number of nodes or vertices.
			Repeat Item 3 NVERT times	
	NodeID	I	-	Node ID
3	XVertex	R	-	X-coordinate of vertex
	YVertex	R	-	Y-coordinate of vertex
			Repeat Item 4 for every model cell	
4	CellID	I	-	Id of cell.

XCellCenter	R	-	X-coordinate of cell center.
YCellCenter	R	-	Y-coordinate of cell center.
NumberOfVertices	I	-	The number of vertices that define this cell. For example, for a square cell, the number of vertices is 4.
Repeat VertexId on line 4 NumberofVertices times. The vertices must be listed in clockwise order.			
VertexID	I	-	The NodeID of the vertex.
Repeat ConnectedCellId on line 4 NumberofVertices times. Listed in the order corresponding to the VertexId.			
ConnectedCellId	I	-	The CellID that shares the cell face for each vertex pair. -999 indicates no connected cell.

1 Square brackets indicate optional or dependent variables

2 C Character
I Integer
R Real

Output File Format

mod-PATH3DU writes a single binary file regardless of simulation type or options. The mod-PATH3DU suite includes a program for writing other file formats from this file. This program, writeP3DOutputs.exe is described in Appendix C. The following table details the format of the binary file.

Table 4 – Binary output file format

Item	Variable[1]	Type[2]	SIZE (BYTES)	Description
1	VERSION	CHAR	16	File version
2	NATTR	INTEGER	4	Number of attributes for each particle point
		Repeat item 3 NATTR times		
3	ATTR_NAME	CHAR	10	Name of attribute
	ATTR_SIZE	INTEGER	4	The size (bytes) of the attribute
	ATTR_TYPE	INTEGER	4	The type of the attribute; 1:Integer, 2:Double, 3:Char
		Item 4 is repeated for each attribute for every particle point		
4	VALUE	INTEGER DOUBLE CHAR	4 8 51	The particle attribute.
		The following are the attributes currently written by mod-PATH3DU to the binary file		
	PID	INTEGER	4	Particle ID
	CELLID	INTEGER	4	Id of cell containing particle
	PTIME	DOUBLE	8	Time
	PX	DOUBLE	8	X coordinate
	PY	DOUBLE	8	Y coordinate
	PZGLO	DOUBLE	8	Z coordinate (global)
	PZLOC	DOUBLE	8	Z coordinate (local)
	PVX	DOUBLE	8	Advective velocity in x-direction
	PVY	DOUBLE	8	Advective velocity in y-direction
	PVZ	DOUBLE	8	Advective velocity in z-direction
	PTERM	CHAR	51	Termination reason
	THREADID	INTEGER	4	ID of thread that calculated point

PLAYER	INTEGER	4	Model layer of CELLID
PHEAD	DOUBLE	8	Head in CELLID at PTIME
PTSTART	DOUBLE	8	Particle start time

Examples

Example 1a

Tracking Near a "Weak" Pumping Well (Structured Grid)

The purpose of this example is to highlight the strength of the Waterloo and SSP&A method in tracking near a pumping well – the analytical correction term implicitly handles weak and strong sinks. The groundwater system is confined (50 feet thick), steady-state and two-dimensional, in plan-view. The ambient groundwater flow field is uniform throughout the whole domain and flow is from left to right. The fully penetrating well is pumped at a constant rate of 50 feet^3/d. Hydraulic conductivity is 10 feet/d and porosity is 0.3. The numerical domain consists of 21 rows, each 5 feet wide, and 87 columns with varying widths ranging from 5 to 20 feet, for a total model width of 500 feet. The conceptual model is shown in Figure 6. The flow balance is presented in Table 5.

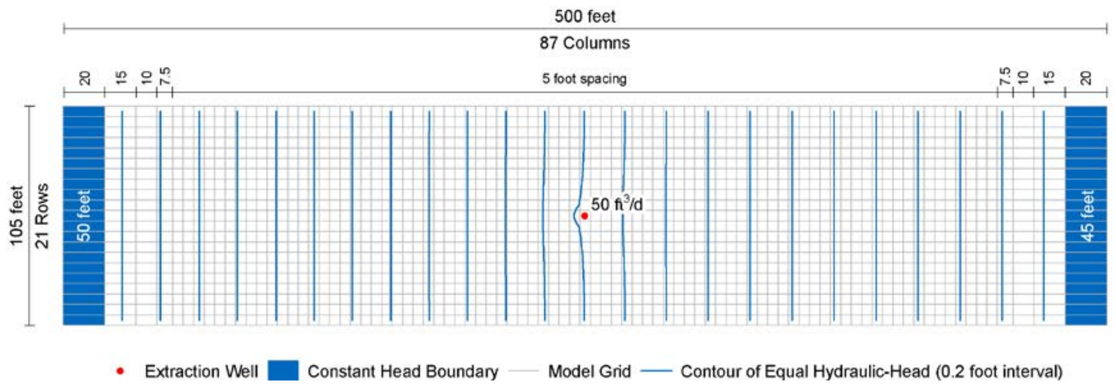


Figure 6 - Conceptual Model - Example 1a

Table 5 - Flow Balance - Example 1a

Item	In (ft^3/d)	Out (ft^3/d)	Net
Constant Head	571.9	521.9	50.0
Well (MNW2)	0.0	50.0	-50.0
Total	571.9	571.9	0.0

Particle paths, near the pumping well, calculated by MODPATH6 and both the Waterloo and SSP&A methods of mod-PATH3DU are shown in Figure 7. The Waterloo method tracks the same as the Pollock method, but also calculates pathlines in the cell containing the weak well – unlike the Pollock method. For both the Waterloo and SSPA methods, pathlines within the analytically calculated capture zone (using Jacob, 1949; p.344.), are captured by the well, while those outside, flow through the cell and continue tracking to their proper terminus. The head contours presented are an expression of what mod-PATH3DU tracks on in the vicinity of a well. They were calculated by interpolating from the MODFLOW calculated heads to a very fine grid using the universal kriging algorithm available with mod-PATH3DU. The shape visible around the well is determined by the linear-log drift term applied in this case.

