Tutorial 2: Unconfined Transient Flow Model

In this example, we will convert the tutorial 1 model into an unconfined, transient flow model with time varying boundaries. Instead of using constant heads for the left and right boundaries (by setting ibound to -1), we will use general head boundaries. . We will have the model run the following simulation.

* Initial conditions – head is 10.0 everywhere
* Period 1 (1 day) – steady state with left and right GHB stage = 10.
* Period 2 (100 days) – left GHB with stage = 10., right GHB with stage set to 0.
* Period 3 (100 days) – pumping well at model center with rate = -10., left and right GHB = 10., and 0.

**Discretization**

**Start by changing the modelname**

* Change it to something more appropriate.

We keep the model dimensions the same i.e. one layer, 50m thick etc but have to add some information about time periods.

**New time step parameters**

*To create a model with multiple stress periods, we need to define nper, perlen, nstp, and steady. This is done in the following block in a manner that allows us to pass these variable directly to the discretization object:*

Copy and paste the following block into a new cell

nper = 3

perlen = [1, 100, 100]

nstp = [1, 100, 100]

steady = [True, False, False]

Relevant documentation <https://modflowpy.github.io/flopydoc/mfdis.html>

What do each of these parameters mean? Answer as comments adjacent (no more than 15 words per comment!)

* Add the time step parameters to the discretization object

**Basic Package**

* Set ibound to equal 1 for every cell – hint you don’t need to add anything!
* Set strt to equal 10 for every cell

Relevant documentation <https://modflowpy.github.io/flopydoc/mfbas.html>

**Layer-Property flow package**

**Add in some new parameters for the unconfined aquifer**

* Change hydraulic conductivity to 1
* Change the vertical hydraulic conductivity to 1
* Add a non-default value of specific yield 0.1
* Add a non-default value of specific storage 1.e-4
* Add a non-default of layer type 1 to make unconfined

Relevant documentation <https://modflowpy.github.io/flopydoc/mflpf.html>

**Transient General-Head Boundary Package –New Addition**

**Copy and paste the following into a new cell**

*# Make list for stress period 1*

stageleft = 10. #General head on the left boundary

stageright = 10. #General head on the right boundary

bound\_sp1 = []

**for** il **in** range(nlay):

condleft = hk \* (stageleft - zbot) \* delc #conductance on the left boundary

condright = hk \* (stageright - zbot) \* delc #conductance on the right boundary

**for** ir **in** range(nrow):

bound\_sp1.append([il, ir, 0, stageleft, condleft])

bound\_sp1.append([il, ir, ncol - 1, stageright, condright])

print ('Adding ', len(bound\_sp1), 'GHBs for stress period 1.')

*# Make list for stress period 2*

stageleft = 10.

stageright = 0.

condleft = hk \* (stageleft - zbot) \* delc

condright = hk \* (stageright - zbot) \* delc

bound\_sp2 = []

**for** il **in** range(nlay):

**for** ir **in** range(nrow):

bound\_sp2.append([il, ir, 0, stageleft, condleft])

bound\_sp2.append([il, ir, ncol - 1, stageright, condright])

print ('Adding ', len(bound\_sp2), 'GHBs for stress period 2.')

*# We do not need to add a dictionary entry for stress period 3.*

*# Flopy will automatically take the list from stress period 2 and apply it to the end of the simulation, if necessary.*

stress\_period\_data = {0: bound\_sp1, 1: bound\_sp2}

*# Create the flopy ghb object*

ghb = flopy.modflow.ModflowGhb(mf, stress\_period\_data=stress\_period\_data)

Relevant documentation <https://modflowpy.github.io/flopydoc/mfghb.html>

**Have a look at the stress\_period\_data variable – does it make sense to you?**

### **Well package - New Addition**

Remember to use zero-based layer, row, column indices!

