GLO-AEM: documentation of the Python script of the Analytical Element Model for the Gloucester subregion

Luk Peeters, 4/11/2014

This report documents the Python script of the Analytical Element Model for the Gloucester. The script is available at:

[\\OSM-07-CDC.it.csiro.au\OSM\_CBR\_LW\_BA\_working\SYD\pee035\GLO\_AEM\GLO\_AEM\_script.py](file:///\\OSM-07-CDC.it.csiro.au\OSM_CBR_LW_BA_working\SYD\pee035\GLO_AEM\GLO_AEM_script.py)

The script is divided in blocks by headings starting with #%%. This report is organised according these blocks

The script is developed using Python2.7 compiled for 64-bit, distributed by Anaconda (<https://store.continuum.io/cshop/anaconda/>)

# #%% Preamble

#%% Preamble

**import** numpy **as** np

**import** os **as** os

**import** ttim **as** ttim

Imports the necessary packages. Numpy and os are part of the Anaconda distribution. The ttim package has the analytic element algorithms. The 64-bit compiled ttim package is obtained from the developer, Mark Bakker (markbak@gmail.com), and is available at:

[\\OSM-07-CDC.it.csiro.au\OSM\_CBR\_LW\_BA\_working\SYD\pee035\GLO\_AEM\ttim](file:///\\OSM-07-CDC.it.csiro.au\OSM_CBR_LW_BA_working\SYD\pee035\GLO_AEM\ttim)

# #%% Set working directory, load data and set general model parameters

#%% Set working directory, load data and set general model parameters

wdir **=** r'\\OSM-07-CDC.it.csiro.au\OSM\_CBR\_LW\_BA\_working\SYD\pee035\GLO\_AEM'

os**.**chdir**(** wdir **)**

# CRDP

wells **=** np**.**loadtxt**(**'GLO\_AEM\_wellfile.csv'**,**delimiter**=**','**,**skiprows**=**1**)**

mine\_print **=** np**.**loadtxt**(**'GLO\_AEM\_mine\_footprint.csv'**,**delimiter**=**','**,**skiprows**=**1**)**

mine\_q **=** np**.**loadtxt**(**'GLO\_AEM\_mine\_waterproduction.csv'**,**delimiter**=**','**,**skiprows**=**1**)**

nwells **=** len**(**wells**)**

nmines **=** len**(**np**.**unique**(**mine\_print**[:,**0**]))**

# Parameters

params **=** np**.**loadtxt**(**'GLO\_AEM\_parameters.csv'**,**delimiter**=**','**,**skiprows**=**1**)**

coalseams **=** np**.**loadtxt**(**r'coalseams\coalseams%04i.csv' **%** int**(**params**[**0**]),**delimiter**=**','**,**skiprows**=**1**)**

majorfaults **=** np**.**loadtxt**(**r'majorfaults\majorfaults%04i.csv' **%** int**(**params**[**1**]),**delimiter**=**','**,**skiprows**=**1**)**

subfaults **=** np**.**loadtxt**(**r'subseismicfaults\subseismicfaults%04i.csv' **%** int**(**params**[**2**]),**delimiter**=**','**,**skiprows**=**1**)**

nseams **=** len**(**coalseams**)**

nfaults **=** len**(**np**.**unique**(**majorfaults**[:,**0**]))**

nsubfaults **=** len**(**np**.**unique**(**subfaults**[:,**0**]))**

# Output locations and time

gwreceptors **=** np**.**loadtxt**(**'GLO\_AEM\_gwreceptors.csv'**,** delimiter**=**','**,** skiprows**=**1**,** usecols**=(**0**,**3**,**4**,**7**))**

gwtargets **=** np**.**loadtxt**(**'GLO\_AEM\_targets.csv'**,** delimiter**=**','**,** skiprows**=**1**)**

outtime **=** 365.25**\***np**.**cumsum**(**np**.**concatenate**((**np**.**ones**(**9**)\***10**,**np**.**array**([**1**,**3**,**10**,**30**,**100**,**300**,**1000**,**3000**,**10000**]))))**

The first and second line define and change to the working directory. Next the CSG wells are read in from *GLO\_AEM\_wellfile.csv*. This file is generated by *GLO\_AEM\_stochastic\_generator.py[[1]](#footnote-1)*. Each line represents a single CSG well in the model. Table 1 describes the columns of the file.