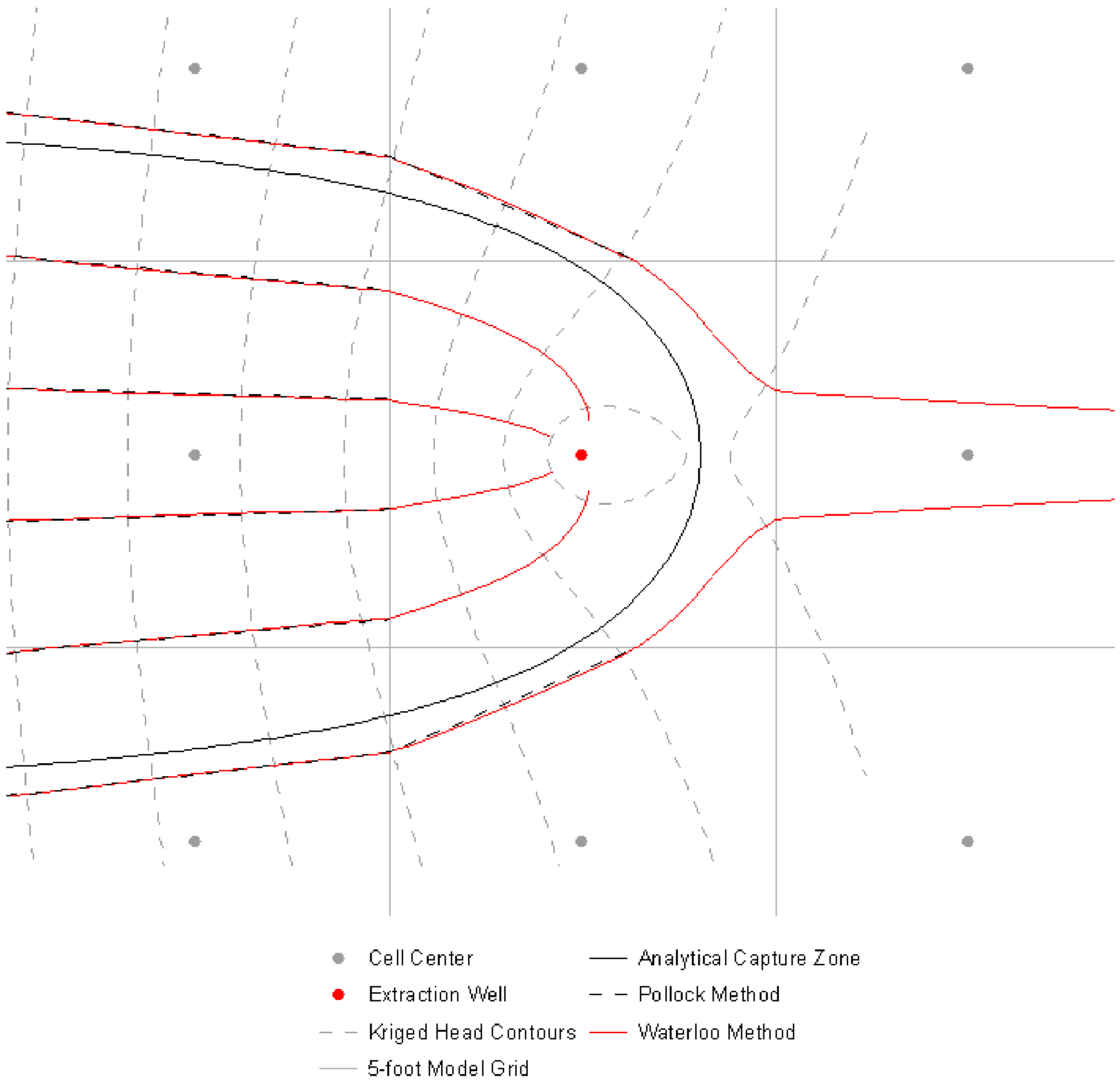


Figure 7 – Particle paths calculated using MODPATH6 and mod-PATH3DU near a "weak" sink.

Because both the Waterloo and SSP&A methods include an analytic solution when tracking near a well, they can mitigate some coarse grid discretization issues. Further, they both can simulate off-center pumping wells. An example of how mod-PATH3DU can be used to calculate capture zones for off-center pumping wells using the Area Based Redistribution (ABRD) approach of Pinales et al. (2003;2005) is presented in Muffels et al. 2011.

Example 1b

Tracking Near a "Weak" Pumping Well (Unstructured Grid)

The modeling exercise in Example 1a is repeated using an unstructured grid defined by Voronoi cells, Figure 8. The resulting flow balance is presented in Table 6. The difference in the flow entering the system between this model and the five-foot structured model is due to the different cell geometry along the boundaries.

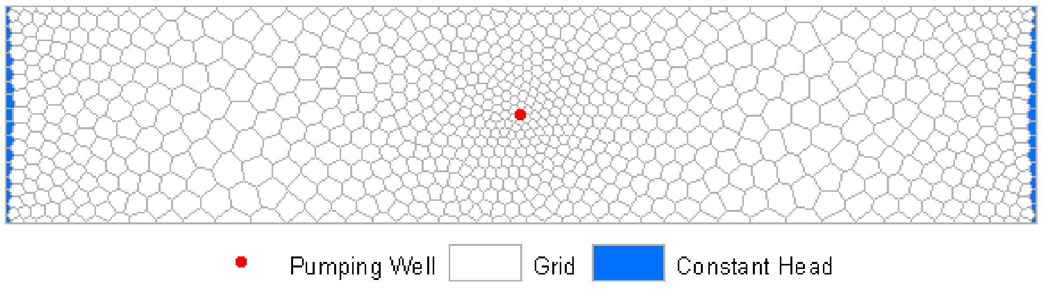


Figure 8 – MODFLOW-USG Voronoi example using AlgoMesh

Table 6 - Flow Balance - Example 1b

Item	In (ft^3/d)	Out (ft^3/d)	Net
Constant Head	532.4	482.4	50.0
Well (MNW2)	0.0	50.0	-50.0
Total	532.4	532.4	0.0

The mod-PATH3DU calculated particle paths, for both the Waterloo and SSP&A methods, for this model are shown on Figure 9.

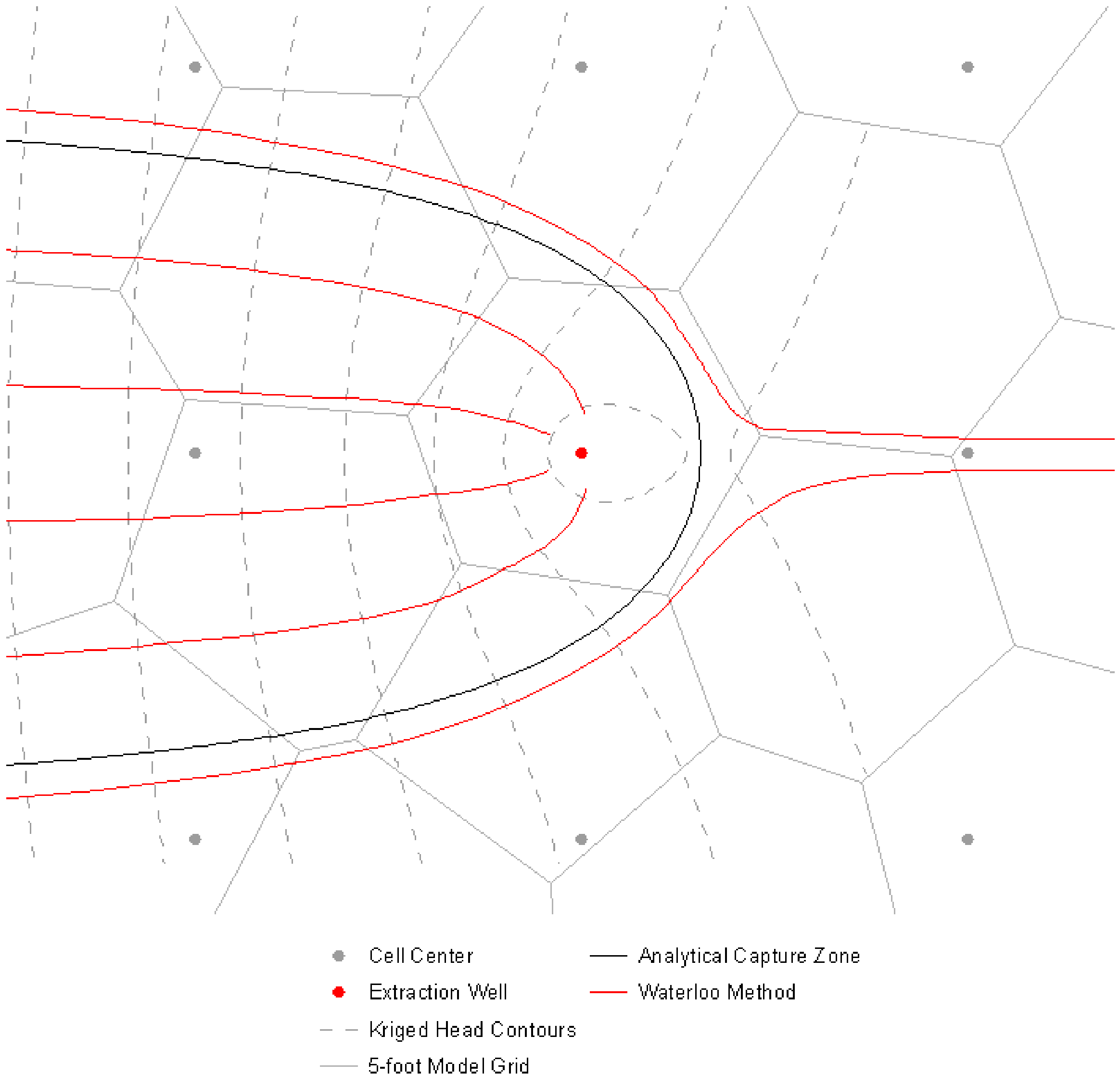


Figure 9 - Particle paths calculated by mod-PATH3DU near a “weak” pumping well for a MODFLOW-USG model with Voronoi cells

