**Applied Hydrogeology Practical Weeks 10 and 11**

**PART 3: MODPATH**

MODPATH is a post-processing model that uses the output from the flow simulator to compute 3D the pathway of individual particles. If you give an initial, starting position for the particle, then you can use numerical techniques to track the particle in 3D over a series of discrete time steps. A particle's path is tracked from one cell to the next until it reaches a boundary, an internal sink/source, or satisfies some other termination criterion. Or you can specify an end point and MODPATH can be used to track a particle back to the source.

Particle tracking helps you to visualize a flow field, detect errors in the flow model and check mass balances for individual cells or for regions. Applications of particle tracking include:

* capture zones in order to specify wellhead protection areas, and calculate strategies for hydraulic control of contaminant plumes
* time of travel to calculate arrival of contaminants, time to clean up and to estimate recharge
* model calibration by comparison with isotope data on groundwater ages.

MODPATH uses a semi-analytical solution to this equation and linear interpolation between point data. We only know head at specific points at specific times; linear interpolation is used to obtain velocity at other points and times.

Advantages of particle tracking:

1. Can determine pathlines in spatially variable parameter domains
2. Can determine three-dimensional pathlines
3. Compute forward and reverse-tracked pathlines
4. Confine, leaky, and unconfined aquifers

It is important to remember that pathlines are not flow nets - they are not solutions to the Laplace equation and there is not equal amounts of flow between individual path lines.

**Q4.** Write a short paragraph giving several reasons why travel times calculated using particle tracking might be shorter than those derived from a field-scale tracer test carried out using an unreactive tracer.

MODPATH input data consists of a combination of MODFLOW input data files, MODFLOW head and flow output files, and other input files specific to MODPATH. MODPATH produces several output files, including a number of particle coordinate output files. These can be used as input data for other programs that process, analyze, and display the results in various ways.

We will need a new executable, the MODPATH engine which is called called mp6, to run these simulations. The processes getting this executable are the same as when we got the first mf2005 executable.

**MAC**

**cd** into the pymake directory (wherever you cloned it to last time)

**cd examples**

**python make\_modpath6.py**

This then creates the mp6 executable in the temp folder. Move it to the place (folder in Applications most likely) where you stored mf2005.

**Windows**

<https://water.usgs.gov/ogw/modpath/>

Download **Version 6** (listed under additional resources) – **DO NOT download the latest release version 7**

Go to the bin

Copy the file mp6 to a folder somewhere locally. I’d recommend putting it a folder in your applications folder or perhaps inside of a folder inside of the modflow folder

**MODPATH Model Creation**

In a new cell **add the following code;**

mp = flopy.modpath.Modpath(modelname='Modflowmod', exe\_name='/YOUR/PATH/TO/mp6',

modflowmodel=mf,

dis\_file='threelayer.dis',

head\_file='threelayer.hds',

budget\_file='threelayer.cbc')

In a new cell bellow, convert to markdown (click on left hand side of cell till it turns blue and press ‘m’), and briefly describe what each of the of the arguments is doing. Try and format your answer using headings (#) so that it is easily readable.

In a new cell **add the following code;**

mpb = flopy.modpath.ModpathBas(mp, hdry=mf.lpf.hdry, laytyp=mf.lpf.laytyp, ibound=1, prsity=0.05) #note prsity = effective porosity

sim = mp.create\_mpsim(trackdir='forward', simtype='pathline', packages='RCH', start\_time=(2, 0, 99.))

mp.write\_input()

mp.run\_model(silent=True)

In a new cell below, convert to markdown, and briefly describe how the arguments to ModpathBas and create\_mpsim are working.

In a new cell **add the following code;**

epobj = flopy.utils.EndpointFile('ModflowMod.mpend')

well\_epd = epobj.get\_destination\_endpoint\_data(dest\_cells=[**(X, X, X)])**

Input an appropriate location into the values for **X**

Print well\_epd - If it is blank you probably have done something wrong!

In a new cell **add the following code;**

fig = plt.figure(figsize=(14, 14))

ax = fig.add\_subplot(1, 1, 1, aspect='equal')

modelmap = flopy.plot.ModelMap(model=mf, layer=2)

quadmesh = modelmap.plot\_ibound()

linecollection = modelmap.plot\_grid()

quadmesh = modelmap.plot\_bc('WEL', kper=2, plotAll=True)

contour\_set = modelmap.contour\_array(hds,

levels=np.arange(np.min(hds),np.max(hds),0.5), colors='b')

plt.clabel(contour\_set, inline=1, fontsize=14)

modelmap.plot\_endpoint(well\_epd, direction='starting', colorbar=True)

We can also add particle pathlines

In a new cell **add the following code;**

pthobj = flopy.utils.PathlineFile('ModflowMod.mppth')

well\_pathlines = pthobj.get\_destination\_pathline\_data(dest\_cells=[(2, 14, 14)])