Table 1: GLO\_AEM\_wellfile.csv

|  |  |
| --- | --- |
| Column Name | Description |
| # WellID | Well ID, integer |
| X | Easting (m) |
| Y | Northing (m) |
| tstart | Start time pumping (days since t0) |
| tend | End time pumping (day since t0) |
| CS1 – CS12 | A column for each coal seam with 1 indicating that the coal seam is screened by this well, 0 otherwise. |

Next the mine footprints are imported in the model from *GLO\_AEM\_mine\_footprint.csv* (Table 2). Each line in the file has a vertex of a polyline describing the circumference of a mine. The water production curve for each mine is read in from *GLO\_AEM\_mine\_waterproduction.csv* (Table 3).

Table 2: GLO\_AEM\_mine\_footprint.csv

|  |  |
| --- | --- |
| Column Name | Description |
| MineID | Mine ID, integer |
| X | Easting (m) |
| Y | Northing (m) |

Table 3: GLO\_AEM\_mine\_waterproduction.csv

|  |  |
| --- | --- |
| Column Name | Description |
| MineID | Mine ID, integer |
| Time (years) | Time in years when a change in water production occurs. |
| Flow rate (m3/yr) | Water production in m3/yr. Last line for each mine should have zero flow rate and represent the date the mine is closed (no active extraction of water). |

The next two lines compute and assign the number of well screens[[2]](#footnote-2) and number of mines to **nwells** and **nmines**.

The following four lines read parameters and stochastic components of the model. First the second column of parameter file *GLO\_AEM\_parameters.csv* (Table 4) is loaded. This file contains all parameters of the model that are to be included in the formal uncertainty analysis.

Table 4: GLO\_AEM\_parameters.csv

|  |  |
| --- | --- |
| Column Name | Description |
| ParamName | Parameter name |
| Value | Parameter value |
| Description | Short description of parameter |
| Python Index | Index of column in params in GLO\_AEM |

The first three entries of GLO\_AEM\_parameters.csv are integers pointing to individual stochastic realisations of the coal seam, major fault and subseismic fault distribution that will be loaded for the simulation. For each of those aspects 1000 stochastic realisations are created with GLO\_AEM\_stochastic\_generation.py and stored in subfolders \coalseams, \majorfaults and \subseismicfaults. Table 5,

Table 6 and Table 7 describe the structure of the files in those folders

Table 5: \coalseams\coalseamsxxxx.csv

|  |  |
| --- | --- |
| Column Name | Description |
| # CS\_ID | Coal seam ID |
| Depth (m) | Depth of center of coal seam (m below surface) |

Table 6: \majorfaults\majorfaultsxxxx.csv

|  |  |
| --- | --- |
| Column Name | Description |
| # FaultID | Fault ID |
| X | Easting (m). Each line in the file is a vertex of a fault polyline |
| Y | Northing (m) |

Table 7: \subseismicfaults\subseismicfaultsxxxx.csv

|  |  |
| --- | --- |
| Column Name | Description |
| # FaultID | Fault ID |
| X | Easting (m). Each line in the file is a vertex of a fault polyline |
| Y | Northing (m) |
| Top Coal Seam | Integer indicating the top of the coal seam intersected by the subseismic fault (assumes subseismic faults intersect 3 subsequent coal seams) |

The number of coal seams, major and subseismic faults are assigned to **nseams**, **nfaults** and **nsubfaults** respectively.

The final subsection reads in the locations for which to produce output. This includes the groundwater receptor locations outside the alluvium (*GLO\_AEM\_gwreceptors.csv*) and the target locations (*GLO\_AEM\_targets.csv*) for the interpolation of GLO\_AEM results to the alluvial MODFLOW models of the Avon and Karuah.

