Python Workshop
How we use python

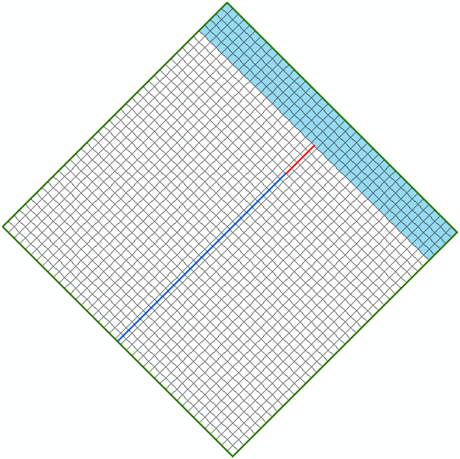
Joseph D. Hughes

U.S. Geological Survey
Florida Water Science Center, Tampa, Florida USA

USGS National Groundwater Workshop, August 2012

Create GHB file from raw data (1)

[xoff,yoff,rot]=[1000,1000,45]



Create GHB file from raw data (2)

CreateCoastalGHB.py

```python
import numpy as np
import MFArrayUtil as au
#--function for calculating equivalent freshwater head
def eqfwh( rho, h, z ):
    return ( rho / 1000. ) * h - ( ( rho - 1000. ) / 1000. ) * min( h, z )
#--main script
#--spatial dimensions
nlay, nrow, ncol  = 1, 41, 40
dx, dy = 500.0, 500.0 #m
#--temporal dimensions
nper = 3
#--boundary condition data
icbc, start_coast = 0, 36
kh    = 10.000 #m/d
cond  = kh * dx * dy
#--read MODFLOW data from external files
ibound  = au.loadArrayFromFile(nrow,ncol,'..\\ref\\ibound.ref')
top     = au.loadArrayFromFile(nrow,ncol,'..\\ref\\top.ref')
```

Create GHB file from raw data (3)

CreateCoastalGHB.py

```
#--ghb dataset
nghb = 0
for ir in range(0,nrow):
    for ic in range(0,ncol):
        if ibound[ir,ic] == 2:
            nghb += 1
f = open('Model.ghb','w')
#--dataset 0
f.write( '#Coastal Aquifer GHB Package\n' )
#--dataset 1
f.write( '{0:10d}{1:10d}  NOPRINT\n'.format( nghb, icbc ) )
#--write header for ghb file -- stress period 1
f.write( '{0:10d}         0          #STRESS PERIOD {1:05d}\n'.format( nghb, 1 ) )
for ir in range(0,nrow):
    for ic in range(0,ncol):
        if ibound[ir,ic] == 2:
            f.write( ' {0:9d} {1:9d} {2:9d} {3:9.5f} {4:9.3g}\n'.format(nlay,ir+1,ic+1, \
                     eqfwh( 1025., 0.0, top[ir,ic]),cond ) )
#--reuse data for the remaining stress period(s)
for iper in range(1,nper):
    f.write( '{0:10d}         0          #STRESS PERIOD {1:05d}\n'.format(-1,iper+1) )
f.close()
```

Output GHB file from raw data (4)

Model.ghb

```
#Coastal Aquifer GHB Package
       205         0  NOPRINT
       205         0          #STRESS PERIOD 00001
         1         1        36   0.00000   2.5e+06
         1         1        37   0.05000   2.5e+06
         1         1        38   0.10000   2.5e+06
         1         1        39   0.15000   2.5e+06
         1         1        40   0.20000   2.5e+06
         1         2        36   0.00000   2.5e+06
         1         2        37   0.05000   2.5e+06
         1         2        38   0.10000   2.5e+06
         1         2        39   0.15000   2.5e+06
         1         2        40   0.20000   2.5e+06

         1        41        36   0.00000   2.5e+06
         1        41        37   0.05000   2.5e+06
         1        41        38   0.10000   2.5e+06
         1        41        39   0.15000   2.5e+06
         1        41        40   0.20000   2.5e+06
        -1         0          #STRESS PERIOD 00002
        -1         0          #STRESS PERIOD 00003
```