Example 2

Flow Path in Heterogeneous Cross-section

This example is adapted from the documentation of PATH3D ver. 4.6. The example was developed to test the calculation of groundwater pathlines in a heterogeneous cross-section. The groundwater model comprises three strata; the hydraulic conductivity of each stratum is shown in Figure 10. The model is discretized into 20 layers. The cross-section is bordered by no-flow boundaries along the left, right and bottom edges. The top boundary is assigned specified-heads to represent a linear water table declining from 41 feet at *x* = 100 feet to 40 feet at *x* = 0 feet. Porosity is 0.2. The resulting flow balance in presented in Table 7.

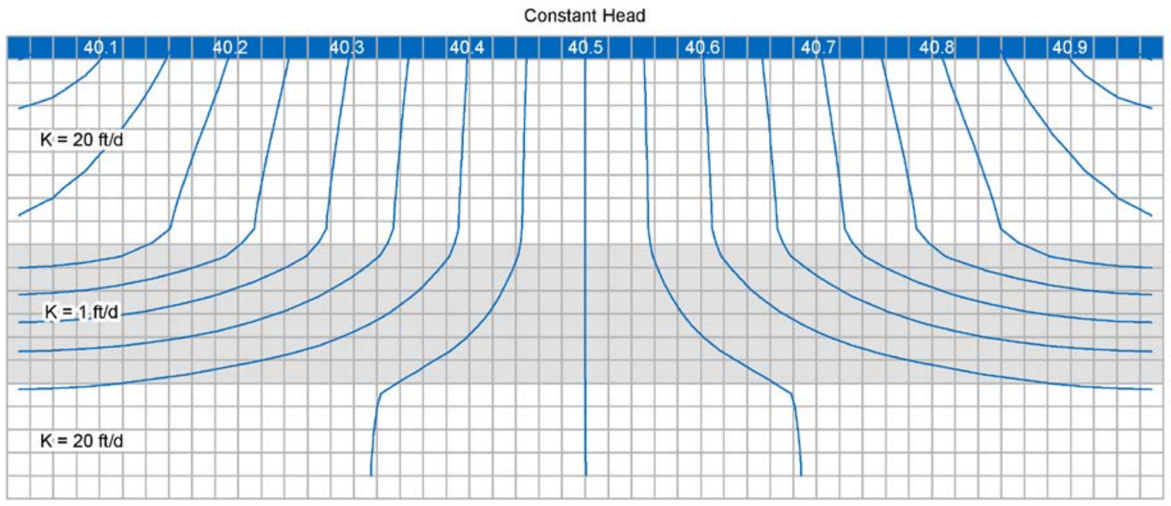


Figure 10 - Conceptual Model - Example 2

Table 7 - Flow Balance - Example 2

Item	In (ft^3/d)	Out (ft^3/d)	Net
Constant Head	3.7	3.7	0.0
Total	3.7	3.7	0.0

A particle is tracked backwards from *x* = 1 foot. The particle paths calculated using PATH3D v.4.6 and Waterloo method in mod-PATH3DU for this particle are shown in Figure 11. The total travel time calculated by each approach are nearly identical; PATH3D calculated 637 days, while mod-PATH3DU calculates 636 days. This example primarily serves to demonstrate the implementation of the Pollock method in the *z*-direction when used in conjunction with the Waterloo method is correct and that the method can be applied to cross-section models. This example, in and of itself, is not particularly challenging for the Pollock method.

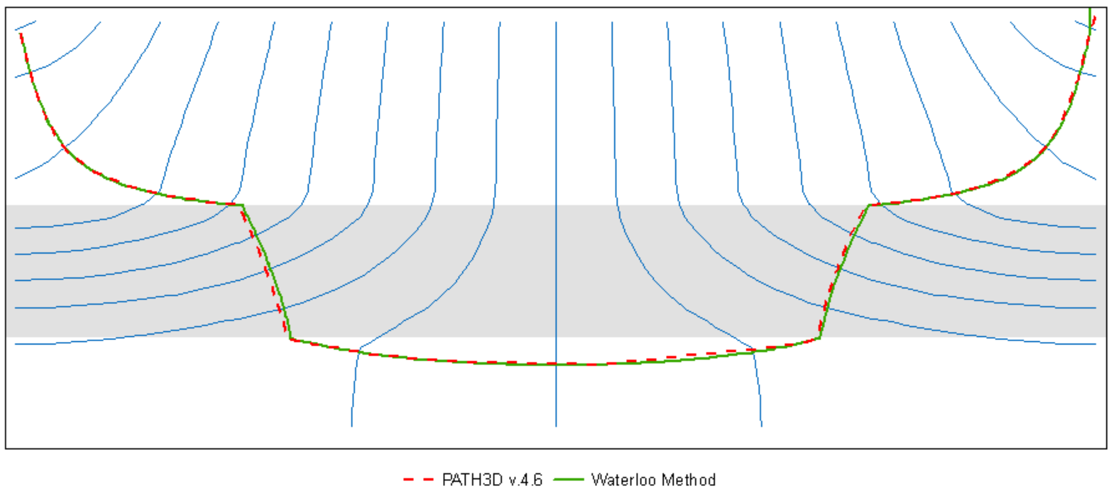


Figure 11 - Comparison of pathlines computed using mod-PATH3DU and PATH3D v4.6 - Example 2

Example 3

Example model from Pollock (2015)

Example 3 is taken from Pollock (2015). It is a two-dimensional steady-state model with a quad-based unstructured grid. The system is confined with a constant hydraulic conductivity (100 feet/day) and porosity (0.25). No-flow boundaries surround the model on all four sides. Flow circulation is induced with constant-head cells in the center of the model domain, from 15 feet to 2 feet. There are 352 cells in the unstructured grid. mod-PATH3DU was used to calculate flow paths for the unstructured grid model and these were compared with the results in Pollock (2015) – Figure 12. The flow paths calculated by mod-PATH3DU are comparable to those calculated by the algorithm described by Pollock (2015).