Table 8: GLO\_AEM\_gwreceptors.csv

|  |  |
| --- | --- |
| Column Name | Description |
| OID\_ | Receptor ID (integer) |
| Bore\_ID | Receptor label (text) |
| Depth\_m | Depth in m below surface, -999 is nodata value |
| Easting | Easting (m) |
| Northing | Northing (m) |
| System | Projection system |
| Source | Source of data |
| Use | Negative numbers indicate output needs to be calculated at this point. Positive values indicate that the receptor is within 200m of another receptor. The integer is the OID\_ from which to take the results. |

Table 9: GLO\_AEM\_targets.csv

|  |  |
| --- | --- |
| Column Name | Description |
| X | Easting (m). Each line in the file is a vertex of a fault polyline |
| Y | Northing (m) |

The last line of the section creates variable **outtime**; the time in days for which to produce output. It produces output every ten years for the first 90 years and then a quasi-logarithmic series (from the groundwater modelling methodology).

# #%% Create layer structure

#%% Create layer structure

thick\_ws **=** 75.0 # thickness weathered zone

thick\_cs **=** 3.0 # thickness of coal seam

naq **=** nseams**+**1 # number of aquifers

nll **=** nseams # number of leaky layers

zaq **=** np**.**zeros**(**2**\***naq**+**1**)**

zaq**[**0**]** **=** 1.0

zaq**[**2**]** **=** **-**thick\_ws # bottom of weathered zone

zaq**[**3**:(**2**\***naq**+**1**):**2**]** **=** **-**coalseams**[:,**1**]+**0.5**\***thick\_cs # top coal seams

zaq**[**4**:(**2**\***naq**+**1**):**2**]** **=** **-**coalseams**[:,**1**]-**0.5**\***thick\_cs # bottom coal seams

zmid **=** zaq**[**0**:-**1**]+**0.5**\***np**.**diff**(**zaq**)** # position of center of each layer

zmid\_aq **=** zmid**[**1**:**len**(**zaq**):**2**]** # center of all aquifers

zmid\_ll **=** zmid**[**0**:**len**(**zaq**):**2**]** # center of all leaky layers separating aquifers

zmid\_ll**[**0**]** **=** zmid\_aq**[**0**]**

This section creates the layer structure of alternating aquifer and leaky layers from the distribution of coal seams. The first layer is a leaky layer with nominal thickness of 1m, followed by the weathered zone. The thickness of the weathered zone and coal seams are set at 75m and 3m. These are not varied during uncertainty analysis.

# #%% Hydraulic properties: P=a1\*exp(a2\*depth)

#%% Hydraulic properties: P=a1\*exp(a2\*depth)

K\_a1 **=** params**[**3**:**5**]** # a1 for [K interburden, K coal seam ]

K\_a2 **=** params**[**5**:**7**]** # a2 for [K interburden, K coal seam ]

S\_a1 **=** params**[**7**:**9**]** # a2 for [S interburden, S coal seam ]

S\_a2 **=** params**[**9**:**11**]** # a2 for [S interburden, S coal seam ]

KvKh **=** params**[**11**]** # Kv over Kh for interburden

Kaq **=** K\_a1**[**1**]\***np**.**exp**(**K\_a2**[**1**]\***zmid\_aq**)**\*365.25 # m/d to m/yr

Kll **=** K\_a1**[**0**]\***np**.**exp**(**K\_a2**[**0**]\***zmid\_ll**)**\*365.25 # m/d to m/yr