Extract heads to create a new initial head array

extractSteadyHead.py

```
import numpy as np
import MFBinaryClass as mfb
#--problem size
nlay, nrow, ncol = 1, 41, 40
nper, nstp = 1, 100
#--name of MODFLOW head file
head_file = '..\\Results.SWI\\CoastalAquifer.hds'
#--read head data
#--create instance of head object from MFBinaryClass
headObj = mfb.MODFLOW_Head(nlay,nrow,ncol,head_file)
#--read array
totim,kstp,kper,h,success = headObj.get_record(nstp,nper)
#--save array or print error message
if success:
    np.savetxt('..\\ref\\steady_ihead.ref',h[0,:,:])
else:
    print 'Could not read Stress Period {0} Time Step {1}\n  from {2}'.format(nper,nstp,\
                                                                               head_file)
```

Binary head data

plotHeads.py

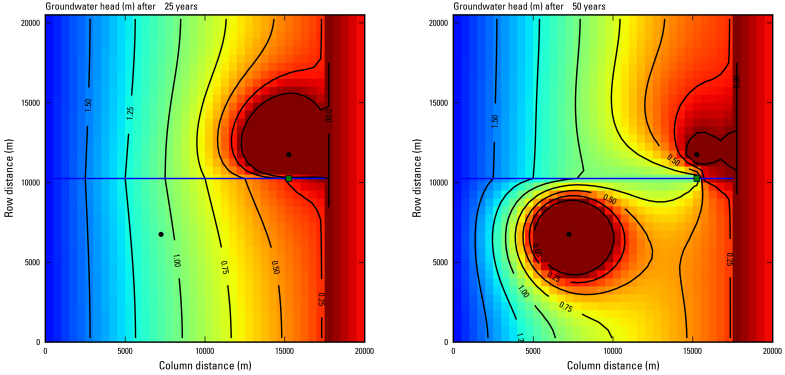
```
#--get available times
headObj = mfb.MODFLOW_Head(nlay,nrow,ncol,head_file)
t = headObj.get_gage(1)
ntimes = t.shape[0]
mf_times = np.zeros( (ntimes), np.float )
for i in range(0,ntimes):
    mf_times[i] = t[i,0]

#--create figures for each output time
for ipos,on_time in enumerate( mf_times ):
    #--build output file name
    output_name = '{0}{1}_{2:05d}.{3}'.format(base_dir,base_name,int(ipos),extension)
    fnames.append( output_name )
    #--read head data
    headObj = mfb.MODFLOW_Head(nlay,nrow,ncol,head_file)
    totim,kstp,kper,h,success = headObj.get_record(on_time)
    hd = np.copy( h[0,:,:] )
```

Figures (1)

plotHeads.py

```
    hp = ax.pcolor(Xedge,Yedge,hd,\
                   vmin=0,vmax=2,cmap='jet_r',alpha=1.0,edgecolors='None')
    ch = ax.contour(xcell,ycell,hd,\
                    levels=hdcontour,colors='k',linewidths=1)
    ax.clabel(ch,inline=1,fmt='%5.2f',fontsize=6)
    ax.plot([xedge[0],xedge[35]],[ycell[20],ycell[20]],linewidth=1,color='b',label='River')
    ax.plot(xcell[struct_loc[1]],ycell[struct_loc[0]],'gs',markersize=4,label='Structure')
    ax.plot(xcell[well_loc[0,1]],ycell[well_loc[0,0]],'ko',markersize=3,label='PW-1')
    ax.plot(xcell[well_loc[1,1]],ycell[well_loc[1,0]],'ko',markersize=3,label='PW-2')
```

