PATH3D

Revisions incorporated in Version 4.6

S.S. Papadopulos & Associates, Inc.
September 12, 2001

Several revisions and enhancements of PATH3D have been incorporated in the most recent version, designated as Version 4.6. The major revisions contained in Version 4.6 are listed below:

1. Allocatable memory for the Y array and TFRONT array;
2. Support for reading of true binary headsave files;
3. Expanded layer placement options;
4. More rigorous particle tracking options;
5. Support for reverse tracking in transient flow simulations;
6. Support for the tracking of individual particles over user-specified times or stress periods;
7. Identification and repair of two sources of potential error; and
8. Revision of the order in which input files are read by PATH3D.

The major revisions and fixes are described in detail in the following sections.

1. Allocatable Memory

In previous versions of PATH3D, the PARAMETERs LENY and NFMAX needed to be specified in advance. In situations where these parameters were exceeded, the source code needed to be altered and then recompiled. By making these arrays allocatable, we have eliminated the need to resize the Y array and TFRONT array when dealing with large models. The revised version of PATH3D can now take advantage of all the available RAM of individual users.

2. Binary Headsave Files

Previous versions of PATH3D could utilize only one form of binary file, the FORTRAN unformatted files created by the Lahey FORTRAN compilers. An option to read true binary headsave files has now been added. This enables PATH3D to work directly with the versions of MODFLOW that are included with the most popular graphical modeling environments.

2.1 Background

Binary files arise in two contexts. First, MODFLOW results are generally saved in binary files. Second, MODFLOW can be directed to read array input from external binary files. There are several formats for storing data in "binary" files. We consider only two: FORTRAN-unformatted files; and true binary files. The key difference between the two files formats is that FORTRAN unformatted files preserve a structure that includes carriage control (that is, end-of-line records). The FORTRAN-unformatted file is the traditional file structure in groundwater simulation, in the sense that this structure has been associated with MODFLOW since its inception. There are two important drawbacks of the FORTRAN-unformatted file structure: first, the file structure is compiler-dependent; second, graphical modeling environments that are not coded in FORTRAN do not always support it. Contemporary graphical modeling environments (for example Groundwater Vistas) instead store MODFLOW results in true binary files.

Previous versions of PATH3D have been able to read MODFLOW head files saved only as FORTRAN-unformatted files. We have developed an alternative version of MODFLOW that creates head files in true binary format. We have also developed an alternative version of PATH3D that can read these true binary head files. The modified version of PATH3D can also read parameter arrays from true binary files.

2.2 Modified version of PATH3D

SSP&A will maintain a version of PATH3D capable of reading both file formats.

1. FORTRAN unformatted format

OPEN(…,FORM='unformatted'); by default, ACCESS='sequential'

This version is capable of reading parameter arrays and MODFLOW head solutions from FORTRAN-unformatted files. These files have the following formats:

- parameter arrays that are saved in FORTRAN-unformatted files are preceded by a header line. The header line is of indeterminate length, it is separated from the array itself by a carriage return; and
- MODFLOW head solutions that are saved in FORTRAN-unformatted files are preceded by a header line with the following contents:
  kstp (I), kper (I), pertim (R), totim (R), text (C), ncol (I), nrow (I), ilay (I);
  where I denotes Integer*4, R denotes Real*4, and C denotes Character*16.

2. True binary format

OPEN(…,FORM='unformatted', ACCESS='transparent')
In fact, FORM='unformatted' is unnecessary. The statement ACCESS='transparent' is sufficient to identify the file as a true binary file.

This version is capable of reading parameter arrays and MODFLOW head solutions from true binary files.

- It is assumed that the parameter arrays that are saved in binary files are not preceded by anything.
- MODFLOW head solutions that are saved in binary are preceded by the following information:
  kstp (I), kper (I), pertim (R), totim (R), text (C), ncol (I), nrow (I), ilay (I);
  where I denotes Integer*4, R denotes Real*4, and C denotes Character*16.
  This preceding information comprises a total of 44 bytes.