Saq **=** S\_a1**[**1**]\***np**.**exp**(**S\_a2**[**1**]\***zmid\_aq**)**

Sll **=** S\_a1**[**0**]\***np**.**exp**(**S\_a2**[**0**]\***zmid\_ll**)**

cll **=** **-**np**.**diff**(**zaq**)[**0**::**2**]** **/** **(**Kll**\***KvKh**)**

fhres **=** 1.0**/**params**[**12**]** # fault entry resistance as d/Kfh with fault width nominally d = 1 m

fvres\_m **=** 1000.0**/**(params**[**13**]**\*365.25) # flow resistance in fault (vertical) as d/Kfv with major fault length nominally d=1000m

fvres\_s **=** 100.0**/**(params**[**13**]**\*365.25) # flow resistance in fault (vertical) as d/Kfv with subseismic fault length nominally d=100m

This section assigns the hydraulic properties to the aquifers and aquitards. Both Kh and S are calculated with a 2-parameter exponential function of depth. The Kh/Kv for aquitards is set as a ratio.

Numpy arrays **Kaq** and **Kll** have the hydraulic conductivity values for aquifers and aquitards respectively and likewise **Saq** and **Sll** have the storage for aquifers and aquitards. In the AEM the permeability of aquitards or leaky layers is expressed as a flow resistance term, **cll**, the ratio of layer thickness over vertical hydraulic conductivity (Kh x KvKh).

The last three lines compute the entry resistance for faults and the within flow resistance for major and subseismic faults from the horizontal and vertical hydraulic conductivity with assumed nominal fault width and length.

# #%% Create basic model to simulate water production curve

This section creates the basic model that will be used to calculate the amount of water pumped by each CSG well.

#%% Create basic model to simulate water production curve

ml **=** ttim**.**ModelMaq**(**kaq**=**Kaq**,**

z**=**zaq**[**1**::],**

c**=**cll**[**1**::],**

Saq**=**Saq**,**

Sll**=**Sll**[**1**::],**

topboundary**=**'imp'**,**

phreatictop**=False,**

tmin **=** .01**,**

tmax **=** wells**[:,**4**].**max**(),**

M **=** 15 **)**

# Add wells

**for** i **in** range**(**nwells**):**

well\_layers **=** np**.**argwhere**(**wells**[**i**,**5**::]>**0**)+**2

ttim**.**HeadWell**(**ml**,**

xw**=**wells**[**i**,**1**],**

yw**=**wells**[**i**,**2**],**

rw**=**.25**,**

tsandh **=** **[**wells**[**i**,**3**],**zmid\_aq**[**well\_layers**[**0**]]+**25**],**

res**=**100.0**,**

layers**=**well\_layers**.**flatten**(),**

label**=**'well\_'**+**str**(**int**(**wells**[**i**,**0**]))** **)**

# Add major faults

**for** i **in** range**(**nfaults**):**

xy\_f **=** majorfaults**[**majorfaults**[:,**0**]==**np.unique(majorfaults[:,0])[i]**,**1**:**3**]**

ttim**.**ZeroMscreenLineSinkString**(**ml**,**xy**=**xy\_f**,**res**=**fhres**,**wh**=**'H'**,**

layers**=**range**(**naq**),**vres**=**fvres\_m**,**wv**=**.1**,**

label**=**'Fault\_v\_'**+**str**(**i**))**

# Add subseismic faults

**for** i **in** range**(**nsubfaults**):**

xy\_f **=** subfaults**[**subfaults**[:,**0**]==**i**+**1**,**1**:**4**]**

cs\_f **=** subfaults**[**subfaults**[:,**0**]==**i**+**1**,**3**][**0**]**

ttim**.**ZeroMscreenLineSinkString**(**ml**,**xy**=**xy\_f**,**res**=**fhres**,**wh**=**'H'**,**

layers**=**range**(**int**(**cs\_f**),**int**(**cs\_f**+**4**)),** vres**=**fvres\_s**,**wv**=**.1**,**

label**=**'SubFault\_v\_'**+**str**(**i**))**

# solve model

ml**.**solve**()**

# compute water production curve

wprodlen **=** 20

Q **=** np**.**zeros**(** **(** **(**wprodlen**),**2**,**nwells**)** **)**

**for** i **in** range**(**nwells**):**

Q**[**0**:**wprodlen**,**0**,**i**]** **=** np**.**linspace**(**wells**[**i**,**3**],**wells**[**i**,**4**],**wprodlen**)**

Q**[**0**:**wprodlen**-**1**,**1**,**i**]** **=** ml**.**strength**(**'well\_'**+**str**(**int**(**wells**[**i**,**0**])),**Q**[**1**:**wprodlen**,**0**,**i**])[**0**,:]**

The first subsection initialises the basic model. See the TTim manual for details on the function variables (Bakker, 2012). The simulation time is limited to the maximum of the planned CSG pumping time.