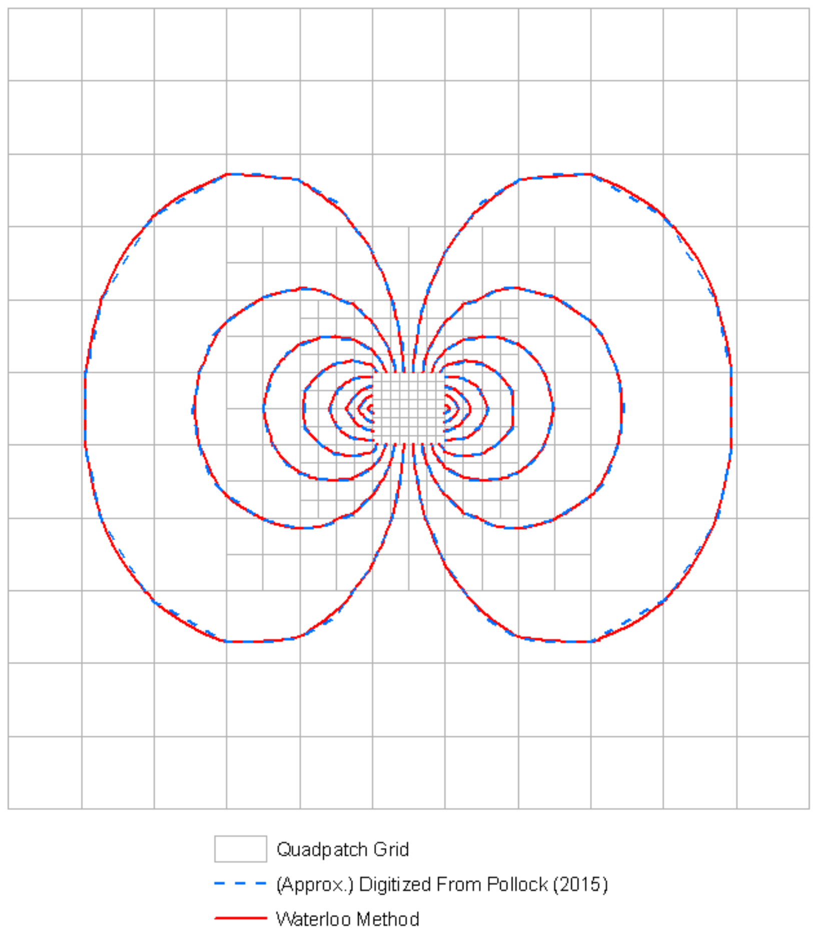
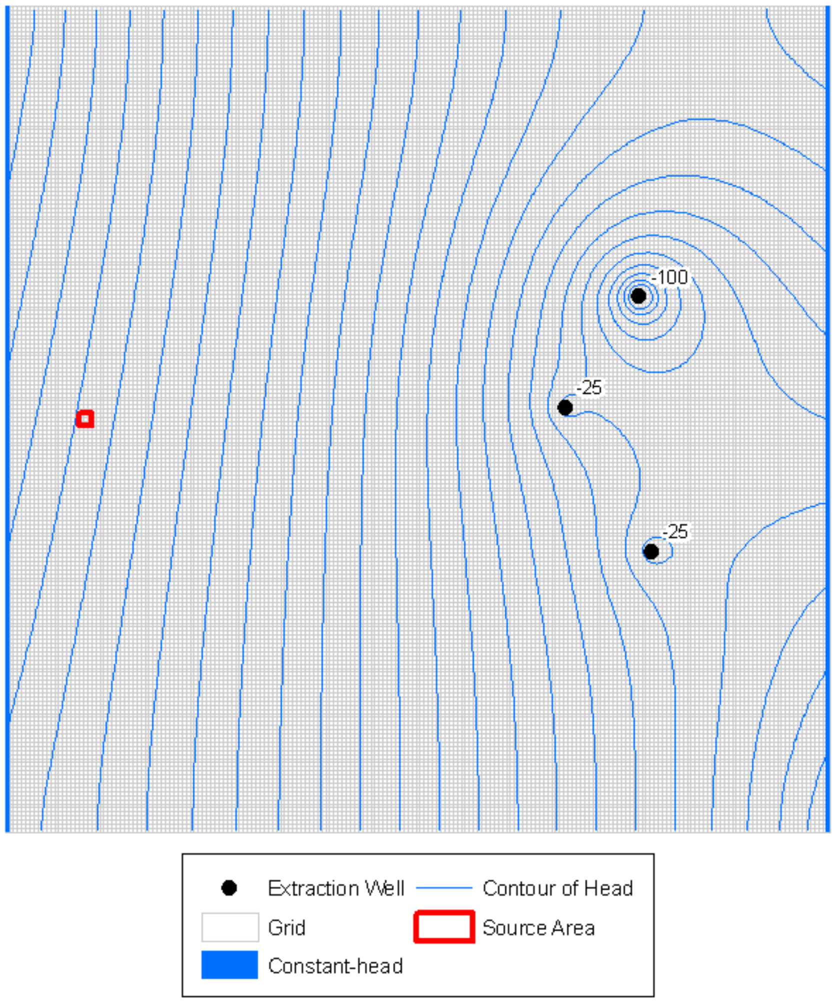


Figure 12 - Comparison of mod-PATH3DU flow paths with those calculated by Pollock (2015) for an unstructured grid

Example 4a

Advection-dispersion with MODFLOW and MEUK

This example illustrates the advection-dispersion functionality of mod-PATH3DU. It is a single layer model with 201 rows and 201 columns; each 100 feet. Hydraulic conductivity is uniform at 100 feet/day. Constant-head boundary conditions line the left and right-hand sides. The values vary linearly - on the left between 80 feet at the top and 75 feet at the bottom, and on the right between 65 and 55 feet at the top and bottom, respectively. The model is steady-state with three extraction wells; the first pumping at a rate of 25,000 feet^3/day, the second at 100,000 feet^3/day, and the third at 25,000 feet^3/day.



The transport of a contaminant following an instantaneous injection of 1000 units in 9 model cells is simulated using MT3D. Dispersivity parameters are as follows, longitudinal 100 feet, and transverse 10 feet. Porosity is 0.3. mod-PATH3DU is used to simulate advection and dispersion by releasing 144 particles over the source area and repeating each particle 5000 times for a total of 720,000 particles. The mass at each particle is calculated, and using the particle count in each cell at different time intervals, concentration distributions are compared against the MT3D results. These comparisons are shown in the figures that follow. The first set of figures result from tracking on the MODFLOW solution, while the second set of figures corresponds to running the equivalent MEUK model. For both models, the correspondence between the two is very good. It is not the intent that mod-PATH3DU be used instead of a transport model like MT3D, rather it is intended to be used as a screening level model.

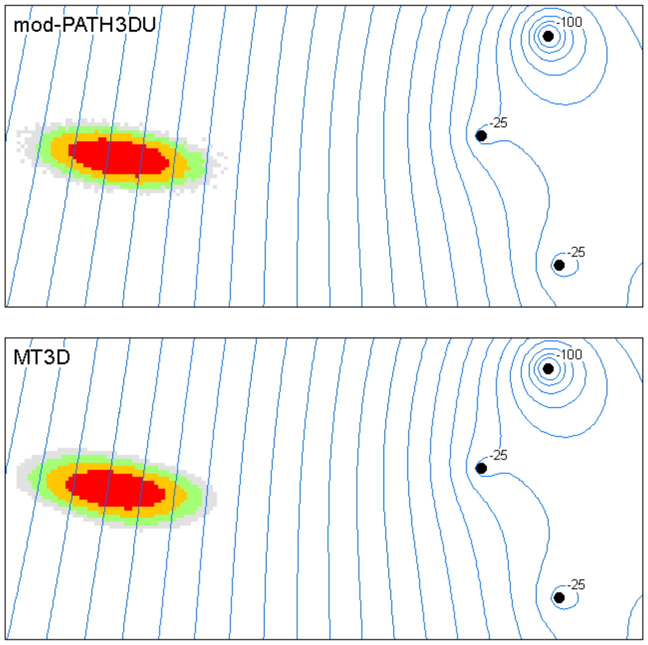


Figure 13 - Comparison of mod-PATH3DU with dispersion to MT3D for an example MODFLOW model (after 5,044 days).