2.3 Revised PATH3D programming

A new variable, FileFormat, and a SELECT CASE construct are added to subroutine OPENFL in order to facilitate future expansion of options:

```
Select Case (FileFormat)
     Case (1)        ! Unformatted Sequential
          FLFORM='UNFORMATTED'
          flaccess='sequential'
     Case (2)        ! Unformatted Transparent (BINARY)
          FLFORM='UNFORMATTED'
          flaccess='transparent'
     Case (3)        ! Formatted Sequential
          FLFORM='FORMATTED'
          flaccess='sequential'
```

2.4 Revised PATH3D prompts

When running the new version of PATH3D, the user now receives the following prompt:

```
Enter U if the head file is UNFORMATTED;
Enter B in the head file is TRUE BINARY;
Otherwise, enter format of the head file:
```

3. Expanded Particle Placement Options

In previous versions of PATH3D, when particle release was specified according to cell location (J,I,K) the only option was release from the center of a cell. We have incorporated additional options that were first added to the version of PATH3D that was included with ModIME (Version 4.00). Particles can now be placed at the top, middle, or bottom of a cell.

3.1 Definitions

INIPOS: code specifying the manner in which the starting locations of particles are specified.

NPCELL: the number of particles started from a particular location

KOPTION: the code specifying the vertical location within a cell from which the particle is released

3.2 INIPOS options

INIPOS =

1 Particle starting locations are assigned by specifying the x, y, and z coordinates of each individual particle. Coordinates are specified in terms of the PATH3D local coordinate system. The z coordinate is measured vertically downwards from HTOP(J=1, I=1).

A single particle is specified with each record in the particle tracking input file.
Record 4 of the particle tracking input file
X Y Z; Format: 3F10.0

2 Particle starting locations are assigned by specifying the column (J), row (I), and layer (K) indices of the finite difference cells in which particles are to be initialized.

The user can specify the number of particles that are released from the cell, with particle locations assigned according to either a fixed pattern or randomly.
The user can specify one of three vertical release positions; if nothing is specified, the particles are released from the mid-elevation of the cell.

Record 4 of the particle tracking input file
J I K NPCELL KOPTION; Format: 5I10

3 Particle starting locations are assigned by specifying the x and y coordinates, and the layer index (K) of each individual particle. The x and y coordinates are specified in terms of the PATH3D local coordinate system.

The user can specify one of three vertical release positions; if nothing is specified, the particle is released from the mid-elevation of the cell.

A single particle is specified with each record in the particle tracking input file.
Record 4 of the particle tracking input file:
X Y K KOPTION; Format: 2F10.0,2I10

3.3 NPCELL

The user can specify the release of multiple particles within a cell, by specifying INIPOS=2 and |NPCELL| > 1.

- NPCELL > 1
  particles are started from a circle at a fixed elevation in the cell. The elevation of the cell depends on the specification for KOPTION.

- NPCELL < 1
  particles are started from randomly generated locations in the cell (horizontally and vertically). The setting of KOPTION is in fact ignored.

- NPCELL = 1
  one particle placed at center of horizontal center of cell, vertical starting elevation set by KOPTION

3.4 KOPTION

KOPTION =

10001 Particle(s) released from top of cell
(HTOP-DZ*ZFAC, where ZFAC=0.01)

10002 Particle(s) released from middle of cell
(HTOP-DZ*0.5)

10003 Particle(s) released from bottom of cell
(HTOP-DZ*ZFAC, where ZFAC=0.99)

3.5 Adjustment of initial particles positions above the water table

A particle cannot be tracked from an initial location that is above the water table. If the specified starting elevation is above the water table, then the initial particle position may be adjusted internally:

- if the gradient is upwards, the particle will be removed automatically; and
- if the gradient is downwards, the initial particle position will be adjusted downwards so that the particle starts at the elevation of the water table.

For KOPTION=10001, particles are initially placed at the top of the cell, not at the water table. If the top of the cell is above the water table, then the particle is automatically moved downwards so that it starts at the water table.