In a new cell **add the following code;**

fig = plt.figure(figsize=(14, 14))

ax = fig.add\_subplot(1, 1, 1, aspect='equal')

modelmap = flopy.plot.ModelMap(model=mf, layer=0)

quadmesh = modelmap.plot\_ibound()

linecollection = modelmap.plot\_grid()

quadmesh = modelmap.plot\_bc('WEL', kper=1, plotAll=True)

#contour\_set = modelmap.contour\_array(hds,

# levels=np.arange(np.min(hds),np.max(hds),0.5), colors='b')

plt.clabel(contour\_set, inline=1, fontsize=14)

modelmap.plot\_endpoint(well\_epd, direction='starting', colorbar=True)

for pid in np.unique(well\_pathlines.particleid):

modelmap.plot\_pathline(pthobj.get\_data(pid), layer='all', colors='red');

Finally we can define areas by travel time.

In a new cell **add the following code;**

fig = plt.figure(figsize=(14, 14))

ax = fig.add\_subplot(1, 1, 1, aspect='equal')

modelmap = flopy.plot.ModelMap(model=mf, layer=2)

quadmesh = modelmap.plot\_ibound()

linecollection = modelmap.plot\_grid()

quadmesh = modelmap.plot\_bc('WEL', kper=2, plotAll=True)

modelmap.plot\_endpoint(well\_epd, direction='starting', colorbar=True)

a=np.full(hds.shape, np.nan)

for i in well\_epd:

a[i[6],i[7],i[8]]=i[4]

travtime\_set=modelmap.contour\_array(a[0],levels=np.arange(0,1000,100))

plt.clabel(travtime\_set, inline=1, fontsize=10)

You can change the arguments to np.arange of 100 to derive contours with greater or smaller contour intervals and 1000 to expand the max value if necessary. I.e. its np.arange(start,stop,interval). np.arange is generating an numpy array. You can also develop a list in place of np.arange() if you would like a truly custom contour solution.

The environment agency is interested defining source protection zones (SPZs) based on groundwater travel times.

**Zone 1** is closest to the well showing the area of highest risk and defined by 50-day travel time from any pollution below the water table to the groundwater source (minimum for decay of some biological contaminants).

**Zone 2** is defined by a 400-day travel time (minimum recharge area required to support 25% of the protected yield).

**Zone 3** is the area within which all groundwater recharge (from precipitation or surface water) is presumed to be discharged at the well - covers the whole of the catchment area of a well based on the area needed to maintain abstraction, assuming that all water will eventually reach the abstraction point. Designed to avoid the long-term degradation of the quality of water in the aquifer.

**Q5.** For your best fit model , generate a map of the source protection zones 1, 2 and 3 around your well using a porosity of 0.05. Explain and illustrate the sensitivity of travel time to the specified porosity.

**PART 4: Running tests using MODPATH**

After 10 years of pumping good quality water from the well, an underground storage tank full of inorganic contaminant E-VIL located 1500 m due west (up-hydraulic-gradient) of the pumping well started to leak. The base of the storage tank sits immediately above the water table. As the contaminant passes through the groundwater system a range of different chemical and biochemical processes act to transform, retard, and attenuate solutes. Sorption, which controls the distribution of mass between the solid and solution phase, was determined in the laboratory using rock material from all three aquifer layers. Experimental data is summarised in Table 3.

Table 3. Experimental data on sorbtion of the contaminant which was spilled onto a

|  |  |  |
| --- | --- | --- |
| Aquifer Layer | Equilibrium concentration of E-VIL mg/L | Contaminant (E-VIL) sorbed ug/g |
| 1 | 0.08 | 13 |
| 1 | 0.10 | 17 |
| 1 | 0.27 | 46 |
| 1 | 0.40 | 67 |
| 2 | 0.53 | 86 |
| 2 | 0.67 | 105 |
| 3 | 0.20 | 31 |
| 3 | 0.32 | 51 |

**Q6a.** Plot and describe the form of the isotherm, and calculate the distribution coefficient Kd (L/kg) from the mass of solute sorbed/dry weight of solid and the concentration of solute in equilibrium with mass of solute sorbed on solid.

**Q6b** Use the distribution coefficient derived from the experimental data in Table 3 to calculate the retardation factor, *rf*

where

d = bulk density of the aquifer material is 1600 mg/cm3  
 = porosity   
Kd = partition coefficient

**Q6c.** Using your retardation factor and the equation below, estimate how long the contaminant will take to reach the pumping well. When would you start monitoring the well for the appearance of E-VIL? Justify your recommendation.

**BONUS POINTS: Using your MODFLOW model, develop a strategy (or strategies) which you could present to the Environment Agency for remediating the effect of the E-VIL contaminant – answer in word document as BONUS 2**