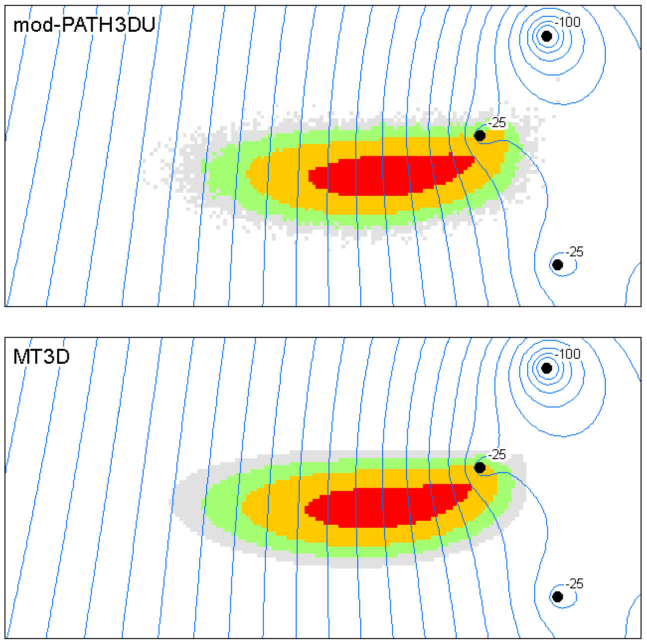


Figure 14 - Comparison of mod-PATH3DU with dispersion to MT3D for an example MODFLOW model (after 23,363 days).

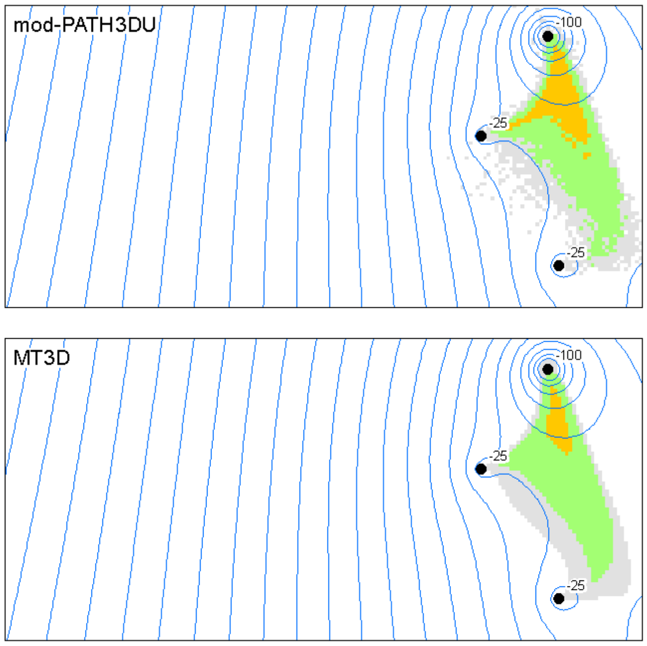


Figure 15 - Comparison of mod-PATH3DU with dispersion to MT3D for an example MODFLOW model (after 50,000 days).

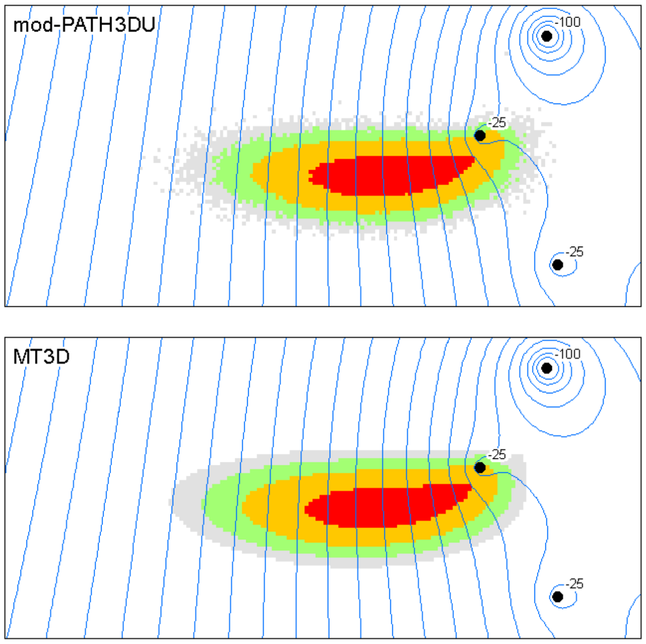


Figure 16 - Comparison of mod-PATH3DU with dispersion to MT3D for an example MEUK model (after 23,363 days).

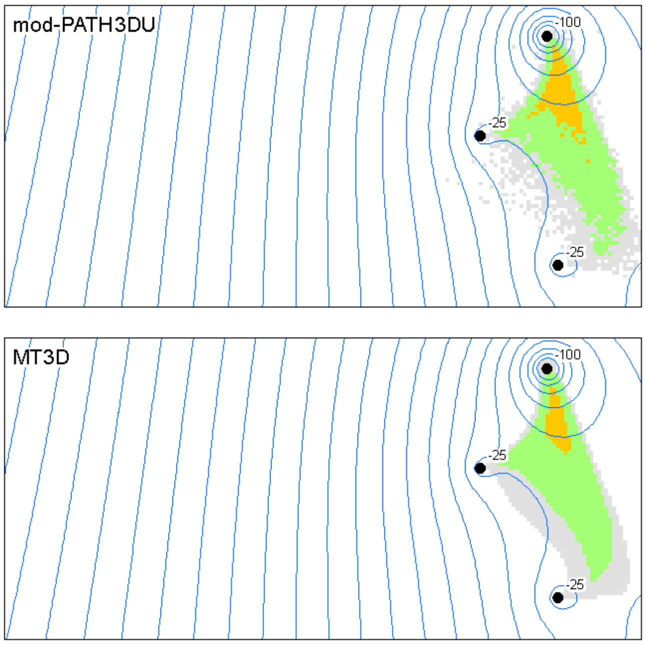


Figure 17 - Comparison of mod-PATH3DU with dispersion to MT3D for an example MEUK model (after 50,000 days).

Example 4b

Advection-dispersion with Variable Hydraulic Conductivity Zones

This example used the same MODFLOW and MT3D models described in Example 4a, with the following changes, 1) no extraction wells were simulated, 2) hydraulic conductivity is varied through zones, 3) transverse dispersivity was tripled from 10 to 30, and 4) fewer particles were tracked - only 2000 per start location for a total of 288,000 particles.

The following figure illustrates the good correspondence between MT3D and the advection-dispersion tracking of mod-PATH3DU. The hydraulic conductivity zone with a value of 10 feet/day is outlined in red to make clear transport in this zone.

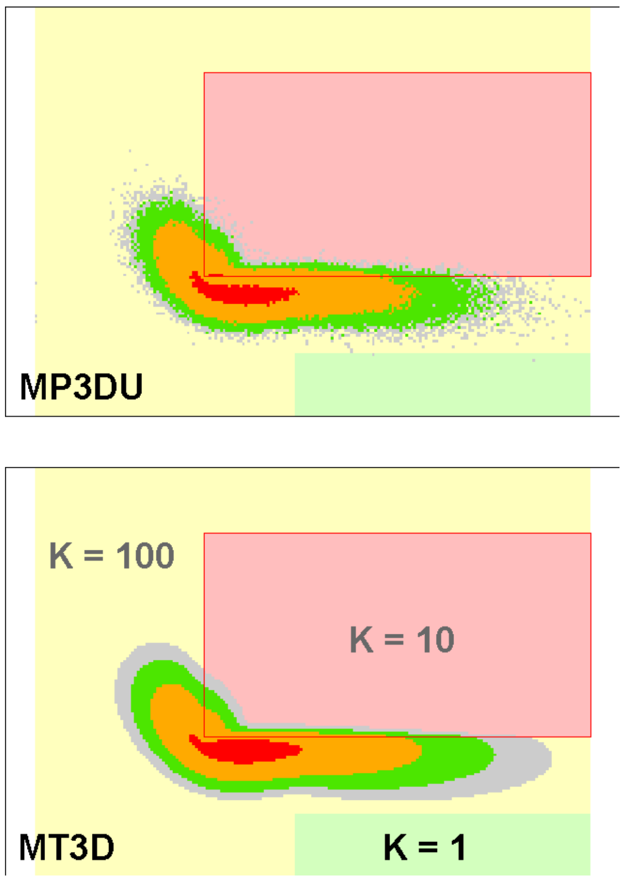


Figure 18 - Comparison of mod-PATH3DU with dispersion to MT3D for an example MODFLOW model with variable zones of hydraulic conductivity (after 50,000 days).