Section ‘# Add Wells’ adds the CSG wells as a head dependent boundary. A single well is screened in all the coal seam layers indicated in the well file. The head is set equal to the top of the uppermost coal seam plus 25m. The well diameter and entry resistance are nominally set to 0.25m and 100d.

In section ‘Add major faults’ all the major faults are implemented as linesinks with entry resistance fhres and fault resistance **fvres\_m**. Major faults hydraulically connect the entire aquifer-aquitard sequence

Section ‘Add subseismic faults’ implements the subseismic faults. This identical to the major faults, except the fault resistance is **fvres\_s** and fault only connect 3 coal seams in the sequence, the top of those three is taken from the *subseismicfaultxxxx.csv* file.

The ml.solve() command solves the set of equations that forms the AEM model. The following section evaluates the model to obtain the flow rate from each individual well.

# #%% Recreate model, wells now specified flux

#%% recreate model, wells now specified flux

## Create model

ml **=** ttim**.**ModelMaq**(**kaq**=**Kaq**,**

z**=**zaq**[**1**::],**

c**=**cll**[**1**::],**

Saq**=**Saq**,**

Sll**=**Sll**[**1**::],**

topboundary**=**'imp'**,**

phreatictop**=False,**

tmin **=** .1**,**

tmax **=** outtime.max()**,**

M **=** 20 **)**

# Add wells

**for** i **in** range**(**nwells**):**

timeq **=** Q**[:,:,**i**]**

well\_layers **=** np**.**argwhere**(**wells**[**i**,**5**::]>**0**)+**2

ttim**.**Well**(**ml**,**

xw**=**wells**[**i**,**1**],**

yw**=**wells**[**i**,**2**],**

rw**=**0.25**,**

tsandQ **=** timeq**,**

res**=**100.0**,**

layers**=**well\_layers**.**flatten**(),**

label**=**'well'**+**str**(**i**))**

# Add major faults

**for** i **in** range**(**nfaults**):**

xy\_f **=** majorfaults**[**majorfaults**[:,**0**]==**np**.**unique**(**majorfaults**[:,**0**])[**i**],**1**:**3**]**

ttim**.**ZeroMscreenLineSinkString**(**ml**,**xy**=**xy\_f**,**res**=**fhres**,**wh**=**'H'**,**

layers**=**range**(**naq**),**vres**=**fvres\_m**,**wv**=**0.1**,**

label**=**'Fault\_v\_'**+**str**(**i**))**

# Add subseismic faults

**for** i **in** range**(**nsubfaults**):**

xy\_f **=** subfaults**[**subfaults**[:,**0**]==**i**+**1**,**1**:**4**]**

cs\_f **=** subfaults**[**subfaults**[:,**0**]==**i**+**1**,**3**][**0**]**

ttim**.**ZeroMscreenLineSinkString**(**ml**,**xy**=**xy\_f**,**res**=**fhres**,**wh**=**'H'**,**

layers**=**range**(**int**(**cs\_f**),**int**(**cs\_f**+**4**)),**vres**=**fvres\_s**,**wv**=**.1**,**

label**=**'SubFault\_v\_'**+**str**(**i**))**

# Add mines

**for** i **in** range**(**nmines**):**

mine\_tsandQ **=** mine\_q**[**mine\_q**[:,**0**]==**np**.**unique**(**mine\_q**)[**i**],**1**::]**