For KOPTION=10002, particles are initially placed at the middle of the cell, not at the middle of the saturated thickness of the cell. If the middle of the cell is above the water table, then the particle is moved downwards so that it begins at the water table.

4. More Rigorous Particle Capture Options

We have noted some inconsistencies in the PATH3D particle capture algorithms. Specifically, with IOPSS=2 or 4, particles could be removed at a cell containing a RIV or STR, even if the feature were inactive or acting as a source. We have corrected this so that particles are removed only if the feature is acting as a sink. PATH3D also did not allow particles to be captured at General-Head Boundary (GHB) cells, although the GHB file was read. We have added support for GHBs.

A detailed re-interpretation of the particle tracking capture options, set with the parameter OPOSS is presented below.

IOPSS =

1 Particles that enter a cell containing a WEL, DRN, RIV, STR, or GHB are removed only if all gradients at the cell interfaces are inwards; the sink must be strong.

2 Particles that enter a cell containing a DRN, RIV, STR, or GHB are removed immediately if the feature is acting as a sink (i.e. Q < 0); the sink can be either strong or weak.
Particles that enter a cell containing a WEL are removed if the well is extracting water.

3 Particles that enter a cell containing a DRN, RIV, STR, or GHB are removed only if all gradients at the cell interfaces are inwards (same as IOPSS = 1).
Particles that enter a cell containing a WEL are removed only if they enter within the analytically-determined capture zone of the WEL (the well must be extracting water).

4 Particles that enter a cell containing a DRN, RIV, STR, or GHB are removed immediately if the feature is acting as a sink (same as IOPSS = 2).
Particles that enter a cell containing a WEL are removed only if they enter within the analytically-determined capture zone of the WEL (the well must be extracting water).

5. Reverse transient tracking

Previous versions of PATH3D supported forward and reverse tracking in steady flow fields, but only forward tracking in transient flow fields. PATH3D now supports reverse particle tracking in transient flow fields.

PATH3D has been modified extensively to support reverse particle tracking in transient flow fields. In particular, PATH3D has modified to now produce temporary files in which the headsave file and source/sink input files are read, and written out in reverse. The temporary files are named `$~temp.XXX`, where `XXX` is the corresponding file extension for the headsave or source/sink input file. For example, the temporary file based on the WEL package input file would be `$~temp.wel`. The temporary files are not automatically deleted by PATH3D, but may be deleted manually, if desired.

6. Transient release of particles

PATH3D has been extended to support the transient release of particles. A new input parameter ITIME is introduced into Record 1 of the particle tracking input file. This parameter allows one to specify either:

- a starting and ending stress period over which specific particles will be tracked; or
- a starting and ending time over which specific particles will be tracked.

If the option is not invoked, the particle tracking times revert to the TIME1 and TIME2 parameters of the particle tracking input file.

6.1 Definitions

ITIME: code specifying the manner in which the starting and ending times of particles are specified.

RELTIME: the release time for a specific particle or group of particles, in simulation time. Tracking starts at RELTIME.

ENDTIME: the ending time for a specific particle or group of particles, in simulation time. Tracking ends at ENDTIME.

RELPER: the release stress period for a specific particle or group of particles. Tracking starts at the beginning of stress period RELPER.

ENDPER: the release stress period for a specific particle or group of particles. Tracking ends at the end of stress period ENDPER.

6.2 ITME options

ITIME =

<0 The starting and ending stress periods for particle tracking will be specified in Record 4 of the particle tracking input file.

0 The starting and ending times for particle tracking will be controlled by parameters TIME1 and TIME2 of Record 2 in the particle tracking input file.

>0 The starting and ending times will be specified in Record 4 of the particle tracking input file.