References

Abrams, D., H. Haitjema, and L. Kauffman, 2012. On Modeling Weak Sinks in MODPATH: Ground Water 51, no. 4: 597-602, doi 10.1111/j.1745-6584.2012.00995.x.

Burnett, R.D., and E.O. Frind, 1987. Simulation of contaminant transport in three dimensions, 2, Dimensionality effects. Water Resources Research, Vol. 23, No. 4, 695.

Cressie, N., 1993, Statistics for Spatial Data, revised ed., 900pp., Wiley, New York.

Deutsch, C. V. and Journel, A. G., 1998, GSLIB: Geostatistical Software Library and User's Guide, 2nd ed. Oxford University Press, Oxford.

Haitjema, H.M., 1995. Analytic Element Modeling of Groundwater Flow, Academic Press Inc., San Diego, California.

Harbaugh, A.W., 2005, MODFLOW-2005, the U.S. Geological Survey modular ground-water model—The Ground-Water Flow Process: U.S. Geological Survey Techniques and Methods 6–A16, variously paginated.

Jacob, C.E., 1949, Flow of ground water, in Engineering Hydraulics, Proceedings of the 4th Hydraulics Conference, Iowa Institute of Hydraulic Research, H. Rouse (ed.), June 12-15, 1949, John Wiley & Sons, Inc., New York.

Jankovic, I., and Barnes R., 1999a, High-order line elements in modeling two-dimensional groundwater flow, Journal of Hydrology 226, pages 224-233.

Jankovic, I., and Barnes R., 1999b, Three-dimensional flow through large numbers of spheroidal inhomogeneities, Journal of Hydrology 226, pages 224-233.

Journel, A.G. and Huijbregts, C.J., 1992, Mining Geostatistics. Academic Press, New York.

Karanovic, M., M. Tonkin, and D. Wilson, 2009. KT3D_H20: A Program for Kriging Water-Level Data Using Hydrologic Drift Terms: Ground Water 45, no. 4: 580-586, doi 10.1111/j.1745-6584.2009.00565.x.

Kinzelbach, W., 1990, Simulation of pollutant transport in groundwater with the random walk method. Groundwater Monitoring and Management (Proceedings of the Dresden Symposium, March 1987). IAHS Publ. no. 173, 1990.

LaBolle, E.M., Quastel, J., Fogg, G.E., and Gravner, J., 2000, Diffusion processes in composite porous media and their numerical integration by random walks: Generalized stochastic differential equations with discontinuous coefficients. Water Resources Research, Vol. 36, No. 3, Pages 651-662.

Muffels, C., Stonebridge, G., Tonkin, M.J., and Karanovic, M., 2011. An Unstructured Version of PATH3D, PATH3DU. Presentation at MODFLOW and More 2011, Integrated Hydrologic Modeling, International Ground Water Modeling Center, Colorado School of Mines, Golden, CO, June 6-8, 2011, v. 1, pp. 272-275.

Panday, S., Langevin, C.D., Niswonger, R.G., Ibaraki, M., and Hughes, J.D., 2013, MODFLOW-USG version 1: An unstructured gird version of MODFLOW for simulating groundwater flow and tightly coupled processes using a control volume finite difference formulation: U.S. Geological Survey Techniques and Methods 6 A45.

Pinales, A., A. Keer, F. Espinosa, L. Manzanares, G. Llerar, and A. Chavez. (2003), An Alternative Approach for Assigning Well Rates in a Block-Centered Finite-Difference Grid. In MODFLOW and More 2003: Understanding through Modeling – Conference Proceedings.

Pinales, A. Chvez, G. Llerar, L. Manzanares, and A. Keer. (2005), An Improved Approach for Assigning Pumping Rates to Heterogeneous Aquifer Models. Ground Water, vol. 43:274-279.

Pollock, D.W., 1988, Semi-analytical computation of pathlines for finite difference models, Ground Water vol. 26, no. 6: 743–750.

Pollock, D.W., 1989, Documentation of a computer program to compute and display pathlines using results from the U.S. Geological Survey modular three-dimensional finite-difference ground-water flow model: U.S. Geological Survey Open-File Report 89–381.

Pollock, D.W., 1994, User’s guide for MODPATH/MODPATH-PLOT, version 3: A particle-tracking post-processing package for MODFLOW, the U.S. Geological Survey finite-difference ground-water flow model: U.S. Geological Survey Open-File Report 94–464.

Pollock, D.W., 2012, User’s guide for MODPATH version 6 A particle-tracking model for MODFLOW: U.S. Geological Survey Techniques and Methods 6–A41, 58p.

Pollock, D.W., 2015, Extending the MODPATH algorithm to Rectangular Unstructured Grids, Ground Water vol. 54, no. 1: 121–125. doi: 10.1111/gwat.12328.

Prickett, T.A., Naymik, T.G., and Lonnquist, C.G., 1981, A “Random-Walk” Solute Transport Model for Selected Groundwater Quality Evaluations. Illinois State Water Survey, Champaign, Bulletin 65, 1981.

Rhamadhan, M., 2015, A Semi-Analytical Particle Tracking Algorithm for Arbitrary Unstructured Grids. Unpublished MASc. Thesis. University of Waterloo, Waterloo Ontario.

Salamon, P., 2006, On modeling contaminant transport in complex porous media using random walk particle tracking. Phd Thesis. Universidad Politecnica de Valencia.

Skrivan, J. A., and Karlinger, M. R., 1980, Semi-variogram estimation and universal kriging program; U. S. Geological Survey Computer Contribution, 98 p. Tacoma, Washington. (Computer Program K603).

Tonkin, M.J., and Larson, S. P., 2002. Kriging Water Levels with a Regional-linear and Point-logarithmic Drift. Ground Water, vol. 40, no. 2: 185-193.

Zheng, C., 1989, PATH3D, A ground-water path and travel-time simulator, version 3.0 user’s manual, S.S. Papadopulos & Associates, Inc., Bethesda, MD.

Zheng, C., 1992, PATH3D, A ground-water path and travel-time simulator, version 3.2 user’s manual, S.S. Papadopulos & Associates, Inc., Bethesda, MD.

Zheng, C., 1994, Analysis of particle tracking errors associated with spatial discretization, Ground Water, vol. 32 no. 5: 821-828.

Zheng, C., and Bennett, G., 2002. Applied Contaminant Transport Modeling, 2nd ed., John Wiley & Sons, New York.