Copy and paste the following block into a new cell:

pumping\_rate = -100. # The abstraction must be negative number

wel\_sp1 = [[0, nrow/2 - 1, ncol/2 - 1, 0.]] # [lay, row, col, flux]

wel\_sp2 = [[0, nrow/2 - 1, ncol/2 - 1, 0.]] # [lay, row, col, flux]

wel\_sp3 = [[0, nrow/2 - 1, ncol/2 - 1, pumping\_rate]]

stress\_period\_data = {0: wel\_sp1, 1: wel\_sp2, 2: wel\_sp3}

wel = flopy.modflow.ModflowWel(mf, stress\_period\_data=stress\_period\_data)

Relevant documentation <https://modflowpy.github.io/flopydoc/mfwel.html>

**Again do you understand what is going on here?**

**Create the output control package object**

**Going to add stress period data to the output**

spd = {(0, 0): ['print head', 'save head', 'print budget', 'save budget']}

oc = flopy.modflow.ModflowOc(mf,stress\_period\_data=spd)

Relevant documentation <https://modflowpy.github.io/flopydoc/mfoc.html>

**Leave the fields about Preconditioned Conjuagte gradient package and writing the modflow data files alone**

## **Running the Modeling**

*Added some flags that can be turned on or off at the users liking for error checking*

success, mfoutput = mf.run\_model(silent=**True**, pause=**False**, report=**True**)

**if** **not** success:

**raise** Exception('MODFLOW did not terminate normally.')

## **Post-Processing the Results**

Once again, we can read heads from the MODFLOW binary output file, using the [flopy.utils.binaryfile](http://modflowpy.github.io/flopydoc/binaryfile.html) module. Included with the HeadFile object are several methods that we will use here: \* get\_times() will return a list of times contained in the binary head file \* get\_data() will return a three-dimensional head array for the specified time \* get\_ts() will return a time series array [ntimes, headval] for the specified cell

Using these methods, we can create head plots and hydrographs from the model results.:

**Copy and paste the following BUT add in functions to the location marked XXXX to print out the min, man and std for HEAD.**

**#Import the libraries**

import matplotlib.pyplot as plt

import flopy.utils.binaryfile as bf

***# Create the headfile object***

headobj = bf.HeadFile(modelname+'.hds')

times = headobj.get\_times()

***# Setup contour parameters***

levels = np.arange(1, 10, 1) # same as tutorial 1

extent = (delr/2., Lx - delr/2., delc/2., Ly - delc/2.) # same as tutorial 1

print ('Levels: ', levels)

print ('Extent: ', extent)

***# Well point***

wpt = ((float(ncol/2)-0.5)\*delr, (float(nrow/2-1)+0.5)\*delc)

wpt = (450., 550.)

***# Make the plots***

mytimes = [1.0, 101.0, 201.0]

for iplot, time in enumerate(mytimes):

print ('\*\*\*\*\*Processing time: ', time)

head = headobj.get\_data(totim=time)

***#Print statistics***

print ('Head statistics')

print (' min: ', **XXXX** )

print (' max: ', **XXXX** )

print (' std: ', **XXXX** )

***#Create the plot***

***#plt.subplot(1, len(mytimes), iplot + 1, aspect='equal')***

plt.subplot(1, 1, 1, aspect='equal')

plt.title('stress period ' + str(iplot + 1)) # plot title

plt.imshow(head[0, :, :], extent=extent, cmap='BrBG', vmin=0., vmax=10.)

plt.colorbar() # plot colourbar

CS = plt.contour(np.flipud(head[0, :, :]), levels=levels, extent=extent,

zorder=10)

plt.clabel(CS, inline=1, fontsize=10, fmt='%1.1f', zorder=11) # plot contour label

mfc = 'None'

if (iplot+1) == len(mytimes):

mfc='black'

plt.plot(wpt[0], wpt[1], lw=0, marker='o', markersize=8,

markeredgewidth=0.5,

markeredgecolor='black', markerfacecolor=mfc, zorder=9)

plt.text(wpt[0]+25, wpt[1]-25, 'well', size=12, zorder=12)

plt.show()

plt.show() # same as tutorial1

***# Plot the head versus time***

idx = (0, nrow/2 - 1, ncol/2 - 1)

ts = headobj.get\_ts(idx)

plt.subplot(1, 1, 1)

ttl = 'Head at cell (*{0}*,*{1}*,*{2}*)'.format(idx[0] + 1, idx[1] + 1, idx[2] + 1)

plt.title(ttl)

plt.xlabel('time')

plt.ylabel('head')

plt.plot(ts[:, 0], ts[:, 1])

plt.show()

Relevant documentation <https://modflowpy.github.io/flopydoc/binaryfile.html>

**Answer the following questions**

1. How does varying the following affect the model?
   * Specific yield
   * Specific storage
   * Hydraulic conductivity/Vertical hydraulic conductivity
   * Show this with various figures or tables with 2-3 sentence descriptions
2. SUPER BONUS POINTS – Modify the code with a time-varying pumping rate