Enhanced FloPy scripts for Constructing and Running MODFLOW-Based Models

Mark Bakker[1], Vincent Post[2], Joseph Hughes[3], Christian Langevin[4], Alain Francés[5], Jeremy White[6]
[1]*Delft University of Technology, mark.bakker@tudelft.nl, Delft, The Netherlands*
[2]*Flinders University, vincent.post@flinders.edu.au, Adelaide, Australia*
[3]*U.S. Geological Survey, jdhughes@usgs.gov, Tampa, FL, USA*
[4]*U.S. Geological Survey, langevin@usgs.gov, Reston, VA, USA*
[5]*University of Twente, frances.alain@gmail.com, Enschede, The Netherlands*
[6]*U.S. Geological Survey, jwhite@usgs.gov, Tampa, FL, USA*

ABSTRACT

The FloPy package is a community project that is being developed by volunteers. It consists of a set of scripts to run MODFLOW, MT3DMS, SEAWAT, SWI, PHT3D, and other MODFLOW-related groundwater programs. The package is written in Python, a powerful scripting language that is free, easy to learn, and rapidly becoming popular among hydrologists. Input scripts are confined to a single, readable file, which makes it easy to build, run, compare, reproduce, and archive models. FloPy is functional for most MODFLOW Packages, but it is still under development. Those interested in FloPy can download the code or binary installer for PC or Mac from flopy.googlecode.com.

INTRODUCTION

The FloPy project started a few years ago — a FloPy example was presented on a poster at the MODFLOW and More 2011 conference. The latest version contains several enhancements over the 2011 version: (1) arrays are now defined in the MODFLOW standard form (nlays,nrows,ncols), (2) the code is redesigned to improve memory management and handle larger models, (3) online documentation is available for all modules, (4) indexing is implemented for post processing of binary output files so that large files can be processed in an efficient manner, (5) a number of examples have been developed using the iPython Notebooks, and (6) default values for many input parameters are set in a configuration file that can be customized by the user.

DEMONSTRATION BY EXAMPLE

FloPy's capabilities are presented through an example. Consider a square model with a specified head equal to `h1` along all boundaries. The head at the cell in the center in the top layer is fixed to `h2`. First, set the name of the model and the parameters of the model: the number of layers `Nlay`, the number of rows and columns `N`, the lengths of the sides of the model `L`, the aquifer thickness `H`, and the hydraulic conductivity `k`

```
name = 'mf2013_example'
h1 = 100     # Head along boundary of model (m)
h2 = 90      # Head at center cell (m)
Nlay = 10    # Number of layers
N = 101      # Number of cells in x and y direction
L = 400.0    # Length of model (m)
H = 50.0     # Aquifer thickness (m)
k = 1.0      # Hydraulic conductivity (m/d)
```

Create a MODFLOW model and store it (in this case in the variable `ml`, but you can call it whatever you want). The modelname will be the name given to all MODFLOW files (input and output). The `exe_name` should be the full path to your MODFLOW executable. The version is either `'mf2k'` for MODFLOW2000 or `'mf2005'` for MODFLOW2005.

```
ml = Modflow(modelname=name, exe_name='/your_path/mf2005', version='mf2005')
```

Define the discretization of the model. In this example, all layers are given equal thickness. The `bot` array consists of the bottom elevations of all layers, and `delrow` and `delcol` are computed from model size `L` and number of cells `N`. Once these are all computed, the Discretization and the LPF packages are entered.

```
bot = linspace(-H/Nlay,-H,Nlay)
delrow = delcol = L/(N-1)
ModflowDis(ml,nlay=Nlay,nrow=N,ncol=N,delr=delrow,delc=delcol,top=0.0,botm=bot
,laycbd=0)
ModflowLpf(ml,hk=k)
```

Next we specify the boundary conditions and starting heads with the Basic package. The `ibound` array is `1` in all cells in all layers, except for along the boundary and in the cell at the center in the top layer where it is set to `-1` to indicate fixed heads. The starting heads are used to define the heads in the fixed head cells. This is a steady simulation, so none of the other starting values matter. The starting heads are set to `h1` everywhere, except for the head at the center of the model in the top layer, which is set to `h2`.

```
Nhalf = (N-1)/2
ibound = ones((Nlay,N,N))
ibound[:,0,:] = -1; ibound[:,-1,:] = -1; ibound[:,:,0] = -1; ibound[:,:,-1]=-1
ibound[0,Nhalf,Nhalf] = -1
start = h1 * ones((N,N))
start[Nhalf,Nhalf] = h2
ModflowBas(ml,ibound=ibound,strt=start)
```