Appendix A: Supported *Key*:*Value* Pairs

Key	Value	Description
FLOW_MODEL_TYPE	Object	Specifies the type of flow model. Required.
Options (one of the following is required)		
MODFLOW		Indicates MODFLOW is the model type.
MEUK		Indicates MEUK is the model type.
MODFLOW	Object	MODFLOW model.
Options		
NAME_FILE	String	The name of the MODFLOW name file. Required.
GSF_FILE	Object	Specifies the type and name of the GSF file. Required.
OUTPUT_PRECISION	String	Output precision of MODFLOW output files, cell-by-cell and head-save. Can be on of SINGLE or DOUBLE. Optional. Default is Single.
IFACE	Vector of objects	List of the different boundary conditions and their respective IFACE values. This is a critical object. Be sure to read the IFACE section of this manual and ensure all boundary condition types are explicitly listed here.
THREAD_COUNT	Integer	Number of threads to use when processing MODFLOW output files. This processing can be time consuming for simulations with many stress periods and time steps. mod-PATH3DU can parallelize aspects of this processing which can reduce run time. Optional. Default is 1.
MEUK	Object	MEUK model.
Options		
GRID	Object	Grid specification for a MEUK model, i.e. number of rows and columns. Required.
GSF_FILE	Object	Specifies the type and name of the GSF file. Required.
OPTIONS	Vector of strings	Simulation options. Optional. No default.
HHK	Object	Horizontal hydraulic conductivity. Required.
POROSITY	Object	Porosity. Required.

DISPERSION	Object	Longitudinal and transverse dispersivities. Required.
DRIFTS	Object	List of the drift objects. Optional. No default.
EVENTS	Object	List of the event objects. Required.
SEARCH_RADIUS	Real	The kriging search radius for the SSP&A method. Optional. Default is 1000.
OPTIONS	Vector of strings	MEUK options
Options		
KRIGE_ONLY		Use kriging interpolation everywhere in model domain.

ENDPOINT	Object	Endpoint simulation type.
PATHLINE	Object	Pathline simulation type.
Options		
NAME	String	A name to identify the simulation. The name of the output binary file is this NAME appended with “_EPT” or “_PTL” for endpoint and pathline simulations, respectively. Required.
THREAD_COUNT	Integer	The number of threads to use to calculate the pathlines. Optional. Default is 1.
INITIAL_STEPSIZE	Real	The size of the initial tracking step size. Optional. Default is 1.
ADAPTIVE_STEP_ERROR	Real	The Runge-kutta adaptive step error control parameter. Optional. Default is 1.e-6. Values between 1.e-4 and 1.e-8 are recommended.
MAX_DT	Real	Maximum tracking step size. Optional. Default is 1.e6.
SIMULATION_END_TIME	Real	The end time of the tracking simulation. When forward tracking, this is the maximum tracking time, and when backward tracking it is the minimum tracking time. Optional. Default is 1.e12 when forward tracking and 0 when backward tracking.
CAPTURE_RADIUS	Real	The distance from any pumping wells within which a particle is considered captured. Optional. Default is 10.
DIRECTION	String	The particle tracking direction; “BACKWARD” or “FORWARD”. Optional. Default is FORWARD.
OPTIONS	Vector of strings	Simulation options. Optional. No default.
PARTICLE_START_LOCATIONS	Object	Indicates the particle starting locations, including SHP filename and options. Required.
OPTIONS	Vector of strings	Simulation options
Options		
DISPERSION		Simulate dispersion. This option must be set to simulate

		dispersion.
TRACK_TO_TERMINATION		This option will track particles on the final stress period of the simulation until they terminate at a boundary. It overwrites SIMULATION_END_TIME.

GSF_FILE	Object	Specifies the type of GSF file. For writeP3DGSF.exe, it indicates the file to convert. For mod-PATH3DU and other programs, it indicates the file to read – this file must be of type “GSF_V.1.1.0”. FILE_NAME is not required when converting a structured DIS file to GSF.
TYPE	String	Supported types: GSF_V.1.0.0, GSF_V.1.1.0, SHAPEFILE, STRUCTURED_DIS, and MODPATH7_MPUGRID.
FILE_NAME	String	The name of the GSF file.

PARTICLE_START_LOCATIONS	Object	Definition of the particle starting locations. Required.
Options		
SHAPEFILE	Object	Specifies the parameters necessary to define particle starting locations from a shapefile. Required.
REPEAT_DT	Real	Repeat each particle’s release position at intervals of REPEAT_DT from release time to end of simulation. For example, if a particle’s release time is 5, and the simulation end time is 10, a REPEAT_DT of 1, will release this particle at time 5, 6, 7, 8, 9, and 10.
REPEAT	Integer	The number of times to repeat each particle. Optional. Default is 1. This parameter can be used to repeat particles, which is especially useful when using the dispersion option.

SHAPEFILE	Object	The particle starting locations shapefile. Required.
Options		
FILE_NAME	String	The filename of the shapefile. Required.
CELLID_ATTR	String	The attribute header in the shapefile that corresponds to the cell id in which a particle starts. Required.
TIME_ATTR	String	The attribute header in the shapefile that corresponds to the release time of a particle. Required.
ZLOC_ATTR	String	The attribute header in the shapefile that corresponds to the

		local z-coordinate of a particle. Required.
TRANSFORMATION	Object	Specifies the transformation to convert between global and local model coordinates. This key is used when WriteP3DGSF converts a structured grid defined by DELR and DELC into a GSF with global coordinates. It can also be used to convert an existing GSF from local to global coordinates. Optional.
Options		
XOFF	Real	The offset in the x-direction when converting between global and local model coordinates. Optional. Default is 0.
YOFF	Real	The offset in the y-direction when converting between global and local model coordinates. Optional. Default is 0.
ROT	Real	The rotation (in degrees) when converting between global and local model coordinates. Optional. Default 0.
IFACE	Vector	List of the IFACE specification for each boundary condition in a MODFLOW model.
Options		
e.g. CONSTANT HEAD, DRAINS, RIVER LEAKAGE, WELLS, any other boundary package listing in the cell-by-cell flow file	Object	Each boundary package in a MODFLOW model must be assigned an appropriate IFACE. Example listing is: { "CONSTANT HEAD" : 6 } Each BC must be separated by a comment and bracketed as in the example.
OUTPUT_FILENAME	String	The name of the output file. Used by writeP3DGSF.exe to supply the name of the new GSF file.
GRID	Object	The designation of the number or rows and columns in a MEUK model. Required.
Options		
NCOLS	Real	Number of columns. Required.
NROWS	Real	Number of rows. Required.
XLLCORNER	Real	The x-coordinate of the lower-left corner of the model domain. Optional/Required by writeP3DGSF.exe.
YLLCORNER	Real	The y-coordinate of the lower-left corner of the model domain. Optional/Required by writeP3DGSF.exe.

CELLSIZE	Real	The dimensions, length in x and y, of a grid cell. Optional/Required by writeP3DGSF.exe.

HHK	Object	Specification of horizontal hydraulic conductivity for a MEUK model. Required.
Options (one of the following is required)		
CONSTANT	Real	Value of conductivity applied to entire MEUK domain.
FILE_NAME	String	Name of file listing conductivity values for each cell in MEUK model domain. Cells are listed in order from cell 1 to cell N, according to GSF file.

POROSITY	Object	Specification of porosity for a MEUK model. Required.
Options (one of the following is required)		
CONSTANT	Real	Value of porosity applied to entire MEUK domain.
FILE_NAME	String	Name of file listing porosity values for each cell in MEUK model domain. Cells are listed in order from cell 1 to cell N, according to GSF file.

DISPERSION	Object	Specification of the dispersivity parameters. Optional.
Options		
LONGITUDINAL	Real	Value of longitudinal dispersivity for entire MEUK domain. Optional. Default is 0.
TRANSVERSE_HORIZONTAL	Real	Value of transverse-horizontal dispersivity for entire MEUK domain. Optional. Default is 0.

DRIFTS	Vector of objects	List of the different drift components in the MEUK model. Optional.
Options		
WELL	Object	Well drift specification.
LINESINK	Object	Linesink drift specification.

WELL	Object	The linkage of mod-PATH3DU parameters designating wells to those in a MEUK simulation.
Options		
FILE_NAME	String	The name of the file listing well drifts.

