SEAWAT with Flopy

Density-driven flow simulation

Harry Lee

3/14/2018

CEE 696

SEAWAT

- A computer program for simulation of 3D variable-density GW flow and transport
- We use SEAWAT version 4 released in 2014.
- Coupled version of MODFLOW and MT3DMS designed to simulate 3D variable-density, saturated ground-water flow.
- In specific, the variable-density ground-water flow equation is solved using a finite-difference approximation similar to the one solved by MODFLOW-2000. The solute- transport equation is solved using one of the approaches available with MT3DMS.

SEAWAT installation

For Windows:

1. Download executable
    - from `https:////water.usgs.gov/ogw/seawat/swt_v4_00_05.zip`
    - or go to SEAWAT webpage: `https://water.usgs.gov/ogw/seawat/index.html` and click "Download SEAWAT Version 4.00.05 program, source code, user guides, and example problems [12.3MB ZIP file] (updated October 19, 2012)"
2. Unzip the folders and copy exe/swt_v4.exe to your flopy working directory

For Mac and Linux:

1. Download or clone pyMake (`https://github.com/modflowpy/pymake`)
2. go to "examples" folder and run make_swtv4.py
3. copy "temp/swtv4" to your working directory

Example 3 - Henry problem (Henry [1965], Voss and Souza [1987])

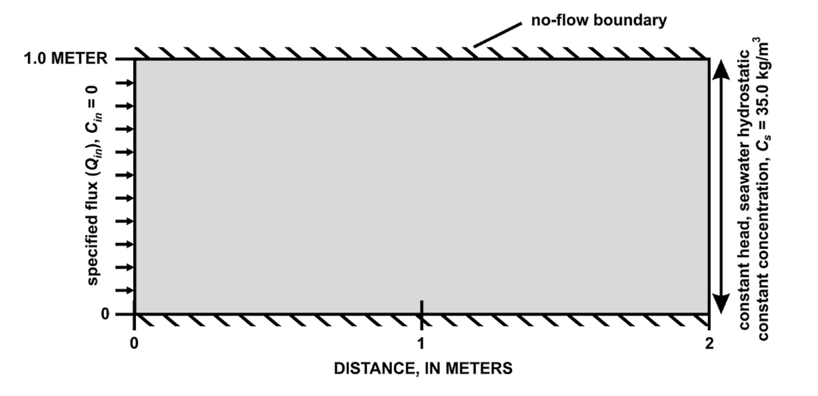


Figure 1: from figure 1 of Langevin and Guo (2006)

Run SEAWAT simulation

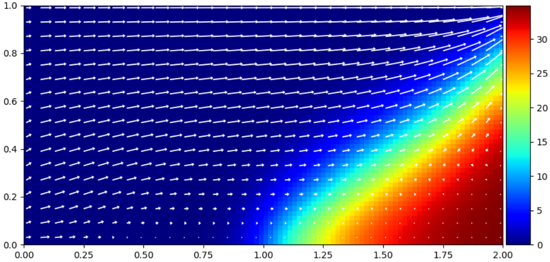
Download a script from

`https://www2.hawaii.edu/~jonghyun/classes/S18/CEE696/files/henry.py`

and run the script.

This script is adapted from `https://github.com/modflowpy/flopy/blob/develop/examples/Notebooks/flopy3_SEAWAT_henry_problem.ipynb`

Result



1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
30
25
20
15
10
5
0

Governing Equations

Hydraulic head h in freshwater is defines as

$$h = \frac{p}{\rho g} + z \tag{1}$$

h hydraulic head $[L]$
p pressure $[ML^{-1}T^{-2}]$
ρ fluid density $[ML^{-3}]$
g gravity $[LT^{-1}]$
z the upward coordinate direction aligned with g $[L]$