The format of Record 4 will depend upon the INIPOS option specified in Record 1 of the particle tracking input file. A summary of the possible input options are given in the following table:

INIPOS	ITIME	Record 4	Format
1	-1	X Y Z RELPER ENDPER	3F10.0, 2I10
	0	X Y Z	3F10.0
	1	X Y Z RELTIME ENDTIME	5F10.0
2	-1	J I K NPCELL KOPTION RELPER ENDPER	7I10
	0	J I K NPCELL KOPTION	5I10
	1	J I K NPCELL KOPTION RELTIME ENDTIME	5F10.0, 2F10.0
3	-1	X Y K KOPTION RELPER ENDPER	2F10.0, 4I10
	0	X Y K KOPTION	2F10.0, 2I10
	1	X Y K KOPTION RELTIME ENDTIME	2F10.0, 2I10, 2F10.0

In order to ensure backward compatibility, PATH3D will function as previously if the ITIME parameter is omitted from Record 1 of the particle tracking input file. That is, the TIME1 and TIME2 parameters will control the particle tracking time for all particles (same as ITIME=0 option).

7. Additional modifications

7.1 Repair of a potential "Divide by Zero" problem

The following code was generating a divide by zero error when compiled with LF90. The statement is really two conditions, either of which may test true: NSTP=1 OR (ISAV>0 AND MOD(NSTP,ISAV)=0). However, even when ISAV=0, MOD(NSTP,ISAV) was always evaluated, leading to the divide by zero error.

```
C
C--STORE INTERMEDIATE RESULTS IF REQUIRED
C--(FIRST STEP IS ALWAYS SAVED)
      IF(NSTP.EQ.1.OR.ISAV.GT.0.AND.MOD(NSTP,ISAV).EQ.0) THEN
        IF(IPRT.EQ.1) THEN
```

The fix consists of ensuring that ISAV>0 before executing the modulus:

```
C
C--STORE INTERMEDIATE RESULTS IF REQUIRED
C--(FIRST STEP IS ALWAYS SAVED)
      if (isav.gt.0) then                                  !MAK
          modcheck=MOD(NSTP,ISAV)                          !MAK
      endif                                                !MAK
      IF(NSTP.EQ.1.AND.modcheck.EQ.0) THEN
        IF(IPRT.EQ.1) THEN
```

7.2 Improvement of tracking accuracy in heterogeneous formations

Chunmiao Zheng has alerted us to potential inaccuracies in particle tracking in heterogeneous formations with large contrasts in hydraulic conductivity. Since PATH3D's adaptive stepsize control is based on *DT* only, it is possible that when a particle moves into a high K cell from a low K cell, *DT**velocity is so large that the particle's next position would be out of model boundary, leading to some problems. The lines below (in lower cases) can be added to prevent a particle from moving more than one cell in one tracking step even if the error criterion is satisfied.

```
C
C--IF TRUNCATION ERROR TOO LARGE, REDUCE STEPSIZE
      IF(ERRMAX.GT.1.) THEN
        DT=SAFETY*DT/ERRMAX**0.2
        GOTO 1

c--else if particle moves more than one cell, cut stepsize by half
      elseif(abs(p(1)-psav(1)).gt.delr(jpsav).or.
     &       abs(p(2)-psav(2)).gt.delc(ipsav).or.
     &       abs(p(3)-psav(3)).gt.dz(jpsav,ipsav,kpsav) ) then
        dt=dt/2.
        goto 1

C
C--ELSE IF STEP SUCCEEDED, COMPUTE SIZE OF NEXT STEP
      ELSE
        IF(ERRMAX.GT.0) DTNEXT=SAFETY*DT/ERRMAX**0.2
        IF(ERRMAX.EQ.0.OR.ABS(DTNEXT).GT.ABS(4.*DT)) DTNEXT=4.*DT
      ENDIF
```

8. Revision of the order in which Packages are read

The order in which PATH3D reads MODFLOW packages was altered in going from Version 3.2 to Version 4.00 (version included with ModIME). The reason for this is unknown. However, the order in which packages are read in Version 4.5 has been reverted back to the original order used in Version 3.2. The following table summarizes the package order:

Version 3.2 / Version 4.5	Version 4.00
BAS	BAS
BCF	BCF
WEL	WEL
DRN	DRN
RIV	RIV
STR	STR
RCH	EVT
EVT	GHB
GHB	RCH