mine\_xy **=** mine\_print**[**mine\_print**[:,**0**]==**np**.**unique**(**mine\_print**)[**i**],**1**::]**

ttim**.**MscreenLineSinkDitchString**(** ml**,**

xy**=**mine\_xy**,**

tsandQ**=**mine\_tsandQ**,**

res**=**0**,** wh**=**'H'**,**layers**=**1**,**

label**=**'Mine\_'**+**str**(**i**+**1**)** **)**

# solve model

ml**.**solve**()**

This section creates and solves the AEM model used for predictions. It is almost identical to section 5, except for **tmax** (which is set to the maximum simulation time needed), the implementation of the wells, which are now flux specified based on the first run of the model.

The final change is adding the mines as a line feature to the first layer from which the mine dewatering volumes are extracted.

# #%% Compute outputs

#%% Compute outputs

# groundwater receptor locations

h\_ind **=** np**.**argwhere**(**gwreceptors**[:,**3**]<**0**)**

h\_gwreceptors **=** np**.**zeros**((**len**(**gwreceptors**),**len**(**outtime**)))**

**for** i **in** h\_ind**:**

h\_gwreceptors**[**i**,:]** **=** ml**.**head**(**gwreceptors**[**i**,**1**],**

gwreceptors**[**i**,**2**],**

outtime**,**layers**=**1**)**

h\_notind **=** np**.**argwhere**(**gwreceptors**[:,**3**]>**0**)**

**for** i **in** h\_notind**:**

h\_gwreceptors**[**i**,:]=**h\_gwreceptors**[**gwreceptors**[:,**0**]==** gwreceptors**[**i**,**3**],:]**

np**.**savetxt**(**'GLO\_AEM\_h\_gwreceptors.csv'**,**

np**.**concatenate**((**gwreceptors**,**h\_gwreceptors**),**axis**=**1**),**

delimiter **=** ','**,**

header **=** 'OID\_,Easting,Northing,Use,'**+** ','**.**join**(**np**.**char**.**mod**(**'t=%i'**,**outtime**)),**

fmt **=** '%3i,%10.2f,%10.2f,%3i'**+**len**(**outtime**)\***',%10.5e' **)**

# interpolation points for modflow alluvial models

h\_targets **=** np**.**ones**((**len**(**gwtargets**),**len**(**outtime**)))**

**for** i **in** range**(**len**(**h\_targets**)):**

h\_targets**[**i**,:]** **=** ml**.**head**(**gwtargets**[**i**,**0**],**gwtargets**[**i**,**1**],**outtime**,**layers**=**1**)**

**print** i

np**.**savetxt**(**'GLO\_AEM\_h\_gwtargets.csv'**,**

np**.**concatenate**((**gwtargets**,**h\_targets**),**axis**=**1**),**

delimiter **=** ','**,**

header **=** 'Easting,Northing,'**+** ','**.**join**(**np**.**char**.**mod**(**'t=%i'**,**outtime**)),**

fmt **=** '%10.2f,%10.2f'**+**len**(**outtime**)\***',%10.5e' **)**

The last section computes the drawdown at the locations specified in **gwreceptors** and **gwtargets** at the times in **outtimes**. These are saved to *‘GLO\_AEM\_h\_gwreceptors.csv’* and *‘GLO\_AEM\_h\_targets.csv’*. The reported values are the heads expressed relative to initial conditions that are assumed to be zero and therefore all values are negative. Note that these files may have computed values beyond the time defined as tmax in section 6. These are not to be trusted (Python will produce warnings as well).

# References

Bakker, M. (2012) TTim. A multi-aquifer transient analytic element model. Version 0.21. Water Resources Section, Civil Engineering and Geosciences. Delft University of Technology, Delft, The Netherlands, <https://code.google.com/p/ttim/>

1. Separate script that generates the stochastic components of the GLO\_AEM model. Documented in *GLO\_AEM\_stochastic\_readme.docx* [↑](#footnote-ref-1)
2. Each well screen is treated as an individual well in the AEM model. [↑](#footnote-ref-2)