We assume negligible viscosity differences (with lots of assumptions!) here. Then, the general for of Darcy's law for variable-density conditions

$$q = -\frac{k}{\mu}\left(\nabla p + \rho g \nabla z\right) \tag{2}$$

where k is the permeability, $K = \frac{k\rho g}{\mu}$

$$\rho \approx \rho_f + \frac{\partial \rho}{\partial C} C \approx \rho_f + denseslp * C \tag{3}$$

Henry problem - Parameters

Table 1
Input and Numerical Solution Parameters for the Henry Problem

	Value
Input parameters	
Q_{in} (Henry)	5.702 m^2/d
Q_{in} (modified Henry)	2.851 m^2/d
C_{in}	0.0 kg/m^3
K_f	864 m/d
n	0.35
α_L, α_T	0 m
D_m	1.62925 m^2/d
C_s	35 kg/m^3
ρ_s	1025 kg/m^3
ρ_f	1000 kg/m^3
Numerical solution parameters	
Cell size (columns 1 to 20); dx, dz	0.1×0.1 m
Cell size (column 21); dx, dz	0.01×0.1 m
Solution of flow equation	
Matrix solution technique	PCG
Head convergence value	1×10^{-7} m
Flow convergence value	1×10^{-7} kg/d
Solution of transport equation	
Advection term	TVD
Dispersion and source terms	Implicit finite difference; generalized conjugate gradient
Time-step length	Calculated during simulation using Courant value of 0.1
Concentration convergence value	1×10^{-6}

Figure 2: from figure 2 of Langevin and Guo (2006) - not exactly same in our example

Example 3 - Henry problem (Henry [1965], Voss and Souza [1987])

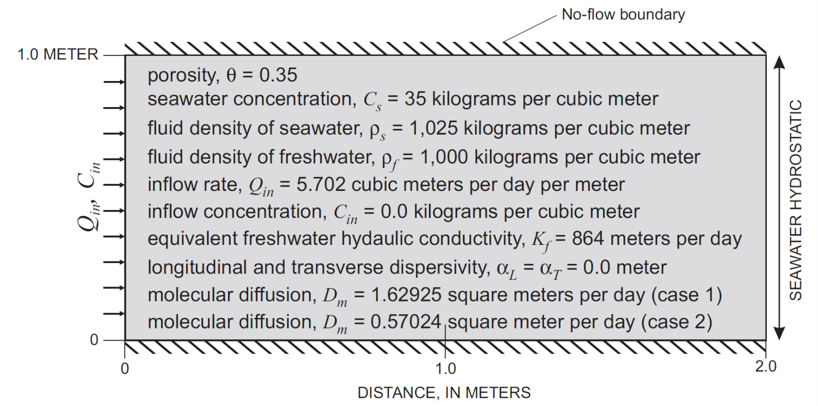


Figure 3: from figure 12 of User's Guide to SEAWAT (2002)

Input variables for the Henry Problem

```
Lx = 2.; Lz = 1.
nlay = 50; nrow = 1; ncol = 100
delr = Lx / ncol; delv = Lz / nlay
delc = 1.0
henry_top = 1.
henry_botm = np.linspace(henry_top-delv,0.,nlay)
qinflow = 5.702  #m3/day
dmcoef = 0.57024 #m2/day
hk = 864.  #m/day
```

Seawat Object

```
# Create the basic Seawat model structure
modelname = 'henry'
swt = flopy.seawat.Seawat(modelname,
                          exe_name='./swt_v4.exe')
```

For Linux and Mac users,

```
swt = flopy.seawat.Seawat(modelname,
                          exe_name='./swtv4')
```

Flow model setup (1)

```python
# Add DIS package to the MODFLOW model
dis = flopy.modflow.ModflowDis(swt, nlay, nrow, ncol,
                               nper=1,delr=delr,
                               delc=delc,laycbd=0,
                        top=henry_top,botm=henry_botm,
                               perlen=1.5, nstp=15)
# Variables for the BAS package
ibound = np.ones((nlay, nrow, ncol), dtype=np.int32)
ibound[:, :, -1] = -1 # right const. head bc
bas = flopy.modflow.ModflowBas(swt, ibound, 0)

# Add LPF package to the MODFLOW model
lpf = flopy.modflow.ModflowLpf(swt, hk=hk, vka=hk,
                               ipakcb=53)
```