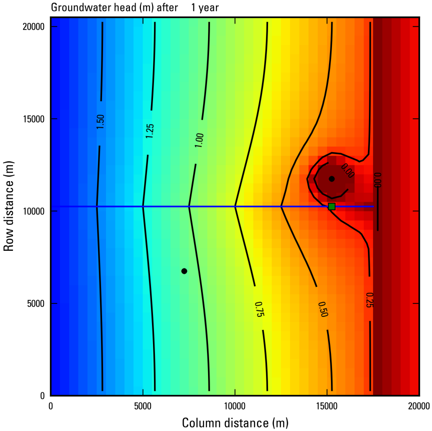
Figures (2)


Groundwater head (m) after 25 years
Row distance (m)
Column distance (m)
0
5000
10000
15000
20000
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Groundwater head (m) after 50 years
0.50
0.25
0.00
1.00
0.75
1.2

Using `ffmpeg.exe`

```
#--animate head data
coutf = '{0}{1}.swf'.format(base_dir,base_name)
cline = 'ffmpeg.exe -i {0}{1}_%05d.png {2} -y'.format( base_dir,base_name,coutf )
try:
    os.remove(coutf)
except:
    print 'could not remove...{0}'.format( coutf )
subprocess.call(cline, stdin=None, stdout=None, stderr=None, shell=False)
```

Using `ffmpeg.exe`



Making shapefiles (1)

`makeShapefileFromDIS.py`

```
import numpy as np
import shapefile
import MFArrayUtil as au

filename = '..\\data\\CoastalAquifer.dis'
offset,nlay,nrow,ncol,delr,delc = load_dis_file(filename)
#--flip the delc since we moved the orgin to lower left
delc = np.flipud( delc )
#--sum the lengths along the distance vectors
delr_cum = np.cumsum(delr)
delc_cum = np.cumsum(delc)
#--flip the delc since we moved the origin to lower left
delc_cum = np.flipud( delc_cum )
#--insert '0' in the first position
xedge = np.hstack((0,delr_cum))
yedge = np.hstack((delc_cum,0))
#--read MODFLOW data from external files
ibound  = au.loadArrayFromFile(nrow,ncol,'..\\ref\\ibound.ref')
top     = au.loadArrayFromFile(nrow,ncol,'..\\ref\\top.ref')
```

Making shapefiles (2)

makeShapefileFromDIS.py

```
#--create polygon shapefile of grid
wr = shapefile.Writer()
wr.field('row',fieldType='N',size=20)
wr.field('column',fieldType='N',size=20)
wr.field('delx',fieldType='N',size=20)
wr.field('dely',fieldType='N',size=20)
wr.field('cellnum',fieldType='N',size=20)
wr.field('ibound',fieldType='N',size=20)
wr.field('elev_m',fieldType='N',size=16,decimal=7)
#--create each polygon
cell_count = 0
for ir in range(0,nrow):
    for ic in range(0,ncol):
        #--calc the box points relative to the grid
        lowleft = [xedge[ic],yedge[ir]]
        lowright = [xedge[ic+1],yedge[ir]]
        upright = [xedge[ic+1],yedge[ir+1]]
        upleft = [xedge[ic],yedge[ir+1]]
        closeit = [xedge[ic],yedge[ir+1]-0.0001]
        this_box = [upleft,upright,lowright,lowleft,closeit]
```

Making shapefiles (3)

makeShapefileFromDIS.py

```
        #--if rotation is non-zero
        if offset[2] != 0.0:
            this_box = rotate(this_box,offset[2])
        #--add the offset in after the rotation
        this_box = add_offset(this_box,offset)
        ibt    = ibound[ir,ic]
        ttop   = top[ir,ic]
        wr.poly(parts=[this_box], shapeType=5)
        wr.record([ir+1,ic+1,delc[ir],delr[ic],cell_count+1,ibt,ttop])
        cell_count += 1
#--save polygon shapefile
wr.save(target='..\\data\\CoastalAquifer_grid')
```