XCOORDS	String	Column header in the MEUK well drift file that corresponds to the x-coordinate of the well. Required.
YCOORDS	String	Column header in the MEUK well drift file that corresponds to the y-coordinate of the well. Required.
VAL	String	Column header in the MEUK well drift file that corresponds to the value (pumping rate) of the well. Required.
TERM	String	Column header in the MEUK well drift file that corresponds to the drift term a well belongs to. Required.
NAME	String	Column header in the MEUK well drift file that corresponds to the name of the well. Required.
EVENT	String	Column header in the MEUK well drift file that corresponds to the event of the well. Required.

LINESINK	Object	The linkage of mod-PATH3DU parameters designating linesinks to those in a MEUK simulation.
Options		
FILE_NAME	String	The name of the file listing linesink drifts. Required.
XCOORDS	String	Column header in the MEUK linesink drift file that corresponds to the first x-coordinate of a linesink. Required.
YCOORDS	String	Column header in the MEUK linesink drift file that corresponds to the first y-coordinate of a linesink. Required.
XCOORDS_2	String	Column header in the MEUK linesink drift file that corresponds to the second x-coordinate of a linesink. Required.
YCOORDS_2	String	Column header in the MEUK linesink drift file that corresponds to the second y-coordinate of a linesink. Required.
VAL	String	Column header in the MEUK linesink drift file that corresponds to the value (strength) of the linesink. Required.
TERM	String	Column header in the MEUK linesink drift file that corresponds to the drift term a linesink belongs to. Required.
NAME	String	Column header in the MEUK linesink drift file that corresponds to the name of the linesink. Required.
EVENT	String	Column header in the MEUK linesink drift file that corresponds to the event of the linesink. Required.

EVENTS	Vector of objects	List of the events in a MEUK simulation. Each event is designated by its name and requires the options listed below.
Options		

FILE_NAME	String	Name of the file containing the kriged surface calculated by MEUK, upon which mod-PATH3DU will track particles.
DURATION	Real	The length of the event.

Appendix B: WriteP3DGSF.exe

Regardless of whether a model is structured or unstructured, this latest release of mod-PATH3DU requires the program specific GSF file for input of the spatial location of each model cell. To facilitate the transition to this new format WriteP3DGSF.exe was developed to:

1. Convert a GSF file in another format into the mod-PATH3DU specific format, or
2. Convert a polygon SHAPEFILE with a cell id attribute (cell ids must begin at 0), or
3. Convert a MODPATH7 MPU grid file, or
4. Convert a structured MODFLOW grid defined by DELR and DELC; and
5. Write the corresponding GSF file for a structured model.

The program is executed from the command line as follows:

```
writeP3DGSF FILENAME
```

where *FILENAME* is the name of the required input file, detailed below, and must be the first input on the command line after the name of the writeP3DGSF executable. Optionally, and recommended, users can supply *colorcode* after FILENAME, i.e.

```
writeP3DGSF FILENAME colorcode
```

which will color the text of the console as the program runs. This color coding is more aesthetic than a true codification, except when there is an error – errors, and only errors, are provided in red text.

The primary input file for writeP3DGSF.exe requires the following keywords:

KEY	VALUE
FLOW_MODEL_TYPE	MODFLOW or MEUK

The following are optional keywords:

KEY	VALUE
TRANSFORMATION	To convert from local to global coordinates.
OUTPUT_FILENAME	To specify a name for the new GSF file. If not present, the user will be prompted for a filename during runtime.

Keywords are detailed in Appendix A. Example files follow.

```
{
  "FLOW_MODEL_TYPE" : {
    "MODFLOW" : {
      "NAME_FILE" : "MODFLOW-USG.nam",
      "GSF_FILE" : {
(1)     "TYPE" : "GSF_V.1.0.0",
        "FILE_NAME" : "MP3DUv100.gsf"
      }
    }
  }
}
```

```
{
   "TRANSFORMATION" : {
       "XOFF" : 1234.,
(2)    "YOFF" : 5678.,
       "ROT"   : 9.
   },
   "FLOW_MODEL_TYPE" : {
      "MODFLOW" : {
         "NAME_FILE" : "MODFLOW2000.nam",
         "GSF_FILE" : {
           "TYPE" : "STRUCTURED_DIS"
         }
      }
   }
}
```

```
{
   "FLOW_MODEL_TYPE" : {
      "MODFLOW" : {
         "GSF_FILE" : {
           "TYPE" : "MODPATH7_MPUGRID",
(3)        "FILE_NAME" : MP7.mpugrid"
         },
         "NAME_FILE" : "MODFLOW-USG.nam"
      }
   }
}
```

The first example (1), will convert an existing GSF (in general format, v.1.0.0) into the updated format for a MODFLOW-USG model, while the second example (2), will write a GSF file from a structured MODFLOW-2000 model, in global coordinates. The third example (3), will convert a MODPATH7 MPUGRID file.

Appendix C: WriteP3DOutput.exe

mod-PATH3DU writes pathline points to a binary file. WriteP3DOutput.exe converts this output file to formats that are supported by GIS programs, i.e. DBF and SHP. The following lists the files it will write:

1. ASCII Table, an ASCII table of all the particle points in the binary file
2. DBF Table, a DBF formatted table of all the particle points in the binary file
3. Pathline Shapefile, a shapefile with a single poyline feature for each particle's path
4. Endpoint Shapefile, a shapefile with a single point feature for each particle point in the binary file. It is recommended this output type be used with the ENDPOINT simulation type.

In addition, there is a "Summary" type that will echo details of the file, including version, to the screen.

The program is executed from the command line as follows:

`writeP3DOUTPUT` *`FILENAME`*, or
`writeP3DOUTPUT` *`FILENAME colorcode`*

where *FILENAME* is the name of the required input file, detailed below, and must be the first input on the command line after the name of the writeP3DOUTPUT executable. Optionally, and recommended, users can supply *colorcode* after FILENAME which will color the text of the console as the program runs. This color coding is more aesthetic than a true codification, except when there is an error – errors, and only errors, are provided in red text.

The following examples are different input files that illustrate the input file:

```
{
  "MP3DU_BIN" : "MEUK_ PATHLINE.bin",
  "OUTPUTS" : [
   { "SUMMARY" : {
   } },
   { "ASCII_TABLE" : {
     "FILENAME" : "MEUK_TEST.dat"
   } },
   { "PATHLINE_WHOLE" : {
     "FILENAME" : "MEUK_TEST_ADVECT.shp",
     "MAX_TIME" : 0. ,
     "MIN_TIME" : 1000.
   } }
```

This input file includes 3 output types: SUMMARY, ASCII_TABLE, and PATHLINE_WHOLE. In this file, the output types are listed within a vector (square-braces) - any combination of types can be listed in any order. Each type is listed as an object, with a FILENAME parameter, that is the name of the file writeP3DOutput will write. PATHLINE_WHOLE has optional parameters MAX_TIME and MIN_TIME – when these are specified only the parts of a particle's path between these times are written to the shapefile.

] }	

{
  "MP3DU_BIN" : "MEUK_ ENDPOINT.bin",
  "OUTPUTS" : [
   { "SUMMARY" : {
   } },
   { "DBF_TABLE" : {
    "FILENAME" : "MEUK_TEST.dbf"
   } },
   { "ENDPOINT" : {
    "FILENAME" : "MEUK_TEST_ADVECT.shp"
   } }
  ]
}
	This input file includes 3 output types: SUMMARY, DBF_TABLE, and ENDPOINT. In this file, the output types are listed within a vector (square-braces), and thus any combination of types can be listed in any order. Each type is listed as an object, with a FILENAME parameter, that is the name of the file writeP3DOutput will write.