Flow model setup (2)

```
# Add PCG Package to the MODFLOW model
pcg = flopy.modflow.ModflowPcg(swt, hclose=1.e-8)

# Add OC package to the MODFLOW model
oc = flopy.modflow.ModflowOc(swt,
stress_period_data={(0, 0): ['save head', 'save budget']
                              compact=True)
```

Almost same as what we did for MODFLOW before!

Boundary Conditions

```
itype = flopy.mt3d.Mt3dSsm.itype_dict()
wel_data = {} # for flow
ssm_data = {} # for transport
wel_sp1 = []
ssm_sp1 = []
for k in range(nlay):
    # Q = totalQ/nlay for each layer
    wel_sp1.append([k, 0, 0, qinflow / nlay])
    # zero concentration at the left boundary
    ssm_sp1.append([k, 0, 0, 0., itype['WEL']])
    # C = 35 at the right boundary
    ssm_sp1.append([k, 0, ncol - 1, 35., itype['BAS6']])
wel_data[0] = wel_sp1
ssm_data[0] = ssm_sp1
wel = flopy.modflow.ModflowWel(swt, stress_period_data=w
```

Treat left flux boundary as well injection

Transport Setup

```
btn = flopy.mt3d.Mt3dBtn(swt, nprs=-5, prsity=0.35,
                         sconc=35., ifmtcn=0,
                         chkmas=False, nprobs=10,
                         nprmas=10, dt0=0.001)
adv = flopy.mt3d.Mt3dAdv(swt, mixelm=0)
dsp = flopy.mt3d.Mt3dDsp(swt, al=0., trpt=1., trpv=1.,
                         dmcoef=dmcoef)
gcg = flopy.mt3d.Mt3dGcg(swt, iter1=500, mxiter=1,
                         isolve=1, cclose=1e-7)
ssm = flopy.mt3d.Mt3dSsm(swt,
                         stress_period_data=ssm_data)
```

Similar to typical MT3DMS setup

Seawat Variable-Density Flow (VDF) package

```
vdf = flopy.seawat.SeawatVdf(swt, iwtable=0, densemin=0,
                              densemax=0, denseref=1000.,
                              denseslp=0.7143, firstdt=1e-3)
```

iwtable a flag used to activate the variable-density water-table corrections (Guo and Langevin, 2002, eq. 82). If 0, the water-table correction will not be applied. If > 0, the water-table correction will be applied.

densemin the minimum fluid density if 0, no limitation

densemax the maximum fluid density if 0, no limitation

denseref the fluid density at the reference concentration, temperature, and pressure

denseslp the slope of the linear equation of state that relates fluid density to solute concentration

firstdt the length of the first transport timestep used to start the simulation if transport time step is larger than firstdt

Run SEAWAT simulation and plot results

```
# write inputs
swt.write_input()
# Run!
v = swt.run_model(report=True)

# Load data
import flopy.utils.binaryfile as bf

cnobj = bf.UcnFile('./MT3D001.UCN', model=swt)
times = ucnobj.get_times()
concentration = ucnobj.get_data(totim=times[-1])
cbbobj = bf.CellBudgetFile('./henry.cbc')
times = cbbobj.get_times()
qx = cbbobj.get_data(text='flow right face',
                     totim=times[-1])[0]
qz = cbbobj.get_data(text='flow lower face',
                     totim=times[-1])[0]
```

References

- SEAWAT version 4 manual https://pubs.usgs.gov/tm/tm6a22/pdf/tm6A22.pdf
- Lengevin and Guo, Groundwater [2006] https://water.usgs.gov/ogw/seawat/langevin_guo_GW2006.pdf