Finally, we need to specify the solver we want to use (PCG with default values), and the output control (using the default values). Then we are ready to write all MODFLOW input files and run MODFLOW.

```
ModflowPcg(ml)
ModflowOc(ml)
ml.write_input()
ml.run_model3()
```

```
                                  MODFLOW-2005-SWI
    U.S. GEOLOGICAL SURVEY MODULAR FINITE-DIFFERENCE GROUND-WATER FLOW MODEL
                             Version 1.8 11/11/2010

 Using NAME file: mf2013_example.nam
 Run start date and time (yyyy/mm/dd hh:mm:ss): 2012/12/14 17:33:33

 Solving:  Stress period:     1    Time step:     1    Ground-Water Flow Eqn.
 Run end date and time (yyyy/mm/dd hh:mm:ss): 2012/12/14 17:33:35
 Elapsed run time:  1.953 Seconds

  Normal termination of simulation
```

MODFLOW ran in less than 2 seconds and no errors were reported. The next step is to read the heads from the output file and make a contour plot. The read statements are being modified at the time this paper is written, so please check the FloPy website to find out how to read the model results. Once the results are read, it is easy to create a contour plot with the Python package matplotlib:

```
# This is a placeholder – the real read statements is under development
h = read_modflow_head_output(name+'.hds')
x = y = linspace(0,L,N)
figure()
# Head at time 0, layer 0 (top), see Figure 1
c = plt.contour(x,y,h[0,0],arange(90,100.1,.2))
clabel(c,fmt='%1.1f')
axis('scaled')
```

```
figure()
# Head at timestep 0 layer -1 (bottom layer), see Figure 1
c = plt.contour(x,y,h[0,-1],arange(90,100.1,.2))
clabel(c,fmt='%1.1f')
axis('scaled')
figure()
z = linspace(-H/Nlay/2,-H+H/Nlay/2,Nlay)
# Vertical cross-section along row 50, see Figure 2
c = plt.contour(x,z,h[0,:,50,:],arange(90,100.1,.2))
```

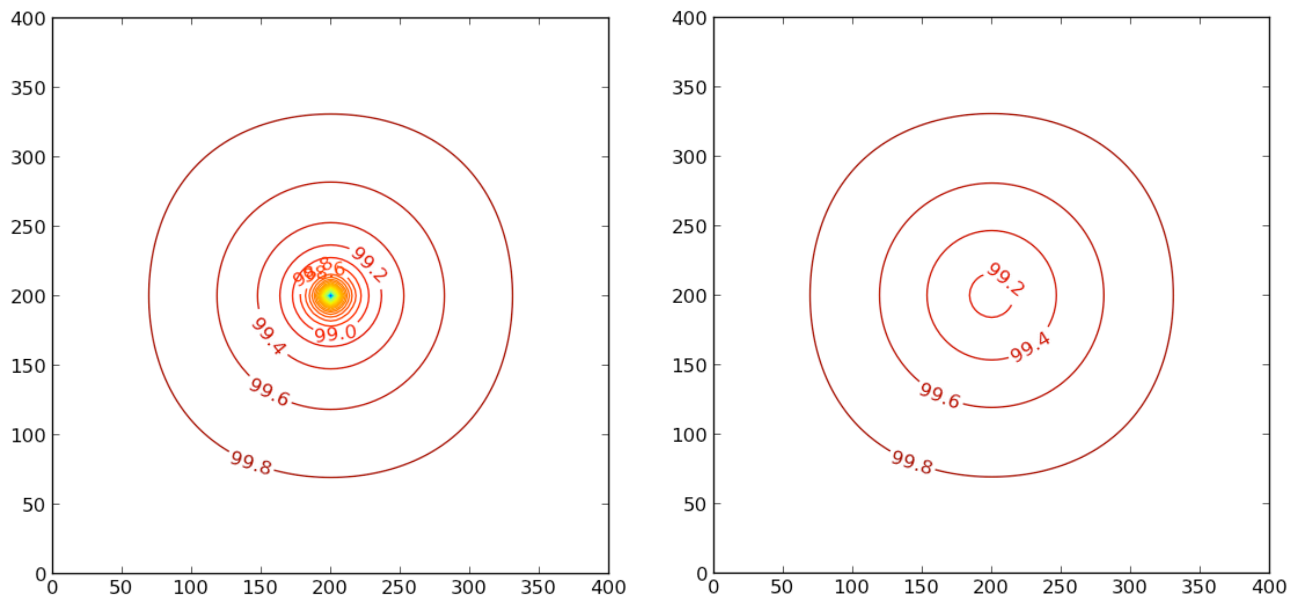

Figure. 1. Contour plot of heads in top layer (left) and bottom layer (right)

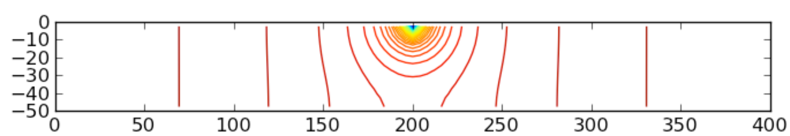

Figure. 2. Contour plot of heads in vertical cross-section through center of model

CONCLUSIONS

The FloPy package is a set of scripts to run MODFLOW and friends. FloPy can be downloaded from flopy.googlecode.com. We encourage contributions of code, documentation, or examples (preferably as iPython Notebooks).